ALGORITMLASH VA DASTURLASH ASOSLARI

B.J. BOLTAYEV, A.R. AZAMATOV, A.D. RAHIMOV,
B.A. AZAMATOV, D.T. ASRAYEVA, SH.Z. QAMBARALIYEV

C++ TILI ASOSLARI

Amaliy qo'llanma

A11+

ALGORITMLASH VA DASTURLASH ASOSLARI

B.J. BOLTAYEV,
A.R. AZAMATOV, A.D. RAHIMOV,
B.A. AZAMATOV, D.T. ASRAYEVA, SH.Z. QAMBARALIYEV

C++ TILI ASOSLARI

AMALIY QO'LLANMA

Nazariya, masalalar, mulohazalar, yechimlar, tavsiyalar

A11+

Toshkent
«Akademnashr»
2021

UO'K: 004.43 (04)
KBK: 32.973.26-018
C 49

C 49 C++ tili asoslari [Matn]: ilmiy-ommabop / B. J. Boltayev [va boshq.]. – Toshkent: Akademnashr, 2021. – 404 b.

ISBN 978-9943-6501-4-5

UO'K: 004.43
KBK: 32.973.26-018

Dasturlash tillariga oid bilimlarini rivojlantiruvchilar uchun qo'llanma

Mualliflar:
B.J. Boltayev, fizika-matematika fanlari nomzodi, O'zbekistonda xizmat ko'rsatgan yoshlar murabbiysi,
A.R. Azamatov, fizika-matematika fanlari nomzodi, oliy toifali informatika va matematika fani o'qituvchisi,
A.D. Rahimov, Innopolis universiteti talabasi,
B.A. Azamatov, dasturchi-muhandis,
D.T. Asrayeva, informatika fani o'qituvchisi,
Sh.Z. Qambaraliyev, dasturchi-muhandis

Mas'ul muharrir:
B.S. XURRAMOV

Ushbu qo'llanmada C++ dasturlash tili va elementlari, dastur tarkibi va tavsifi, unda dasturlash imkoniyatlari va usullari, tuzilmalar mohiyati erkin tarqatiladigan Codeblocks IDE asosida ochib berishga qaratilgan. Shu bilan birga, umumiy va nazariy ma'lumotlar, dastur namunalaridan tashqari takrorlash va mustaqil ishlash uchun ko'p sonli vazifalar keltirilgan. Keltirilgan vazifalarni hal etishda turli yondashuvlar va usullarning mohiyati ochib berishga harakat qilingan. Ma'lumotlar va vazifalar "oddiydan murakkabga" mantiqiy ketma-ketlikda bayon etilgan. Vazifalar qiyinlik darajasi bo'yicha A, B, C va D turkumlarga ajratilgan.

Qo'llanmadan o'qituvchilar, o'quvchilar, talabalar, umuman, dasturlash san'ati bilan qiziquvchilar foydalanishlari mumkin. Qo'llanma 11 va undan katta yoshli qiziquvchilar uchun mo'ljallangan.

TAQRIZCHILAR:
S.I. Rahmonqulova, fizika-matematika fanlari nomzodi,
G.A. Ishanxodjayeva, informatika fani o'qituvchisi

ISBN 978-9943-6501-4-5

Bevosita yoki bilvosita ustozlik qilgan
aziz insonlar xotirasiga bag'ishlanadi!

SO'ZBOSHI

Ushbu qo'llanma "Algoritmlash va dasturlash asoslari" seriyasiga bag'ishlangan navbatdagi kitob bo'lib, u dasturlash tillaridan biri bo'lgan C++ tilining elementlari, dastur tarkibi va tavsifi, unda dasturlash imkoniyatlari va usullari, tuzilmalarining mohiyati erkin tarqatiladigan Codeblocks IDE asosida ochib berishga bag'ishlangan. Qo'llanmada masalalar yechishda foydalaniladigan matematika fanidagi tayanch va qiziqarli ma'lumotlar keltirilgan bo'lib, foydalanuvchi o'z bilim doirasiga mos qismini ushbu va qo'shimcha qo'llanmalar asosida to'ldirishi hamda rivojlantirishi mumkin.

Ma'lumotlar va vazifalar "oddiydan murakkabga" mantiqiy ketma-ketlikda bayon etilgan. Keltirilgan masalalar mualliflar nazaridagi qiyinlik darajasi bo'yicha A, B, C va D guruhlarga ajratildi. Shu sababli kitobxonlarga qo'llanmadan quyidagicha foydalanishni tavsiya etamiz:

- Dasturlash bilan ilk marta tanishayotgan foydalanuvchilar faqat A guruhi masalalarini ishlab chiqishlari;
- Dasturlash haqida boshlang'ich ma'lumotga ega bo'lgan foydalanuvchilar A va B guruhi masalalarini ishlab chiqishlari;
- Dasturlash bo'yicha tajribaga ega foydalanuvchilar A, B va C guruhi masalalarini ishlab chiqishlari;
- A, B va C guruhi masalalarini osonlikcha yecha olgan foydalanuvchilar D guruhi masalalarini ishlab chiqishlari maqsadga muvofiqdir.

Ma'lumki, dasturlash masalalari turli maxsus internet saytlarida ham berilgan bo'lib, foydalanuvchi tuzgan dastur to'g'riligini ularda yuzlab test orqali tekshirish imkoni ham mavjud. Shu sababli D guruhi masalalarini erkin yecha boshlagan foydalanuvchilarga https://codeforces.com saytining arxiviga kiritilgan, qiyinlik darajasi 1200 dan kichik bo'lgan (matni rus va ingliz tillarida bayon etilgan) masalalarni yechib chiqishni maslahat beramiz. Mazkur saytda masalalar yechish uchun tahlil ma'lumotlari ham keltirilgan.

Adabiyotlar ro'yxatida foydalanuvchilar e'tiboriga havola etilgan saytlarda ham tahliliy ma'lumotlar va juda ko'p masalalar tavsiya etilgan bo'lib, bu saytlar foydalanuvchilar o'z bilimlarini mustahkamlash va rivojlantirishlari uchun foydali va qiziqarli manba sifatida xizmat qiladi.

E'tiboringizga havola etilayotgan ushbu amaliy qo'llanma Sizga C++ dasturlash tilini oson o'rganishda va uning sir-asrorini bilib olishingizda muhim ko'makchi bo'ladi deb umid qilamiz.

Mualliflar

KIRISH

C++ dasturlash tili tarixi

Birinchi ishlab chiqarilgan operatsion sistemalar har bir kompyuter platformasi uchun alohida yozilar edi. Bir kompyuter uchun yozilgan operatsion sistema kodlarini boshqa kompyuter platformasiga o'tkazish juda ko'p vaqt va mehnat talab qiladigan ish hisoblanadi.

Shu kamchilikni bartaraf etish yo'lida 1965-yildan boshlab **Bell Telephone Laboratories, General Electric Company** va Massachusets texnologiya instituti tomonidan yuzlab foydalanuvchilarga xizmat ko'rsata oladigan **Multics** operatsion sistemasini ishlab chiqishga kirishildi. Ammo 1969-yilda Bell Telephone Laboratories loyihadan chiqib ketgach bu ish amalga oshmadi. Shunga qaramay **Bell laboratoriyasi** xodimlari Denis Ritchi (Dennis Ritchie) va Ken Tompson (Kenneth Thompson) loyiha ustida ishlashni davom ettirishdi hamda 1971-yili kodlari to'liq assemblerda (assembly language) yozilgan, Multiks so'ziga ohangdosh **UNIX** (o'qilishi: Yuniks) nomli operatsion sistemasini ishlab chiqishdi.

Dasturlash jarayonini osonlashtirish uchun Ken Tompson **B** (o'qilishi: bi) nomli tilni ishlab chiqdi. Denis Ritchi esa keyinchalik bu tilni o'zgartirib, 1972-yili **C** (o'qilishi: si) tilini ishlab chiqdi. Ma'lumki, C tilining sintaksisi C++ (o'qilishi: si plyus plyus), C# (o'qilishi: si sharp), Java (o'qilishi: java) tillari uchun asos bo'lib xizmat qiladi. 1974-yilda e'lon qilingan UNIX operatsion sistemasi dunyo dasturchilari tan olgan juda kuchli operatsion sistemalardan biri hisoblandi. Ko'p foydalanuvchili UNIX operatsion sistemasining o'zagi yuqori darajali **C** dasturlash tilida va faqat 10 foizga yaqini (bir necha sahifasi, deyarli 1000 ta satri) assemblerda yozilgan edi. Shu sababli bir necha oyda uni boshqa kompyuter platformalariga o'tkazish mumkin edi, qo'shimchalar va o'zgartirishlar kiritish esa juda osonlashdi. Ta'kidlash mumkinki, UNIX birinchi ko'chirib o'tkazish mumkin bo'lgan operatsion sistema bo'ldi. Uning ishlab chiqarilgan barcha naqllariga o'zgartirishlar kiritish oson edi.

C dasturlash tilida imperativ (dastur holatini o'zgartiruvchi ko'rsatmalar orqali ifodalash) va strukturali (bloklarni shajarali struktura ko'rinishida ifodalash) dasturlash paradigmalarini (dasturlash **paradigma**si – dasturlashga bo'lgan **yondoshuv**) qo'llangan. C dasturlash tilining ANSI-C, ISO C, C99, C11 standartlari mavjud. C dasturlash tili operatsion sistemalar yozishda eng ko'p qo'llanadigan til hisoblanadi.

C++ dasturlash tili C dasturlash tiliga asoslangan bo'lib, **Bell laboratoriyasi** xodimi, daniyalik Byorn Straustrup (Bjarne Stroustrup) tomonidan 1985-yili ishlab chiqilgan. Stroustrup xizmat vazifasini bajarish vaqtida Simula-67 dasturlash tilida ishlaydi. Simula-67 obyektga yo'naltirilgan dasturlash tili bo'lsa-da, juda sekin ishlar hamda asosan simulyatsiyaga mo'ljallangan edi. Shu sababli Stroustrup 1979-yilda juda tezkor bo'lgan C dasturlash tiliga asoslangan va obyektga yo'naltirilgan dasturlashni o'z ichiga olgan yangi "C sinflar

bilan" ("C with Classes") dasturlash tili ustida ishlay boshlaydi. 1983-yilda yangi til nomi C++ kabi o'zgartirilgan va bir qator yangi imkoniyatlar (virtual funksiyalar, const kalit so'zi va boshqalar) qo'shildi.

C++ dasturlash tili birinchi marta 1985-yili keng foydalanishga chiqarilgan. Shundan keyin ham tilga bir qator yangi o'zgartirishlar kiritildi. C++ dasturlash tili shunchalik ommalashib ketdiki, 1998-yili C++ uchun birinchi xalqaro standart C++98 ishlab chiqildi. Bu standart STL (Standard Template Library – Standart shablonlar kutubxonasi)ni o'z ichiga olgan.

Keyinchalik, C++03, C++11, C++14 va C++ 17 standartlari ishlab chiqildi. Har bir yangi standartda avvalgi standartlardagi xato va kamchiliklar tuzatilib, yangi imkoniyatlar qo'shildi yoki ba'zi bir eski (keraksiz deb topilgan) imkoniyatlar (funksiyalar, konstantalar, kutubxonalar) olib tashlandi. C++ dasturlash tilining boshqa dasturlash tillaridan afzal tomonlari – bu uning o'ta tezkorligi, foydali funksiyalarga boyligi, obyektga yo'naltirilgan va umumlashgan dasturlash (generic) paradigmalarni qo'llashi, xotira adreslariga to'g'ridan to'g'ri murojaat eta olishidir. Bu va boshqa ko'pgina sifatlari bilan C++ tili boshqa dasturlash tillaridan ajralib turadi. C++ dasturlash tili 3D o'yinlar, GUI (Graphic User Interface – foydalanuvchining grafikli interfeysi) dasturlar, web brauzerlar, ma'lumotlar omborini boshqarish sistemalari, kompilyatorlar, tibbiyotga oid dasturlar, operatsion sistemalarni ishlab chiqarishda keng qo'llaniladi.

Kompilyatorlar va IDE

C++ dasturlash tilida yozilgan dastur kompilyator deb ataluvchi dastur yordamida kompyuter tushunadigan tilga tarjima qilinadi. **Kompilyator** – bir tilda yozilgan dasturni boshqa tilga tarjima qiladigan dastur. Odatda, kompilyatorlar yuqori darajali dasturlash tillarida (C++) yozilgan dasturni quyi darajali dasturlash tillariga (Assembly) yoki to'g'ridan to'g'ri mashina kodiga tarjima qiladi. **Mashina kodi – kompyuter tushunadigan yagona til.** Bu til protsessor arxitekturasiga bog'liq bo'ladi. Masalan, Intel x86 (x64 – x86 ning kengaytirilgan varianti), ARM, MIPS kabi arxitekturalar mavjud.

C++ dasturlash tili yordamida ixtiyoriy dastur faqatgina C++ kompilyatori va matn muharriri yordamida tuzilishi mumkin. Ammo dasturchiga qulayliklar yaratish va ishini tezlashtirish maqsadida turli xil **integrallashgan takomillashtirish muhit**lar chiqarilgan.

IDE. Dasturchilar uchun muhim atama bo'lgan **integrallashgan takomillashtirish muhiti** (ITM, ingl. IDE – Integrated Development Environment), qisqacha **muhit** deb atash ham mumkin (shu birgina **muhit** so'zi bilan integrallashgan ekanligi ta'kidlanadi), dasturlash tillarining takomillashib borishi hamda dastur interfeysi yoki dasturlash muhitini takomillashib borishi bilan bog'liqdir. IDE o'z ichiga turli oyna va boshqaruv elementlarini olgan bo'lib, ular dasturchi va dasturlash muhiti orasidagi bog'lanishni ta'minlashga xizmat qiladi.

Dasturchi muhit yordamida osongina amaliy dasturlar va ilovalar tayyorlashda ilovaning interfeys qismini loyihalashi, dastur kodini yozishi va uni boshqaruv elementlari bilan bog'lashi mumkin. Turli xil IDE lar turli xil dasturlash tillarida ishlash uchun moslashtirilgan bo'ladi.

C++ dasturlash tili bilan ishlash uchun ham juda ko'p IDE lar mavjud. Masalan, Codeblocks, Codelite, C-Lion va Visual Studio shular jumlasidandir. Bunday IDE larning afzallik tomonlari quyidagilardan iborat:

- dasturchi tomonidan yozilgan kod ranglar bilan ajratib ko'rsatiladi va buning natijasida dasturchi tezlik bilan kerakli nuqtani topishi mumkin;
- kodni to'ldirish funksiyasi – funksiya, o'zgaruvchi, o'zgarmas va boshqa nomlar boshlang'ich qismi yozilishi bilan, shu prefix ga mos keladigan nomlarni tanlash uchun ko'rsatadi va buning natijasida dasturchi o'sha nomni oxirigacha yozishi shart emas;
- dasturdagi xatoni topish uchun qulayliklar;
- kompilyatsiya qilish va dasturni ishga tushirish o'ta qulay.

Bu qo'llanmadagi mashqlar asosan Codeblocks IDE asosida bajarilishi ko'zda tutilgan.

C++ kompulyatorlari orasida eng keng tarqalganlari – GNU GCC, Microsoft Visual C++ Compiler, CygWin, MinGW va boshqalar. Bulardan eng keng tarqalgani va ushbu qo'llanmada biz asoslanadigan kompilyator **GNU GCC** (MinGW) **kompilyatori**dir.

Codeblocks IDE va MinGW kompilyatorini kompyuterga yuklash va o'rnatish

Bepul Codeblocks IDE va MinGW kompilyatorini kompyuterga yuklash uchun Code::Blocks rasmiy saytining quyidagi sahifasini ochamiz:

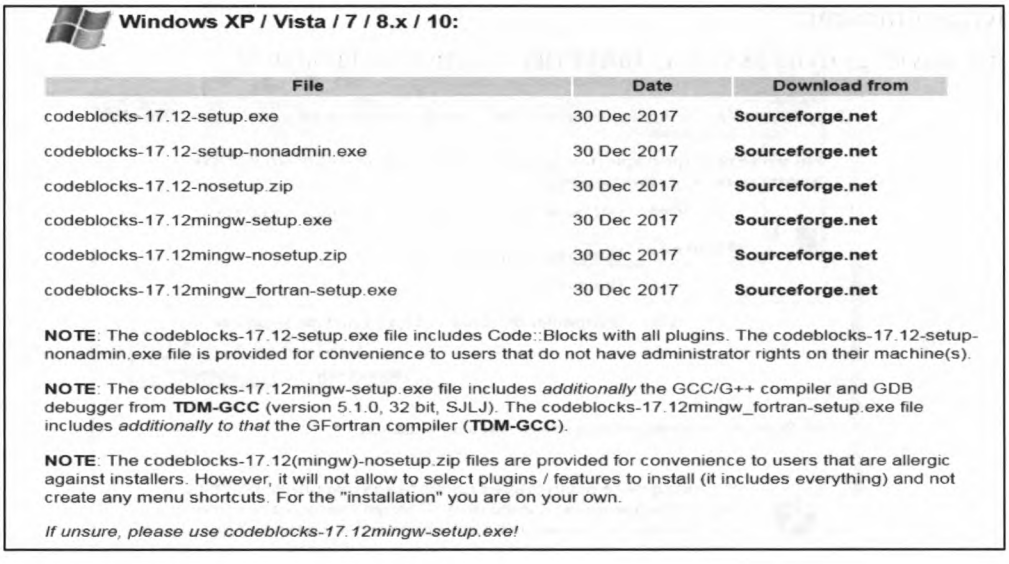
http:/ /www.codeblocks.org/ downloads

Natijada quyidagi ko'rinishdagi sahifa ochiladi:

1) **Download the binary release** (yuqoridagi rasmda shtrixli strelka bilan ko‘rsatilgan) havolani tanlab quyidagi ko‘rinishdagi sahifaga o‘tamiz:

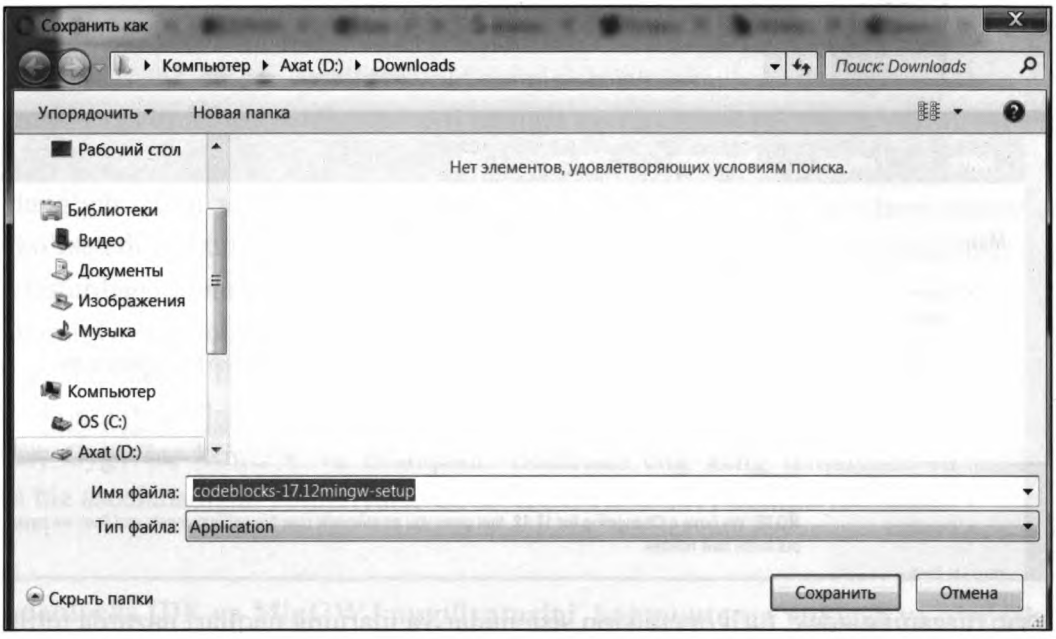
Ko‘rib turganingizdek, turli operatsion sistemalar va ularning naqllari nazarda tutilgan.

2) **Windows XP / Vista / 7 / 8.x / 10** havolani tanlagach, biz uchun quyidagi imkoniyatlar tavsiya qilinadi:



Bu sahifadagi ma'lumotlar yillarga mos yangilab boriladi, shuning uchun ba'zi fayllar yili bilan farqlanishi mumkin. Taklif etilganlar ichidan **codeblocks-17.12mingw-setup.exe** ga mos Sourceforge.net yuklash havolasini tanlaymiz.

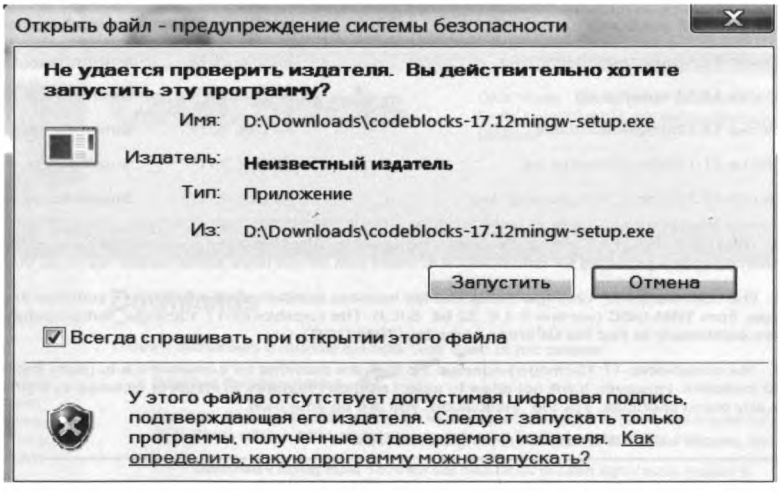
3) Endi quyidagi muloqot oynasi orqali fayllarni qaysi papkaga saqlashni tanlashimiz mumkin:



Yuklanayotgan fayllar hajmi katta emas, masalan, fayllar hajmi 86,2 Mbayt.

4) Yuklab olingan **codeblocks-17.12mingw-setup.exe** faylni ishga tushirib kompyuterga o'rnatamiz.

a) agar quyidagi oyna aks etsa, **Запустить** tugmasini tanlaymiz:



b) keyingi tanlash oynasidan **Next** (ma'nosi: Keyingi) tugmasini tanlaymiz;

d) keyingi tanlash oynasidan **I Agree** (ma'nosi: Men Roziman) tugmasini tanlaymiz;

e) keyingi tanlash oynasidan **Next** tugmasini tanlaymiz;

f) keyingi tanlash oynasidan **Install** (ma'nosi: O'rnatish) tugmasini tanlaymiz;

g) keyingi tanlash oynasidan **Нет** (ma'nosi: Йўқ) tugmasini tanlaymiz;

h) va nihoyat tanlash oynasida aks etgan **Finish** (ma'nosi: Tamom) tugmasini tanlaymiz.

Shu bilan kerakli **CodeBlocks** dasturi to'liq o'rnatildi.

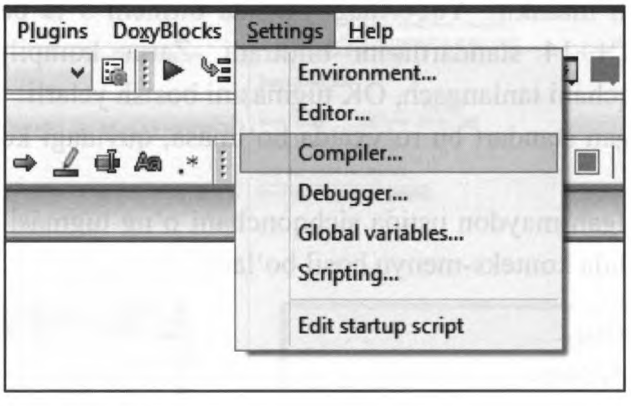
Dasturni ishga tushirish uchun **Ish stoli**dagi yorliqdan yoki Bosh menyuning Dasturlar bo'limi **CodeBlocks** papkasidagi 

tugmasi tanlanadi.

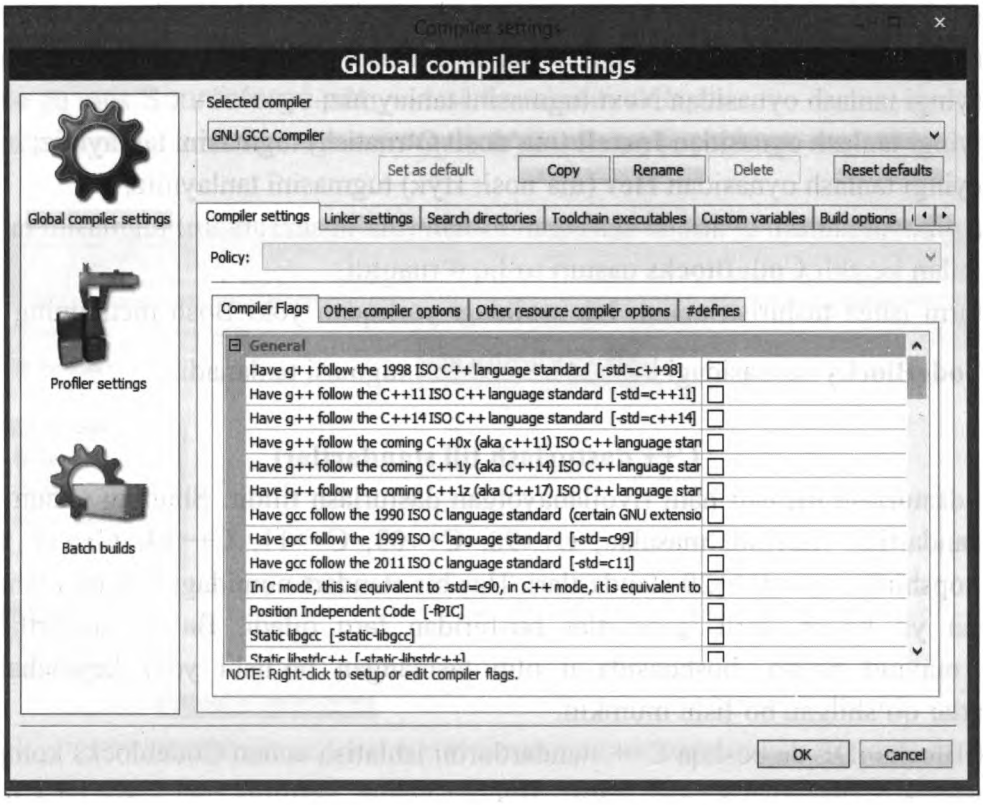
C++ dasturlash tili standartlari

C++ dasturlash tili hali ham rivojlanayotgan dasturlash tilidir. Shuning uchun uning bir nechta standartlari mavjud, masalan, C++98, C++03, C++11, C++14, C++17 va foydalanishga topshirilmagan C++20 standartlari. Har bir standart nomidagi son bu o'sha standart chiqarilgan yil bo'lib, turli standartlar bir-biridan farq qiladi. Ba'zi standartlarda biror funksiya mavjud bo'lsa, boshqasida u olib tashlangan bo'lishi yoki keyinchalik yangi imkoniyatlar qo'shilgan bo'lishi mumkin.

CodeBlocks IDE da boshqa C++ standartlarini ishlatish uchun Codeblocks kompilyatsiya parametrlariga o'zgartirish kiritish zarur. Buning uchun **Settings** menyusining **Compiler...** bo'limi tanlanadi:



Bunda quyidagi oyna aks etadi:



Yuqoridagi oynada ko‘rinib turibdiki, “Compiler Flags” bo‘limidan kompilyator bayroqchalarini tanlash mumkin. Yuqoridagi rasmda birinchi 3 ta bayroqcha mos ravishda C++98, C++11 va C++14 standartlarini bildiradi. Zarur kompilyatorni tanlash uchun ro‘yxatdan mos bayroqchani tanlangach, OK tugmasini bosish yetarli.

Agar kerakli bo‘lgan standart bu ro‘yxatda bo‘lmasa, quyidagi ko‘rsatmalarni ketma-ket bajarish talab etiladi:

1) Bayroqlar yozilgan maydon ustida sichqonchani o‘ng tugmasi bosiladi, natijada chap rasmdagi kabi ko‘rinishda konteks-menyu hosil bo‘ladi:

2) Konteks-menyudan “New flag…” bo‘limi tanlanadi va natijada yuqoridagi o‘ng rasmdagi kabi ko‘rinishda oyna hosil bo‘ladi;

3) Oynaning “Name” bo‘limiga qo‘shish zarur bo‘lgan bayroqqa berilayotgan nom yoziladi. Bu nom o‘rnida istalgan ibora bo‘lishi mumkin. “Compiler flags” bo‘limiga esa qo‘shish zarur bo‘lgan bayroq yoziladi. Masalan, C++14 standartini qo‘shmoqchi bo‘lganda - std=c++14 kabi, C++17 standartini qo‘shmoqchi bo‘lganda - std=c++17 kabi yoziladi.

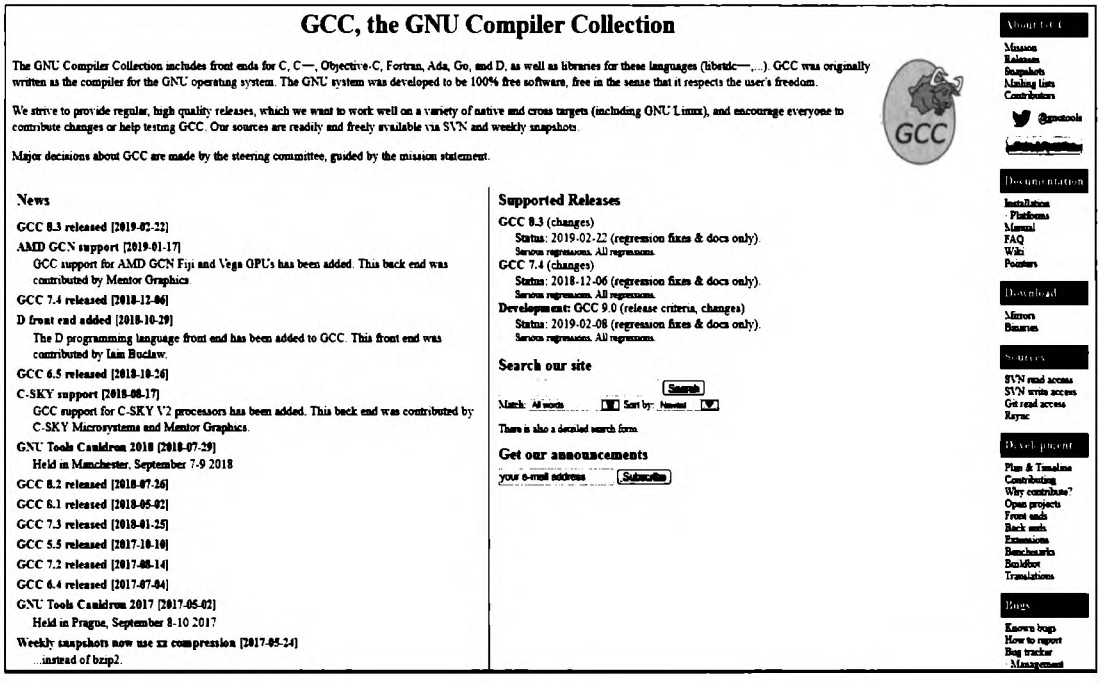
4) OK tugmasi bosiladi.

5) Asosiy oynada hosil bo‘lgan yangi bayroqni tanlang va OK tugmasini bosing.

Shuni ta’kidlab o‘tish joizki, agar yangi qo‘shilgan bayroq yoki qo‘shilayotgan C++ standarti o‘rnatilgan C++ kompilyatori tarkibida bo‘lmasa, qo‘shilgan bayroq hech narsani o‘zgartirmaydi.

C++ kompilyatorlari qanday standartlarni qo‘llashi haqida http://gcc.gnu.org/ saytidan ma’lumotga ega bo‘lish mumkin.

Quyida http://gcc.gnu.org/ saytining bosh sahifasi aks ettirilgan.



1-BOB. C++ TILINING BOSHLANG'ICH ELEMENTLARI

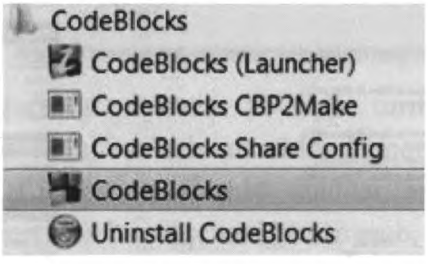
C++ dasturlash tili alifbosi, tarkibi, funksiyalari, operatorlari va boshqa elementlarini Codeblocks IDE orqali izohlaymiz. Codeblocks IDE tuzilgan har bir dasturni foydalanuvchi tanlagan nom bilan saqlash uchun (loyiha sifatida hisoblab) shu nomdagi papka hosil qiladi. Bu papkada bir nechta papka va turli kengaytmali fayllar hosil bo'ladi.

C++ tilida yozilgan dasturlarni alohida saqlash uchun biror, masalan, "C++ dasturlari" kabi, papka hosil qilib olishni tavsiya etamiz.

1-§. CONSOLE APPLICATION ILOVASINI ISHGA TUSHIRISH

C++ muhitida Console Application ilovasini ishga tushirish uchun quyidagi ketma-ketlikda tanlashlar bajariladi:

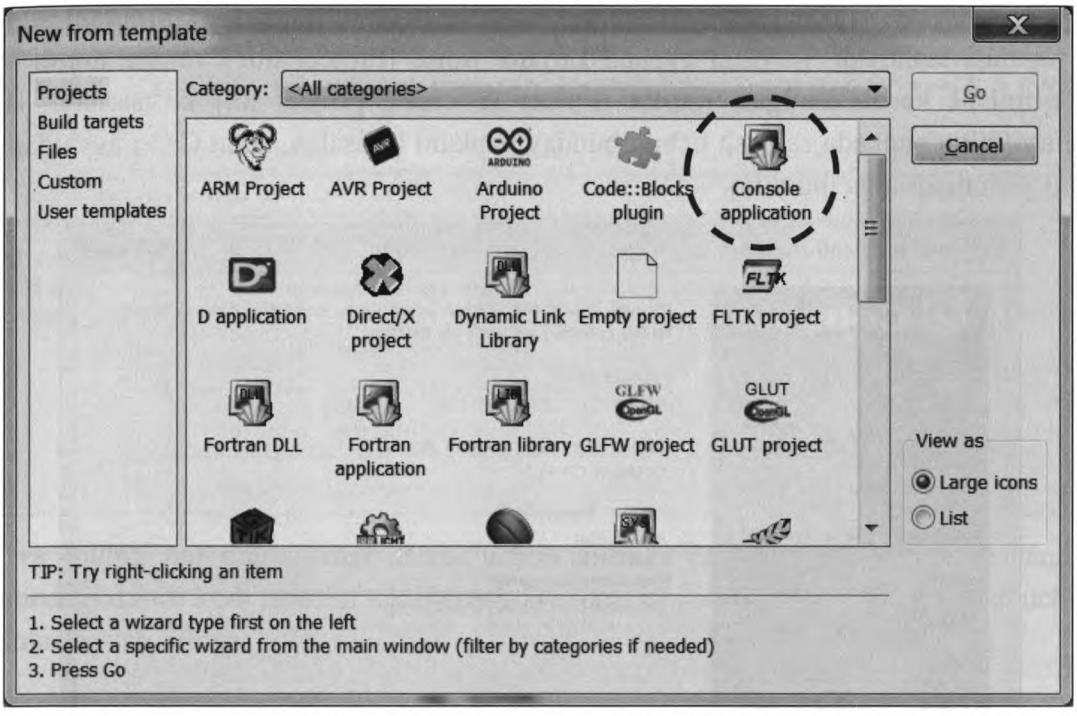
1) **Пуск → Программы →** CodeBlocks va shu bo'limdan:



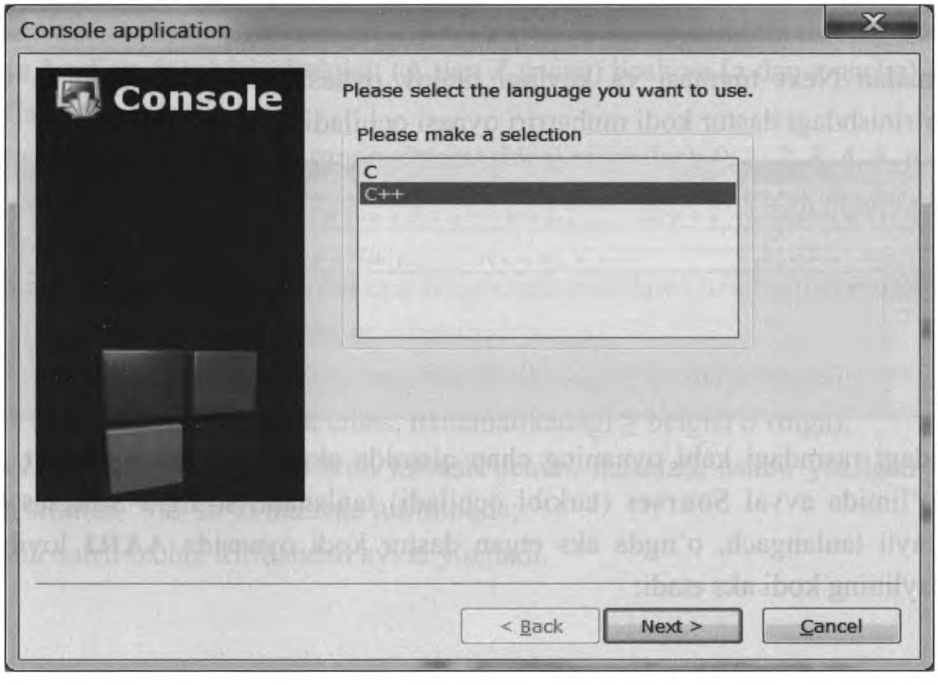
2) Ochilgan **Start here** oynasi yordamida **yangi loyiha hosil qilish** (**Create a new project**) yoki **mavjud loyihani ochish** (**Open an existing project**) mumkin.

Izoh 1: Avval hosil qilingan loyihani ochish uchun loyiha papkasidagi **.cbp** kengaytmali fayl tanlanadi.

Yangi loyiha hosil qilish tanlangach, quyidagi ko'rinishdagi tanlov oynasi ochiladi:

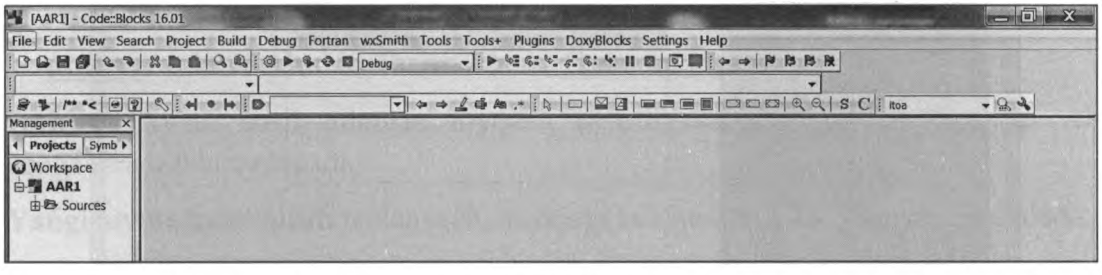


Bu oynada (rasmda shtrixli aylana orqali chegaralab ko‘rsatilgan) **Console Application** ilovasini tezkor 2 marta tanlangach (yoki Console Application ilovasi belgilanganidan so‘ng Go tugmasi tanlangach) dasturlash tilini tanlash oynasi ochiladi:



Dasturlash tilini tanlash oynasida muhit tavsiya etgan C++ dasturlash tilini tanlash uchun **Next** tugmasi tanlanadi. Keyingi oynada **Loyiha nomi** (Project title) va shu nomli **loyiha hosil qilinishi kerak bo'lgan papka** (Folder to create project in:) ko'rsatiladi. Barcha loyihalarni bitta papkada saqlash uchun bunday papkani (masalan, Axat C++) avvaldan hosil qilib qo'yish maqsadga muvofiq.

Bu oynadan **Next** tugmasi va keyingi tanlov oynasidan **Finish** tugmasi tanlangach, quyidagi ko'rinishdagi dastur kodi muharriri oynasi ochiladi:



Yuqoridagi rasmdagi kabi oynaning chap qismida aks etgan **Management** oynasining **Projects** bo'limida avval **Sourses** (tarkibi ochiladi) tanlanadi, so'ngra **Sourses** tarkibidagi **main.cpp** fayli tanlangach, o'ngda aks etgan dastur kodi oynasida **AAR1** loyihasiga mos **main.cpp** faylining kodi aks etadi:

C++ muhiti har doim yangi dastur uchun **andaza** (shablon) sifatida yuqoridagi kabi mazmundagi dastur kodi matnini taklif etadi. Dasturni ishga tushirish va natija olish uchun **F9** klavishini bosish kifoya.

2-§. C++ DASTURLASH TILI ALIFBOSI VA DASTUR TARKIBI

C++ tili alifbosi quyidagi belgilarni o‘z ichiga olgan:

- **52 ta harf va tagchiziq belgisi**: (A dan Z gacha) bosh va (a dan z gacha) kichik lotin harflari va _ belgisi;
- **10 ta arab raqami** (10 lik sanoq sistemasidagi raqamlar): 0, 1, 2, 3, 4, 5, 6, 7, 8, 9;
- **maxsus belgilar**: + - * / . , : ; = > < ' " ( ) [ ] { } # ! & ? | % $ ^ va orachiq (ing. space, rus. пробел).

Ba’zi maxsus belgilar birlashmasi orqali qo‘shma belgilarni hosil qilish mumkin, masalan:

- != – teng emas (matematikadagi ≠ belgisi o‘rniga);
- <= – kichik yoki teng (katta emas, matematikadagi ≤ belgisi o‘rniga);
- >= – katta yoki teng (kichik emas, matematikadagi ≥ belgisi o‘rniga);
- /* va */ – bir necha satrli izoh kiritish uchun, masalan, ushbu yozilgan ko‘rinishda izoh sifatida “va” so‘zi nazarda tutilmoqda;
- // – bir satrli izohni kiritishdan avval yoziladi.

IDENTIFIKATOR

Identifikator deganda lotin harflari yoki tagchiziq belgisidan boshlangan faqat lotin harflari, tagchiziq belgisi va raqamlar ishtirok etgan belgilar ketma-ketligi tushuniladi. Ya'ni **3iden** yoki **#iden** identifikator bo'la olmaydi. Identifikatordagi harflarni quyi yoki yuqori registrda yozilishining farqi bor (ing. case-sensitive). Masalan, **NOM, nom** va **Nom** turli identifikatorlarni aniqlaydi.

Identifikatorlar funksiyalar, parametrlar, o'zgaruvchilar, konstantalar, sinflar, turlar va nishonlar nomi bo'ladi. Identifikatordagi belgilar uzunligiga chegara qo'yilmagan, lekin birinchi 31 ta belgi farqlanturuvchi bo'ladi. Identifikatorlar **zaxira, standart** va **foydalanuvchi** turlarga bo'linadi.

C++ tilida **reserved,** ya'ni **zaxira** identifikatorlar tilning tarkibiy qismi hisoblanib, aniq ko'rinishga va mazmunga ega. Bu identifikatorlar mazmunini **foydalanuvchi** (ya'ni dasturchi) o'zgartira olmaydi. Zaxira identifikatorlarga quyidagilarni misol sifatida keltirish mumkin.

alignas (since C++11)	double	reflexpr (reflection TS)
alignof (since C++11)	dynamic_cast	reinterpret_cast
and	else	requires (since C++20)
and_eq	enum	return
asm	explicit	short
atomic_cancel (TM TS)	export(1)	signed
atomic_commit (TM TS)	extern(1)	sizeof(1)
atomic_noexcept (TM TS)	false	static
auto(1)	float	static_assert (since C++11)
bitand	for	static_cast
bitor	friend	struct(1)
bool	goto	switch
break	if	synchronized (TM TS)
case	import (modules TS)	template
catch	inline(1)	this
char	int	thread_local (since C++11)
char16_t (since C++11)	long	throw
char32_t (since C++11)	module (modules TS)	true
class(1)	mutable(1)	try
compl	namespace	typedef
concept (since C++20)	new	typeid
const	noexcept (since C++11)	typename
constexpr (since C++11)	not	union
const_cast	not_eq	unsigned
continue	nullptr (since C++11)	using(1)

co_await (coroutines TS)	operator	virtual
co_return (coroutines TS)	or	void
co_yield (coroutines TS)	or_eq	volatile
decltype (since C++11)	private	wchar_t
default(1)	protected	while
delete(1)	public	xor
do	register(2)	xor_eq

Izoh 2: https://en.cppreference.com/w/cpp/keyword saytidan olindi.

Izoh 3: (1) yozuvi mazmuni o'zgarganini yoki C++11 da yangi mazmun qo'shilganini bildiradi; (2) yozuvi C++17 da mazmuni o'zgarganini bildiradi. Qavs ichidagi (since C++11) yoki (since C++20) yozuvlari esa C++11 yoki C++20 standartlaridan so'ng ishlatila boshlandi degan ma'noni anglatadi.

Standart identifikatorlar dasturlash tili tuzilmalarini belgilash uchun ishlab chiqaruvchilar tomonidan avvaldan aniqlangan bo'ladi, masalan, quyidagilar uchun:

ma'lumotlar turi	**konstantalar**	**funksiyalar**

Standart identifikatorlarni dasturchi tomonidan qo'shimcha nomlashning imkoni bor, lekin bu nomlash ko'pincha xatoliklarga olib kelishi mumkin. Standart identifikatorlarga misol sifatida quyidagilarni keltirish mumkin: **sin, max**.

Foydalanuvchi identifikatorlari dasturchi tomonidan aniqlanadi, lekin u **identifikator ta'rifiga mos bo'lishi shart**.

DASTUR TARKIBI

C++ tilida dastur asosan quyidagi tarkibda tashkil etiladi:

- qo'llanadigan kutubxona fayllarini **qo'shish**;
- (zarur bo'lsa) global o'zgaruvchilar, konstantalar va funksiyalarni **tavsiflash**;
- funksiyalar matni (agar dasturga funksiyalar qo'shilsa);
- **main ()** nomli bosh funksiya va uning tarkibida:
  - lokal o'zgaruvchilar va konstantalar tavsifi;
  - dastur operatorlari.

C++ tilida vazifalari turlicha bo'lgan funksiyalar juda ko'p bo'lib, ular vazifasi hamda boshqa xususiyatlariga asosan turli nomdagi kutubxonalarga birlashtirilgan. Biror kutubxonaga tegishli operator yoki funksiyani qo'llashdan avval shu kutubxona fayli dasturga **qo'shilishi** (ingl. **include**) shart. Kutubxona faylini dasturga qo'shish uchun dastur boshida alohida satrda

#include <fayl nomi.kengaytma>

direktivasi (ko'rsatmasi) #include xizmatchi so'zi yordamida yozilishi shart.

Dastur tarkibini C++ tilida yozilgan quyidagi sodda dastur orqali izohlashni davom ettiramiz:

```
#include <iostream>

using namespace std;

int main(){
    cout << "Salom O'zbekiston!";
    return 0;
}
```

Bu dastur ishlashi natijasida ekranga faqatgina **Salom O'zbekiston!** matni (qo'sh-tirnoqsiz) chiqariladi. Dastur matni quyidagilardan iborat:

- **`#include <iostream>`** – dasturga kiritish-chiqarish oqimi (i – ingl. input – kirish, o – ingl. output – chiqish, ingl. **stream** – **oqim**) kutubxonasini qo'shadi. Bu kutubxona dasturda **`cout`** (o'qilishi: "siout", ya'ni "c da chiqarish" ma'nosida) operatori ishlatilgani uchun kerakdir. Aytib o'tilganidek, C++ tilida standart kutubxonalar ko'p bo'lib, turli xil kutubxonalar turli foydali funksiyalar va imkoniyatlardan foydalanish uchun xizmat qiladi. Masalan, **cmath** kutubxonasi matematik funksiyalarni o'z ichiga olgan.
- **`using namespace std;`** – dasturda **standart nomlar fazosi** ham foydalanilayotgani uchun yozilgan, chunki maxsus **cout** nomi shu fazoga tegishli. Ta'kidlash joizki, C++ tilida ishlatiladigan asosiy nomlar (masalan, konstantalar, funksiya nomlari) global nomlar fazosida ko'rsatilgan. Global nomlar fazosini **std** nomlar fazosidan farqli o'laroq, hech qanday qo'shimcha ko'rsatmalarsiz ishlatish mumkin.
- **`int main()`** – bosh funksiya. C++ tilida dastur **main** funksiyasini ishlatish bilan o'z ishini boshlaydi. C++ tilidagi har bir dasturda bosh funksiyasi bo'lishi shart. **main** funksiyasi qaytaradigan qiymat butun son (ammo ba'zi kompilyatorlarda hech qanday qiymat qaytarmasligi ham mumkin). C++ tilida funksiya tanasi (funksiya bajaradigan ko'rsatmalar ketma-ketligi) { } orasida yoziladi.
- **`cout`** – ma'lumotlarni ekranga chiqarish operatori. Bu operator va uning ishlatilishi haqida keyingi mavzularda alohida to'xtalamiz.
- **`return 0`** – main funksiyasining qiymati sifatida 0 qiymatni qaytaradi. C++ tilida xatosiz tugagan dastur 0 qiymatini qaytarishi kerak. 0 dan boshqa qiymat dastur ishlashida qandaydir xatolik bo'lganidan darak beradi.

C++ tilida har qanday ko‘rsatma oxirida ; (nuqtali vergul) yoziladi. Bu ushbu ko‘rsatmaning ta’siri tugaganini bildiradi va dastur keyingi ko‘rsatmani bajarishga o‘tadi. Shu sababli C++ tilida ko‘rsatmalar ketma-ketligining bir satrda yozilishi yoki har bir ko‘rsatma yangi satrda yozilishi dastur natijasini o‘zgartirmaydi. Yuqoridagi dasturni bir qatorda ham quyidagicha yozishimiz mumkin:

```
#include <iostream>
using namespace std;int main(){cout <<"Salom O'zbekiston!";return 0;}
```

Bu va yuqoridagi dasturni taqqoslashdan ko‘rish mumkinki, ikkala dastur matnining mazmuni bir xil. E’tibor bering, **#include** direktivalari dasturning qolgan qismidan **ajralib** turishi **shart**.

PREPROTSESSOR, KOMPILYATOR, YIG‘UVCHI

C++ tilida yozilgan dastur bajariluvchi mashina kodiga – **.exe** kengaytmali faylga o‘tkazilishi uchun 3 ta jarayondan o‘tishi zarur: preprotsessorlash, kompilyatsiya, yuklash.

1. Preprotsessorning vazifasiga, zaruratga qarab, ushbu dasturga dasturdagi **#include** direktivalarida ko‘rsatilgan tashqi fayllarni bog‘lash kiradi. Preprotsessor bu direktivalar o‘rniga fayllar matnini joylashtiradi.
2. Kompilyator bir necha qadamda preprotsessor hosil qilgan sintaksis va semantik xatolardan xoli bo‘lgan obyekt faylini **.o** kengaytmali optimallashtirilgan mashina kodiga tarjima (translyatsiya) qiladi. Bunda agar dasturda sintaksis va semantik xatolar uchrasa, u holda dasturchiga bu xatolar ro‘yxati chiqariladi.
3. Yig‘uvchi (ingl. linker, rus. компановщик) kompilyator hosil qilgan obyekt faylini kutubxonalardagi boshqa dasturlar yoki, zarur bo‘lsa, boshqa fayllar bilan bog‘laydi. Natijada esa .exe kengaytmali fayl hosil bo‘ladi.

3-§. C++ TILIDA MA’LUMOTLAR VA ULARNING TURLARI

C++ tilidagi, umuman, har qanday dasturlash tilidagi, dastur hayotimizdagi (hosil bo‘lgan yoki berilgan) ma’lumotlar ustida qandaydir amallar bajarib ma’lum bir ma’lumot hosil qiladi, ya’ni qayta ishlaydi. Inson hayotidagi ma’lumotlar esa asosan sonlar va matnlar orqali ifodalanadi. O‘z navbatida, sonlar raqamlar va maxsus (ishora va kasr qism) belgilar yordamida, matnlar esa harflar va maxsus (xizmatchi) belgilar yordamida hosil qilinadi. Raqam va harflar insoniyat tomonidan aniq va qat’iy kelishuv asosida qabul qilingan, shu sababli ularning mazmuni barcha uchun aniq va yagonadir. Masalan, 21 yoki -23, ‘A’ harfi yoki

"ABC" matni aniq va yagona mazmunga ega. Ma'lumki, raqamlar sonlar to'plamiga tegishlidir. Shu sababli, raqam va sonlar emas, umumiy holda sonlar haqida so'z yuritish mumkin.

Dasturchilar qabul qilgan ba'zi so'zlar borki, ularning mazmuni tushunarli bo'lib, alohida ahamiyat kasb etadi. Masalan, mantiqiy qiymatlar: rost (ing. true) va yolg'on (ing. false).

LITERALLAR

Literallar – bu dastur matniga kiritilgan son, belgi va matnlardir. Bu tushunchani quyidagi dastur orqali izohlaymiz:

```
#include <iostream>
using namespace std;
int main(){
   const int x = 2*5; const char c='!';
   int y = x; bool z = true;
   string s = " marta Assalom";
   cout << x << y << z << s <<c;
   return 0;
}
```

Dastur A

Bu yerda **2** va **5** sonli literallar, **!** (' ' orasida yozilgan) belgili literal, **marta Assalom** (" " orasida yozilgan, marta so'zi oldida orachiq bor) esa satrli (matnli deb ham atashadi) literal. Dasturdagi **true** ham literal bo'lib, u **mantiqiy literal** deyiladi. C++ da, oddiy holatda, ekranga **true** so'zi o'rniga **1** raqami chiqarilgani uchun, yuqoridagi dastur ishlashi natijasida ekranda **10101 marta Assalom!** aks etadi.

O'ZLASHTIRISH OPERATORI

C++ tilidagi eng sodda operator, ya'ni o'zlashtirish operatorini izohlab o'tamiz. **O'zlashtirish operatori** berilgan yoki hosil qilingan qiymatni o'zlashtirilishini anglatadi. Qiymatni o'zlashtirish = (matematikadagi tenglik belgisi) orqali bajariladi. Yuqoridagi **Dastur A** da = (o'zlashtirish) operatori yordamida **x** ga sonli literallar ko'paytmasi natijasi bo'lgan **10** qiymat o'zlashtirildi, **c** ga belgili literal **!** qiymati, **y** ga **x** ning qiymati, **z** ga mantiqiy literal **true** qiymati, **s** ga matnli literal **marta Assalom** qiymati o'zlashtirildi.

O'zlashtirish operatori sodda bo'lsa-da, har qanday dasturlash tilida muhim ahamiyatga ega va keng qo'llaniladi. C++ da quyidagicha o'zlashtirish imkoniyatlari bor:

```
#include <iostream>
using namespace std;
int main(){
    int a, b, c, d, e=0, f=1, g=2, h=22;
    char h1, h2;
    a = b = c = d = 1+2*5;
    h1 = h2 = '!';
    e+=a, f-=2, g*=5; h/=c;
    cout << h1 << a << b << c << d << e << f << g << h << h2;
    return 0;
}
```

Bu dasturda bitta son (1+2*5=11) qiymatni bir necha, ya'ni a, b, c va d larga, belgili literal ! qiymati h1 va h2 ga o'zlashtirilmoqda. Dasturda yana bir necha operator (ya'ni `e+=a, f-=2, g*=5;`) bitta blok sifatida , belgisi yordamida birlashtirilgan. Bundan tashqari amallarni yozishda quyidagicha ekvivalent imkoniyatlar qo'llangan:

`e+=a;` yozuvi `e=e+a;` ga ekvivalent, qiymati e=0+11= 11;

`f-=2;` yozuvi `f=f-2;` ga ekvivalent, qiymati f=1–2= –1;

`g*=5;` yozuvi `g=g*5;` ga ekvivalent, qiymati g=2*5= 10;

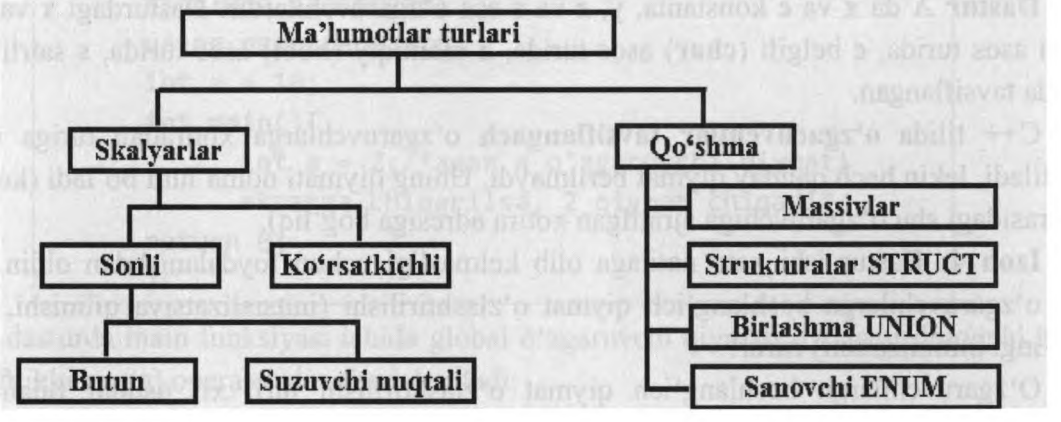
`h/=c;` yozuvi `h=h/c;` ga ekvivalent, qiymati h=22/11= 2;

Bu dastur ishlashi natijasida ekranda **!1111111111-1102!** aks etadi.

MA'LUMOTLARNING TURLARI

C++ tilida ma'lumotlar turkumiga quyidagilarni kiritish mumkin: **konstanta, o'zgaruvchilar, strukturalar (butun va haqiqiy sonlarni o'z ichiga olgan), matn (belgi va satrlar), adreslar (o'zgaruvchilar va strukturalar).**

Ularni quyidagi sxema orqali tasvirlash mumkin:



KONSTANTALAR VA O‘ZGARUVCHILAR

Dasturlash tilida xotirada saqlanishi kerak bo‘lgan ma’lumotlarni **miqdorlar** deb ataymiz. Xotirada saqlanishi kerak bo‘lgan **miqdorlar** xotirada egallaydigan joyini aniq hisoblash mumkin bo‘lishi uchun dasturda avval **tavsiflanishi**, ya’ni turi ko‘rsatilishi **shart**.

Dasturlash tillarida **tur tushunchasi** mazkur dasturlash tilida har bir **miqdor turi** uchun xotiradan ajratilishi kerak bo‘lgan **joy hajmi** va shu tur ustida bajarilishi mumkin bo‘lgan **amal**larni aniqlaydi.

Miqdorlar turini tavsiflash uchun maxsus xizmatchi so‘zlardan foydalaniladi. C++ dasturlash tilida miqdorlarning konstanta va o‘zgaruvchi turlari mavjud.

Konstantalar – dastur ishlashi davomida qiymati o‘zgarmaydigan miqdorlar (o‘zgarmaslar).

Konstantaning qiymatini dasturda o‘zgartirib bo‘lmaydi!

O‘zgaruvchilar – dastur ishlashi davomida qiymati o‘zgarishi mumkin bo‘lgan miqdorlar.

O‘zgaruvchi qiymati dastur ishlashi davomida o‘zgarsa ham, uning qiymatlari doimo mos turda bo‘lishi shart.

O‘z navbatida, konstanta va o‘zgaruvchilar ham bir necha turlarga ajraladi. Biz bu bo‘limda faqat **asos turlar** haqida so‘z yuritamiz. **Asos turlarga butun va haqiqiy sonli, belgili, mantiqiy turlar kiradi**. Dasturdagi konstantalar va o‘zgaruvchilarni farqlash juda oson, chunki konstantalar tavsifi albatta **const** xizmatchi so‘zidan boshlanadi. Demak, dasturda tavsiflanayotgan o‘zgaruvchini konstanta turiga o‘tkazish uchun o‘zgaruvchi tavsifi oldiga **const** xizmatchi so‘zni yozish kifoya ekan. Umumiy holda tavsifni quyidagicha yozish mumkin:

```
asos_tur identifikator; // bu tavsif o'zgaruvchi uchun
   const asos_tur identifikator = ifoda; // bu tavsif konstanta uchun
```

Dastur A da **x** va **c** konstanta, **y**, **z** va **s** esa o‘zgaruvchilardir. Dasturdagi **x** va **y** butun (**int**) asos turida, **c** belgili (**char**) asos turida, **z** mantiqiy (**bool**) asos turida, **s** satrli (**string**) turida tavsiflangan.

C++ tilida **o‘zgaruvchilar tavsiflangach** o‘zgaruvchlarga xotiradan turiga mos joy ajratiladi, lekin hech qanday qiymat berilmaydi. Uning qiymati noma’lum bo‘ladi (kompyuter xotirasidagi shu o‘zgaruvchiga ajratilgan xotira adresiga bog‘liq).

Izoh 4: Dastur ishi xato natijaga olib kelmasligi uchun foydalanishdan oldin lokal o‘zgaruvchilarga boshlang‘ich qiymat o‘zlashtirilishi (initsializatsiya qilinishi, ya’ni ing. initialization) zarur!

O‘zgaruvchilarga boshlang‘ich qiymat o‘zlashtirilishi turli xil usullar bilan amalga oshirilishi mumkin. Masalan:

```
int y = 10; int y = 2 * 5;
int y {10};//C++11 standartidan boshlab
```

LOKAL VA GLOBAL O‘ZGARUVCHILAR VA KONSTANTALAR

Dastur tarkibi haqida so‘z yuritilganda global va lokal o‘zgaruvchilar tushunchalari ishlatilgan edi. Bu tushunchalar o‘zgaruvchilarni dasturning qayerida tavsiflanishiga bog‘liqdir.

Lokal o‘zgaruvchi – biror-bir funksiya yoki {} bloki orasida tavsiflangan o‘zgaruvchidir.

Izoh 5: **Lokal o‘zgaruvchilar { } blokidan tashqarida hech qanday ma’noga ega bo‘lmaydi, ya’ni tavsiflanmagan hisoblanadi.**

Global o‘zgaruvchilar – dasturning barcha qismlarida ma’noga ega bo‘lgan o‘zgaruvchilar. Global o‘zgaruvchilarga dasturdagi har qanday blok yoki funksiyadan murojaat qilish va uni qiymatini o‘zgartirish mumkin. Global o‘zgaruvchilar asosan **main** funksiyasidan avval tavsiflanadi. Masalan:

```
#include <iostream>
using namespace std;
int a = 10; //global o'zgaruvchi
int main(){
      int b = 23; //lokal o'zgaruvchi
return 0;
}
```

Yuqoridagi fikrlar konstantalarga ham taalluqlidir.

Agar **a** global o‘zgaruvchi tavsiflangan bo‘lsa va biror-bir funksiya tanasida yana **a** o‘zgaruvchi tavsiflansa, dastur ishlashi jarayonida shu funksiya ichida **a** o‘zgaruvchiga murojaat etish natijasida funksiya ichida tavsiflangan o‘zgaruvchi ishlaydi, ya’ni:

```
#include <iostream>
using namespace std;
int a = 10;
int main(){
      int a = 2;/*agar a o'zgaruvchi qiymati
      ekranga chiqarilsa, 2 qiymat chiqadi*/
return 0;
}
```

Shu dasturda main funksiyasi ichida global o‘zgaruvchi qiymati o‘zlashtirilmoqchi bo‘lsa, **::** (juft ikki nuqta) operatoridan foydalaniladi:

```
#include <iostream>
using namespace std;
int a = 10;
int main(){
      int a = 2;
      /* agar a o'zgaruvchi qiymati
      ekranga chiqarilsa, 2 qiymat chiqadi*/
      int b = a;//b o'zgaruvchi qiymati 2 ga teng bo'ladi
      int d = ::a;//d o'zgaruvchi qiymati 10 ga teng bo'ladi
      return 0;
}
```

4-§. C++ TILIDA MIQDORLARNING MANTIQIY VA BUTUN TURI

MIQDORLARNING MANTIQIY TURI

Mantiqiy turdagi miqdorlar tavsifi **bool** xizmatchi so‘zi orqali amalga oshirilib, mantiqiy miqdorlar faqat 2 ta qiymatni qabul qilishi mumkin: **true (1)** yoki **false (0)**. Masalan:

```
bool a = false; // qiymati false
bool b {true}; // qiymati true
bool d {-123}; // qiymati true
bool u {0}; //qiymati false
```

Mantiqiy turdagi o‘zgaruvchiga 0 dan farqli qiymat o‘zlashtirilganda **true** qiymatni qabul qiladi, faqat 0 o‘zlashtirilsagina **false** qiymatni qabul qiladi.

Informatikadan ma’lumki, asosiy 3 ta mantiqiy amal, ya’ni **mantiqiy ko‘paytirish** (o‘zb. VA, ingl. AND), **mantiqiy qo‘shish** (o‘zb. YOKI, ingl. OR) va **mantiqiy inkor** (o‘zb. EMAS, ingl. NOT), quyidagi rostlik jadvallari bilan aniqlanadi (rost=1 va yolg‘on=0 belgilashga ko‘ra):

A	B	A AND B
1	1	1
1	0	0
0	1	0
0	0	0

A	B	A OR B
1	1	1
1	0	1
0	1	1
0	0	0

A	NOT A
1	0
0	1

Bu amallarning asosiy xossalari quyidagicha:

A AND A =A	A AND B = B AND A
A AND TRUE=A	A OR B = B OR A
A AND FALSE=FALSE	A AND (B AND C)=(A AND B) AND C
A AND NOT A=FALSE	A OR (B OR C)=(A OR B) OR C
A OR A =A	A OR (B AND C)=(A OR B) AND (A OR C)
A OR TRUE=TRUE	A AND (B OR C)=(A AND B) OR (A AND C)
A OR FALSE=A	A AND (A OR B)=A
A OR NOT A=TRUE	A OR (A AND B)=A
NOT NOT A=A	(A OR B) AND (NOT A OR B)=B
NOT TRUE=FALSE	(A AND B) OR (NOT A AND B)=B

Mantiqiy ko'paytirish va mantiqiy qo'shish amallarini boshqa ikki mantiqiy amal bilan quyidagicha bog'lash mumkin (de Morgan formulalari):

A AND B=NOT(NOT A OR NOT B)

A OR B=NOT(NOT A AND NOT B)

Ba'zi dasturlash tillariga XOR mantiqiy amali kiritilgan bo'lib, bu amalni **o'xshashlikni inkor etish** yoki **2 modul bo'yicha qo'shish** deb ham atashadi. Bu amal C++ tilida A XOR B yoki A ^ B kabi yozilishi mumkin. Uning rostlik jadvali va asosiy xossalari quyidagicha:

A	B	A XOR B
1	1	0
1	0	1
0	1	1
0	0	0

1) A XOR FALSE = A
2) A XOR TRUE = NOT A
3) A XOR A = FALSE
4) A XOR B = B XOR A
5) (A XOR B) XOR B = A

O'xshashlikni inkor etish amalini boshqa mantiqiy amallar orqali quyidagicha ifodalash mumkin:

A XOR B = NOT A AND B OR A AND NOT B

C++ tilida mantiqiy literallar, konstantalar va o'zgaruvchilar ustida yuqoridagi mantiqiy amallarga qo'shimcha ravishda taqqoslashga oid quyidagi amallarni ham bajarish mumkin:

Amal	Mazmuni	Vazifasi
AND yoki &&	Mantiqiy ko'paytirish	AND kabi
OR yoki ||	Mantiqiy qo'shish	OR kabi
NOT yoki !	Mantiqiy inkor	NOT kabi
XOR yoki ^	O'xshashlikning inkori	XOR kabi
= =	Teng (ekvivalent)	Tenglik o'rinli bo'lsa **1**, aks holda **0** qaytaradi
!=	Teng (ekvivalent) emas	Tengsizlik o'rinli bo'lsa **1**, aks holda **0** qaytaradi

Masalan:

```
bool a = true, b = false;
bool d = (a == b) ^ a; // d = (1 == 0) ^ 1 = 0 ^ 1 = 1
bool u = a && (b or true); // u = 1 && (0 or 1) = 1 && 1 = 1
bool r = a || (u != b); //r = 1 || (1 != 0) = 1 || 1 = 1
```

MIQDORLARNING BUTUN TURI

C++ tilida butun turlar qabul qiladigan qiymatlarining chegarasiga (diapazoniga) va ko'rinishiga (ishorali, ishorasiz) qarab ajratiladi. Ular quyidagilardan iborat:

Tur nomi	Xotirada egallagan hajmi (baytda)	Qabul qiladigan qiymat diapazoni
signed char	1	-128..+127
short short int signed short signed short int	2	-32768..32767
int signed signed int long long int signed long signed long int	4	-2 147 483 648..+2147483647
long long long long int signed long long signed long long int	8	-9223372036854775808.. +9223372036854775807

Izoh 6: Diapazon deganda (.. nuqta orqali ajratilgan) chap chegaradan o'ng chegaragacha bo'lgan barcha qiymatlarni qabul qilinishi tushuniladi.

Yuqoridagi jadvalda **signed** xizmatchi so'zi turning qiymati manfiy ham, musbat ham bo'lishi mumkinligini bildiradi. Agar tur manfiy ham, musbat ham qiymat qabul qilishi nazarda tutilsa, bu xizmatchi so'zni yozish shart emas. Lekin, agar tur uchun faqat musbat sonlar qabul qilishi yetarli bo'lsa, u holda **unsigned** xizmatchi so'zini ishlatish foydali:

Tur nomi	Xotirada egallagan hajmi (baytda)	Qabul qiladigan qiymat diapazoni
unsigned short unsigned short int	2	0..65535
unsigned int unsigned long unsigned long int	4	0..4294967295
unsigned long long unsigned long long int	8	0..18446744073709551615

Bu holda tur qabul qilishi mumkin bo'lgan yuqori chegara qiymati deyarli 2 marta kattalashdi. Buning sababini quyidagi misol orqali izohlaymiz.

Xotiradan 2 bayt=16 bit joy egallaydigan **signed short** turida **yuqori bit** (eng chapdagi bit) ishorani belgilash uchun, qolganlari sonning qiymati uchun xizmat qiladi. Agar yuqori bit 1 bo'lsa, u holda son manfiy, yuqori bit 0 bo'lsa, u holda son musbat. Son manfiy bo'lganda qiymat uchun ajratilgan 15 bit orqali -32768..-1 (jami 2^{15}=32768 ta) diapazondagi sonlarni, son manfiy bo'lmaganda qiymat uchun ajratilgan 15 bit orqali 0..32767 (jami 2^{15}=32768 ta) diapazondagi sonlarni ifodalash mumkin.

Xotiradan 2 bayt=16 bit joy egallaydigan **unsigned short** turida barcha bit sonning qiymati uchun xizmat qiladi. Demak, 0..65535 (jami 2^{16}=65536 ta) diapazondagi sonlarni ifodalash mumkin.

BUTUN O'ZGARUVCHILAR VA KONSTANTALAR TAVSIFI

Butun turdagi o'zgaruvchilar quyidagicha tavsiflanadi:

```
int butun_son;
long long katta_butun_son;
short int kichik_butun_son;
```

Bir vaqtning o'zida bir nechta o'zgaruvchini tavsiflash ham mumkin:

```
int butun_son1, butun_son2;
long long katta_butun_son1, katta_butun_son2;
```

O'zgaruvchilarga boshlang'ich qiymatni o'zlashtirish turli xil usullar bilan amalga oshirilishi mumkin. Masalan:

```
int butun_son = 10;
short int kichik_butun_son = 2 * 5;
long long katta_butun_son {10};//C++11 standartidan boshlab
```

Albatta, o'zgaruvchiga qiymatni dasturda ishlatilishidan avval ham berish mumkin:

```
int butun_son;
```

```
butun_son = 10;
```

Butun turdagi konstantalar quyidagicha tavsiflanadi:

```
const int konstanta_butun_son = 10;
const long long konstanta_katta_butun_son = 10000000000;
const short int konstanta_kichik_butun_son = 128;
```

Dastur A da **x** butun **int** turidagi konstanta, **y** esa butun **int** turidagi o'zgaruvchi. Quyidagi dastur orqali turni to'g'ri tanlashning ahamiyati ko'rsatilgan:

```
#include <iostream>
using namespace std;
int main(){
    signed char a=10, b=30;
    signed char c; c = a*b; //c=300 bo'lishi kerak
    short d, e; d= a*b; //d=300 bo'lishi kerak
    e= d*d;//e=90000 bo'lishi kerak
    int f; f= d*d; //f=90000 bo'lishi kerak
    cout << c << " " << d << " " << e << " " << f;
    return 0;
}
```

Dastur natijasi izohlardagi qiymatlar emas, quyidagicha bo'ladi: `, 300 24464 90000`
Demak, **c** va **e** uchun tur xato tanlangan.

BUTUN QIYMATLI LITERALLAR

10 lik literallar

O'nlik literallarga misol tariqasida quyidagilarni keltirish mumkin:

```
102   123U  -49L  -10ULL      721ul       217LU
```

Son oxirida (ya'ni son **suffiksi** sifatidagi) **u** yoki **U** harfining yozilishi sonning **unsigned** turida ekanligini, **l** yoki **L** – **long** turida, **ll** yoki **LL** – **long long** turida ekanligini bildiradi. **U, L, LL** harflarining qanday ketma-ketlikda yoki qaysi registrda yozilishining ahamiyati yo'q.

Masalan, agar **long long** turidagi o'nlik literalni yozish kerak bo'lsa `102LL`, agar literal **unsigned** bo'lsa (manfiy bo'lmasa): `102ULL` `102LLU`

C++14 standartidan boshlab butun literallar razryadlari (xonalari) qulay ko'rinishi uchun qog'ozda yozilgani kabi, ajratuvchi belgilarni qo'yishimiz ham mumkin, ya'ni masalan, `15000000` sonini quyidagicha yozish mumkin:

```
int son = 15'000'000;
```

Izoh 7: Literallar aniq (son) qiymatga ega bo'lsa, u holda nima uchun literal oxirida qandaydir tur ko'rsatilishi zarur degan savol yuzaga keladi. Bu literallar ustida amallar bajarish bilan bog'liqdir.

16 lik literallar

O‘n oltilik literallar o‘n oltilik son oldiga **0x** (nol bilan x) qo‘shish bilan hosil qilinadi:

16 lik ko‘rinishi	0xA	0x123U	0xAB9L
10 lik ko‘rinishi	10	291U	171L

8 lik literallar

Sakkizlik literallar sakkizlik son oldiga **0** (nol) qo‘shish bilan hosil qilinadi.

8 lik ko‘rinishi	012	0123U	04321LL
10 lik ko‘rinishi	10	83U	2257LL

Bundan ko‘rinib turibdiki, o‘nlik sonning oldida 0 raqamini yozish mumkin emas, aks holda kompilyator uni sakkizlik soni deb hisoblaydi.

2 lik literallar

Ikkilik literallar faqat C++14 standartidan boshlab qo‘shiladi. Ikkilik literallar ikkilik son oldiga 0b yoki 0B ni qo‘shib yozish orqali hosil qilinadi.

2 lik ko‘rinishi	0b010	0B11100U	0b10011LL
10 lik ko‘rinishi	2	28U	19LL

BUTUN SONLAR USTIDA BAJARILADIGAN SODDA AMALLAR

C++ tilida butun literallar, konstantalar va o‘zgaruvchilar ustida quyidagi arifmetik amallarni bajarish mumkin:

Amal	Ifodada bajarilish navbati	Vazifasi
+	2	Qo‘shish
-	2	Ayirish
*	1	Ko‘paytirish
/	1	Butun bo‘lish (ya’ni 7 / 3 = 2)
%	1	Qoldiq hisoblash (ya’ni 7 % 3 = 1)

C++ tilidagi sodda amallar yordamida quyidagi kabi turli ifodalarni yozish mumkin:

```
a = b + c + 7;          c = a * b + 21;
a = a / b - 23;         c = a % b - 1;
```

Matematika fanidan ma’lumki, qavslar yordamida amallarning bajarilish tartibini o‘zgartirish mumkin. C++ tilida qavslar shu kabi vazifani ham bajaradi:

```
a = a * (b + c);
c = ((a % 2) + (b % 2)) % 2;
b = ((a * 10) + 20 / (10 + d)) * (1 + t);
```

BUTUN SONLAR USTIDA BAJARILADIGAN BITLI AMALLAR

C++ tilida butun literallar, konstantalar va o‘zgaruvchilar ustida quyidagi (ikkilik ko‘rinishidagi ifodasida) bitli amallarni bajarish mumkin:

Amal	Vazifasi
&	AND mantiqiy amali kabi (bitga mos mantiqiy ko‘paytirish)
|	OR mantiqiy amali kabi (bitga mos mantiqiy qo‘shish)
^	XOR mantiqiy amali kabi (bitga mos o‘xshashlikni inkor etish)
~	NOT mantiqiy amali kabi (bitga mos inkor)
<<	Razryadli chapga surish
>>	Razryadli o‘ngga surish

Masalan, $2_{10}=10_2$, $4_{10}=100_2$, $5_{10}=101_2$, $8_{10}=1000_2$, $10_{10}=1010_2$, $12_{10}=1100_2$ ekanligini e’tiborga olsak:

4 & 5 = 0…100 & 0…101 = 0…100 = 4

4 | 5 = 0…100 | 0…101 = 0…101 = 5

2 ^ 4 = 0…010 ^ 0…100 = 0…110 = 6

~1 = ~0000…00001 = 1111…11110 = -2 (signed int turida)

10 & 12 = 0…1010 & 0…1100 = 0…1000 = 8

~1u = ~0000…00001u = 1111…11110u =4294967294u (unsigned int turida)

2 << 2 = 10 << 2 = 100 << 1 = 1000 = 8 yoki $2 << 2 = 2 * 2^2 = 2 * 4 = 8$

8 >> 2 = 1000 >> 2 = 100 >> 1 = 10 = 2 yoki $8 >> 2 = 8 / 2^2 = 8 / 4 = 2$

Bitli XOR amalining arifmetik mazmuni ham bor bo‘lib, u sonlari ikkilikdagi ifodasida har bir bit bo‘yicha qo‘shib 2 ga bo‘lgandagi qoldig‘i olinadigan natija kabi ham tushuniladi.

Quyida butun literallar bilan bog‘liq bo‘lgan yuqoridagi **Izoh 7** uchun misol keltiramiz:

```
#include <iostream>
using namespace std;
int main(){
      long long a, b, c, d;
      a = 1<<31; //a ning qiymati -2147483648 ga teng bo'ladi
      b = 1LL<<31; // b ning qiymati 2147483648 ga teng bo'ladi
      c = 1<<32; //c ning qiymati 0 ga teng bo'ladi
      d = 1LL<<32; // d ning qiymati 4294967296 ga teng bo'ladi
          return 0;
    }
```

5-§. HAQIQIY VA BELGILI TURLAR

HAQIQIY TURLAR

Dasturlashda haqiqiy sonlar o'nli kasr ko'rinishida tasvirlanadi. O'nli kasrning butun va kasr qismini ajratuvchi belgi sifatida vergul emas, nuqta qo'llanadi. Ixtiyoriy haqiqiy sonni kompyuter xotirasida to'liq shaklda tasvirlab bo'lmaganligi uchun **qo'zg'aluvchan nuqtali sonlar** kabi tasvirlash usuli kiritilgan. Qo'zg'aluvchan nuqta tushunchasi o'nli kasrning quyidagi ko'rinishda tasvirlanishiga mos kiritilgan: $1,5=0,15\cdot10^{1}=15\cdot10^{-1}=0,0015\cdot10^{3}=150\cdot10^{-2}$.

C++ tilida haqiqiy sonlarni aniqlik nuqtayi nazaridan tasvirlashning 3 xil turi mavjud:

Tur nomi	Xotirada egallagan hajmi (baytda)	Qabul qiladigan qiymat diapazoni
float	4	±1.17549e-038.. ±3.40282e+038
double	8	±2.22507e-308.. ±1.79769e+308
long double	10-12 (kompilyatorga bog'liq)	±3.3621e-4932.. ±1.18973e+4932 (kompilyatorga bog'liq)

Masalan:

```
float a{10.23};   const double b = 10.1222;
long double c = 1312312.122L;
```

Qo'zg'aluvchan nuqtali sonlar literallari uchun L yoki l (**long double** uchun) va f yoki F (**float** uchun) suffikslarini ishlatish mumkin. Agar hech qanday suffiks bo'lmasa, demak, u **double** turida bo'ladi. Zaruratga qarab qo'zg'aluvchan nuqtali sonlarni eksponensial ko'rinishida yozish ham mumkin (e yoki E):

```
1e+5  123E-13L    12e-1f
```

Qo'zg'aluvchan nuqtali sonlarga butun turlarga qo'llash mumkin bo'lgan arifmetik amallardan % (qoldiq) dan boshqa barchasini qo'llash mumkin, mantiqiy amallarni qo'llab bo'lmaydi.

TURNI O'ZGARTIRISH

C++ tilida butun va haqiqiy qiymatdagi o'zgaruvchilar qiymatlarini boshqa turdagi o'zgaruvchiga o'zlashtirish mumkin. Bunda o'zlashtirayotgan o'zgaruvchi turining chegarasi o'zlashtirilayotgan qiymat turgan o'zgaruvchi chegarasidan katta bo'lsa, hech qanday muammo bo'lmaydi. Turlar diapazoniga mos C++ tilida quyidagicha saralangan:

1. **long double**	2. **double**	3. **float**
4. **unsigned long long**	5. **long long**	6. **unsigned long**
7. **long**	8. **unsigned int**	9. **int**

Masalan, quyidagicha yozish mumkin:

```
int y = 10; long long b = y;
```

Turni boshqa turga **static_cast** yordamida ham o'zgartirish mumkin:

```
float b = 12.12;
int a = static_cast<int>(b);
```

Bunda o'tilayotgan tur < > belgilari orasida ko'rsatiladi. Bunday usul ancha samarali hisoblanadi. Chunki oddiy usulda o'tilayotgan turni kompilyator aniq farqlay olmasligi va buning natijasida xatoliklar kelib chiqishi mumkin.

MIQDORLARNING BELGILI TURI

C++ tilida belgili miqdorlar **char** turida aniqlangan. Belgilar asosan ASCII jadvalidagi belgilardir (kompilyatorga ham bog'liq). Ularni tavsiflash va qiymat o'zlashtirish quyidagicha:

```
char a;
char b{'s'};
const char c = 'c';
```

Turi **char** bo'lgan o'zgaruvchiga butun turdagi qiymatni o'zlashtirish ham mumkin. Bundan tashqari, butun turdagi o'zgaruvchilar ustida bajariladigan amallarni ham qo'llash mumkin. Ammo natija baribir belgi bo'ladi. Chunki bu holda char turidagi o'zgaruvchida belgining ASCII kodi saqlanadi. Masalan:

```
char a{65}; //a o'zgaruvchi qiymati 'A' belgisi
a = a + 1; //a o'zgaruvchi qiymati 'B' belgisi
```

KO'RSATKICHLAR

Aytib o'tilganidek, dasturda tavsiflangan har bir konstanta yoki o'zgaruvchi o'z **nomi**ga va kompyuterning tezkor xotirasi – RAM da ajratilgan **joy**ga ega, har bir xotiradan ajratilgan joyning esa o'z adresi (manzili) bo'ladi. Tezkor xotirani ketma-ket joylashgan xotira uyachalari ko'rinishida tushunish mumkin, ketma-ket kelgan uyachalar manzili bittaga farq qiladi. Har bir xotira uyachasi o'zida 1 bayt axborotni saqlay oladi. Xotira adreslari, odatda, 16 lik sanoq sistemasida ifodalanadi:

0x00	0x00000001	0x00000002	0x00000003	...

C++ dagi **bool**, **char** kabi turdagi o'zgaruvchilar shu xotira adreslarining bittasiga joylashadi, **signed int**, **unsigned int**, **float** kabi (4-baytli) turlar esa xotiraning 4 ta ketma-ket joylashgan uyachasini, **double**, **long long** lar esa 8 ta uyachani egallaydi.

C++ ning boshqa yuqori darajali dasturlash tillaridan eng katta farqlaridan biri bu uning **xotiraga to'g'ridan to'g'ri murojaat qila olish** xususiyatidir. Ya'ni biror-bir xotira adre-

sidagi qiymatni olish, uni o'zgartirish kabi amallarni bajarish mumkin. Bu amallar esa **ko'rsatkich** deb ataluvchi C++ dasturlash tili elementi orqali amalga oshiriladi.

Ko'rsatkich – biror xotira adresini o'zida saqlaydigan o'zgaruvchi yoki konstantadir.

Har bir ko'rsatkich ma'lum bir turdagi o'zgaruvchi yoki konstanta adresini o'zida saqlashi mumkin. Bu tur ko'rsatkich tavsifida ko'rsatiladi. Ko'rsatkichlar tavsifiga quyidagilarni misol qilish mumkin:

```
int *a; // int turidagi o'zgaruvchi adresini saqlovchi ko'rsatkich
const long* b; // const long(long turidagi konstanta) adresini saqlovchi ko'rsatkich
unsigned int *c; //unsigned int turidagi o'zgaruvchi adresini saqlovchi ko'rsatkich.
```

Yuqoridagi tavsiflardan ko'rinib turibdiki, ko'rsatkichlar tavsifi o'zgaruvchilar tavsifiga o'xshash va faqat identifikator oldidan * belgisi qo'yilishi bilan farqlanadi.

```
int *a,d;
```

bu yerda a – int turidagi o'zgaruvchiga ko'rsatkich, d – int turidagi o'zgaruvchi.

Bu holda * belgisi a identifikatorga tegishli. Ikkala o'zgaruvchini ko'rsatkich kabi tavsiflash uchun quyidagicha yozish kerak bo'ladi:

```
int *a, *d;
```

Istalgan o'zgaruvchi yoki konstanta adresini olish uchun & operatoridan foydalaniladi. Masalan, uning yordamida ko'rsatkichlarga boshlang'ich qiymat berishimiz ham mumkin:

```
int a=10;
int *b{&a};
int *c = &a;
```

bu holda **b** va **c** ko'rsatkichlar o'zida **a** o'zgaruvchi adresini saqlaydi.

Ko'rsatkichlarga ham o'zgaruvchilar kabi har doim boshlang'ich qiymat berish yuzaga kelishi mumkin bo'lgan ba'zi xatolarning oldini olishga xizmat qiladi. Agar ko'rsatkich tavsiflanayotganda qanday boshlang'ich qiymat berilishi aniq bo'lmasa, u holda uni **NULL** qiymati bilan instalyatsiya qilish maqsadga muvofiq. **NULL** qiymati – ko'rsatkich hech qanday o'zgaruvchi yoki konstanta adresiga ega emasligini bildiradi. C++ da buni **NULL** yoki **nullptr** (**nullptr** – C++11 standartida qo'shilgan va dasturda shundan foydalanish maqsadga muvofiq) orqali ifodalash mumkin.

```
int *a{NULL}; //maslahat berilmaydi
int *b{nullptr}; //maslahat beriladi
```

Ko'rsatkichda saqlanayotgan adresdagi o'zgaruvchi yoki konstanta qiymatini olish yoki o'zgaruvchi qiymatini o'zgartirish * operatori yordamida amalga oshiriladi.

```
int *b{nullptr}; // nullptr yordamida b ko'rsatkichga boshlang'ich qiymat berish
int a{10};
```

```
b = &a; // b ko‘rsatkichga a o‘zgaruvchisi adresini o‘zlashtiradi
int c = *b;
*b = 12;
```

Oxirgi ifoda b ko‘rsatkich ko‘rsatgan o‘zgaruvchi (ya’ni a o‘zgaruvchi) qiymatini 12 ga o‘zgartiradi. Bunda b o‘zgaruvchisining qiymati (a o‘zgaruvchi adresi) o‘zgarmaydi. Yuqoridagi ko‘rsatmalardan so‘ng b o‘zgaruvchi qiymati a o‘zgaruvchining adresi, c o‘zgaruvchi qiymati 10 (a o‘zgaruvchining qiymati nusxasi), a o‘zgaruvchining qiymati 12 bo‘ladi.

Ko‘rsatkichlar ham, yuqorida aytib o‘tilganidek, o‘zgaruvchi yoki konstanta bo‘ladi. Yuqoridagilarning barchasi o‘zgaruvchan ko‘rsatkichlarga misol bo‘ladi. Konstanta ko‘rsatkichlar oddiy konstantalar kabi **const** xizmatchi so‘zini qo‘shish yordamida tavsiflanadi:

```
int c{11};
int* const a{&c};
```

Konstanta ko‘rsatkichlarda ham oddiy konstantalardagi kabi instalyatsiya qilish shart. Keyinchalik bu ko‘rsatkichga boshqa o‘zgaruvchi adresini o‘zlashtirish mumkin emas, ya’ni masalan:

```
int b{11}, c{21};
int* const a{&b};
a = &c; // Mumkin emas!
```

MIQDORLARNING SATRLI TURI

Satrli konstantalar qiymati belgilardan tashkil topgan ketma-ketlik bo‘lib, ikki tomondan qo‘shtirnoq bilan chegaralangan. C++ tilida satrli konstantalar **const char*** xizmatchi so‘zlar yordamida tavsiflanadi. Masalan:

```
const char* a = "Salom";
const char* b {"Toshkent"};
```

C++ kompilyatori satrli konstantalarning oxiriga avtomatik ravishda '\0' belgisini qo‘shadi. Demak, bu belgi satr tugaganligini bildiradi.

Satrli o‘zgaruvchilar haqida keyingi mavzularda kengroq so‘z yuritiladi.

MIQDORLARNING VOID TURI VA VOID TURIDAGI KO‘RSATKICHLAR

C++ da miqdorlarning **void** turi ham bo‘lib, **void** turidagi o‘zgaruvchilar hech qanday qiymat qabul qilmaydi. Ular haqida keyingi mavzularda kengroq so‘z yuritiladi.

Void turidagi ko‘rsatkichlar boshqa barcha turdagi ko‘rsatkichlar qiymatlarini qabul qila oladi. Ko‘rsatkichlar turini boshqa turga o‘tkazish uchun quyidagicha yo‘l tutiladi:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  int b = 10;`
`  int *a = &b;`
`  float *c = (float*)a;`
`  void *d = a;`
`  int *q = (int*)d;`
`  cout << *q << endl;`
`  return 0;`
`}`
	`10`

6-§. MA'LUMOTLARNI KLAVIATURADAN KIRITISH VA EKRANGA CHIQARISH OPERATORLARI

C++ dasturlash tilida ma'lumotlarni klaviaturadan kiritish va ma'lumotlarni ekranga chiqarish uchun turli xil operatorlar mavjud. Ularga misol tariqasida **cin** va **cout** operatorlarini keltirish mumkin. Shuni ta'kidlab o'tish joizki, **cin** va **cout** operatorlari tuzilishi va ishlash prinsiplari, ularning negizida yotgan tushunchalar ancha murakkab. Umuman olganda bu operatorlarning imkoniyatlari juda keng. Qo'llanma C++ dasturlash tilining boshlang'ich qisminigina qamrab olishi ko'zda tutilgan bo'lib, mazkur operatorlarning ko'pgina imkoniyatlari qo'llanma doirasiga kirmaydigan mavzularga bog'liq. Shu sababli qo'llanmada bu operatorlarning boshlang'ich, ya'ni asosiy imkoniyatlari haqida so'z yuritamiz, qo'shimcha imkoniyatlarni esa kitobxonning o'zi mustaqil o'rganishini tavsiya etamiz.

Avval aytib o'tilganidek, **cin** va **cout** operatorlaridan foydalanish uchun dastur boshida (hech bo'lmasa) quyidagi direktiva yozilishi shart: `#include <iostream>`

ASOSIY IMKONIYAT: COUT OPERATORI

Ma'lumotlarni ekranga chiqarish uchun **cout** operatoridan foydalanish mumkin. Bu operatorning umumiy ko'rinishi quyidagicha:

```
cout << ifoda_1 << ifoda_2 << … << ifoda_m;
```

Bu yerda ifoda_1, ifoda_2, …, ifoda_m larning qiymati butun (int, long long, ….), o'nli kasrli (float, double, long double), satrli konstanta (masalan, "satr"), belgili ('a', 'b', ….) yoki mantiqiy (bool) turida bo'lishi kerak. Quyidagi dastur **cout** operatori bilan bog'liq ba'zi imkoniyatlarni ochib beradi:

Dastur

```
#include <iostream>
using namespace std;
int main(){
 cout << "Assalom O'zbekiston!";
 cout << 2 + 4 - 1 << ' '; cout << 1.2 + 3.2 << " ";
 cout <<'a'<<'b'<<'c'<<' '; cout <<true<<" "<<false<<' ';
 int a = 10; float b = 2.4;
 char d = 'a'; bool t = true;
 cout << a << b << d << t;
 return 0;
}
```

Natijasi

```
Assalom O'zbekiston!5 4.4 abc 1 0 102.4a1
```

Dastur natijasida ekranga quyidagi ma'lumotlar ketma-ketligi chiqarilishi kerak edi:

```
Assalom O'zbekiston! 5 4.4 a b c 1 0 10 2.4  a 1
```

Lekin natijadan ko'rinib turibdiki, ba'zi ma'lumotlar boshqa ma'lumotlarga ulanib ekranga chiqarilgan. Chiqarish operatori tarkibida ' ' yoki " " yozib ekranga chiqarilgan orachiq belgilari esa ma'lumotlarni biridan ikkinchisini ajratib turibdi. Demak, agar chiqarilayotgan ma'lumotlarni ajralgan holda chiqarmoqchi bo'lsak, u holda orachiq belgisini qo'shishimiz kerak ekan.

Dastur natijasini chiqarishda juda ko'p kerak bo'ladigan amal, ya'ni yurgichni yangi satrga o'tkazish amali uchun C++ tili andozasida taklif etilgan **endl** funksiyasidan foydalanish mumkin:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  cout << "Assalom O'zbekiston!"<<endl;`
`  cout << "5*5=" <<5*5<<endl<<"9+1="<<9+1;`
`  return 0;`
`}`
	`Assalom O'zbekiston!`
	`5*5=25`
	`9+1=10`

QO'SHIMCHA IMKONIYAT: ESCAPE BELGILARI

Ma'lumotlarni boshqa usullar yordamida ham ekranda ajralgan holda chiqarish mumkin. Buning uchun asosan **Escape belgilari** deb ataluvchi (\ belgidan boshlanadigan) belgilar ketma-ketligidan foydalaniladi. Escape belgilari, odatda, apostrof yoki qo'shtirnoq belgilari orasida joylashtirilgan bo'ladi. Escape belgilariga mos jadval quyidagicha:

Escape belgilari	Bajaradigan vazifasi
\\	Teskari og‘ma (slesh) chiziq chiqarish
\'	Apostrof chiqarish
\"	Qo‘shtirnoq chiqarish
\?	So‘roq belgisini chiqarish
\a	Tovush signali chiqaradi
\b	Yurgichni bir belgi orqaga suradi
\n	Keyingi satrga o‘tish
\r	Yurgichni satrning boshiga qaytarish
\t	Tabulyatsiya berish
\xxx	x lar o‘rniga 8 lik raqamlar yozib ASCII jadvalidagi shu o‘rinda turgan belgini chiqarish
\xNN	N lar o‘rniga 16 lik raqamlar yozib, ASCII jadvalidagi shu o‘rinda turgan belgini chiqarish

Demak, Escape belgilaridan ‘\n’ belgisi yordamida ham yurgichni yangi satrga o‘tkazish mumkin ekan:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  cout << "Assalom O'zbekiston!\n";`
`  cout << "5*5=" << 5*5 <<"\n9+1="<<9+1;`
`  return 0;`
`}`
	`Assalom O'zbekiston!`
	`5*5=25`
	`9+1=10`

Quyida Escape belgilarining ba’zilarini izohlovchi dastur keltirilgan:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  cout<<"\"Assalom"<<"\t"<<"O'zbekiston\"" <<'\n';`
`  cout << '\101' << '\n';`
`  cout << '\x41' << '\n';`
`  cout << "abc" << '\b' << "abc";`
`  return 0;`
`}`
	`"Assalom`
	`O'zbekiston"`
	`A`
	`A`
	`ababc`

Dastur ishini tahlil qilamiz:

a) birinchi chiqarish operatorining boshlanishida qo‘shtirnoqdan keyingi \" belgisi qo‘shtirnoq chiqarishga, `\t` belgisi `Assalom` va `O'zbekiston` so‘zlari orasida tabulyatsiya joy-

lashtirishga, \" belgisi qo‘shtirnoq chiqarishga va \n belgisi yurgichni yangi satrga o‘tkazishga xizmat qiladi;

b) ikkinchi chiqarish operatorida (3 xonali bo‘lgani uchun) 8 lik raqamlari orqali yozilgan \101 belgisi (8 lik sanoq sistemasidagi 101 soni 10 lik sanoq sistemasidagi 65 ga teng bo‘lib, u lotin A harfining ASCII kodi) A harfini ekranga chiqarishga, \n belgisi yurgichni yangi satrga o‘tkazishga xizmat qiladi;

c) uchinchi chiqarish operatorida (x dan boshlangani uchun) 16 lik raqamlari orqali yozilgan \41 belgisi (16 lik sanoq sistemasidagi 41 soni 10 lik sanoq sistemasidagi 65 ga teng) A harfini ekranga chiqarishga, \n belgisi yurgichni yangi satrga o‘tkazishga xizmat qiladi;

d) to‘rtinchi chiqarish operatori abc satrni ekranga chiqargach, \b Escape belgisi yurgichni bitta belgi o‘rniga mos orqaga qaytaradi, ya’ni yurgich c belgisi turgan o‘ringa o‘tadi va keyingi abc satr shu joydan boshlab chiqarilgani uchun avvalgi abc satrning c belgisi o‘chib ketib, faqat ab qismigina qoladi.

QO‘SHIMCHA IMKONIYAT: COUT OPERATORIDA FORMATLI CHIQARISH

Ma’lumotlarni ekranga chiqarish operatori **cout** yordamida turli xil formatdagi natijalarni chiqarish mumkin. Masalan, haqiqiy sonlar qanday aniqlikda chiqarilayotganini bilish va aniqlik darajasini o‘zgartirish mumkin. Buning uchun **precision** funksiyasidan foydalaniladi:

`int p = cout.precision();` – funksiya shu vaqtda foydalanilayotgan aniqlik darajasini qaytaradi;

`cout.precision(n);` – aniqlik darajasini n ga o‘zgartiradi.

Quyidagi misol ushbu funksiyalar ishini izohlaydi:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  double a=20,b=30,c; int p;`
`  p=cout.precision();`
`  cout<<"aniqlik "<<p<<" xona:"<<endl;`
`  c=a/b;`
`  cout<<"c= "<< c <<endl;`
`  cout.precision(9);`
`  p=cout.precision();`
`  cout<<"aniqlik "<<p<<" xona:"<<endl;`
`  cout<<"c= "<< c;`
`  return 0;`
`}`
	`aniqlik 6 xona:`
	`c= 0.666667`
	`aniqlik 9 xona:`
	`c= 0.666666667`

Quyidagi format o‘zgartiruvchi manipulyatorlar asosni o‘zgartirib chiqarish uchun xizmat qiladi:

Manipulyator	Bajaradigan vazifasi
`dec`	Keyingi barcha butun qiymatlarni 10 lik sanoq sistemasida chiqaradi
`hex`	Keyingi barcha butun qiymatlarni 16 lik sanoq sistemasida chiqaradi
`oct`	Keyingi barcha butun qiymatlarni 8 lik sanoq sistemasida chiqaradi

Masalan, 10 lik sanoq sistemasidagi 27 va 63 sonlari 16 lik sanoq sistemasida 1B va 3F ga, 8 lik sanoq sistemasida 33 va 77 ga teng ekanligini bilgan holda quyidagi dastur natijasini tushunib olish qiyin emas:

Dastur	**Natijasi**

`#include <iostream>`
`using namespace std;`
`int main(){`
`  int a=27,b=63;`
`  cout<<hex;`
`  cout<<"a= "<<a<<endl;`
`  cout<<"b= "<<b<<endl;`
`  cout<<oct<<"a= "<<a<<endl;`
`  cout<<"b= "<<b<<endl;`
`  cout<<dec<<"a= "<<a<<endl;`
`  cout<<"b= "<<b<<endl;`
`  return 0;`
`}`
	`a= 1b`
	`b= 3f`
	`a= 33`
	`b= 77`
	`a= 37`
	`b= 63`

Quyidagi manipulyatorlar ham qiziqarli vazifalarni bajaradi:

Manipulyator	Bajaradigan vazifasi
`boolalpha`	**bool** turidagi o‘zgaruvchi qiymatlarini ekranga **true** yoki **false** ko‘rinishida chiqarish
`showbase`	Bundan keyin chiqarilayotgan 16 lik va 8 lik butun sonlarning asosi ko‘rsatib chiqariladi
`showpoint`	Haqiqiy sonlarni ‘.’ bilan chiqaradi, sonning kasr qismi 0 bo‘lsa ham ‘.’ chiqariladi
`showpos`	Manfiymas sonlar oldiga “+” belgisi qo‘yib chiqariladi, 0 soni ham +0 kabi chiqariladi
`uppercase`	Butun son 16 lik ko‘rinishda yoki haqiqiy sonlar eksponensial ko‘rinishda chiqarilayotganda belgilar yuqori registrda chiqariladi

Yuqoridagi formatli chiqarishni “bekor qilish” uchun mos ravishda **noboolalpha, noshowbase, noshowpoint, noshowpos, nouppercase** manipulyatorlaridan foydalaniladi.

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  int a=10; bool t=true; double x=100;`
`  cout<<showbase<<hex<<a<<endl;`
`  cout<<uppercase<<a<<endl;`
`  cout<<boolalpha<<t<<endl;`
`  cout<<showpoint<<x<<endl;`
`  cout<<showpos<<a<<" "<<t<<" "<<x<<endl;`
`  cout<<nouppercase<<noboolalpha;`
`  cout<<noshowpoint<<noshowbase;`
`  cout<<dec<<a<<" "<<t<<" "<<x<<endl;`
`  return 0;`
`}`
	`0xa`
	`0XA`
	`true`
	`100.000`
	`0XA true +100.000`
	`+10 +1 +100`

Haqiqiy sonlarni turli ko‘rinishlarda chiqarish manipulyatorlari:

Manipulyator	Bajaradigan vazifasi
`defaultfloat`	Standart chiqarish turiga qaytaradi
`fixed`	Qo‘zg‘aluvchan nuqtali sonni qo‘zg‘almas nuqtali son ko‘rinishida chiqaradi
`scientific`	Keyingi barcha haqiqiy sonlarni qo‘zg‘aluvchan nuqtali son ko‘rinishida chiqaradi
`setprecision(n)`	Kasr sonlarni n xona aniqlikda chiqarish (<iomanip> kutubxonasi dasturga qo‘shiladi)

Quyidagi dasturda shu manipulyatorlarni qo‘llash ko‘rsatilgan:

Dastur	Natijasi

`#include <iostream>`
`#include <iomanip>`
`using namespace std;`
`int main(){`
`  double a=1.2345;`
`  cout<<"0: "<<a<<endl;`
`  cout<<setprecision(10);`
`  cout<<showpoint;`
`  cout<<"1: "<<a<<endl;`
`  cout<<fixed<<"2: "<<a<<endl;`
`  cout<<defaultfloat<<"3: "<<a<<endl;`
`  cout<<scientific<<"4: "<<a<<endl;`
`  return 0;`
`}`
	`0: 1.2345`
	`1: 1.234500000`
	`2: 1.2345000000`
	`3: 1.234500000`
	`4: 1.2345000000e+00`

Ba'zan chiqarilayotgan ma'lumotlar uchun biror kenglikdagi soha ajratish, sohada ma'lumotlarni chapdan yoki o'ngdan tekislash kabi amallarni ham bajarish kerak bo'ladi. Quyidagi manipulyatorlar shu kabi vaziyatlarda qo'llaniladi.

Manipulyator	Bajaradigan vazifasi
`setw(n)`	Kengligi n ta belgi bo'lgan soha hosil qiladi, soha chiqarilayotgan faqat bitta qiymatga tegishli bo'ladi (<iomanip> kutubxonasi dasturga qo'shiladi)
`setfill(c)`	Agar chiqarilayotgan qiymat sohani to'liq qamrab olmasa, bo'sh joylar c belgisi bilan to'ldiriladi (<iomanip> kutubxonasi dasturga qo'shiladi)
`left`	Natija sohaning chap tomoniga tekislanadi
`right`	Natija sohaning o'ng tomoniga tekislanadi
`internal`	Sonning ishorasi yoki asosi chap tomonga, sonning magnitudasi o'ng tomonga tekislab chiqariladi.

Quyidagi dastur formatli chiqarish imkoniyatlarini aks ettirgan.

Dastur	Natijasi

`#include <iostream>`
`#include <iomanip>`
`using namespace std;`
`int main(){`
`long double a = 1.0L / 9.0L; int k=-21;`
`cout<<setw(10)<<internal<<k<<endl;`
`cout<<setprecision(25)<<fixed<< a << '\n';`
`cout<<setprecision(25)<<scientific<<a<<endl;`
`cout <<setprecision(25)<<defaultfloat<<a;`
`cout <<endl<<setprecision(5);`
`cout << setw(10) << left << a << endl;`
`cout <<setw(10)<<setfill('#')<<left<<a<<'\n';`
`cout << setw(10) << right << a << endl;`
`cout << setw(10) << setfill('*') << a <<`
`endl;`
`int b = 0x123;`
`cout << dec << b << endl;`
`cout << oct << b << endl;`
`cout << hex << b << endl;`
`cout << hex << showbase << b << endl;`
`cout << showbase << b << endl;`
`return 0;`
`}`
	- 21
	0.11111111111111111111096053
	1.11111111111111111110960527e-001
	0.11111111111111111111096053
	0.11111
	0.11111###
	###0.11111
	*****0.11111**
	291
	443
	123
	0x123
	0x123
	0123

CIN OPERATORI

Ma'lumotlarni klaviaturadan kiritish uchun **cin** operatoridan foydalaniladi. Uning umumiy ko'rinishi quyidagicha:

```
cin >> o'zgaruvchi_1 >> o'zgaruvchi_2 >> … >> o'zgaruvchi_n;
```

Bu yerda o'zgaruvchi_1, o'zgaruvchi_2, …, o'zgaruvchi_n – butun (int, long long, …), haqiqiy (float, double, long double), belgili (char), mantiqiy (bool) turidagi o'zgaruvchilardir.

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  int a; float b;char c; bool t;`
`  cout<<"Kiruvchi ma'lumotlar:\n";`
`  cin >> a >> b; cin >> c; cin >> t;`
`  cout<<"Chiquvchi ma'lumotlar:\n";`
`  cout << a << " " << b << endl;`
`  cout << c << endl;`
`  cout << t << endl;`
`  return 0;`
`}`
	`Kiruvchi ma'lumotlar:`
	`1 2.3`
	`a`
	`true`
	
	`Chiquvchi ma'lumotlar:`
	`1 2.3`
	`a`
	`0`

Bu dasturda **cin** va **cout** operatorlarini qo'llash bir necha usullarda ifodalangan:

a) dastur ishga tushirilganda **a** o'zgaruvchiga qiymat kiritib **orachiq** klavishi bosilgan (C++ tili bu holda keyingi o'zgaruvchiga qiymat kiritilayotganini biladi);

b) **b**, **c** va **t** o'zgaruvchilarning har biriga qiymat kiritgandan keyin esa **Enter** klavishi bosilgandagi holatda ifodalangan.

Umuman olganda, har bir o'zgaruvchiga qiymat kiritgach, Enter klavishini bosish ham mumkin.

Izoh 8: Mantiqiy **t** o'zgaruvchining ekranga chiqarilgan qiymati **0** (ya'ni false ma'nosida) bo'ladi, chunki klaviatura orqali kiritilayotgan qiymat **true**, ya'ni 0 dan farqli bo'lgan biror **sonli qiymat** emas. Mantiqiy **true** va **false** qiymatlar faqat literal kabi qo'llangandagina C++ tili 1 va 0 deb qabul qiladi. **Klaviatura orqali kiritilayotgan mantiqiy qiymat 0 dan farqli son yoki 0 kabi bo'lishi shart**.

Demak, **bool** tipidagi o'zgaruvchilarni o'qishda **true** qiymati kiritilsa ham, o'zgaruvchiga **false**, ya'ni 0 qiymat o'zlashtiriladi. Klaviaturadan **true** yoki **false** qiymatlar kiritilganda mos ravishda **true** va **false** qiymatlar kabi o'zlashtirilishi uchun **boolalpha** manipulyatoridan foydalanish mumkin:

```
cin >> boolalpha >> t;
```

Bu usulda agar qiymat sifatida 1 kiritilsa, t o‘zgaruvchisiga **false** qiymati o‘zlashtiriladi. Odatiy holatga qaytarish uchun **noboolalpha** manipulyatoridan foydalaniladi:

```
cin >> noboolalpha >> t;
```

QO‘SHIMCHA IMKONIYAT: CIN OPERATORI UCHUN

Kiritish operatori **cin** bilan quyidagi funksiyalarni qo‘llash orqali turli xil yordamchi imkoniyatlarga ega bo‘lish mumkin:

Funksiya	Bajaradigan vazifasi
`int get()`	Kiritilgan belgini o‘qiydi va uni ASCII kodini qiymat sifatida qaytaradi
`get(char &c)`	Bitta belgini o‘qiydi va uni c o‘zgaruvchisiga o‘zlashtiradi
`get(char *s, signed int n, char delim)`	Uzunligi eng ko‘pi bilan **n** bo‘lgan **s** satrni o‘qiydi, delim – satrni ajratuvchi belgi: ya’ni shu belgi uchrashi bilan belgi o‘qilmasdan satrni o‘qish to‘xtatiladi (delimga qiymat bermasdan funksiyani ishlatish ham mumkin, bunda delim sifatida ‘\n’ - “yangi qator” olinadi). Agar kiritilgan satr uzunligi **n** dan katta bo‘lsa, u holda birinchi (**n-1**) ta belgi satrga o‘zlashtiriladi
`getline(char *s, signed int n, char delim)`	Uzunligi eng ko‘pi bilan **n** bo‘lgan **s** satrni o‘qiydi, delim – satrni ajratuvchi belgi: ya’ni shu belgi o‘qilgan zahoti satrni o‘qish to‘xtatiladi (delimga qiymat bermasdan funksiyani ishlatish ham mumkin, bunda delim sifatida ‘\n’ - “yangi qator” olinadi). Agar kiritilgan satr uzunligi **n** dan katta bo‘lsa, u holda birinchi (**n-1**) ta belgi satrga o‘zlashtiriladi
`ignore(signed int n, char delim)`	n ta belgini yoki delim belgisi uchraguncha kiritilgan belgilarni e’tiborsiz qoldiradi, ya’ni ularni “chetlab o‘tib” ketadi
`unget()`	O‘zlashtirilgan oxirigi belgini o‘qiladigan belgilar qatoriga qayta joylashtiradi
`putback(char c)`	c belgisini o‘qiladigan belgilar qatoriga joylashtiradi
`signed int gcount()`	O‘qib olingan belgilar sonini natija sifatida qaytaradi, bu funksiya faqat get, getline, putback, unget, ignore funksiyalari ishlashi natijasida o‘qilgan belgilar sonini qaytaradi, ya’ni cin >> s; orqali o‘qilgan belgilar soni ko‘rsatilmaydi

Quyidagi dasturga foydalanuvchi tomonidan bitta belgi kiritilgan. Dastur esa 3 marta yangi belgi o‘qib oladi. Bunga **unget** va **putback** funksiyalari sabab bo‘lib, **putback** funksiyasi **a** belgisini o‘qiladigan belgilar qatoriga qo‘shib qo‘ygani uchun oxirgi qatorda **a** belgisi chiqadi:

Dastur

```
#include <iostream>
using namespace std;
int main(){
  char c;
  cout<<"Kiritilgan qiymat: "<<endl;
  c=cin.get(); //belgini o'qidi va c ga o'zlashtirdi
  cout<<"kiritilgan belgi: "<<c<<endl;
  cin.unget(); //c ga o'zlashtirilgan belgi o'qiladigan belgilar
            //qatoriga qaytarildi qo'yish, bunda c ga o'zlashtirigan
            // qiymat saqlab qolinadi!
  cin.get(c); //c ga belgini o'qidi
  cout<<"o'qilgan belgi: "<<c<<endl;
  cin.putback('a'); //o'qiladigan belgilar qatoriga 'a' belgisi qo'shildi
  cin.get(c); //navbatdagi belgi o'qildi
  cout<<"qo'shilgan belgi: "<<c<<endl;
  return 0;
}
```

Natijasi

```
Kiritilgan qiymat:
r
kiritilgan belgi: r
o'qilgan belgi: r
qo'shilgan belgi: a
```

Quyidagi dasturda **getline** funksiyasining ishlashi ko‘rsatilgan.

Dastur

```
#include <iostream>
using namespace std;
int main(){
      int x;
      char s[100];//eng ko'pida 99 ta belgini
      //o'zida saqlay oladigan satr
      cout<<"Kiritilgan qiymat: "<<endl;
      cin.getline(s, 10, '2');
      cout<<"birinchi satr: "<<s<<endl;
      cin.getline(s, 10, '1');
      cin>>x;
      cout<<"ikkinchi satr: "<<s<<endl;
      cout<<"son: "<<x<<endl;
      return 0;
}
```

Natijasi

```
Kiritilgan qiymat:
abcdef2jklmn1234
birinchi satr: abcdef
ikkinchi satr: jklmn
son: 234
```

Dasturda kiritilgan qiymat `abcdef2jklmn1234` bo'lib, birinchi **`cin.getline(s, 10, '2');`** yordamida **s** ga eng ko'pi bilan **10** ta belgini yoki **`'2'`** belgisigacha bo'lgan belgilarni o'qish ko'zda tutilgan. Shu sababli o'qilgan 7-belgi **`'2'`** belgisi bo'lgani uchun birinchi satrga `abcdef` o'zlashtirildi. Ikkinchi **`cin.getline(s, 10, '1');`** yordamida **s** ga eng ko'pi bilan **10** ta belgini yoki **`'1'`** belgisigacha bo'lgan belgilarni o'qish ko'zda tutilgan. Shu sababli **`'2'`** belgisidan keyin o'qilgan 6-belgi **`'1'`** belgisi bo'lgani uchun ikkinchi satrga `jklmn` o'zlashtirildi. Shundan keyin ***x*** o'zgaruvchiga qolgan qiymatlar o'qildi, ya'ni ***x*** ning qiymati `234` ga teng bo'ldi.

Quyidagi dasturda esa **ignore** funksiyasidan foydalanish imkoniyati ifodalangan.

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  char s[100];`
`  cout<<"Kiritilgan qiymat: "<<endl;`
`  cin.ignore(2);`
`  cin.get(s,10);`
`  cout<<"Natija: "<<endl;`
`  cout<<s<<endl;`
`  return 0;`
`}`
	`Kiritilgan`
	`qiymat:`
	`nodavlat`
	`Natija:`
	`davlat`

Dasturda kiritilgan qiymat `nodavlat` bo'lib, `ignore(2)` kiritishda 2 ta belgini "chetlab o'tib" ketadi, shu sababli natija `davlat` bo'ladi.

Agar cin.ignore(); deb yozilsa, faqat bitta belgi chetlab o'tiladi. Bu funksiya quyidagicha muammolarni hal etishda ayniqsa foydalidir: agar dasturda

```
...
cin.get(s, 10);
cin.get(s, 10);
...
```

yozilgan bo'lsa va birinchi satr uchun 9 tadan kam belgi kiritilgan bo'lsa, ya'ni masalan:

```
Toshkent
Buxoro
```

qiymatlar kiritilgan bo'lsa, u holda ikkinchi satrga hech qanday qiymat o'zlashtirilmaydi.

Bunga quyidagi sabab bo'ladi: birinchi funksiya bajarilgandan so'ng kiritiladigan qiymatlarning navbatdagisi '\n' - "yangi qator" bo'ladi. Yuqorida aytib o'tilganidek, **get** funksiyasi '\n' belgisi uchraguncha belgilarni o'qishni davom ettiradi. Shundan kelib chiqib, ikkinchi safar **get** funksiyasi qo'llanganda hech qanday qiymat o'zlashtirilmaydi.

Bu muammoni yechish uchun **ignore** funksiyasidan foydalanish mumkin:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  char a[10], b[10], c[10], d[10];`
`  cout<<"Kiritilgan qiymatlar: "<<endl;`
`  cin.get(a, 10);`
`  cin.ignore();`
`  cin.get(b, 10);`
`  cin.ignore();`
`  cin.get(c, 10);`
`  cin.get(d, 10);`
`  cout<<"Natija: "<<endl;`
`  cout << "a = " << a << endl;`
`  cout << "b = " << b << endl;`
`  cout << "c = " << c << endl;`
`  cout << "d = " << d;`
`  return 0;`
`}`
	`Kiritilgan qiymatlar:`
	`Toshkent`
	`Samarqand`
	`Buxoro`
	`Natija:`
	`a = Toshkent`
	`b = Samarqand`
	`c = Buxoro`
	`d =`

Dasturdan ko‘rinib turibdiki, **d** satrini o‘qishdan avval **ignore** funksiyasi qo‘llanilmaganligi **d** satrga hech qanday qiymat o‘zlashirilmasligiga olib kelgan.

Agar har bir yangi qiymat yangi qatordan kiritiladigan bo‘lsa, butun, haqiqiy, belgili, mantiqiy turdagi qiymatlar o‘qilgandan so‘ng **get** yoki **getline** funksiyalarini ishlatish zarur bo‘lsa, u holda bu funksiyalardan avval **ignore** funksiyasini ishlatish zarur. Aks holda yuqorida aytib o‘tilgan muammoli holat yuzaga keladi.

Dasturda ba’zi belgilarni “chetlab o‘tish” (o‘qimaslik) zarur bo‘lib qoladi. Manipulyatorlardan **skipws** (belgilarni chetlab o‘tish) va **noskipws** (belgilarni chetlab o‘tmaslik) kiritilayotgan qiymatlar orasidan **Whitespace** belgilarni chetlab o‘tish yoki chetlab o‘tmaslikni bosh-qarishga xizmat qiladi. Whitespace belgilarga: `' '`, `'\n'`, `'\v'`, `'\t'`, `'\r'`, `'\f'` belgilari kiradi. Odatda, **skipws** manipulyatori qo‘llangan deb hisoblash mumkin.

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  char a, b, c;`
`  cout<<"Kiritilgan qiymatlar: "<<endl;`
`  cin >> noskipws >> a >> b >> c;`
`  cout<<"Natija: "<<endl;`
`  cout << a << b << c << endl;`
`  cout<<"Kiritilgan qiymatlar: "<<endl;`
`  cin >> skipws >> a >> b >> c;`
	`Kiritilgan`
	`qiymatlar:`
	`r a`
	`Natija:`
	`r a`
	`Kiritilgan`
	`qiymatlar:`
	`r a d`
	`Natija:`
	`rad`

```
        cout<<"Natija: "<<endl;
        cout << a << b << c;
        return 0;
}
```

Bu dasturda ikkinchi marta belgilar o‘qilayotganda orachiq belgisi “chetlab” o‘tilgan.

Ba’zan o‘qib olingan belgi qanday belgi ekanligini bilish va shunga asosan ish ko‘rishga to‘g‘ri keladi. Bunda <cctype> kutubxonasiga tegishli funksiyalardan foydalanish mumkin.

Funksiya	Tekshiradi
isalnum	10 lik raqam, katta yoki kichik registrdagi harf
isalpha	Katta yoki kichik registrdagi harf
isblank	‘\t’ yoki ‘ ’ belgilari
iscntrl	ASCII jadvalidagi 0x00..0x1f diapazondagi va 0x7f belgilari
isdigit	Raqam
isgraph	Belgining grafik ko‘rinishi borligi
islower	Kichik registrdagi harf
isprint	Ekranga chiqarish mumkin bo‘lgan belgi
ispunct	Punktuatsiya belgisi (isalnum ga kirmaydigan belgilar)
isspace	‘ ’, ‘\t’, ‘\v’, ‘\n’, ‘f’, ‘\r’ belgilaridan biri
isupper	Yuqori registrdagi harf
isxdigit	O‘n oltilik sanoq sistemasidagi raqam (0..9,A..F,a..f)

Agar tekshirish natijasi ijobiy bo‘lsa, bu funksiyalar 0 dan farqli son qaytaradi, aks holda 0 ni qaytaradi.

Dastur	Natijasi

`#include <iostream>`
`#include <cctype>`
`using namespace std;`
`int main(){`
`  char c = 'a';`
`  cout << isalpha(c) << endl;`
`  cout << isalnum(c) << endl;`
`  cout << isdigit(c) << endl;`
`  cout << isspace(c) << endl;`
`  cout << islower(c) << endl;`
`  cout << isupper(c) << endl;`
`  c = ' ';`
`  cout << isblank(c) << endl;`
`  cout << isspace(c) << endl;`
`  cout << ispunct(c) << endl;`
`  return 0;`
`}`
	1024
	8
	0
	0
	512
	0
	1
	8192
	0

Belgining registrini o‘zgartirish uchun quyidagi funksiyalardan foydalanish mumkin:

Funksiya	Bajaradigan vazifasi
`int tolower(int c)`	c belgisini quyi registrga o‘tkazadi, agar belgining quyi registrga mos ko‘rinishi bo‘lmasa, o‘zgarishsiz qoldiradi
`int toupper(int c)`	c belgisini yuqori registrga o‘tkazadi, agar belgining quyi registrga mos ko‘rinishi bo‘lmasa, o‘zgarishsiz qoldiradi

Bu funksiyalarni qo‘llashga quyidagi dastur misol bo‘ladi.

Dastur	Natijasi

`#include <iostream>`
`#include <cctype>`
`using namespace std;`
`int main(){`
`  char c = 'a', d = 'B', r = '4';`
`  char cc = toupper(c), dd = tolower(d);`
`  cout << cc << " " << dd << endl;`
`  cc = tolower(c);dd = toupper(d);`
`  cout << cc << " " << dd << endl;`
`  char r1 = toupper(r), r2 = tolower(r);`
`  cout << r1 << " " << r2 << endl;`
`  return 0;`
`}`
	`A b`
	`a B`
	`4 4`

PRINTF VA SCANF FUNKSIYALARI

C++ tilida ma’lumotlarni xotiraga klaviaturadan kiritish va ekranga chiqarishning C uslubidagi imkoniyati ham mavjud. Bu imkoniyatni C uslubidagi turi deyilishiga sabab, bu funksiyalar aslida C dasturlash tilining elementlari hisoblanadi. Bu uslubda ma’lumotlarni ekranga chiqarish funksiyasi – **printf**, ma’lumotlarni klaviaturadan kiritish funksiyasi esa – **scanf**. Bu funksiyalardan foydalanish uchun dasturga **cstdio** yoki **stdio.h** kutubxonasini qo‘shish zarur. Bunda (agar dasturdagi boshqa funksiyalar talab qilmasa) **using namespace std;** ko‘rsatmasini yozish shart emas, chunki **printf** va **scanf** funksiyalari global nomlar fazosida aniqlangan.

CHIQARISH: printf funksiyasi

Bu funksiyaning umumiy ko‘rinishi quyidagicha:

```
int printf(const char* format_satri, …);
```

bunda **format_satri** – chiqariladigan matn va argumentlar formatidan tashkil topgan bo‘ladi. Argumentlar formati **% prefiksidan** (so‘zoldi qo‘shimchasidan) boshlab **maxsus belgi – spetsifikator** orqali yoziladi. Spetsifikator shu o‘rinda chiqariladigan argument qiymatining

turini bildiradi. Format_satridan keyin esa vergul bilan ajratilgan argumentlar keladi. Ular format_satrida ko'rsatilgan tartibda kelishi shart.

Prefiks bilan yozilgan spetsifikatorlar quyidagilardan iborat:

Spetsifikator	Ma'nosi
%c	Belgi
%d	O'nli butun son
%i	O'nli butun son
%e	Eksponensial ko'rinishdagi son (quyi registrdagi e bilan)
%E	Eksponensial ko'rinishdagi son (yuqori registrdagi E bilan)
%f	Qo'zg'aluvchan nuqtali son
%g	%e yoki %f dan qaysi biri qisqa bo'lsa, shu qisqasini
%G	%E yoki %F dan qaysi biri qisqa bo'lsa, shu qisqasini
%o	8 lik unsigned (manfiymas) butun son
%s	Satr
%u	O'nlik unsigned (manfiymas) son
%x	O'n oltilik unsigned (manfiymas) son (quyi registrda)
%X	O'n oltilik unsigned (manfiymas) son (yuqori registrda)
%p	Ko'rsatkich
%%	% belgisi

Quyidagi dastur spetsifikatorlar berishi mumkin bo'lgan imkoniyatlarni aks ettirgan.

Dastur	Natijasi

`#include <cstdio>`
`int main(){`
`  printf("%s - %i\n", "Toshkent", 2018);`
`  printf("6 + 4 %c %i", '=', 10);`
`  return 0;`
`}`
	`Toshkent - 2018`
	`6 + 4 = 10`

Dasturda "%s - %i\n" yozilgani uchun undagi %s turgan o'ringa Toshkent, keyin oradagi ikki tomonida orachiq bo'lgan - belgisi, so'ng %i ning o'rniga 2018 chiqariladi. Zarur bo'lsa, ekranga chiqarilgan belgilar sonini aniqlash mumkin. Ya'ni dasturdagi **printf** funskiyasi o'rniga **int a = printf("%s - %i\n", "Toshkent", 2018);** kabi yozilsa, ekranga `Toshkent - 2018` chiqariladi va ekranga chiqarilgan belgilar miqdoriga teng 16 esa **int** turidagi **a** o'zgaruvchiga o'zlashtiriladi.

Chiqarilayotgan qiymatning uzunligi kamida **n** ta bo'lishi uchun **%nT** kabi yoziladi, bu yerda **T** – biron-bir spetsifikator. Agar chiqarilayotgan qiymat uzunligi ko'rsatilgan **n** dan kam bo'lsa, u holda qolgan joylar orachiq bilan to'ldiriladi. Bunda chiqarilayotgan qiymat uning uchun ajratilgan n ta belgi uzunligidagi sohaning o'ng tomonidan tekislanadi, chap tomondan tekislash uchun esa **%-nT** kabi yozish zarur.

Dastur	Natijasi

`#include <cstdio>`
`int main(){`
`  const char* a = "Salom!";`
`  printf("%10s\n", a);`
`  printf("%-10s\n", a);`
`  return 0;`
`}`
	`  Salom!`
	`Salom!`

Qo‘zg‘aluvchan nuqtali sonlarni verguldan keyin **n** ta raqam aniqlikda ekranga chiqarish uchun **%.nf** kabi yozish kerak:

Dastur	Natijasi

`#include <cstdio>`
`int main(){`
`  float a = 123.0 / 7.0;`
`  printf("%.20f", a);`
`  return 0;`
`}`
	`17.57142829895019531250`

Butun sonlarni uzunligi **n** taga yetguncha prefiks 0 lar bilan to‘ldirish uchun **%0nd** kabi yozish kerak. Masalan:

Dastur	Natijasi

`#include <cstdio>`
`int main(){`
`  printf("%010d", 10);`
`  return 0;`
`}`
	`0000000010`

Satr uzunligini **n** dan oshmaydigan holda cheklash uchun **%.ns** kabi yoziladi.

Dastur	Natijasi

`#include <cstdio>`
`int main(){`
`  printf("%.5s\n", "Salom Toshkent");`
`  printf("%10.5s\n", "Salom Toshkent");`
`  return 0;`
`}`
	`Salom`
	`  Salom`

Chiqarilayotgan qiymat **long, long long, long double** bo‘lsa, spetsifikator oldiga l yoki ll qo‘shib yoziladi, ya’ni **%lT** kabi.

Dastur	Natijasi

`#include <cstdio>`
`int main(){`
`  printf("%d\n", 10'000'000'000LL);`
`  printf("%lld\n", 10'000'000'000LL);`
`  return 0;`
`}`
	`1410065408`
	`10000000000`

Dasturdagi birinchi **printf** funksiyasidagi spetsifikator oldiga l yoki ll qo‘shilmagani uchun ekranga xato qiymat, ikkinchi **printf** funksiyasidagi spetsifikator oldiga l yoki ll qo‘shilgani uchun ekranga to‘g‘ri qiymat chiqarildi. Eslatib o‘tamiz, C++14 standartidan boshlab butun literallar razryadlari (xonalari) qulay ko‘rinishi uchun 10'000'000'00 kabi yozish imkoniyati kiritilgan.

KIRITISH: scanf funksiyasi

Bu funksiyaning umumiy ko‘rinishi quyidagicha:

```
int scanf(const char* format_satri, …);
```

bunda **format_satri** da kiritiladigan qiymatlar % prefiks va turiga mos spetsifikatorlar yoziladi.

Prefiks bilan yozilgan spetsifikatorlar quyidagilardan iborat:

Spetsifikator	Kiritiladigan qiymat turi
%c	Belgili
%d	O‘nlik butun son
%i	Butun son (umumiy)
%e	Haqiqiy son
%f	Haqiqiy son
%F	Haqiqiy son
%g	Haqiqiy son
%o	8 lik butun son
%s	Satr
%x	16 lik butun son
%p	Ko‘rsatkich
%u	O‘nlik unsigned (nomanfiy) son
%%	% belgisini

Bunda **scanf** funksiyasiga argument sifatida qiymatlarni joylashtirish kerak bo‘lgan o‘zgaruvchilarning adreslari ko‘rsatiladi, ya’ni **&a** orqali a o‘zgaruvchi uchun ajratilgan adresga joylashtirish talab etiladi. Format_satri da orachiqlar qo‘yilsa, o‘sha o‘rinda kiritilayotgan qiymatlar orasidagi orachiqlar qiymat sifatida qabul qilinmaydi, ya’ni tashlab o‘tiladi.

Dastur	Natijasi

`#include <cstdio>`
`int main(){`
`  int a1, a2;  char c1, c2;`
`  printf("Kiruvchi ma'lumotlar:\n");`
`  scanf("%d %c", &a1, &c1);`
`  scanf("%d%c", &a2, &c2);`
`  printf("Chiquvchi ma'lumotlar:\n");`
`  printf("%d %c\n", a1, c1);`
`  int d=printf("%d%c", a2, c2);`
`  printf("\n%s%i","d=", d);`
`  return 0;`
`}`
	`Kiruvchi ma'lumotlar:`
	`21 A`
	`23 M`
	
	`Chiquvchi ma'lumotlar:`
	`21 A`
	`23`
	`d=3`

Ikkinchi **scanf("%d%c", &a2, &c2);** ning format_satri da orachiq qo‘yilmagan, shuning uchun kiritilayotgan 23 qiymat a2 o‘zgaruvchiga va undan keyingi orachiq c2 o‘zgaruvchiga o‘zlashtirildi. Shu sababli ekranga a2 o‘zgaruvchi qiymati sifatida 23 va c2 o‘zgaruvchi qiymati sifatida orachiq chiqarilgan. Buni ekranga chiqarilgan belgilar sonini ko‘rsatuvchi d o‘zgaruvchining qiymatidan bilish mumkin.

O‘zgaruvchilarning **long**, **long long**, **long double** turlaridan foydalanish uchun spetsifikator oldiga l yoki ll yozilishi kerak. Quyidagi dasturda shunga ko‘ra xatolik kelib chiqqan.

Dastur	Natijasi

`#include <cstdio>`
`int main(){`
`  long long a, b;`
`  scanf("%d %lld", &a, &b);`
`  printf("%lld %lld", a, b);`
`  return 0;`
`}`
	`10000000000 10000000000`
	`263403070464 10000000000`

7-§. QO‘SHMA AMALLAR

C++ tilidagi dasturda qo‘shma amallardan foydalanish dastur matnini qisqartiradi va qulayliklar beradi. Qo‘shma amallarga quyidagilar misol bo‘ladi:

Qo‘shma amal	Vazifasi
+=	Qo‘shib o‘zlashtirish
-=	Ayirib o‘zlashtirish
*=	Ko‘paytirib o‘zlashtirish
/=	Butun bo‘lib o‘zlashtirish
%=	Qoldiqni hisoblab o‘zlashtirish
&=	Bitga mos mantiqiy ko‘paytirib o‘zlashtirish
|=	Bitga mos mantiqiy qo‘shib o‘zlashtirish
^=	Bitga mos o‘xshashlikni inkor etib o‘zlashtirish
<<=	Razryadli chapga surib o‘zlashtirish
>>=	Razryadli o‘ngga surib o‘zlashtirish

Qo‘shma amallarda avval birinchi (chap) amal bajariladi, so‘ngra o‘zlashtirish bajariladi. Ma’lumki, a = a + 1; ifodada avval a o‘zgaruvchining (eski, ya’ni xotiradagi) qiymatiga 1 qo‘shiladi va a o‘zgaruvchiga natijaviy (yangi) qiymat o‘zlashtiriladi (xotiraga yoziladi). Qulaylik uchun yoki xohishga qarab a = a + 1; ifodaning o‘rniga a += 1; kabi yozish ham mumkin.

Agar dasturda a <<= 1; ifoda yozilgan bo‘lsa, u holda avval a o‘zgaruvchining (eski) qiymati 1 razryad chapga suriladi va natijaviy (yangi) qiymat a o‘zgaruvchiga o‘zlashtiriladi.

INKREMENT VA DEKREMENT AMALLARI

Inkrement (qiymatni 1 taga oshirish) va dekrementlar (qiymatni 1 taga kamaytirish) o‘zgaruvchining qaysi tomoniga yozilganiga qarab turlicha nomlanadi:

O‘qilishi	Yozilishi
Postfix-inkrement (keyin yozilsa)	`a++;`
Prefix-inkrement (oldin yozilsa)	`++b;`
Postfix-dekrement (keyin yozilsa)	`c--;`
Prefix-dekrement (oldin yozilsa)	`--d;`

Prefix-inkrement a = ++b; va a = b++; postfix-inkrementning qiymatlari turlicha bo‘ladi. Buning sababi quyidagicha: a = ++b; ifodada avval b o‘zgaruvchining qiymati bittaga oshiriladi va keyin a o‘zgaruvchiga o‘zlashtiriladi; a = b++; ifodasida esa avval a o‘zgaruvchiga b o‘zgaruvchining qiymati o‘zlashtiriladi va keyin b o‘zgaruvchining qiymati bittaga oshiriladi:

Boshlang‘ich qiymatlar	Ifoda	Natijaviy qiymatlar
a = 10; b = 8;	a = ++b;	a = 9, b = 9;
a = 10; b = 8;	a = b++;	a = 8, b = 9;
a = 10; b = 8;	a = --b;	a = 7, b = 7;
a = 10; b = 8;	a = b--;	a = 8, b = 7;

Inkrement va dekrement amallari qo‘zg‘aluvchan nuqtali sonlar qiymatini 1.0 ga o‘zgartiradi.

Inkrement va dekrement amallaridan, masalan, sanoq uchun foydalanish mumkin.

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  int a = 0, b = 5, c = 0, d = 5;`
`  a++; b--; ++c; --d;`
`  cout<<a<<" "<<b<<" "<<c<<" "<<d<<endl;`
`  a++; b--; ++c; --d;`
`  cout<<a<<" "<<b<<" "<<c<<" "<<d<<endl;`
`  a++; b--; ++c; --d;`
`  cout<<a<<" "<<b<<" "<<c<<" "<<d<<endl;`
`  a++; b--; ++c; --d;`
`  cout<<a<<" "<<b<<" "<<c<<" "<<d<<endl;`
`  a++; b--; ++c; --d;`
`  cout<<a<<" "<<b<<" "<<c<<" "<<d<<endl;`
`  return 0;`
`}`
	1 4 1 4
	2 3 2 3
	3 2 3 2
	4 1 4 1
	5 0 5 0

2-BOB. BUTUN VA MANTIQIY TURLARGA OID MASALALAR

Dastur tuzishga oid masalalarda aniq qiymatlar beriladi yoki "berilgan" iborasi uchraydi. Agar qiymatlar aniq berilgan bo'lsa, bu qiymatlarni dasturga bevosita literallar orqali yozish mumkin yoki o'zgaruvchilarga shu qiymatlarni klaviatura orqali kiritish mumkin. Lekin "berilgan" iborasi uchragan masalalarda o'zgaruvchilarga qiymatlar klaviatura orqali kiritiladi.

Butun va mantiqiy turga oid masalalarda hisoblash amallari butun sonlar ustida amalga oshiriladi. Butun sonlar ustida esa qo'shish, ayirish, ko'paytirish, butun bo'lish va qoldiq hisoblash amallarini bajarish mumkin. Qaralayotgan masalalarda sanab o'tilgan amallar yordamidagina natijaga erishish talab qilinadi.

Ba'zi masalalarning yechimi bob mavzusidan tashqariga chiqib ketadigandek, biror funksiya yoki operatorni qo'llash majburiydek ko'rinadi. Lekin mazkur masalalarni mantiqiy fikrlash va o'ziga xos algoritm yordamida bob mavzusidan chiqmasdan yechish mumkin. Shu maqsadda 1-paragrafda bir necha namunaviy masalalar, bu masalalar uchun zarur bo'lgan ma'lumotlar, o'ziga xos algoritmik usullar, C++ tilining masala yechimidagi imkoniyatlari va masala yechimiga mos dasturlar keltirib o'tildi.

Mavzu masalalarini yecha olish uchun matematika, fizika va informatikaga oid quyidagi o'lchov birliklarni bilish talab etiladi:

1 kilometr=1000 metr, 1 metr=10 detsimetr=100 santimetr, 1 santimetr=10 millimetr;

1 tonna=10 sentner =1000 kilogramm, 1 kilogramm =1000 gramm;

1 yil=365 sutka, 1 sutka=24 soat, 1 soat=60 minut=3600 sekund, 1 minut=60 sekund;

1 gigabayt=1024 megabayt, 1 megabayt=1024 kilobayt, 1 kilobayt=1024 bayt.

Sonlar nazariyasiga oid ma'lumotlar:

Natural sonlar to'plami **N**	**N={1; 2; 3; ...}**
Manfiymas butun sonlar to'plami Z_0	Z_0**={0; 1; 2; 3; ...}**
Butun sonlar to'plami **Z**	**Z={...;–3; –2; –1; 0; 1; 2; 3; ...}**
Ratsional sonlar to'plami **Q**	$Q=\{\frac{p}{q} \mid p\in Z, q\in N\}$
Irratsional sonlar to'plami **I**	$p\in Z$, $q\in N$ bo'lganda $\frac{p}{q}$ ko'rinishida tasvirlab bo'lmaydigan sonlar to'plami
Haqiqiy sonlar to'plami **R**	$R=(-\infty; +\infty)=Q\cup I$

Haqiqiy sonning butun va kasr qismlarini tasvirlashda turli belgilashlar va tushunchalar qo'llanadi:

- $\lfloor x \rfloor$ – **quyi butun qism**, ya'ni x dan katta bo'lmagan eng katta butun son, masalan: a) x=2,7 bo'lsa, u holda $\lfloor 2,7 \rfloor=2$; b) x= –2,7 bo'lsa, u holda $\lfloor -2,7 \rfloor= -3$;

- $\lceil x \rceil$ – **yuqori butun qism**, ya'ni x dan kichik bo'lmagan eng kichik butun son, masalan: a) x=2,7 bo'lsa, u holda $\lceil 2,7 \rceil = 3$; b) x= –2,7 bo'lsa, u holda $\lceil -2,7 \rceil = -2$;
- $\{x\}$ – sonning kasr qismi, ya'ni $\{x\} = x - \lfloor x \rfloor$, masalan, a) x=2,7 bo'lsa, u holda $\{2,7\} = 2,7 - \lfloor 2,7 \rfloor = 2,7 - 2 = 0,7$; b) x=–2,7 bo'lsa, u holda $\{-2,7\} = -2,7 - \lfloor -2,7 \rfloor = -2,7 - (-3) = 3 - 2,7 = 0,3$.

Izoh 9: Ko'p adabiyotlarda **quyi butun qismni sonning butun qismi** deb ataladi va [x] kabi belgilanadi. Biz ham, agar zarurat bo'lmasa, quyi butun qismni sonning butun qismi deb qo'llaymiz. Ta'riflardan, agar **n** soni butun bo'lsa, quyidagi xossalar kelib chiqadi:

$$\lfloor n \rfloor = n = \lceil n \rceil = [n]$$
$$n = \lfloor n/2 \rfloor + \lceil n/2 \rceil.$$

Haqiqiy sonlar uchun quyidagi xossalar o'rinli ($x, y \in R$, $n, m \in Z$):

$$x - 1 < \lfloor x \rfloor \le x \le \lceil x \rceil < x + 1$$
$$\lfloor -x \rfloor = -\lceil x \rceil$$
$$\lceil -x \rceil = -\lfloor x \rfloor$$
$$\lfloor x + n \rfloor = \lfloor x \rfloor + n$$
$$\lfloor x + y \rfloor = \lfloor x \rfloor + \lfloor y \rfloor + \lfloor \{x\} + \{y\} \rfloor$$
$$\left\lfloor \frac{x+m}{n} \right\rfloor = \left\lfloor \frac{\lfloor x \rfloor + m}{n} \right\rfloor$$
$$\left\lceil \frac{x+m}{n} \right\rceil = \left\lceil \frac{\lceil x \rceil + m}{n} \right\rceil$$
$$0 \le \{x\} < 1.$$

Natural n va m sonlar nisbatini **n/m**, qoldig'ini **n mod m** kabi belgilasak, nisbatning butun qismi $\lfloor \mathbf{n/m} \rfloor$ ga tengligini hisobga olgan holda quyidagi tengliklarni yozish mumkin:

$$\mathbf{n = \lfloor n/m \rfloor \cdot m + n \bmod m}$$
$$\mathbf{n = \left\lfloor \frac{n}{m} \right\rfloor + \left\lfloor \frac{n-1}{m} \right\rfloor + \left\lfloor \frac{n-2}{m} \right\rfloor + \ldots + \left\lfloor \frac{n-m+1}{m} \right\rfloor}$$
$$\mathbf{n = \left\lfloor \frac{n}{m} \right\rfloor + \left\lfloor \frac{n+1}{m} \right\rfloor + \left\lfloor \frac{n+2}{m} \right\rfloor + \ldots + \left\lfloor \frac{n+m-1}{m} \right\rfloor.}$$

Ma'lumki, qoldiq uchun quyidagilar o'rinli:

$$\mathbf{n \bmod m = n - \lfloor n/m \rfloor \cdot m}$$
$$\mathbf{0 \le n \bmod m < m}$$
$$\mathbf{0 \le n \bmod m \le m - 1.}$$

Butun n va m ($m \neq 0$) sonlar uchun qoldiqni $\mathbf{n \bmod m = n - \lfloor n/m \rfloor \cdot m}$ kabi aniqlash mumkin. U holda quyidagilar o'rinli:

$$\mathbf{0 \le n \bmod m < m,} \text{ agar } \mathbf{m > 0}$$
$$\mathbf{m < n \bmod m \le 0,} \text{ agar } \mathbf{m < 0.}$$

Qoldiq uchun quyidagi formulalar ham o'rinli ($c \in Z$):

$$(n + c) \bmod m = (n \bmod m + c \bmod m) \bmod m$$

$$c \cdot (n \bmod m) = (c \cdot n) \bmod (c \cdot m).$$

Quyidagi qoidalar ham sonlarning xossalari bilan bog'liq:

1) Sanoq qoidasi: A butun sondan B butun songacha sanalganda (B–A)+1 ta son sanaladi;

2) Juftlik qoidasi: Juft son 2 ga qoldiqsiz bo'linadi;

3) Juft va toq sonlar qoidalari (t, t1, t2 – qandaydir toq va j, j1, j2 – qandaydir juft sonlar):
t1+t2=j; t1+j=t2; j1+j2=j; t1–t2=j; t1–j=t2; j1–j2=j; t1·t2=t; t·j1=j2; j1·j2=j.

Sonlarni tasvirlash usullari quyidagilar:

1) **Ixcham** (oddiy) ko'rinish – son raqamlari razryadi bo'yicha ketma-ket yoziladi:

$$\text{son} = \overline{a_k a_{k-1} \dots a_0 a_{-1} a_{-2} \dots a_{-n}}_p,$$

bu yerda a_k, a_{k-1},..., a_0, a_{-1}, a_{-2},..., a_{-n} – berilgan sonni tashkil etuvchi raqamlar, p – sanoq sistemasi asosi (matematikada son ustiga chiziq chizilishi son raqamlari qiymati oshkormas, ya'ni umumiy ko'rinishda berilgandagina qo'llanadi), masalan: $19501{,}902_{10}$, $210719{,}63AA_{16}$;

2) **Yoyiq** ko'rinish – son raqamlari va sanoq sistemasi asosini raqamlar razryadlariga mos darajalariga ko'paytmalari yig'indisi ko'rinishida yoziladi:

$$\text{son} = \overline{a_k a_{k-1} \dots a_0 a_{-1} a_{-2} \dots a_{-n}}_p =$$

$$= a_k \cdot p^k + a_{k-1} \cdot p^{k-1} + \dots + a_1 \cdot p^1 + a_0 \cdot p^0 + a_{-1} \cdot p^{-1} + a_{-2} \cdot p^{-2} + \dots + a_{-n} \cdot p^{-n}$$

bu yerda a_k, a_{k-1},..., a_0, a_{-1}, a_{-2},..., a_{-n} – berilgan sonni tashkil etuvchi raqamlar, p – sanoq sistemasi asosi.

Ketma-ketliklar uchun kerakli ta'riflarni keltiramiz:

1) $a_1, a_2, \dots, a_n$ o'suvchi ketma-ketlik deyiladi, agar quyidagi tengsizliklar bajarilsa:

$$a_1 < a_2 < \dots < a_n$$

2) $a_1, a_2, \dots, a_n$ kamayuvchi ketma-ketlik deyiladi, agar quyidagi tengsizliklar bajarilsa:

$$a_1 > a_2 > \dots > a_n$$

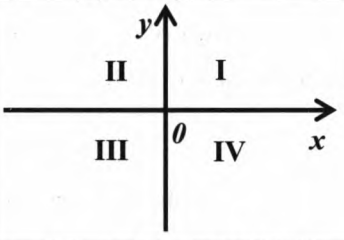
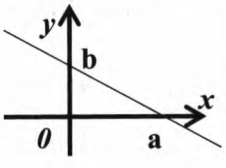
3) $a_1, a_2, \dots, a_n$ kamaymaydigan ketma-ketlik deyiladi, agar quyidagi tengsizliklar bajarilsa:

$$a_1 \le a_2 \le \dots \le a_n$$

4) $a_1, a_2, \dots, a_n$ o'smaydigan ketma-ketlik deyiladi, agar quyidagi tengsizliklar bajarilsa:

$$a_1 \geq a_2 \geq \ldots \geq a_n$$

Ba'zi masalalarni yechish uchun quyidagi ma'lumotlarni bilish zarur bo'ladi:

Tomonlari a va b bo'lgan to'g'ri to'rtburchak perimetri P_{tt}	$P_{tt}=2\cdot(a+b)$
Tomonlari a va b bo'lgan to'g'ri to'rtburchak yuzi S_{tt}	$S_{tt}=a\cdot b$
Tomonlari a, b va c bo'lgan uchburchak perimetri P_{uchb}	$P_{uch}=a+b+c$
Ox o'qidagi A(x) va B(x) nuqtalar orasidagi masofa dAB	$dAB=\lvert x-y\rvert$
Oxy tekisligidagi A(x1; y1) va B(x2; y2) nuqtalar orasidagi dAB masofa kvadrati	$dAB^2=(x1-x2)^2+(y1-y2)^2$
Katetlari a va b, gipotenuzasi c bo'lgan to'g'ri burchakli uchburchak uchun Pifagor teoremasi	$c^2=a^2+b^2$
Tomonlari a, b va c bo'lgan uchburchakning mavjudlik sharti (uchburchak qoidasi)	$\begin{cases} a+b>c \\ a+c>b \\ b+c>a \end{cases}$
Tomonlari a, b, c bo'lgan va o'sish tartibida saralangan ($a\leq b\leq c$) uchburchakning mavjudlik sharti	$a+b>c$
Oxy (Dekart) koordinata sistemasida kvadrantlarning (choraklarning) tartiblanishi	
Markazi (x0; y0) nuqtada bo'lgan radiusi R ga teng aylana tenglamasi	$(x-x0)^2+(y-y0)^2=R^2$
Markazi (x0; y0) nuqtada bo'lgan radiusi R ga teng doira tenglamasi	$(x-x0)^2+(y-y0)^2\leq R^2$
Ikki (x1; y1) va (x2; y2) nuqtalardan o'tuvchi to'g'ri chiziq tenglamasi	$\frac{x-x1}{x2-x1}=\frac{y-y1}{y2-y1}$
Oxy koordinata sistemasida Ox o'qida **a** uzunlikdagi, Oy o'qida **b** uzunlikdagi kesmalar ajratuvchi to'g'ri chiziq tenglamasi (boshqa nomi: to'g'ri chiziqning kesmalar orqali tenglamasi) 	$\frac{x}{a}+\frac{y}{b}=1$

1-§. BUTUN VA MANTIQIY TURLARGA OID MASALALAR YECHISH NAMUNALARI

Namuna 1. Alida har biriga m litr suv sig'adigan chelaklar bor edi. U suvga to'la N litr sig'imli idishdagi barcha suvni chelaklarga to'kib olishi kerak. Aliga kerak bo'ladigan eng kam sondagi chelaklar sonini chiqaruvchi dastur tuzing.

Yechim. Masala shartida "eng kam sondagi" deyilgani uchun har bir chelakni to'ldirish maqsadga muvofiq. Demak, agar N soni m soniga qoldiqsiz bo'linsa (ya'ni, $\frac{N}{m}$ son butun son bo'lib, barcha chelaklar to'la bo'ladi), u holda $\frac{N}{m} = \left[\frac{N}{m}\right]$ dona chelak, aks holda, ya'ni oxirgi chelak to'lmasa (lekin bu chelak ham band bo'ladi) $\left[\frac{N}{m}\right] = \left[\frac{N}{m}\right] + 1$ dona chelak kerak bo'ladi.

Muammo hosil bo'ldi: masala yechimi shartga bog'liq bo'lib qoldi. Qanday yo'l tutish kerak?

Ma'lumki, N sonini m soniga bo'lgandagi butun qismi $b=\left[\frac{N}{m}\right]$ ga, qoldiq esa C++ tilida q=N%m ga teng. Ya'ni $N = b \cdot m + q$ va m ga bo'lishda q qoldiq uchun $0 \le q \le m-1$ shart o'rinli.

Quyidagi ifodani tahlil etamiz:

$$\left[\frac{N+m-1}{m}\right].$$

1-hol. $q = N \% m = 0$ bo'lsin. U holda $\frac{N}{m}$ soni butun son bo'lgani uchun butun qismlar uchun keltirilgan yuqoridagi xossadan:

$$\left[\frac{N+m-1}{m}\right] = \left[\frac{N}{m} + \frac{m-1}{m}\right] = \left[b + \frac{m-1}{m}\right] = b + \left[\frac{m-1}{m}\right] = b + 0 = \left[\frac{N}{m}\right].$$

2-hol. $q = N \% m > 0$, ya'ni $1 \le q \le m-1$ bo'lsin. Shunga ko'ra $m \le q+m-1 \le 2 \cdot m-2$ tengsizlikka ega bo'lamiz. Bu tengsizlik hadlarini m ga bo'lsak, $1 \le \frac{q+m-1}{m} \le \frac{2 \cdot m-2}{m} < 2$ tengsizlik hosil bo'ladi. Oxirgi tengsizlikka ko'ra:

$$\left[\frac{N+m-1}{m}\right] = \left[\frac{b \cdot m + q+m-1}{m}\right] = \left[b + \frac{q+m-1}{m}\right] = b + \left[\frac{q+m-1}{m}\right] = b + 1 = \left[\frac{N}{m}\right] + 1.$$

Keltirib chiqarilgan 1-, 2-hollardagi ifodalarga ko'ra sonning butun qismlari uchun yana bir xossaga ega bo'ldik, ya'ni:

$$\left[\frac{N}{m}\right] = \left[\frac{N+m-1}{m}\right] = \left[\frac{N+m-1}{m}\right].$$

Demak, masalaning yechimi uchun birgina ifoda kifoya ekan.

Berilayotgan qiymatlar butun son bo‘lgani uchun dasturda barcha n, m, chs (bu yerda chs – chelaklar sonini bildiradi) o‘zgaruvchilarni **int** turida tavsiflaymiz.

Butun o‘zgaruvchilarni C++ tilida bo‘lish xossasini hisobga olib masala shartiga javob beruvchi dasturni izohlar bilan quyidagicha yozamiz:

```
#include <iostream>
using namespace std;
int main(){
   int n, m, chs;
   cout << "Idishdagi suv hajmi N= "; cin >> n;
   cout << "Chelakni hajmi m= ";cin >> m;
   chs = (n+m-1)/m;
   cout <<"Chelaklar soni "<< chs << " dona" << endl;
   return 0;
}
```

Namuna 2. Asfalt mashinasi to‘liq kun davomida uzunligi A metr yo‘lni asfalt bilan qoplay oladi. B kilometr yo‘lni asfaltlash uchun asfalt mashinasi qanchadir to‘liq kun davomida ishladi. Asfaltlashga qolgan yo‘lga bir to‘liq kun ko‘plik qilishi ma’lum bo‘lsa, qolgan yo‘lni metrlarda chiqaruvchi dastur tuzing.

Yechim. Asfalt qilinishi kerak bo‘lgan masofa kilometr o‘lchov birligida bo‘lgani uchun metr o‘lchov birligiga o‘tkazamiz: B kilometr=B·1000 metr.

Kuniga A metr yo‘lni asfalt bilan qoplaydigan mashina 1 kunda barcha yo‘lni 1·A metrini, 2 kunda barcha yo‘lni 2·A metrini, … metrini asfalt bilan qoplab bo‘lgan. Masala shartiga ko‘ra asfalt mashinasi qanchadir to‘liq kun, bu noma’lum to‘liq kunni **x** deb belgilaymiz, davomida ishlagan, ya’ni x kunda barcha yo‘lni x·A metrini asfalt bilan qoplab bo‘lgan. U holda asfaltlashga qolgan yo‘l q=(B·1000–x·A) metr bo‘ladi. Shartga ko‘ra, qolgan yo‘lni asfaltlashga bir to‘liq kun ko‘plik qiladi, demak: (B·1000–x·A) metr < A metr.

Endi a sonni b songa qoldiqli bo‘lishni eslaylik: a=k·b+q, bu yerda q qoldiq, k esa a sonni b songa bo‘lgandagi butun qism. Bo‘linmaning butun qismini quyidagicha yozish mumkin: k=[a/b], bu yerda [t] – t sonning butun qismi. Natijada qoldiq uchun quyidagilarga ega bo‘lamiz:

1) q=a–k·b=a–[a/b]·b;

2) 0≤q<b.

Qoldiq formulasiga ko‘ra (B·1000–x·A) metr < A metr tengsizlik mazmunidan quyidagilarga ega bo‘lamiz: 1) x= [B·1000/A]; 2) q=B·1000–x·A= B·1000–[B·1000/A]·A. Oxirgi formuladan esa masala yechimi uchun x noma’lumni aniqlash shart emasligi kelib chiqadi.

Berilayotgan qiymatlar butun son bo‘lgani uchun dasturda barcha a, b, q o‘zgaruvchilarni **int** turida tavsiflaymiz. Shartga asosan **b** o‘zgaruvchiga kilometr o‘lchov birligidagi B qiymatni, **a** o‘zgaruvchiga metr o‘lchov birligidagi A qiymatni kiritishni tashkil etamiz:

```
int a, b, q;
cin>>b; cin>>a;
```

Kilometr o‘lchov birligidagi B qiymatni metr o‘lchov birligiga o‘tkazish uchun yana b o‘zgaruvchidan foydalanamiz:

```
b=b*1000;
```

C++ da bo‘linma qoldig‘ini hisoblash uchun % amal qo‘llanishini e’tiborga olib masala shartiga javob beruvchi dasturni izohlar bilan quyidagicha to‘liq yozamiz:

```
#include <iostream>
using namespace std;
int main(){
   int a, b, q;
   cout << "To'liq yo'l (kilometrda) B= "; cin >> b;
   cout << "1 kunda asfaltlanadigan yo'l (metrda) A= ";
   cin >> a;
   b = b * 1000;
   q = b % a;
   cout <<"Asfaltlashga qolgan yo'l "<<  q << " metr" << endl;
   return 0;
}
```

Namuna 3. 1 qop undan 300 dona non tayyorlanadi. 1 qop un 45 kilogramm og‘irlikka ega. A kilogramm undan necha dona non tayyorlash mumkinligini chiqaruvchi dastur tuzing.

Yechim. Masaladagi birinchi gapdan 1 qop undan 300 dona non tayyorlanishini, ikkinchi gapdan 1 qop un 45 kilogramm ekanligini bilib, shunday xulosa qilish mumkin: 45 kilogramm=45000 gramm undan 300 dona non tayyorlanadi. Endi 1 dona non uchun ishlatiladigan un miqdorini aniqlash mumkin: 45000/300 gramm. Endi A kilogramm unni grammlarga o‘tkazib 1 dona nonga sarflanadigan un miqdoriga bo‘lish yetarli.

Kiritiladigan un miqdori uchun **a** o‘zgaruvchisini, nonlar soni uchun **soni** o‘zgaruvchisini, 1 qop unga mos non soni uchun **non** o‘zgarmasini, 1 qop unning og‘irlik miqdori uchun **un** o‘zgarmasini butun **int** turida tavsiflaymiz.

Butun o‘zgaruvchilarni bo‘lish uchun C++ da / butun bo‘lish amali qo‘llanishini e’tiborga olib barcha hisoblash ishlarini dasturga yuklaymiz:

```
#include <iostream>
using namespace std;
int main(){
   const int non = 300, un = 45*1000;
   int a, soni;
   cout << "Un miqdori (kilogrammda) A= "; cin >> a;
   a = a * 1000;
   soni = a / (un / non);
   cout << "Tayyorlanadigan non "<< soni << " dona" << endl;
   return 0;
}
```

Namuna 4. Har birining o‘lchami A baytga teng bo‘lgan ko‘p fayllar bor. Hajmi 500 gigabayt bo‘lgan bo‘sh qattiq diskka shu fayllardan nechtasi sig‘ishini va ortib qolgan bo‘sh joy hajmini baytlarda chiqaruvchi dastur tuzing.

Yechim. Ma’lumki, kompyuter xotirasiga ko‘chiriladigan fayl yoki to‘liqligicha yozıladi yoki umuman ko‘chirilmaydi. Demak, masalada gap to‘liq ko‘chiriladigan fayllar soni ustida bormoqda. Fayllar sonini aniqlash uchun butun bo‘lish, ortib qolgan bo‘sh joyni aniqlash esa bo‘lgandagi qoldiqni hisoblash bajarilishini bildiradi.

Masalada qattiq disk (ya’ni vinchester) hajmi gigabaytlarda, fayllar hajmi esa baytlarda berilgan. Shu sababli masalani yechishda qattiq disk hajmini gigabayt o‘lchov birligidan bayt o‘lchov birligiga o‘tkazamiz. O‘tkazishda qattiq disk hajmi uchun tanlangan o‘zgarmasning o‘zidan foydalanamiz. Bunda 500·1024·1024·1024 juda katta son bo‘lishini hisobga olishımız zarur.

Hisoblanadigan sonlar kattaligini e’tiborga olib, kiritiladigan fayl hajmi uchun **a** o‘zgaruvchisini, fayllar soni uchun **soni** o‘zgaruvchisini, qattiq disk hajmi uchun **disk** o‘zgarmasini, ortib qolgan bo‘sh joy hajmi uchun **joy** o‘zgaruvchisini **long long int** turida tavsiflaymiz.

Butun o‘zgaruvchilar uchun C++ da / butun bo‘lish, qoldiq uchun % amali qo‘llanishını e’tiborga olib barcha hisoblashni dasturga quyidagicha yuklaymiz:

```
#include <iostream>
using namespace std;
int main(){
   const long long int disk = 500ll*1024*1024*1024;
   long long int a, soni, joy;
   cout << "Fayllar hajmi (baytda) A= "; cin >> a;
   soni = disk / a;
   joy = disk % a;
   cout << "Disk hajmi "<< disk << " bayt" <<endl;
   cout << "Diskka sig'adigan fayllar "<< soni << " ta" << endl;
   cout << "Diskda qolgan bo'sh joy "<< joy << " bayt" << endl;
   return 0;
}
```

Agar dasturdagi disk o'zgarmasi ifodasida yozilgan birinchi literalga ll (long long) suffiksi yozilmasa, dasturda disk hajmi xato hisoblanadi (masalan, bu holda 0 ga tenglashib qoladi!).

Namuna 5. Uch xonali toq son berilgan. Shu sonning birinchi va uchinchi raqamlari o'rnini almashtirib yangi son hosil qilindi. Ikkala son ko'paytmasini chiqaruvchi dastur tuzing.

Yechim. Masaladagi yangi sonni hosil qilish uchun avval berilgan uch xonali toq sonning raqamlarini ajratib olishimiz kerak. Berilgan son "toq" bo'lishi uning birlik raqami 0 dan farqliligini va natijaviy yangi son ham uch xonali bo'lishini bildiradi.

Belgilash kiritamiz: berilgan son **son**, berilgan sonni yuzlik raqami **uch** (ya'ni uchinchi raqam ma'nosida), o'nlik raqami **ik** (ikkinchi raqam), birlik raqami **bi** (birinchi raqam) bo'lsin. Bu belgilashlar asosida berilgan sonni yoyiq ko'rinishda quyidagicha ifodalash mumkin:

son=uch·100+ik·10+bi.

Yoyiq ko'rinishda yozadigan bo'lsak, **yson** kabi belgilangan masala shartidagi yangi son quyidagicha hisoblanadi:

yson=bi·100+ik·10+uch.

Endi berilgan sonni raqamlarini ajratamiz. Buning uchun bo'lishdagi butun qism va qoldiq xossalarini o'rganamiz. Quyidagi misolni qaraylik:

Son	Matematik ifoda	Qiymati
son=123	birlik=son–[son /10]·10	birlik=123–[123/10]·10=123–[12,3]·10= =123–12·10=123–120=3
son=123	son=[son/10]	son=[123/10]=[12,3]=12
son=12	o'nlik= son–[son/10]·10	o'nlik=12–[12/10]·10=12–[1,2]·10= =12–1·10=12–10=2
son=12	son=[son/10]	son=[12/10]=[1,2]=1
son=1	yuzlik=son–[son/10]·10	yuzlik=1–[1/10]·10=1–[0,1]·10= =1–0·10=1–0=1
son=1	son=[son/10]	son=[1/10]=[0,1]=0

Demak, sonni 10 ga bo'lgandagi qoldiq shu sonning birlik raqamini, butun qism esa shu sonning birlik raqamidan boshqa barcha raqamlarini ajratadi. Masalan, boshqa son=721 uchun, 721=72·10+1 bo'ladi. Birlik raqami 1 va o'nliklar soni 72 ajratib olindi. Xuddi shu mulohazalarni o'nliklar soni 72 ga ham tatbiq etsak, 72=7·10+2 ifodadan 7 va 2 raqamlarini ajratib olamiz.

C++ dasturida butun turlardagi butun bo'lish va qoldiq hisoblash amallari dastur tuzish uchun qulayliklar beradi. Xotirani tejashda bi, ik, uch o'zgaruvchilar o'zida faqatgina raqam qiymatini saqlagani uchun, ya'ni 0..9 diapazonda bo'lganligi uchun **char** turida tavsiflash mumkin. Aytilganlarga asosan quyidagi dasturni tuzamiz:

```
#include <iostream>
using namespace std;
int main(){
   int son, yson; char bi, ik, uch;
   cout << "Uch xonali toq son kiriting: "; cin >> son;
   bi = son % 10;
   son = son / 10;
   ik = son % 10;
   uch = son / 10;
   yson = bi * 100 + ik * 10 + uch;
   cout << "Izlanayotgan son: "<< yson << endl;
   return 0;
}
```

Namuna 6. To‘qqizlik sanoq sistemasidagi 3 xonali son berilgan. Shu sonni o‘nlik sanoq sistemasidagi qiymatini chiqaruvchi dastur tuzing.

Yechim. Informatikadan ma’lumki, har qanday sanoq sistemasidagi sonni o‘nlik sanoq sistemasiga o‘tkazish uchun son raqamlarini asosning aniq darajalariga ko‘paytirib, yig‘indisini hisoblash yetarli. To‘qqizlik sanoq sistemasidagi sonni son9 va shu sonni o‘nlik sanoq sistemasidagi qiymatini son10 kabi yozsak, aytib o‘tilgan mulohazani masalaga mos quyidagi ixcham va yoyiq ko‘rinishlarda yozsa bo‘ladi:

$\mathbf{son9=\overline{abc}_9=a\cdot 9^2+b\cdot 9^1+c\cdot 9^0=son10}$

Bu yerda $\overline{abc}$ yozuvi son9 uchta a, b va c raqamlardan tashkil topganini bildiradi. Ma’lumki, a, b va c raqamlar 0..8 diapazondagi qiymatlarni qabul qiladi.

Bu masalada ham **Namuna 5** masalasining yechimidagi kabi son9 ning raqamlarini ajratib olish zarur ekan. U holda masala yechimi quyidagi ko‘rinishda bo‘ladi.

```
#include <iostream>
using namespace std;
int main(){
   int son9, son10; char a, b, c;
   cout << "Uch xonali to'qqizlik son kiriting: "; cin >> son9;
   c = son9 % 10;
   son9 = son9 / 10;
   b = son9 % 10;
   a = son9 / 10;
   son10 = a * 9 * 9 + b * 9 + c;
   cout << "O'nlikdagi son: "<< son10 << endl;
   return 0;
}
```

Namuna 7. O‘nlik sanoq sistemasidagi A son berilgan (7<A<16). Shu sonni ikkilik sanoq sistemasidagi qiymatini chiqaruvchi dastur tuzing.

Yechim. Informatikadan ma'lumki, 10 lik sanoq sistemasidagi sonni 2 lik sanoq sistemasiga o'tkazish uchun sonni 2 ning darajalari bo'yicha yoyish kerak bo'ladi, ya'ni:

$$A=k_{n-1}\cdot 2^{n-1}+k_{n-2}\cdot 2^{n-2}+\ldots+ k_0\cdot 2^0,$$

bu yerda k_{n-1}, k_{n-2}, ..., k_0 koeffitsiyentlar 0 yoki 1 ga teng. Bu koeffitsiyentlarni aniqlash uchun quyidagi usuldan ham foydalanish mumkin:

Berilgan sonni 2 ga qoldiqli bo'linmalaridan biri 2 dan kichik bo'lguncha 2 ga ketma-ket qoldiqli bo'linadi va qoldiqlar o'ngdan chapga qarab yozib olinadi.

Masala shartiga ko'ra A sonning chegaraviy qiymatlari 8 va 15 bo'lib, bu sonlarni 2 lik sanoq sistemasiga o'tkazsak, 1000 va 1111 ga teng bo'ladi. Demak, berilgan A sonning 2 lik ko'rinishi aniq 4 ta raqamdan iborat, ya'ni n=4 bo'ladi. Ikkilik sanoq sistemasidagi sonning raqamlari uchun k0, k1, k2 va k3 o'zgaruvchilarni **char** turida tavsiflash yetarli. Raqamlarni aniqlash uchun esa 2 ga butun bo'lish va qoldiq hisoblash amallaridan foydalanish qulay. Yuqoridagi mulohazalar asosida quyidagi dasturni tuzamiz.

```
#include <iostream>
using namespace std;
int main(){
   int a, son2; char k0, k1, k2, k3;
   cout << "A sonni kiriting: "; cin >> a;
   k0 = a % 2; a = a / 2;
   k1 = a % 2; a = a / 2;
   k2 = a % 2; a = a / 2;
   k3 = a % 2;
   son2 = k3 * 1000 + k2 * 100 + k1 * 10 + k0;
   cout << "Ikkilikdagi son: "<< son2 << endl;
   return 0;
}
```

Namuna 8. A va B butun turdagi o'zgaruvchilar qiymatini almashtirib ekranga chiqaruvchi dastur tuzing, ya'ni: A=3 va B=4 kiritilsa, u holda ekranga A=4 va B=3 kabi chiqarilishi kerak.

Yechim. Masalani A=3, B=4 holda ko'raylik.

Ko'pincha "sodda" dasturchi masala shartidagi "o'zgaruvchilar qiymatini almashtirib" qismiga e'tibor bermay quyidagicha yechim taklif etadi:

```
int a = 3, b = 4;
cout << "A="<< b << " B="<< a << endl;
```

Holbuki, "o'zgaruvchilar qiymatini almashtirib" deganda kompyuter xotirasidagi qiymatlarni almashtirish ko'zda tutilgan bo'ladi, chiqarishdagi "aldov"ni emas.

O'ylab qaralsa, masala yechimi baribir soddadek ko'rinadi:

```
int a=3, b=4;
```

```
a = b; b = a;
cout << "A="<< b << " B="<< a << endl;
```

Dastur ishlatib ko‘rilganda quyidagicha natija beradi:

```
A=4; B=4
```

Natija xato-ku, 3 qayerga ketdi?! Xatolikni aniqlash uchun dasturni tahlil qilamiz:

`a = 3; b = 4;` bu o‘zlashtirish operatorlari bajarilgach xotirada **a** o‘zgaruvchi uchun ajratilgan joyga 3, **b** o‘zgaruvchi uchun ajratilgan joyga 4 yoziladi. Keyingi qadamda `a = b;` o‘zlashtirish operatori bajarilganda **a** o‘zgaruvchi uchun ajratilgan xotiradagi joy tozalanib, u joyga **b** o‘zgaruvchining 4 qiymati yoziladi. Demak, xotirada **a=4** va **b=4** hosil bo‘ladi, ya’ni 3 qiymat esa hech qayerda yo‘q, ya’ni xotiradan o‘chib ketdi. Endi `b = a;` o‘zlashtirish operatori bajarilgach, **b** o‘zgaruvchi uchun ajratilgan xotiradagi joy tozalanib, u joyga **a** o‘zgaruvchining qiymati 4 yoziladi. Demak, dastur xato tuzilgan!

Masalani hayotiy misol kabi qaraylik: ikkita bola (3 va 4) ikkita (**a** va **b**) stulda o‘tirishibdi. Bolalar stullarda almashib o‘tirishlari kerak. Lekin, bir vaqtda ikkala bolaning o‘rnidan turishi mumkin emas. O‘ylab ko‘rib, shunday fikr bildirish tabiiy: bolalardan biri (masalan, **a** stuldagi 3) biror bo‘sh stulga (masalan, **d** ga) o‘tadi, keyin “bo‘shagan” stulga (ya’ni **a** ga) ikkinchi bola (endi bu **b** stuldagi 4 bo‘ladi) o‘tadi, oxirida “bo‘shagan” stulga (ya’ni **b** ga) birinchi bola (endi bu **d** stuldagi 3) o‘tadi.

Demak, masalaga mos dasturda qo‘shimcha biror **d** o‘zgaruvchidan foydalanish kerak ekan.

```
#include <iostream>
using namespace std;
int main(){
   int a, b, d;
   cout << "A sonni kiriting: "; cin >> a;
   cout << "B sonni kiriting: "; cin >> b;
   d = a;
   a = b;
   b = d;
   cout << "A="<< a << " B="<< b << endl;
   return 0;
}
```

Namuna 9. A, B va C mulohazalar qiymati berilgan. “EMAS A VA (B YOKI EMAS C)” murakkab mulohaza natijasini chiqaruvchi dastur tuzing.

Yechim. Ma’lumki, mulohazalar qiymat **rost** yoki **yolg‘on** bo‘ladi. C++ tilida klaviaturadan **rost** o‘rniga 1 va **yolg‘on** o‘rniga 0 qiymatlar kiritiladi. Klaviaturadan **true** yoki **false** qiymatlar kiritilishi hamda natija ham **true** yoki **false** kabi ko‘rinishda chiqishi uchun **boolalpha** manipulyatoridan foydalanish zarur.

Hisoblash talab qilingan mulohazani C++ tilidagi mantiqiy amallar orqali 2 xil usulda (ya'ni and yoki &&, or yoki ||, not yoki !) yozib quyidagi dasturga ega bo'lamiz:

```
#include <iostream>
using namespace std;
int main(){
   bool a, b, c, natija;
   cout << "A mulohaza qiymatini kiriting: "; cin >> a;
   cout << "B mulohaza qiymatini kiriting: "; cin >> b;
   cout << "C mulohaza qiymatini kiriting: "; cin >> c;
   natija = not a and (b or not c);
   cout << "Murakkab mulohaza qiymati = "<< natija << endl;
   natija = !a && (b || !c);
   cout << "Murakkab mulohaza qiymati = "<< natija << endl;
   return 0;
}
```

Namuna 10. A son berilgan. "A musbat va besh xonali son" mulohaza natijasini chiqaruvchi dastur tuzing.

Yechim. Ma'lumki, agar A soni musbat bo'lsa, A>0 shart bajariladi va besh xonali son [10000; 99999] kesmaga tegishli bo'ladi. Kesmaga tegishlilik shartini 10000≤A≤99999 kabi qo'sh tengsizlik ko'rinishida yozish mumkin. Lekin dasturlash tilida qo'sh tengsizlik yozib bo'lmaydi, shuning uchun, odatda, qo'sh tengsizlik o'rniga quyidagicha yoziladi: 10000≤A VA A≤99999. Masala shartidagi mulohazani hosil qilingan tengsizliklar va mantiqiy amallar yordamida quyidagicha yozamiz: A>0 VA 10000≤A VA A≤99999.

Hisoblash talab qilingan mulohazani C++ tilidagi mantiqiy amallar orqali 2 xil usulda yozib, quyidagi dasturga ega bo'lamiz:

```
#include <iostream>
using namespace std;
int main(){
   int a; bool natija;
   cout << "A sonni kiriting: "; cin >> a;
   natija = a > 0 and 10000 <= a and a <= 99999;
   cout << "Mulohaza qiymati = "<< natija << endl;
   natija = a > 0 && 10000 <= a && a <= 99999;
   cout << boolalpha << "Mulohaza qiymati = "<< natija << endl;
   return 0;
}
```

Namuna 11. "Berilgan uchta sonning har biri musbat" mulohaza natijasini chiqaruvchi dastur tuzing.

Yechim. Ma'lumki, agar A soni musbat bo'lsa, A>0 shart bajariladi. Demak, berilgan uchala son uchun shu tengsizlikni tekshirish kerak ekan. Barcha tengsizlikning birdaniga bajarilishi esa VA (and yoki &&) mantiqiy amali orqali birlashtirib ifodalanadi.

```
#include <iostream>
using namespace std;
int main(){
   int a, b, c; bool natija;
   cout << "A sonni kiriting: "; cin >> a;
   cout << "B sonni kiriting: "; cin >> b;
   cout << "C sonni kiriting: "; cin >> c;
   natija = a > 0 and b > 0 and c > 0;
   cout << boolalpha << "Mulohaza qiymati = "<< natija << endl;
   natija = a > 0 && b > 0 && c > 0;
   cout << "Mulohaza qiymati = "<< natija << endl;
   return 0;
}
```

Namuna 12. "Berilgan uch xonali son raqamlarining yozilishi tartibida kamaymaydigan ketma-ketlik tashkil etadi" mulohaza natijasini chiqaruvchi dastur tuzing.

Yechim. Berilgan uch xonali sonning raqamlari kamaymaydigan ketma-ketlik tashkil etishini bilish uchun avval shu son raqamlarini ajratib olishimiz kerak. Bu ishni Namuna 5 va Namuna 6 da bajargan edik. Sonning raqamlarini qulaylik uchun r1 (birlik), r10 (o'nlik) va r100 (yuzlik) kabi belgilaymiz. Masala shartiga ko'ra $r100 \le r10 \le r1$ shartni tekshirishimiz yetarli.

```
#include <iostream>
using namespace std;
int main(){
   int son;
   char r100, r10, r1;
   bool mulohaza;
   cout << "Uch xonali son kiriting: "; cin >> son;
   r1 = son % 10;
   son = son / 10;
   r10 = son % 10;
   r100 = son / 10;
   mulohaza = r100 <= r10 && r10 <= r1;
   cout << "Mulohaza qiymati "<< mulohaza << endl;
   return 0;
}
```

Namuna 13. Turli qiymatli A, B va C sonlar berilgan. Shu sonlarni taqqoslashga oid bajariladigan qo'sh tengsizlikni chiqaruvchi dastur tuzing.

Yechim. Masala shartiga ko'ra, masalan, A=7, B=3, C=9 bo'lsa, 3<7<9 qo'sh tengsizlikni ekranga chiqarish talab etilmoqda (sonlar turli qiymatli bo'lgani uchun tenglik bajarilmaydi).

Buning uchun avval sonlardan eng kichigini aniqlab olish zarur. Bu son qolgan ikki sondan albatta kichik bo'ladi. Demak, (A<B VA A<C) yoki (B<A VA B<C) yoki (C<A VA C<B) mulohazadan faqat bittasi rost. C++ tilida mantiqiy mulohaza qiymati 0 yoki 1 ekanligini e'tiborga olib eng kichik son uchun quyidagi ifodani yozamiz:

son1 = (A<B VA A<C)*A + (B<A VA B<C)*B + (C<A VA C<B)*C.

Xuddi shunday ifodani eng katta son uchun ham yoza olamiz:

son3 = (B<A VA C<A)*A + (A<B VA C<B)*B + (A<C VA B<C)*C;

O'rtadagi son quyidagi tengsizliklarning biridan aniqlanadi:

(B<A VA A<C) YOKI (C<A VA A<B) murakkab shart bajarilsa A,

(A<B VA B<C) YOKI (C<B VA B<A) murakkab shart bajarilsa B,

(A<C VA C<B) YOKI (B<C VA C<A) murakkab shart bajarilsa C.

Shu mulohazalar asosida quyidagi dasturni tuzamiz:

```
#include <iostream>
using namespace std;
int main(){
   int a, b, c, son1, son2, son3;
   cout << "A sonni kiriting: "; cin >> a;
   cout << "B sonni kiriting: "; cin >> b;
   cout << "C sonni kiriting: "; cin >> c;
   son1 = (a<b && a<c)*a + (b<a && b<c)*b + (c<a && c<b) * c;
   son3 = (b<a && c<a)*a + (a<b && c<b)*b + (a<c && b<c) * c;
   son2 = ((b<a && a<c)||(c<a && a<b))*a;
   son2 = son2 + ((a<b && b<c)||(c<b && b<a))*b;
   son2 = son2 + ((a<c && c<b)||(b<c && c<a))*c;
   cout << son1 << '<' << son2 << '<' << son3 << endl;
   return 0;
}
```

Namuna 14. A va B sonlar berilgan. "Ax+B=0 tenglama yechimga ega emas" mulohaza natijasini chiqaruvchi dastur tuzing.

Yechim. Matematika fanidan ma'lumki, Ax+B=0 tenglama yechimi haqida quyidagilarni aytish mumkin:

1) A≠0 bo'lsa, u holda Ax+B=0 tenglama yagona $x = -\frac{B}{A}$ yechimga ega;

2) A=0 va B≠0 bo'lsa, u holda Ax+B=0 tenglama yechimga ega emas (chunki 0 ga bo'lish mumkin emas);

3) A=0 va B=0 bo'lsa, u holda Ax+B=0 tenglama cheksiz ko'p yechimga ega (chunki bu holda 0·x+0=0 tenglama uchun har qanday $x \in R$ yechim bo'la oladi).

Demak, masala shartiga mos dastur tuzish uchun 2-holdagi shartlarni tekshirish yetarli ekan:

```
#include <iostream>
using namespace std;
int main(){
   int a, b; bool mulohaza;
   cout << "A sonni kiriting: "; cin >> a;
   cout << "B sonni kiriting: "; cin >> b;
   mulohaza = a == 0 && b != 0;
   cout << boolalpha << "Mulohaza qiymati "<< mulohaza << endl;
   return 0;
}
```

Namuna 15. A, B va C sonlar berilgan (A noldan farqli). "y= $A \cdot x^2+B \cdot x+C$ funksiya grafigi Ox o'qidan pastda yotadi" mulohaza natijasini chiqaruvchi dastur tuzing.

Yechim. Matematikadan ma'lumki, funksiya grafigi, ya'ni parabola, OY o'qini albatta kesib o'tadi. Parabola holatini $A \cdot x^2+B \cdot x+C=0$ tenglamani yechish bilan bog'laymiz:

1) $D=B^2 - 4 \cdot A \cdot C<0$ bo'lsa tenglama yechimga ega emas, ya'ni parabola grafigi OX o'qi bilan umumiy nuqtaga ega bo'lmaydi. U holda parabola grafigi OX o'qidan $A>0$ bo'lganda to'liq yuqorida joylashadi, aks holda, ya'ni $A<0$ bo'lganda, to'liq quyida joylashadi;

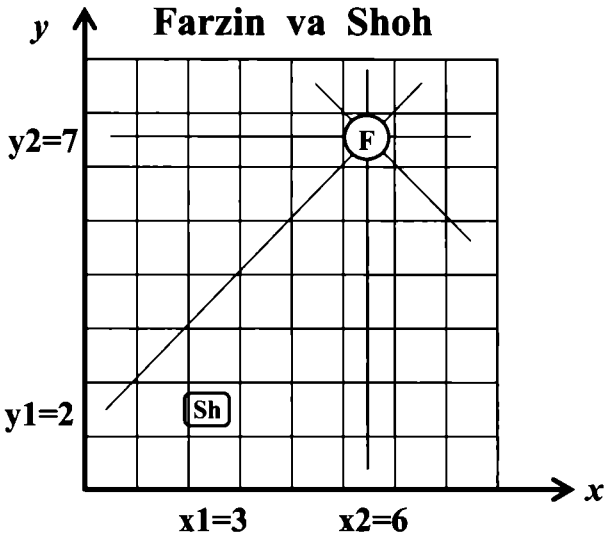
2) $D=B^2 - 4 \cdot A \cdot C=0$ bo'lsa, tenglama yagona yechimga ega. Bu holda parabola grafigi Ox o'qiga 1 ta nuqtada tegib o'tadi;

3) $D=B^2 - 4 \cdot A \cdot C>0$ bo'lsa, tenglama ikkita yechimga ega, ya'ni parabola grafigi OX o'qini 2 ta nuqtada kesib o'tadi.

Demak, masala shartiga mos dastur tuzish uchun 3-holdagi shartlarni tekshirish yetarli ekan:

```
#include <iostream>
using namespace std;
int main(){
   int a, b, c, d;
   bool mulohaza;
   cout << "A sonni kiriting: "; cin >> a;
   cout << "B sonni kiriting: "; cin >> b;
   cout << "C sonni kiriting: "; cin >> c;
   d = b * b - 4 * a * c;
   mulohaza = a < 0 && d < 0;
   cout << "Mulohaza qiymati "<< mulohaza << endl;
   return 0;
}
```

Namuna 16. 1..8 diapazondagi x1, y1, x2 va y2 sonlar berilgan. Shaxmat doskasining (x1; y1) katagida Shoh va (x2; y2) katagida Farzin turgan bo'lsin. "Shoh zarba ostida" mulohaza natijasini chiqaruvchi dastur tuzing.



Yechim: Masala shartini shaxmat figuralari Shoh va Farzin uchun tahlil qilish (yuqoridagi) rasm orqali qulay bo'ladi. Berilgan shaxmat doskasi 1..8×1..8 jadval kabidir.

Rasmda **Sh** belgili to'rtburchak Shoh turgan (x1, y1) katakni, **F** belgili aylana Farzin turgan (x2, y2) katakni bildiradi. Farzin turgan katakdan chiqqan chiziqlar Farzin zarba beradigan kataklarni aniqlaydi. Rasm orqali Farzin zarbasiga uchraydigan kataklarni aniqlab olamiz.

Rasmdan quyidagilarni ko'rish mumkin:

1-hol. Vertikal chiziqqa asosan birinchi koordinatasi x2 ga teng ikkinchi koordinatasi esa ixtiyoriy bo'lgan kataklar Farzin zarbasiga uchraydi. Bu holda Shoh Farzinning zarbasiga uchrashi uchun x1–x2=0 bo'lishi zarur va yetarli.

2-hol. Gorizontal chiziqqa asosan ikkinchi koordinatasi y2 ga teng birinchi koordinatasi ixtiyoriy bo'lgan kataklar Farzin zarbasiga uchraydi. Bu holda Shoh Farzinning zarbasiga uchrashi uchun y1–y2=0 bo'lishi zarur va yetarli.

3-hol. Bu hol nisbatan murakkabroq. Rasmda Farzin koordinatasi (6; 7) kabi berilgan.

a) O'suvchi (/ yo'nalishdagi) og'ma chiziqqa ko'ra (7; 8), (5; 6), (4; 5), (3; 4), (2; 3), (1; 2) koordinatali kataklar Farzin zarbasiga uchraydi. Farzin zarbasiga uchragan barcha kataklar koordinatasi bilan Farzin turgan katak orasida bog'liqlik borligini ko'rish mumkin: 6–7=7–8, 6–5=7–6, 6–4=7–5, 6–3=7–4, 6–2=7–3, 6–1=7–6. Bundan shunday xulosaga kelamiz: bu holda Shoh Farzin zarbasiga uchrashi uchun x2–x1=y2–y1 shart bajarilishi zarur va yetarli.

b) Kamayuvchi (\ yo‘nalishdagi) og‘ma chiziqqa ko‘ra (5; 8), (7; 6), (8; 5) koordinatali kataklar Farzin zarbasiga uchraydi. Bu holda ham Farzin zarbasiga uchragan barcha kataklar koordinatasi bilan Farzin turgan katak orasida bog‘liqlik bor: 6–5= –(7–8), 6–7= –(7–6), 6–8= –(7–5). Bundan shunday xulosaga kelamiz: bu holda Shoh Farzinning zarbasiga uchrashi uchun x2–x1= –(y2–y1) shart bajarilishi zarur va yetarli.

Demak, Shoh Farzin zarbasiga uchrashi uchun quyidagi shartlardan birining bajarilishi zarur va yetarli:

x1–x2=0 yoki y1–y2=0 yoki x2–x1=y2–y1 yoki x2–x1= –(y2–y1).

Berilgan x1, y1, x2 va y2 koordinatalar 1..8 diapazonda ekanligini hisobga olib, **char** turda tavsiflash mumkin. Aniqlangan shartlar asosida mantiqiy ifoda tuzilgan dasturni yozish mumkin.

```
#include <iostream>
using namespace std;
int main(){
   char x1, y1, x2, y2;
   bool mulohaza;
   cout << "Shohning koordinatalarini kiriting:" << endl;
   cout << "x1= "; cin >> x1;
   cout << "y1= "; cin >> y1;
   cout << "Farzinning koordinatalarini kiriting:" << endl;
   cout << "x2= "; cin >> x2;
   cout << "y2= "; cin >> y2;
   mulohaza =x1-x2=0||y1-y2=0||x2-x1=y2-y1||x2-x1=-(y2-y1);
   cout << "Mulohaza qiymati "<< mulohaza << endl;
   return 0;
}
```

Namuna 17. Uchburchakning a, b va c tomonlari berilgan. “Uchburchak o‘tkir burchakli” mulohaza natijasini chiqaruvchi dastur tuzing.

Yechim: Masalani yechish uchun geometriyaga murojaat qilamiz, ya’ni kosinuslar teoremasiga ko‘ra:

$a^2=b^2+c^2-2\cdot b\cdot c\cdot cosA$, $b^2=a^2+c^2-2\cdot a\cdot c\cdot cosB$, $c^2=a^2+b^2-2\cdot a\cdot b\cdot cosC$,

bu yerda A, B va C mos tomonlar qarshisidagi burchaklar. Bu formulalarni tahlil qilib quyidagi xulosalarni hosil qilamiz.

1-hol. Agar uchburchak to‘g‘ri burchakli bo‘lsa, u holda A yoki B yoki C burchaklardan birortasi 90^0 ga teng. Bu holda, $cos90^0=0$ bo‘lgani uchun quyidagi shartlardan biri o‘rinli bo‘ladi:

$$\mathbf{a^2+b^2=c^2} \text{ yoki } \mathbf{a^2+c^2=b^2} \text{ yoki } \mathbf{b^2+c^2=a^2}$$

Demak, yuqoridagi tengliklardan bittasi o'rinli bo'lishi uchburchakning to'g'ri burchakli bo'lishiga teng kuchli ekan.

2-hol. Agar uchburchak o'tkir burchakli bo'lsa, u holda A yoki B yoki C burchaklarning barchasi 90^0 dan kichik. Ma'lumki, agar $\alpha < 90^0$ bo'lsa, u holda $\cos\alpha > 0$. Shunga ko'ra o'tkir burchakli uchburchakning barcha tomonlari uchun quyidagi tengsizliklar o'rinli:

$a^2=b^2+c^2-2\cdot b\cdot c\cdot\cos A < b^2+c^2$, $b^2=a^2+c^2-2\cdot a\cdot c\cdot\cos B < a^2+c^2$, $c^2=a^2+b^2-2\cdot a\cdot b\cdot\cos C < a^2+b^2$.

Demak, quyidagi tengsizliklarning bir vaqtda o'rinli bo'lishi uchburchakning o'tkir burchakli bo'lishiga teng kuchli ekan:

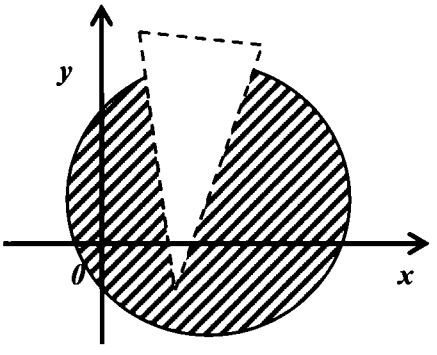
$$\mathbf{a^2 < b^2+c^2} \text{ va } \mathbf{b^2 < a^2+c^2} \text{ va } \mathbf{c^2 < a^2+b^2.}$$

3-hol. Agar uchburchak o'tmas burchakli bo'lsa, u holda A yoki B yoki C burchaklarning birortasi 90^0 dan katta. Uchburchakning ichki burchaklari yig'indisi 180^0 ga teng bo'lganligidan qolgan ikki burchak yig'indisi 90^0 dan kichik. U holda 2-holdagi tengsizliklardan bittasi o'rinli bo'lmaydi.

Mulohazalar asosida olingan shartga ko'ra dastur tuzamiz:

```
#include <iostream>
using namespace std;
int main(){
   int a, b, c; bool mulohaza;
   cout << "A tomonni kiriting: "; cin >> a;
   cout << "B tomonni kiriting: "; cin >> b;
   cout << "C tomonni kiriting: "; cin >> c;
   mulohaza = a*a < b*b+c*c && b*b < a*a+c*c && c*c < a*a+b*b;
   cout << boolalpha << "Mulohaza qiymati "<< mulohaza << endl;
   return 0;
}
```

Namuna 18. Oxy koordinata tekisligida markazi (1,5; 1) nuqtada bo'lgan 2 radiusli doira va uchlari (0,5; 4), (1; –0,5) va (2; 3,5) nuqtalarda bo'lgan uchburchak chizilgan. Chizmada shtrixlangan doira sohasiga uchburchak chegarasi bilan kirmaydi. "(x; y) nuqta shtrixlangan sohaga tegishli" mulohaza natijasini chiqaruvchi dastur tuzing.



Yechim: Masalani yechish uchun geometriyaga murojaat qilamiz. Bu bobda faqat butun sonlar bilan ish ko'rilganligi uchun sonlarni butun ko'rinishga o'tkazib olamiz.

Doiraning tenglamasi quyidagicha bo'ladi:

$$(x-1,5)^2+(y-1)^2\le 2^2$$

yoki tengsizlikning ikki tomonini 2^2 ga ko'paytirsak:

$$(2x-3)^2+(2y-2)^2\leq 4^2$$

Uchburchakning chap tomoni (ya'ni to'g'ri chiziq kabi qaraymiz) tenglamasi:

$$\frac{x-0,5}{1-0,5}=\frac{y-4}{-0,5-4} \text{ yoki } \frac{x-0,5}{0,5}+\frac{y-4}{4,5}=0 \text{ yoki } 9x+y-8,5=0 \text{ yoki } 18x+2y-17=0$$

Uchburchakning chap tomoni (ya'ni, to'g'ri chiziq kabi qaraymiz) tenglamasi:

$$\frac{x-1}{2-1}=\frac{y+0,5}{3,5+0,5} \text{ yoki } \frac{x-0,5}{1}-\frac{y+0,5}{4}=0 \text{ yoki } 4x-y-2,5=0 \text{ yoki } 8x-2y-5=0$$

Uchburchakning chap tomonidan o'tgan to'g'ri chiziqdan chapda yotgan nuqtalar quyidagi tengsizlikka bo'ysunadi (chunki koordinata boshi tengsizlikni qanoatlantiradi: $0+0-5<0$):

$$18x+2y-17<0.$$

Uchburchakning o'ng tomonidan o'tgan to'g'ri chiziqdan o'ngda yotgan nuqtalar quyidagi tengsizlikka bo'ysunadi (chunki koordinata boshi tengsizlikni qanoatlantirmaydi: $0+0-5<0$):

$$8x-2y-5>0.$$

Uchburchakning chap tomonidan o'tgan to'g'ri chiziqdan chapda yotgan doira nuqtalarini quyidagi ifoda orqali aniqlaymiz:

$$((2x-3)(2x-3)+(2y-2)(2y-2)\leq 16) \text{ VA } (18x+2y-17<0).$$

Uchburchakning o'ng tomonidan o'tgan to'g'ri chiziqdan o'ngda yotgan doira nuqtalarini quyidagi ifoda orqali aniqlaymiz:

$$((2x-3)(2x-3)+(2y-2)(2y-2)\leq 16) \text{ VA } (8x-2y-5>0).$$

Bizga kerak bo'lgan sohani aniqlash uchun quyidagi ifodani tuzamiz:

$$((2x-3)(2x-3)+(2y-2)(2y-2)\leq 16) \text{ VA } (18x+2y-17<0) \text{ YOKI}$$
$$((2x-3)(2x-3)+(2y-2)(2y-2)\leq 16) \text{ VA } (8x-2y-5>0).$$

Oxirgi ifodani ketma-ket hisoblash orqali tashkil etilgan quyidagi dasturni yozamiz:

```
#include <iostream>
using namespace std;
int main(){
   int x, y;
   bool mulohaza, mulohaza1, mulohaza2;
   cout << "x koordinatani kiriting: "; cin >> a;
   cout << "y koordinatani kiriting: "; cin >> b;
   mulohaza1 = (2*x-3)*(2*x-3)+(2*y-2)*(2*y-2)<16;
   mulohaza1 = mulohaza1 && (18*x+2*y-17<0);
   mulohaza2 = (2*x-3)*(2*x-3)+(2*y-2)*(2*y-2)<16;
   mulohaza2 = mulohaza2 && (8*x-2*y-5>0);
   mulohaza = mulohaza1 || mulohaza2;
   cout << "Mulohaza qiymati "<< mulohaza << endl;
   return 0;
}
```

2-§. BUTUN TURGA OID MASALALAR

Bu paragrafdagi masalalarda kiritilayotgan va chiqarilayotgan barcha qiymatlar butun sonlardir. Xonalar (razryadlar) soni (ikki xonali, uch xonali, ...) berilgan masalalardagi barcha sonlar musbat hisoblanib, yuqori razryad (masalan, uch xonali 204 da 2 yuqori) 0 dan farqli. Sonlar diapazoni aniq bo'lgan yoki chegarasi berilgan masalalarda xotirani tejaydigan butun turni tanlash talab etiladi, aks holda, ya'ni chegara berilmagan holda sonlarni **int** turda tavsiflash mumkin.

= A =

Butun 1. Musbat A va B sonlar berilgan (A>B). A sonni B songa bo'lgandagi butun qism va qoldiqni chiqaruvchi dastur tuzing (yo'llanma: butun bo'lish va qoldiq hisoblash).

Butun 2. A kilometrni metr, detsimetr, santimetr va millimetr o'lchov birliklari ko'rinishida chiqaruvchi dastur tuzing (yo'llanma: birliklar, butun bo'lish).

Butun 3. A santimetr masofa berilgan. Shu masofa necha to'liq metr, metrdan ortib qolgani necha santimetr bo'lishini chiqaruvchi dastur tuzing (yo'llanma: birliklar, butun bo'lish va qoldiq hisoblash).

Butun 4. Uzunligi A santimetr bo'lgan taxta berilgan. Narvonning pillapoyalarini yasash uchun B santimetrli taxta bo'laklari kerak. Berilgan taxtadan yasash mumkin bo'lgan pillapoyalar sonini chiqaruvchi dastur tuzing (yo'llanma: butun bo'lish).

Butun 5. A kilogramm shokolad berilgan. Shokoladlar necha to'liq tonna bo'lishini va to'liq tonnadan ortib qolgan kilogrammni chiqaruvchi dastur tuzing (yo'llanma: birliklar, butun bo'lish va qoldiq hisoblash).

Butun 6. 1 dona shokolad tayyorlash uchun B gramm kakao ishlatiladi. A kilogramm kakaodan tayyorlash mumkin bo'lgan shokoladlar sonini chiqaruvchi dastur tuzing (yo'llanma: birliklar, butun bo'lish).

Butun 7. A sentnerni necha to'liq tonna bo'lishini, tonnadan ortib qolgani necha kilogramm bo'lishini chiqaruvchi dastur tuzing (yo'llanma: birliklar, butun bo'lish va qoldiq hisoblash).

Butun 8. O'lchamlari A baytga teng bo'lgan N ta fayl berilgan. Barcha fayllar to'liq necha kilobayt bo'lishini chiqaruvchi dastur tuzing (yo'llanma: birliklar, butun bo'lish).

Butun 9. O'lchamlari A kilobaytga teng bo'lgan juda ko'p fayllar bor. Hajmi B megabayt bo'lgan CD diskka shu fayllardan nechtasi to'liq sig'ishini chiqaruvchi dastur tuzing (yo'llanma: birliklar, butun bo'lish).

Butun 10. O'lchamlari A baytga teng bo'lgan juda ko'p fayllar bor. Hajmi B gigabayt bo'lgan fleshkaga shu fayllardan nechtasi sig'ishini va ortib qolgan bo'sh joyni baytlarda chiqaruvchi dastur tuzing (yo'llanma: birliklar, butun bo'lish va qoldiq hisoblash).

Butun 11. A va B butun sonlar berilgan (B<A<125). Bitta qo'lqop juftini tikish uchun B santimetr material kerak. A metrli materialdan nechta qo'lqop juftini tikish mumkinligini va ortib qolgan materialni millimetrlarda chiqaruvchi dastur tuzing (yo'llanma: birliklar, butun bo'lish va qoldiq hisoblash).

Butun 12. A va B butun sonlar berilgan (B<A<50000). A uzunlikdagi kesmada nechta B uzunlikdagi kesma borligini va ortib qolgan kesma uzunligini chiqaruvchi dastur tuzing (yo'llanma: butun bo'lish va qoldiq hisoblash).

Butun 13. Ikki xonali son berilgan. Shu sonni birinchi (o'nlik) va ikkinchi (birlik) raqamlarini bitta satrda orachiq bilan ajratib chiqaruvchi dastur tuzing (yo'llanma: butun bo'lish va qoldiq hisoblash).

Butun 14. Ikki xonali son berilgan. Shu son raqamlarining yig'indisini va ko'paytmasini alohida satrlarda chiqaruvchi dastur tuzing (yo'llanma: butun bo'lish va qoldiq hisoblash).

Butun 15. Ikki xonali son berilgan. Shu son raqamlarining o'rnini almashtirib hosil qilingan sonni chiqaruvchi dastur tuzing (yo'llanma: butun bo'lish va qoldiq hisoblash).

Butun 16. Sutka boshidan N sekund o'tdi. Sutka boshidan necha to'liq minut o'tganini chiqaruvchi dastur tuzing (yo'llanma: butun bo'lish).

Butun 17. Sutka boshidan N sekund o'tdi. Sutka boshidan necha to'liq soat o'tganini chiqaruvchi dastur tuzing (yo'llanma: butun bo'lish).

Butun 18. Sutka boshidan N sekund o'tdi. Oxirgi soatdan keyin necha to'liq sekund qolganini chiqaruvchi dastur tuzing (yo'llanma: butun bo'lish va qoldiq hisoblash).

Butun 19. A va B son berilgan (-100<A, B<100). M=21·A+19·B ifoda qiymatini chiqaruvchi dastur tuzing.

Butun 20. To'g'ri to'rtburchakning tomonlari A va B bo'lsa, uning perimetri va yuzini chiqaruvchi dastur tuzing.

Butun 21. Tomoni A santimetr, B detsimetr va R metr bo'lgan uchburchakning perimetri qiymatini detsimetr o'lchov birligida chiqaruvchi dastur tuzing.

Butun 22. To'g'ri chiziqda koordinatalari A va B bo'lgan (A<B) nuqtalar berilgan. A va B nuqtalar orasidagi butun koordinatali nuqtalar sonini chiqaruvchi dastur tuzing (sanoq qoidasi).

Butun 23. Tezligi V km/soat bo'lgan samolyot T soatda uchib o'tgan S yo'lni metrlarda chiqaruvchi dastur tuzing (yo'llanma: birliklar).

Butun 24. Uyning bo'yi A metr, eni B santimetr, balandligi esa R detsimetr bo'lsa, uning hajmini santimetr o'lchov birligida chiqaruvchi dastur tuzing (yo'llanma: birliklar).

Butun 25. Xaridorning A so'm puli bo'lib, u kilogrami B so'm bo'lgan konfet sotib olmoqchi. U necha kilogramm konfet olishi mumkinligini chiqaruvchi dastur tuzing (yo'llanma: butun bo'lish).

= B =

Butun 26. Uch xonali son berilgan. Shu sonning barcha raqamlarini alohida satrlarda chiqaruvchi dastur tuzing (yo'llanma: butun bo'lish va qoldiq hisoblash).

Butun 27. Uch xonali son berilgan. Shu sonning avval ikkinchi (o'nlik) raqamini, keyin birinchi (yuzlik) raqamini, so'ng oxirgi (birlik) raqamini ajratib chiqaruvchi dastur tuzing (yo'llanma: butun bo'lish va qoldiq hisoblash).

Butun 28. Uch xonali son berilgan. Shu sonning barcha raqamlari yig'indisini va ko'paytmasini alohida satrlarda chiqaruvchi dastur tuzing (yo'llanma: butun bo'lish va qoldiq hisoblash).

Butun 29. Uch xonali toq son berilgan. Shu sonni o'ngdan chapga qarab o'qilgandagi sonni chiqaruvchi dastur tuzing (yo'llanma: butun bo'lish va qoldiq hisoblash, soning yoyiq ko'rinishi).

Butun 30. Uch xonali son berilgan. Shu sonning ikkinchi (o'nlik) raqami o'chirilib, shu raqam sonning oxiriga yozildi. Berilgan va hosil bo'lgan sonlarni ajratib chiqaruvchi dastur tuzing (yo'llanma: butun bo'lish va qoldiq hisoblash, sonning yoyiq ko'rinishi).

Butun 31. Uch xonali toq son berilgan. Shu sonning birlik raqami o'chirilib, shu raqam sonning boshiga yozildi. Berilgan va hosil bo'lgan yangi sonlarni ajratib chiqaruvchi dastur tuzing (yo'llanma: butun bo'lish va qoldiq hisoblash, sonning yoyiq ko'rinishi).

Butun 32. Uch xonali ikkinchi (o'nlik) raqami 0 dan farqli bo'lgan son berilgan. Shu sonning birinchi (yuzlik) va ikkinchi raqamlari o'rnini almashtirib, yangi son hosil qilindi. Berilgan va hosil bo'lgan sonlarni ajratib chiqaruvchi dastur tuzing (yo'llanma: butun bo'lish va qoldiq hisoblash, sonning yoyiq ko'rinishi).

Butun 33. Uch xonali ikkinchi (o'nlik) raqami 0 dan farqli bo'lgan toq son berilgan. Shu sonning oxirgi (birlik) va ikkinchi raqamlari o'rnini almashtirib, yangi son hosil qilindi. Berilgan va hosil bo'lgan yangi sonlarni ajratib chiqaruvchi dastur tuzing (yo'llanma: butun bo'lish va qoldiq hisoblash, sonning yoyiq ko'rinishi).

Butun 34. Uch xonali x sonidan birlik raqami ayirildi. Hosil bo'lgan sonni 10 ga bo'lgandagi bo'linmasining chap tomoniga x sonining birlik raqami yozildi. Natijada 237 soni hosil bo'ldi. x sonini aniqlab chiqaruvchi dastur tuzing (yo'llanma: butun bo'lish va qoldiq hisoblash, sonning yoyiq ko'rinishi).

Butun 35. Uch xonali x sonining yuzlik raqami o'chirildi. Hosil bo'lgan sonni 10 ga ko'paytirib, hosil bo'lgan natijaga x sonining yuzlik raqami qo'shildi. Natijada 564 soni hosil bo'ldi. x sonini aniqlab chiqaruvchi dastur tuzing (yo'llanma: butun bo'lish va qoldiq hisoblash, sonning yoyiq ko'rinishi).

Butun 36. A soni berilgan (A>999). Butun bo'lish va qoldiq hisoblash amallarini bir martadan qo'llab, yuzlik raqamini chiqaruvchi dastur tuzing.

Butun 37. A soni berilgan (A>999). Butun bo‘lish va qoldiq hisoblash amallarini bir martadan qo‘llab minglik raqamini chiqaruvchi dastur tuzing.

Butun 38. Sutka boshidan N sekund o‘tdi. Oxirgi soatdan keyin necha to‘liq minut o‘tganini va oxirgi minutdan keyin necha sekund qolganini chiqaruvchi dastur tuzing (yo‘llanma: birliklar, butun bo‘lish va qoldiq hisoblash).

Butun 39. Tomoni B bo‘lgan kvadrat berilgan. Uning ichiga joylashtirish mumkin bo‘lgan A tomonli kvadratlar sonini chiqaruvchi dastur tuzing (yo‘llanma: butun bo‘lish).

Butun 40. Tomoni B bo‘lgan kvadrat berilgan. Uning ichiga joylashtirish mumkin bo‘lgan R radiusli aylanalar sonini chiqaruvchi dastur tuzing (yo‘llanma: butun bo‘lish).

Butun 41. Sakkizlik sanoq sistemasidagi 3 xonali son berilgan. Shu sonni 10 lik sanoq sistemasidagi qiymatini chiqaruvchi dastur tuzing (yo‘llanma: butun bo‘lish va qoldiq hisoblash, sonning yoyiq ko‘rinishi).

Butun 42. Ikkilik sanoq sistemasidagi 5 xonali son berilgan. Shu sonning 10 lik sanoq sistemasidagi qiymatini chiqaruvchi dastur tuzing (yo‘llanma: butun bo‘lish va qoldiq hisoblash, sonning yoyiq ko‘rinishi).

Butun 43. O‘nlik sanoq sistemasidagi A son berilgan (15<A<32). Shu sonning ikkilik sanoq sistemasidagi qiymatini chiqaruvchi dastur tuzing (yo‘llanma: butun bo‘lish va qoldiq hisoblash, sonning yoyiq ko‘rinishi).

Butun 44. O‘nlik sanoq sistemasidagi A son berilgan (15<A<32). Shu sonning sakkizlik sanoq sistemasidagi raqamlari yig‘indisini o‘nlik sanoq sistemasida hisoblab chiqaruvchi dastur tuzing (yo‘llanma: butun bo‘lish va qoldiq hisoblash).

= C =

Butun 45. Uch xonali x sonidan birlik raqami ayirildi. Hosil bo‘lgan sonni 10 ga bo‘lgandagi bo‘linmasining chap tomoniga x sonining birlik raqami yozildi. Natijada N soni hosil bo‘ldi. Berilgan N sonining o‘nlik raqami 0 dan farqli bo‘lib [10; 999] kesmada yotsa, u holda x sonini aniqlab chiqaruvchi dastur tuzing (yo‘llanma: butun bo‘lish va qoldiq hisoblash, sonning yoyiq ko‘rinishi).

Butun 46. Uch xonali x sonining yuzlik raqami o‘chirildi. Hosil bo‘lgan sonni 10 ga ko‘paytirib, hosil bo‘lgan natijaga x sonining yuzlik raqami qo‘shildi. Natijada N soni hosil bo‘ldi. Berilgan N soni [1; 999] kesmada yotsa, u holda x sonini aniqlab chiqaruvchi dastur tuzing (yo‘llanma: butun bo‘lish va qoldiq hisoblash, sonning yoyiq ko‘rinishi).

Butun 47. Uch xonali x sonining o‘nlik raqami o‘chirildi. Hosil bo‘lgan sonning chap tomoniga x sonining o‘nlik raqami yozildi. Natijada N soni hosil bo‘ldi. Berilgan N sonining o‘nlik raqami 0 dan farqli bo‘lib [10; 999] kesmada yotsa, u holda x sonini aniqlab chiqaruvchi dastur tuzing (yo‘llanma: butun bo‘lish va qoldiq hisoblash, sonning yoyiq ko‘rinishi).

Butun 48. Uch xonali x sonining o'nlik raqami o'chirildi. Hosil bo'lgan sonning o'ng tomoniga x sonining o'nlik raqami yozildi. Natijada N soni hosil bo'ldi. Berilgan N soni [100; 999] kesmada yotsa, u holda x sonini aniqlab chiqaruvchi dastur tuzing (yo'llanma: butun bo'lish va qoldiq hisoblash, sonning yoyiq ko'rinishi).

Butun 49. Uch xonali x sonining birlik raqami o'chirildi. Hosil bo'lgan ikki xonali sonning raqamlari o'rni almashtirilib, chap tomoniga x sonining birlik raqami yozildi. Natijada N soni hosil bo'ldi. Berilgan N sonining birlik raqami 0 dan farqli bo'lib [1; 999] kesmada yotsa, u holda x sonini aniqlab chiqaruvchi dastur tuzing (yo'llanma: butun bo'lish va qoldiq hisoblash, sonning yoyiq ko'rinishi).

Butun 50. Qo'shimcha o'zgaruvchidan foydalanmasdan a va b o'zgaruvchilar qiymatini almashtirib ekranga chiqaruvchi dastur tuzing. Masalan, a=3 va b=4 kiritilsa, u holda ekranga a=4 va b=3 kabi chiqarilishi kerak (yo'llanma: faqat qo'shish va ayirish yoki faqat ko'paytirish va butun bo'lish yoki faqat 3 marta XOR amalidan foydalanish).

Butun 51. Tomoni A va B bo'lgan to'g'ri to'rtburchak berilgan. Uning ichiga joylashtirish mumkin bo'lgan C tomonli kvadratlar soni va ortib qolgan soha yuzini chiqaruvchi dastur tuzing (yo'llanma: butun bo'lish va qoldiq hisoblash).

Butun 52. Hafta kunlariga quyidagicha tartib raqami berilgan: 0 – yakshanba, 1 – dushanba, 2 – seshanba, 3 – chorshanba, 4 – payshanba, 5 – juma, 6 – shanba. Yilning 1..365 diapazonidagi K soni berilgan. Agar 1-yanvar dushanba bo'lsa, u holda yilning K-kuni haftaning qaysi kuniga mos kelishini chiqaruvchi dastur tuzing (yo'llanma: butun bo'lish va qoldiq hisoblash).

Butun 53. Hafta kunlariga quyidagicha tartib raqamlari berilgan: 0 – yakshanba, 1 – dushanba, 2 – seshanba, 3 – chorshanba, 4 – payshanba, 5 – juma, 6 – shanba. Yilning 1..365 diapazonidagi K soni berilgan. Agar 1-yanvar payshanba bo'lsa, u holda yilning K-kuni haftaning qaysi kuniga mos kelishini chiqaruvchi dastur tuzing (yo'llanma: butun bo'lish va qoldiq hisoblash).

Butun 54. Hafta kunlariga quyidagicha tartib raqamlari berilgan: 1 – dushanba, 2 – seshanba, 3 – chorshanba, 4 – payshanba, 5 – juma, 6 – shanba, 7 – yakshanba. Yilning 1..365 diapazonidagi K soni berilgan. Agar 1-yanvar seshanba bo'lsa, u holda yilning K-kuni haftaning qaysi kuniga mos kelishini chiqaruvchi dastur tuzing (yo'llanma: butun bo'lish va qoldiq hisoblash).

Butun 55. Hafta kunlariga quyidagicha tartib raqamlari berilgan: 1 – dushanba, 2 – seshanba, 3 – chorshanba, 4 – payshanba, 5 – juma, 6 – shanba, 7 – yakshanba. Yilning 1..365 diapazonidagi K soni berilgan. Agar 1-yanvar shanba bo'lsa, u holda yilning K-kuni haftaning qaysi kuniga mos kelishini chiqaruvchi dastur tuzing (yo'llanma: butun bo'lish va qoldiq hisoblash).

Butun 56. Hafta kunlariga quyidagicha tartib raqamlari berilgan: 1 – dushanba, 2 – seshanba, 3 – chorshanba, 4 – payshanba, 5 – juma, 6 – shanba, 7 – yakshanba. Yilning 1..365 diapazonidagi K soni va 1..7 diapazondagi N soni berilgan. Agar 1-yanvar haftaning N-kuni bo'lsa, u holda yilning K-kuni haftaning qaysi kuniga mos kelishini chiqaruvchi dastur tuzing (yo'llanma: butun bo'lish va qoldiq hisoblash).

Butun 57. Deylik, A yilning tartibi bo'lsin (0<A<50000). Shu yil tegishli bo'lgan asr tartibini chiqaruvchi dastur tuzing. Masalan, 1963 yil 20 asrda (yo'llanma: butun bo'lish va qoldiq hisoblash).

3-§. TO'PLAMLAR USTIDAGI AMALLAR VA MANTIQIY AMALLAR

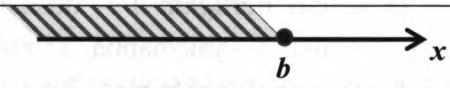
Matematika fanida ko'pgina to'plamlarni geometrik (chizmalar yordamida) va algebraik (ifodalar yordamida) usullarda tasvirlash umkoniyati bo'lib, bu usullar turli matematik masalalar yechishni osonlashtiradi.

To'plamlar ustida kesishma, birlashma, ayirma amallarini bajarib yangi to'plamlar hosil qilish mumkin. Bu amallar $\cap$ (kesishma), $\cup$ (birlashma), / (ayirma) kabi belgilanadi.

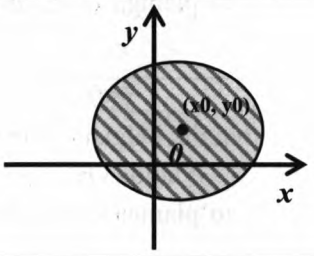
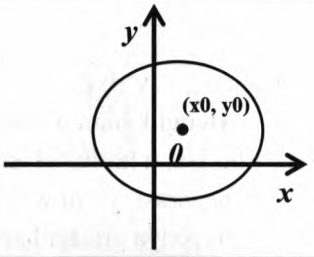
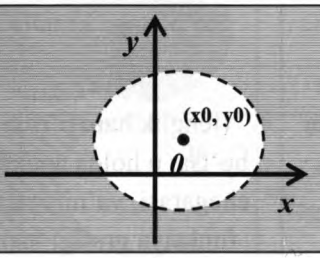
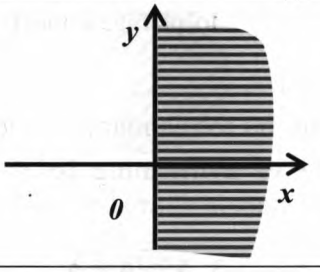
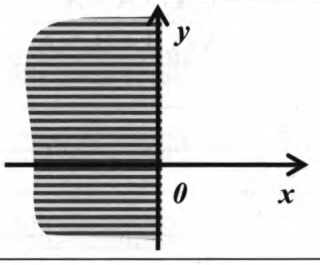
Bizning maqsad to'plamlarni ifodalashda dasturlash tillari uchun qulay bo'lgan uchinchi usul, ya'ni mantiqiy ifoda hosil qilish imkoniyatlarini ko'rib chiqishdir. Mantiqiy ifodalar sodda to'plamlar (bitta ifoda orqali tasvirlanadigan) ustida amallar bajarib murakkab ko'rinishdagi to'plamlarni hosil qilish uchun muhimdir.

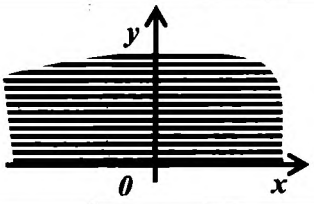
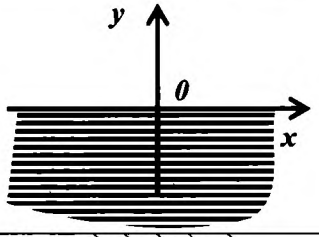
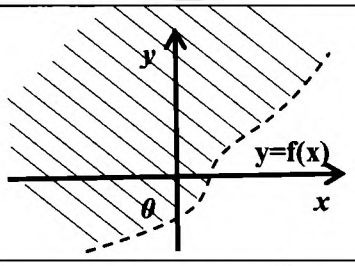
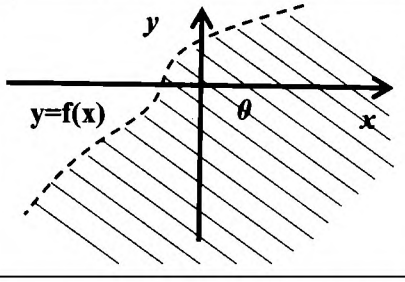
Har qanday chekli yoki qisman chekli to'plam qandaydir cheksiz to'plamning qismi bo'lishi mumkin. Masalan:

R=(–∞; +∞) to'plam – bu haqiqiy sonlar to'plami (boshqa nomi: Ox son o'qi) cheksiz to'plam bo'lib, quyidagilar uning qism to'plamlariga misol bo'ladi:

Chizmasi	Sodda to'plam	Algebraik ifodasi
	[a; +∞) nur yoki o'ng yarim o'q	a≤x
	(a; +∞) o'ng cheksiz interval	a<x
	(–∞; b] nur yoki chap yarim o'q	x≤b
	(–∞; b) chap cheksiz interval	x<b

$R^2=R\oplus R=(-\infty;\ +\infty)\oplus(-\infty;\ +\infty)$ to‘plam – bu Oxy (Dekart) koordinatalar sistemasi aniqlagan tekislik bo‘lib, quyidagilar uning qism to‘plamlariga misol bo‘ladi:

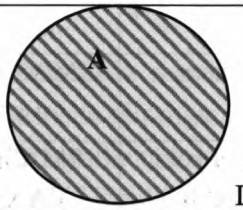
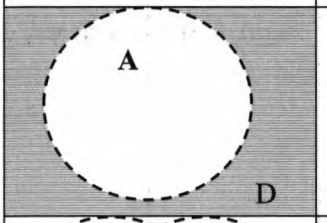
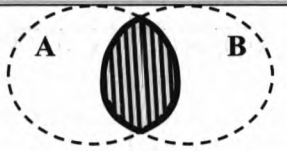
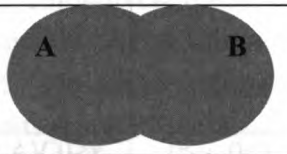
Chizmasi	Sodda to‘plam	Algebraik ifodasi
	Doira	$(x-x0)^2+(y-y0)^2\le r^2$ (tenglik belgisi bo‘lmasa, u holda doira chegarasi to‘plamga kirmaydi)
	Aylana	$(x-x0)^2+(y-y0)^2=r^2$
	Doiraning tashqarisi	$(x-x0)^2+(y-y0)^2>r^2$ (tenglik belgisi bo‘lsa, u holda doira chegarasi ham to‘plamga kiradi)
	Oxy o‘ng yarim tekisligi	$x\ge 0$ (tenglik belgisi bo‘lmasa, u holda Oy o‘qi to‘plamga kirmaydi)
	Oxy chap yarim tekisligi	$x\le 0$ (tenglik belgisi bo‘lmasa, u holda Oy o‘qi to‘plamga kirmaydi)

	Oxy yuqori yarim tekisligi	$y \geq 0$ (tenglik belgisi bo‘lmasa, u holda Ox o‘qi to‘plamga kirmaydi)
	Oxy quyi yarim tekisligi	$y \leq 0$ (tenglik belgisi bo‘lmasa, u holda Ox o‘qi to‘plamga kirmaydi)
	y=f(x) funksiya grafigidan yuqorida (yoki chapda) yotgan yarim tekislik	$y > f(x)$ (tenglik ham o‘rinli bo‘lsa, u holda tekislik chegarasi, ya’ni y=f(x) funksiya grafigi ham to‘plamga kiradi)
	y=f(x) funksiya grafigidan quyida (yoki o‘ngda) yotgan yarim tekislik	$y < f(x)$ (tenglik ham o‘rinli bo‘lsa, u holda tekislik chegarasi, ya’ni y=f(x) funksiya grafigi ham to‘plamga kiradi)

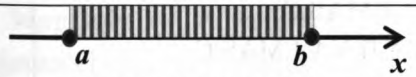
Berilgan ikki ixtiyoriy sodda to‘plamlarni A va B bilan, bu to‘plamlarni o‘z ichiga olgan cheksiz to‘plamni D bilan belgilab, kesishma, birlashma va ayirmaning ba’zi xossalarini quyida sanab o‘tamiz:

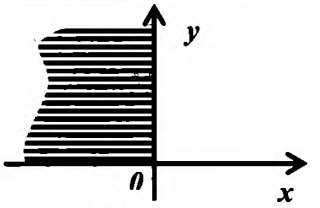
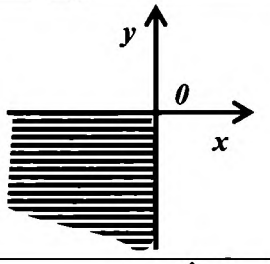
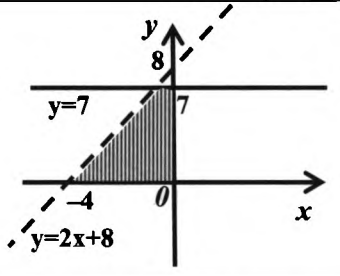
1. $A \cup A = A$	2. $A \cup D = D$	3. $A \cap A = A$
4. $A \cap D = A$	5. $A \cup B = B \cup A$	6. $A \cap B = B \cap A$
7. $(A \cup B) \cup C = A \cup (B \cup C)$		
8. $(A \cap B) \cap C = A \cap (B \cap C)$		
9. $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$		
10. $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$		

Yuqoridagi A, B va D to‘plamlar ustidagi amallarni chizma, algebraik ifoda va mantiqiy ifodalar orqali quyidagicha tasvirlash mumkin:

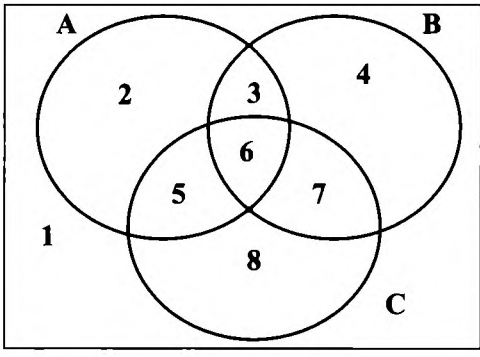
Chizma	Algebraik ifoda (va mazmuni)	Mantiqiy ifoda
	A to‘plam (to‘plamning tengsizlik orqali algebraik ifodasi)	A
	D\A (A to‘plamdan tashqari soha)	EMAS A
	A∩B (bir vaqtda ham A ga va ham B ga tegishli bo‘lgan elementlar to‘plami)	A VA B
	A∪B (A va B to‘plamning barcha elementlari to‘plami)	A YOKI B
	A\B (A to‘plamning B to‘plamga tegishli bo‘lmagan elementlari to‘plami)	A VA EMAS B
	B\A (B to‘plamning A to‘plamga tegishli bo‘lmagan elementlari to‘plami)	EMAS A VA B

Bu amallar yordamida nisbatan murakkab to‘plamlar hosil qilish mumkin. Masalan:

Chizmasi	Murakkab to‘plam	Algebraik ifodasi	Mantiqiy ifodasi
	Kesma [a; b] = = [a; +∞) ∩ (–∞; b]	a≤x≤b	a≤x VA x≤b
	Interval [a; b) = = [a; +∞) ∩ (–∞; b)	a≤x<b	a≤x VA x<b
	Interval (a; b] = = (a; +∞) ∩ (–∞; b]	a<x≤b	a<x VA x≤b

	Interval (a; b) = = (a; +∞) ∩ (–∞; b)	a<x<b	a<x VA x<b
	R\[a; +∞)= (–∞; a)	x<a	EMAS x≥a = = x<a
	R\[a; b]= (–∞; a) ∪ (b; +∞)	x<a yoki x>b	EMAS(a≤x VA x≤b)
	II kvadrant	x≤0, y≥0	x≤0 VA y≥0
	III kvadrant	x≤0, y≤0	x≤0 VA y≤0
	Trapetsiya	x≤0, y≥0, y≤7, y<2x+8	x≤0 VA y≥0 VA y≤7 VA y<2x+8

To‘plamlar soni ortib borgani sari ular ustidagi amallarni yozish murakkablashib boraveradi. Quyida 3 ta to‘plamdan hosil qilingan sohalar uchun mantiqiy ifodalar keltirilgan:



1: EMAS A VA EMAS B VA EMAS C
2: A VA EMAS B VA EMAS C
3: A VA B VA EMAS C
4: EMAS A VA B VA EMAS C
5: A VA EMAS B VA C
6: A VA B VA C
7: EMAS A VA B VA C
8: EMAS A VA EMAS B VA C

4-§. MANTIQIY TURGA OID MASALALAR

Bu paragrafda masala shartidagi mulohazaga mos mantiqiy ifoda tuzish va kiritilgan qiymatga asosan ekranga shu ifoda natijasi 1 (true) yoki 0 (false) ni chiqarish ko'zda tutilgan. Xonalar (razryadlar) soni (ikki xonali, uch xonali, ...) berilgan masalalardagi barcha sonlar butun musbat hisoblanib, yuqori razryad (masalan, uch xonali 204 da 2 yuqori) 0 dan farqli. Sonlar diapazoni aniq bo'lgan yoki chegarasi berilgan masalalarda xotirani tejaydigan butun turni tanlash talab etiladi, aks holda, ya'ni chegarasi berilmagan holda sonlarni **int** turida tavsiflash mumkin.

= A =

Mantiqiy 1. A mulohaza rost bo'lsin. "EMAS A YOKI A" mulohaza natijasini chiqaruvchi dastur tuzing.

Mantiqiy 2. Mulohazalar A=rost, B=yolg'on bo'lsin. "A VA EMAS B" mulohaza natijasini chiqaruvchi dastur tuzing.

Mantiqiy 3. Mulohazalar A=rost, B=yolg'on, C=yolg'on bo'lsin. "A YOKI B VA EMAS C" mulohaza natijasini chiqaruvchi dastur tuzing.

Mantiqiy 4. Mulohazalar A=rost, B=yolg'on, C=yolg'on bo'lsin. "EMAS (A VA B) YOKI C" mulohaza natijasini chiqaruvchi dastur tuzing.

Mantiqiy 5. A, B va C mulohazalar qiymati berilgan. "A VA B YOKI EMAS C" mulohaza natijasini chiqaruvchi dastur tuzing.

Mantiqiy 6. A son berilgan. "A musbat son" mulohaza natijasini chiqaruvchi dastur tuzing.

Mantiqiy 7. A son berilgan. "A manfiy emas" mulohaza natijasini chiqaruvchi dastur tuzing.

Mantiqiy 8. A son berilgan. "A juft son" mulohaza natijasini chiqaruvchi dastur tuzing (yo'llanma: juftlik qoidasi).

Mantiqiy 9. A son berilgan. "A toq son" mulohaza natijasini chiqaruvchi dastur tuzing (yo'llanma: juftlik qoidasi).

Mantiqiy 10. A va B sonlar berilgan. "$A > 7$ va $B \le 21$ tengsizliklar o'rinli" mulohaza natijasini chiqaruvchi dastur tuzing.

Mantiqiy 11. A va B sonlar berilgan. "$A \ge -1$ va $B \le -23$ tengsizliklar o'rinli" mulohaza natijasini chiqaruvchi dastur tuzing.

Mantiqiy 12. A va B sonlar berilgan. "$A^2 \cdot (A+B) - B \cdot (B-A)^2$ ifoda qiymati musbat" mulohaza natijasini chiqaruvchi dastur tuzing.

Mantiqiy 13. A va B sonlar berilgan. "A soni manfiy emas va B sonidan kichik" mulohaza natijasini chiqaruvchi dastur tuzing.

Mantiqiy 14. "Berilgan uchala sonlar bir-biriga teng" mulohaza natijasini chiqaruvchi dastur tuzing.

Mantiqiy 15. "Berilgan ikki xonali sonning birinchi raqami ikkinchi raqamidan katta" mulohaza natijasini chiqaruvchi dastur tuzing (yo'llanma: butun bo'lish va qoldiq hisoblash).

Mantiqiy 16. A va B sonlar berilgan. "Ax+B=0 tenglama yagona yechimga ega" mulohaza natijasini chiqaruvchi dastur tuzing.

Mantiqiy 17. A, B va C sonlar berilgan (A noldan farqli). "$Ax^2+Bx+C=0$ tenglama yagona yechimga ega" mulohaza natijasini chiqaruvchi dastur tuzing (yo'llanma: $D=B^2-4\cdot A\cdot C$ diskriminantni 0 bilan taqqoslash).

Mantiqiy 18. A, B va C sonlar berilgan (A noldan farqli). "$Ax^2+Bx+C=0$ tenglama haqiqiy yechimlarga ega" mulohaza natijasini chiqaruvchi dastur tuzing (yo'llanma: $D=B^2-4\cdot A\cdot C$ diskriminantni 0 bilan taqqoslash).

Mantiqiy 19. x soni berilgan. "Ox o'qida x koordinatali nuqta 0 nuqtadan chapda yotadi" mulohaza natijasini chiqaruvchi dastur tuzing.

Mantiqiy 20. Uchburchakning A, B va C tomonlari berilgan. "Uchburchak teng tomonli" mulohaza natijasini chiqaruvchi dastur tuzing.

Mantiqiy 21. A, B, x va y sonlar berilgan. "Oxy tekisligida (x; y) koordinatali nuqta $y=Ax+B$ funksiya grafigida yotadi" mulohaza natijasini chiqaruvchi dastur tuzing.

Mantiqiy 22. A, B, C, x va y sonlar berilgan. "Oxy tekisligida (x; y) koordinatali nuqta $y=Ax^2+Bx+C$ funksiya grafigida yotadi" mulohaza natijasini chiqaruvchi dastur tuzing.

Mantiqiy 23. x va y sonlar berilgan. "Oxy tekisligida (x; y) koordinatali nuqta $x^2+y^2=19^2$ aylanada yotadi" mulohaza natijasini chiqaruvchi dastur tuzing.

Mantiqiy 24. x va y sonlar berilgan. "Oxy tekisligida (x; y) koordinatali nuqta $x^2+y^2=19^2$ aylana ichida yotadi" mulohaza natijasini chiqaruvchi dastur tuzing.

= B =

Mantiqiy 25. A, B va C sonlar berilgan. "A < B < C qo'sh tengsizlik o'rinli" mulohaza natijasini chiqaruvchi dastur tuzing.

Mantiqiy 26. A, B va C sonlar berilgan. "B soni A va C sonlar orasida yotadi" mulohaza natijasini chiqaruvchi dastur tuzing.

Mantiqiy 27. A va B sonlar berilgan. "A va B sonlarning har biri juft" mulohaza natijasini chiqaruvchi dastur tuzing (yo'llanma: qoldiq hisoblash).

Mantiqiy 28. A va B sonlar berilgan. "A va B sonlarning har biri toq" mulohaza natijasini chiqaruvchi dastur tuzing (yo'llanma: qoldiq hisoblash).

Mantiqiy 29. A va B sonlar berilgan. "A va B sonlarning hech bo'lmasa bittasi toq" mulohaza natijasini chiqaruvchi dastur tuzing (yo'llanma: qoldiq hisoblash).

Mantiqiy 30. A va B sonlar berilgan. "A va B sonlarning hech bo'lmasa bittasi juft" mulohaza natijasini chiqaruvchi dastur tuzing (yo'llanma: qoldiq hisoblash).

Mantiqiy 31. A va B sonlar berilgan. "A va B sonlarning faqat bittasi toq" mulohaza natijasini chiqaruvchi dastur tuzing (yo'llanma: qoldiq hisoblash).

Mantiqiy 32. A va B sonlar berilgan. "A va B sonlarning faqat bittasi juft" mulohaza natijasini chiqaruvchi dastur tuzing (yo'llanma: qoldiq hisoblash).

Mantiqiy 33. A va B sonlar berilgan. "A va B sonlarning ikkalasi bir vaqtda juft yoki toq" mulohaza natijasini chiqaruvchi dastur tuzing (yo'llanma: qoldiq hisoblash).

Mantiqiy 34. A, B va C sonlar berilgan. "A, B va C sonlardan hech bo'lmasa bittasi musbat" mulohaza natijasini chiqaruvchi dastur tuzing.

Mantiqiy 35. A, B va C sonlar berilgan. "A, B va C sonlardan faqat bittasi musbat" mulohaza natijasini chiqaruvchi dastur tuzing.

Mantiqiy 36. A, B va C sonlar berilgan. "A, B va C sonlardan faqat ikkitasi musbat" mulohaza natijasini chiqaruvchi dastur tuzing.

Mantiqiy 37. A soni berilgan. "A soni ikki xonali va juft" mulohaza natijasini chiqaruvchi dastur tuzing.

Mantiqiy 38. A soni berilgan. "A soni uch xonali va toq" mulohaza natijasini chiqaruvchi dastur tuzing (yo'llanma: qoldiq hisoblash).

Mantiqiy 39. "Berilgan uchta sondan hech bo'lmasa ikkitasi teng" mulohaza natijasini chiqaruvchi dastur tuzing.

Mantiqiy 40. "Berilgan uchta sondan hech bo'lmasa ikkitasi teskari ishorali" mulohaza natijasini chiqaruvchi dastur tuzing.

Mantiqiy 41. "Berilgan uch xonali sonni ikkinchi raqami birinchi va uchinchi raqamidan katta" mulohaza natijasini chiqaruvchi dastur tuzing (yo'llanma: butun bo'lish va qoldiq hisoblash).

Mantiqiy 42. "Berilgan uch xonali sonning barcha raqamlari bir-biridan farqli" mulohaza natijasini chiqaruvchi dastur tuzing (yo'llanma: butun bo'lish va qoldiq hisoblash).

Mantiqiy 43. "Berilgan uch xonali sonning raqamlari yozilish tartibida o'suvchi ketma-ketlik tashkil etadi" mulohaza natijasini chiqaruvchi dastur tuzing (yo'llanma: butun bo'lish va qoldiq).

Mantiqiy 44. A va B sonlar berilgan. "Ax+B=0 tenglama cheksiz ko'p yechimga ega" mulohaza natijasini chiqaruvchi dastur tuzing.

Mantiqiy 45. x va y sonlar berilgan. "Oxy tekisligida (x; y) koordinatali nuqta ikkinchi kvadrantda yotadi" mulohaza natijasini chiqaruvchi dastur tuzing.

Mantiqiy 46. x va y sonlar berilgan. "Oxy tekisligida (x; y) koordinatali nuqta to'rtinchi kvadrantda yotadi" mulohaza natijasini chiqaruvchi dastur tuzing.

Mantiqiy 47. x va y sonlar berilgan. "Oxy tekisligida (x; y) koordinatali nuqta ikkinchi yoki uchinchi kvadrantda yotadi" mulohaza natijasini chiqaruvchi dastur tuzing.

Mantiqiy 48. x va y sonlar berilgan. "Oxy tekisligida (x; y) koordinatali nuqta birinchi yoki uchinchi kvadrantda yotadi" mulohaza natijasini chiqaruvchi dastur tuzing.

Mantiqiy 49. x0, y0, x1, y1, x2 va y2 sonlar berilgan. "Oxy tekisligida (x0; y0) koordinatali nuqta tomonlari koordinata o'qlariga parallel, qarama-qarshi uchlari (x1; y1) va (x2; y2) nuqtalarda bo'lgan to'g'ri to'rtburchakning chegarasiga tegishli" mulohaza natijasini chiqaruvchi dastur tuzing.

Mantiqiy 50. x1, y1, x2 va y2 sonlar berilgan. "Oxy tekisligida (x1; y1) va (x2; y2) nuqtalar orasidagi masofa juft son" mulohaza natijasini chiqaruvchi dastur tuzing (yo'llanma: qoldiq hisoblash, juft va toq sonlar qoidalari, juftlik qoidasi).

Mantiqiy 51. Uchburchakning A, B va C tomonlari berilgan. "Uchburchak teng yonli" mulohaza natijasini chiqaruvchi dastur tuzing.

Mantiqiy 52. 1..8 diapazondagi x va y sonlar berilgan. Shaxmat doskasining chap quyi katagi (ya'ni (1; 1) katak) qora ekanligini hisobga olib "(x; y) katak oq rangda" mulohaza natijasini chiqaruvchi dastur tuzing (yo'llanma: joylashish tahlili).

Mantiqiy 53. 1..8 diapazondagi x1, y1, x2 va y2 sonlar berilgan. Shaxmat doskasining (x1; y1) va (x2; y2) kataklari turlicha bo'lsin. "Shoh berilgan bir katakdan ikkinchisiga bir yurishda o'tadi" mulohaza natijasini chiqaruvchi dastur tuzing (yo'llanma: joylashish tahlili).

Mantiqiy 54. 1..8 diapazondagi x1, y1, x2 va y2 sonlar berilgan. Shaxmat doskasining (x1; y1) va (x2; y2) kataklari turlicha bo'lsin. "Rux berilgan bir katakdan ikkinchisiga bir yurishda o'tadi" mulohaza natijasini chiqaruvchi dastur tuzing (yo'llanma: joylashish tahlili).

= C =

Mantiqiy 55. Ikki xonali A va B sonlar berilgan. "A·B ifoda qiymati juft son" mulohaza natijasini chiqaruvchi dastur tuzing (yo'llanma: qoldiq hisoblash, juftlik qoidasi, juft va toq sonlar qoidalari).

Mantiqiy 56. Ikki xonali A, B va C sonlar berilgan. "A·B·C ifoda qiymati juft son emas" mulohaza natijasini chiqaruvchi dastur tuzing (yo'llanma: qoldiq hisoblash, juftlik qoidasi, juft va toq sonlar qoidalari).

Mantiqiy 57. "Berilgan uch xonali sonning raqamlari yozilishi tartibida o'suvchi yoki kamayuvchi ketma-ketlik tashkil etadi" mulohaza natijasini chiqaruvchi dastur tuzing (yo'llanma: butun bo'lish va qoldiq hisoblash).

Mantiqiy 58. "Berilgan to'rt xonali son chapdan ham, o'ngdan ham bir xil o'qiladi" mulohaza natijasini chiqaruvchi dastur tuzing (yo'llanma: butun bo'lish va qoldiq hisoblash, sonning yoyiq ko'rinishi).

Mantiqiy 59. x va y sonlar berilgan. "Ox o'qida x va y koordinatali nuqtalar orasidagi masofa 19 ga teng" mulohaza natijasini chiqaruvchi dastur tuzing.

Mantiqiy 60. x0, y0, x1, y1, x2, y2 sonlar berilgan. "Oxy tekisligida (x0; y0) koordinatali nuqta tomonlari koordinata o'qlariga parallel, chap yuqori uchi (x1; y1) va o'ng quyi (x2; y2) nuqtalarda bo'lgan to'g'ri to'rtburchakka tegishli" mulohaza natijasini chiqaruvchi dastur tuzing.

Mantiqiy 61. Uchburchakning A, B va C tomonlari berilgan. "Uchburchak to'g'ri burchakli" mulohaza natijasini chiqaruvchi dastur tuzing (yo'llanma: Pifagor teoremasi).

Mantiqiy 62. A, B va C sonlar berilgan. "Tomonlari A, B, va C bo'lgan uchburchak mavjud" mulohaza natijasini chiqaruvchi dastur tuzing (yo'llanma: uchburchak qoidasi).

Mantiqiy 63. 1..8 diapazondagi x1, y1, x2 va y2 sonlar berilgan. Shaxmat doskasiga oid "(x1; y1) va (x2; y2) kataklar bir xil rangda" mulohaza natijasini chiqaruvchi dastur tuzing (yo'llanma: joylashish tahlili).

Mantiqiy 64. 1..8 diapazondagi x1, y1, x2 va y2 sonlar berilgan. Shaxmat doskasining (x1; y1) va (x2; y2) kataklari turlicha bo'lsin. "Fil berilgan bir katakdan ikkinchisiga bir yurishda o'tadi" mulohaza natijasini chiqaruvchi dastur tuzing (yo'llanma: joylashish tahlili).

Mantiqiy 65. 1..8 diapazondagi x1, y1, x2 va y2 sonlar berilgan. Shaxmat doskasining (x1; y1) va (x2; y2) kataklari turlicha bo'lsin. "Farzin berilgan bir katakdan ikkinchisiga bir yurishda o'tadi" mulohaza natijasini chiqaruvchi dastur tuzing (yo'llanma: joylashish tahlili).

Mantiqiy 66. 1..8 diapazondagi x1, y1, x2 va y2 sonlar berilgan. Shaxmat doskasining (x1; y1) va (x2; y2) kataklari turlicha bo'lsin. "Ot berilgan bir katakdan ikkinchisiga bir yurishda o'tadi" mulohaza natijasini chiqaruvchi dastur tuzing (yo'llanma: joylashish tahlili).

Mantiqiy 67. A, B, C va D kesmalar berilgan. Shu kesmalardan yasash mumkin bo'lgan uchburchaklar sonini chiqaruvchi dastur tuzing (yo'llanma: uchburchak qoidasi).

Mantiqiy 68. Tekislikda koordinatalari orqali A(x1, y1), B(x2, y2), G(x3, y3) nuqtalar berilgan. "B nuqta koordinata boshiga yaqin" mulohaza natijasini chiqaruvchi dastur tuzing (yo'llanma: nuqtalardan koordinata boshigacha bo'lgan masofalar kvadrati).

Mantiqiy 69. Noldan farqli A soni va A soniga karrali noldan farqli B son berilgan. "y=A·x+B to'g'ri chiziq va koordinata o'qlari bilan chegaralangan uchburchak ikkinchi kvadrantda yotadi" mulohaza natijasini chiqaruvchi dastur tuzing (yo'llanma: koordinata o'qlari bilan kesishish nuqtalari).

Mantiqiy 70. A, B, C va D sonlar berilgan. "Tomonlari A, B, C, D ketma-ketlik orqali aniqlangan qabariq to'rtburchak romb bo'ladi" mulohaza natijasini chiqaruvchi dastur tuzing (yo'llanma: to'rtburchakning barcha tomonlari teng bo'lsa, u kvadrat yoki romb bo'ladi; kvadratning diagonallari uzunligi teng; rombning diagonallari uzunligi teng emas).

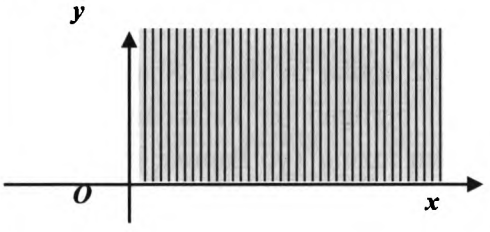
Mantiqiy 71. 3 ta tanganing A, B va C og‘irligi berilgan. Ulardan ikkitasi haqiqiy, ya’ni og‘irligi teng, qalbakisining og‘irligi esa haqiqiy tangalar o‘g‘irligidan farqlanadi. Qalbaki tanganing og‘irligini chiqaruvchi dastur tuzing.

Mantiqiy 72. Berilgan butun sonning modulini chiqaruvchi dastur tuzing.

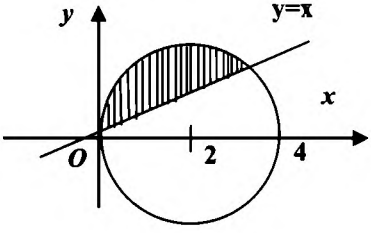
Mantiqiy 73. Musbat 3 xonali son berilgan. Shu sonning raqamlaridan hosil qilingan eng katta sonni chiqaruvchi dastur tuzing (yo‘llanma: butun bo‘lish va qoldiq hisoblash, saralash, sonning yoyiq ko‘rinishi).

Mantiqiy xx. x0 va y0 sonlar va shtrixlangan sohasi bo‘lgan chizma berilgan. “Koordinatalari (x0; y0) bo‘lgan nuqta shtrixlangan sohaga tegishli” mulohaza natijasini chiqaruvchi dastur tuzing (yo‘llanma: shtrixlangan sohani tengsizliklar orqali aniqlash).

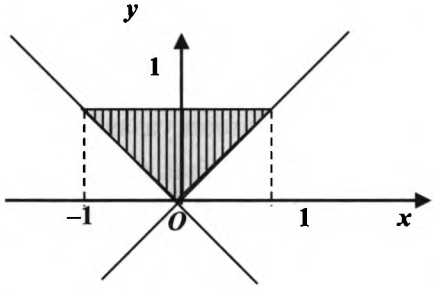
74.



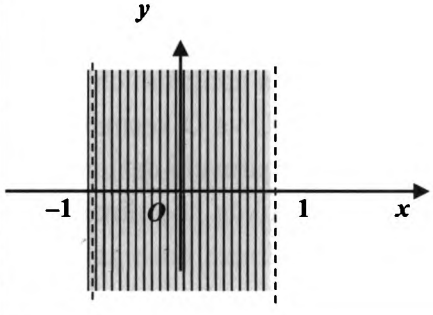
75.



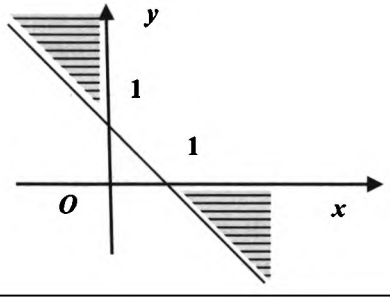
76.



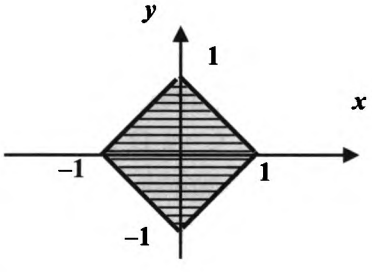
77.



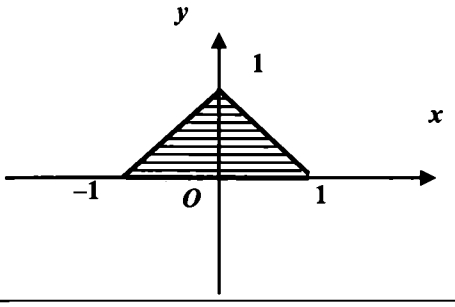
78.



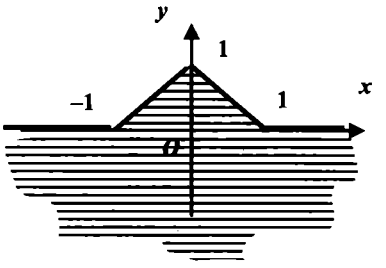
79.



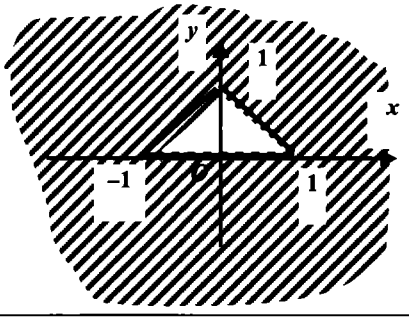
80.



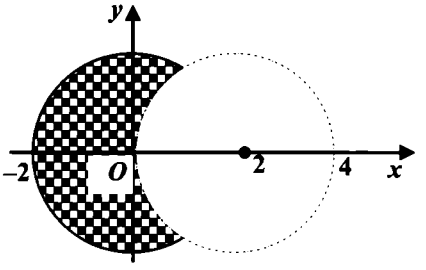
81.



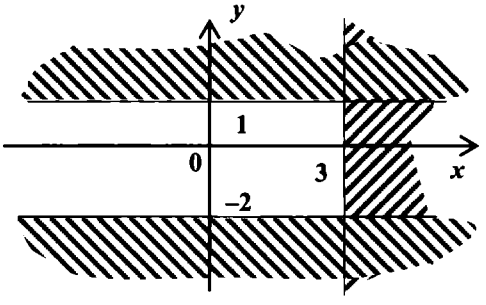
82.



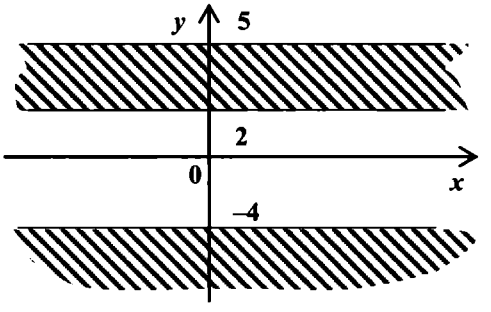
83.



84.



85.



3-BOB. C++ TILIDA MATEMATIK FUNKSIYALAR VA CHIZIQLI DASTURLAR

Bu bobda C++ tilida foydalanish mumkin bo‘lgan asosiy funksiyalar sintaksisi misollar bilan izohlangan. Masalan, funksiyani quyidagi ko‘rinishda tasvirlaymiz:

qaytuvchi_tur funk_nomi(arg1_turi arg1_iden, arg2_turi arg2_iden, ...)

bu yerda **arg1_iden**, **arg2_iden**, ... – argumentlar nomiga mos identifikatorlarni, **arg1_turi**, **arg2_turi**, ... – argumentlar turini, **funk_nomi** – funksiya nomini, **qaytuvchi_tur** – funksiya qaytaradigan ma’lumot turini bildiradi. Albatta, har bir qaytariladigan qiymatni o‘zlashtiradigan o‘zgaruvchi turi **qaytuvchi_tur** da bo‘lishi shart.

Shuningdek, turli amallar va funksiyalarni qo‘llab chiziqli algoritmli dastur tuzish uchun ko‘plab misol va masalalar, turdosh masalalardan ba’zilari esa yechimi bilan keltirilgan. Misol yoki masala shartida aytilmagan bo‘lsa, u holda barcha sonlar haqiqiy turda, aniqlik darajasiga bog‘liq ravishda **float** yoki **double** yoki **long double** deb tushuniladi.

Misol yoki masala shartida “a ni toping” yoki “b ni aniqlang” yoki “c ni hisoblang” talablari dasturda talab qilingan natijani ekranga so‘ralgan ko‘rinishda chiqarishni nazarda tutadi. Ba’zi misol yoki masalalarga tuzilgan dastur ishini birlamchi tekshirish uchun testlar bilan ta’minlangan. Testlarda shartda berilgan yoki kiruvchi (K:) ma’lumotlarga mos chiquvchi (CH:) ma’lumotlar keltirilgan.

Quyida avvalgi bobdagi formulalarga qo‘shimcha ravishda matematikaning turli sohalariga oid foydali ma’lumotlar va formulalar keltirilgan.

Sonlar nazariyasiga oid

A sonining absolyut qiymati (moduli)	$\lvert A\rvert = \begin{cases} -A, \text{agar } A < 0 \\ A, \text{agar } A \geq 0 \end{cases}$
A1, **A2**, …, **An** sonlarining o‘rta arifmetigi	$\frac{A1 + A2 + \cdots + An}{n}$
A1, **A2**, …, **An** sonlarining o‘rta geometrigi	$\sqrt[n]{A1 \cdot A2 \cdot \ldots \cdot An}$
Birinchi hadi $\mathbf{a_1}$, ayirmasi **d** bo‘lgan arifmetik progressiyaning n-hadi $\mathbf{a_n}$	$a_n=a_1+(n-1)\cdot d$
Birinchi hadi $\mathbf{a_1}$, ayirmasi **d** bo‘lgan arifmetik progressiyaning birinchi n ta hadi yig‘indisi $\mathbf{S_n}$	$S_n = \frac{2 \cdot a_1 + d(n-1)}{2} \cdot n$
Birinchi hadi $\mathbf{a_1}$, n-hadi $\mathbf{a_n}$ bo‘lgan arifmetik progressiyaning birinchi n ta hadi yig‘indisi $\mathbf{S_n}$	$S_n = \frac{a_1 + a_n}{2} \cdot n$
Birinchi hadi $\mathbf{b_1}$, maxraji **q** bo‘lgan geometrik progressiyaning n-hadi $\mathbf{b_n}$	$b_n=b_1 \cdot q^{n-1}$
Birinchi hadi $\mathbf{b_1}$, maxraji **q** bo‘lgan geometrik progressiyaning birinchi n ta hadi yig‘indisi $\mathbf{S_n}$	$S_n = \frac{b_1 \cdot (q^n - 1)}{q - 1}$
Birinchi hadi $\mathbf{b_1}$, maxraji **q** ($\lvert q\rvert<1$) bo‘lgan cheksiz geometrik progressiya yig‘indisi **S**	$S = \frac{b_1}{1 - q}$

Uchburchak uchun ma'lumotlar

α radian o'lchov birligidan **gr** gradus o'lchovi birligiga o'tish formulasi	$gr = \alpha \cdot \frac{180^0}{\pi}$
α^o gradus o'lchov birligidan **rad** radian o'lchov birligiga o'tish formulasi	$rad = \alpha^0 \cdot \frac{\pi}{180^0}$
Uchburchakning α, β va γ ichki burchaklari yig'indisi	$\alpha+\beta+\gamma=180^o$
Tomonlari **a**, **b** va **c**, shu tomonlar qarshisidagi burchaklar, mos ravishda, **A**, **B** va **C** bo'lgan uchburchak uchun kosinuslar teoremasi	$a^2=b^2+c^2-2\cdot b\cdot c\cdot cosA$ $b^2=a^2+c^2-2\cdot a\cdot c\cdot cosB$ $c^2=a^2+b^2-2\cdot a\cdot b\cdot cosC$
Uchburchakning **AB** tomoniga parallel **MN** o'rta chizig'i uzunligi	$MN = \frac{AB}{2}$
Tomonlari **a**, **b** va **c** bo'lgan uchburchak yarim perimetri	$p = \frac{a+b+c}{2}$
Tomonlari **a**, **b** va **c** bo'lgan uchburchak yuzi (Geron), bu yerda **p** – yarim perimetr	$S_{uch} = \sqrt{p(p-a)(p-b)(p-c)}$
Tomonlari **a**, **b** va **c** bo'lgan uchburchak yuzi (Geron, yarim perimetrsiz ifodasi)	$S_{uch} = \frac{1}{4}\sqrt{4a^2b^2-(c^2-a^2-b^2)^2}$
Tomonlari **a** va **b**, ular orasidagi burchak **C** bo'lgan uchburchak yuzi	$S_{uch} = \frac{1}{2}a\cdot b\cdot sinC$
Tomoni **a** va shu tomonga tushirilgan balandligi **ha** bo'lgan uchburchak yuzi	$S_{uch} = \frac{1}{2}a\cdot ha$
Tomonlari **a**, **b**, **c** va tashqi chizilgan aylana radiusi **R** bo'lgan uchburchak yuzi	$S_{uch} = \frac{a\cdot b\cdot c}{4\cdot R}$
Tomonlari **a**, **b**, **c** va ichki chizilgan aylana radiusi **r** bo'lgan uchburchak yuzi	$S_{uch} = \frac{a+b+c}{2}\cdot r$
Tomoni **a** bo'lgan teng tomonli uchburchak yuzi	$S_{uch} = \frac{\sqrt{3}}{4}a^2$
Katetlari **a** va **b** bo'lgan to'g'ri burchakli uchburchak yuzi	$S_{uch} = \frac{1}{2}a\cdot b$
Uchburchakning **A** uchidan **a** tomonga tushirilgan **ma** mediana uzunligi	$ma = \frac{1}{2}\sqrt{2b^2+2c^2-a^2}$
Uchburchakning **A** uchidan **a** tomonga tushirilgan **ha** balandlik uzunligi	$ha = \frac{2\sqrt{p(p-a)(p-b)(p-c)}}{a}$
Tomonlari **a**, **b** va **c** bo'lgan uchburchakda mos tomonlarga tushirilgan **ha**, **hb**, **hc** balandliklar orasidagi bog'lanish	$ha:hb:hc = \frac{1}{a}:\frac{1}{b}:\frac{1}{c}$
Uchburchakning **A** uchidan **a** tomonga tushirilgan **la** bissektrisa uzunligi	$la = \frac{2\sqrt{bcp(p-a)}}{b+c}$

Tomoni **a** bo‘lgan teng tomonli uchburchak mediana, balandlik va bissektrisasi uzunliklari	$ma = ha = la = \frac{\sqrt{3}}{2}a$
Tomoni **a** bo‘lgan teng tomonli uchburchakka tashqi va ichki chizilgan aylana radiuslari	$R = \frac{\sqrt{3}}{3}a, \quad r = \frac{\sqrt{3}}{6}a$

Ba’zi geometrik shakllar uchun formulalar

Tomoni **a** va shu tomonga tushirilgan balandligi **ha** bo‘lgan parallelogramm yuzi	$S_{par}=a\cdot ha$
Tomonlari **a** va **b**, ular orasidagi burchak α bo‘lgan parallelogramm yuzi	$S_{par}=a\cdot b\cdot \sin\alpha$
Diagonallari **d1** va **d2**, ular orasidagi burchak α bo‘lgan parallelogramm yuzi	$S_{par} = \frac{1}{2}\cdot d1\cdot d2\cdot \sin\alpha$
Tomonlari **a** va **b** bo‘lgan parallelogrammning **d1** va **d2** diagonallari orasidagi bog‘lanish	$d1^2+d2^2=2(a^2+b^2)$
Tomoni **a** va tomonlar orasidagi burchak α bo‘lgan romb yuzi	$S_{romb}=a^2\cdot \sin\alpha$
Diagonallari **d1** va **d2** bo‘lgan romb yuzi	$S_{romb} = \frac{1}{2}\cdot d1\cdot d2$
Tomoni **a** bo‘lgan rombning **d1** va **d2** diagonallari orasidagi bog‘lanish	$d1^2+d2^2=4a^2$
Tomonlari **a** va **b** bo‘lgan to‘g‘ri to‘rtburchakning **d** diagonali orasidagi bog‘lanish	$d = \sqrt{a^2 + b^2}$
Tomoni **a** bo‘lgan kvadratning yuzi	$S_{kv}=a^2$
Tomoni **a** bo‘lgan kvadratning **d** diagonali orasidagi bog‘lanish	$d = \sqrt{2}\cdot a$
Tomoni **a** bo‘lgan kvadratga ichki chizilgan aylananing **r** radiusi	$r = \frac{1}{2}a$
Tomoni **a** bo‘lgan kvadratga tashqi chizilgan aylananing **R** radiusi	$R = \frac{\sqrt{2}}{2}a$
Asoslari **a** va **b** bo‘lgan trapetsiyaning **MN** o‘rta chizig‘i uzunligi	$MN = \frac{a+b}{2}$
Asoslari **a** va **b**, asosga tushirilgan balandligi **h** bo‘lgan trapet-siyaning yuzi	$S_{tr} = \frac{a+b}{2}\cdot h$
Radiusi **r** bo‘lgan aylana uzunligi	$L = 2\cdot\pi\cdot r$
Radiusi **r** bo‘lgan doiraning yuzi	$S_d = \pi\cdot r^2$
Asosining yuzi **Sa** va balandligi **H** bo‘lgan to‘g‘ri parallelepipedning hajmi	$V=Sa\cdot h$
Tomonlari **a**, **b** va **c** bo‘lgan to‘g‘ri burchakli parallelepipedning hajmi	$V=a\cdot b\cdot c$
Tomoni **a** bo‘lgan kubning hajmi	$V=a^3$
Tomoni **a** bo‘lgan kub to‘la sirtining yuzi	$S_{kub\text{-}to‘la}=6\cdot a^2$

Determinant va vektorlarga oid ma'lumotlar

Ikki o'lchovli determinantning yoyilmasi	$\begin{vmatrix} a1 & b1 \\ a2 & b2 \end{vmatrix} = a1 \cdot b2 - b1 \cdot a2$
Uch o'lchovli determinantning yoyilmasi	$\begin{vmatrix} a1 & b1 & c1 \\ a2 & b2 & c2 \\ a3 & b3 & c3 \end{vmatrix} = a1 \cdot b2 \cdot c3 + b1 \cdot c2 \cdot a3 + c1 \cdot a2 \cdot b3 - a1 \cdot c2 \cdot b3 - b1 \cdot a2 \cdot c3 - c1 \cdot b2 \cdot a3$
c(cx;cy) vektorning uzunligi (tekislikda)	$\lvert c \rvert = \sqrt{cx^2 + cy^2}$
c(cx;cy;cz) vektorning uzunligi (fazoda)	$\lvert c \rvert = \sqrt{cx^2 + cy^2 + cz^2}$
a va **b** vektorlarning **(a,b)** skalyar ko'paytmasi (φ – a va b vektorlar orasidagi burchak)	$(a,b) = \lvert a \rvert \cdot \lvert b \rvert \cdot \cos\varphi$
a(ax;ay) va **b(bx;by)** vektorlarning **(a,b)** skalyar ko'paytmasi (tekislikda)	$(a,b) = ax \cdot bx + ay \cdot by$
a(ax;ay;az) va **b(bx;by;bz)** vektorlarning **(a,b)** skalyar ko'paytmasi (fazoda)	$(a,b) = ax \cdot bx + ay \cdot by + az \cdot bz$
a(ax;ay;az) va **b(bx;by;bz)** vektorlarning **[a×b]** vektor ko'paytmasi (**i**, **j** va **k** – koordinata o'qlari bo'ylab yo'nalgan birlik ortogonal vektorlar)	$[a \times b] = \begin{vmatrix} i & j & k \\ ax & ay & az \\ bx & by & bz \end{vmatrix}$
a(ax;ay;az) va **b(bx;by;bz)** vektorlar hosil qilgan parallelogrammning S yuzi (φ – a va b vektorlar orasidagi burchak, $0<\varphi<\pi$)	$S = \lvert [a \times b] \rvert = \lvert a \rvert \cdot \lvert b \rvert \cdot \sin\varphi$
a(ax;ay;az), **b(bx;by;bz)** va **c(cx;cy;cz)** vektorlarning **(a,[b×c])** (=([a×b],c])) aralash ko'paytmasi	$([a \times b], c) = \begin{vmatrix} ax & ay & az \\ bx & by & bz \\ cx & cy & cz \end{vmatrix}$
a(ax;ay;az), **b(bx;by;bz)** va **c(cx;cy;cz)** vektorlar hosil qilgan parallelepipedning V_{pp} hajmi	$V_{pp} = \left\lvert \begin{vmatrix} ax & ay & az \\ bx & by & bz \\ cx & cy & cz \end{vmatrix} \right\rvert$
A1(x1, y1, z1), A2(x2, y2, z2), A3(x3, y3, z3), A4(x4, y4, z4) nuqtalar hosil qilgan parallelepipedning V_{pp} hajmi	$V_{pp} = \left\lvert \begin{vmatrix} x2-x1 & y2-y1 & z2-z1 \\ x3-x1 & y3-y1 & z3-z1 \\ x4-x1 & y4-y1 & z4-z1 \end{vmatrix} \right\rvert$
a(ax;ay;az), **b(bx;by;bz)** va **c(cx;cy;cz)** vektorlar hosil qilgan piramidaning **V** hajmi	$V_{pir} = \frac{1}{6} \left\lvert \begin{vmatrix} ax & ay & az \\ bx & by & bz \\ cx & cy & cz \end{vmatrix} \right\rvert$
A1(x1, y1, z1), A2(x2, y2, z2), A3(x3, y3, z3), A4(x4, y4, z4) nuqtalar hosil qilgan piramidaning **V** hajmi	$V_{pir} = \frac{1}{6} \left\lvert \begin{vmatrix} x2-x1 & y2-y1 & z2-z1 \\ x3-x1 & y3-y1 & z3-z1 \\ x4-x1 & y4-y1 & z4-z1 \end{vmatrix} \right\rvert$

To‘g‘ri chiziqlarga oid formulalar

Ox o‘qidagi A(x1) va B(x2) nuqtalarning o‘rtasidagi C(x0) nuqta koordinatasi	$x0=\frac{x1+x2}{2}$
Oxy tekisligida A(x1; y1) va B(x2; y2) nuqtalarning o‘rtasidagi C(x0; y0) nuqta koordinatalari ([A; B] kesma o‘rtasi C nuqtaning koordinatalari)	$x0=\frac{x1+x2}{2}$ $y0=\frac{y1+y2}{2}$
Oxy tekisligida to‘g‘ri chiziqning umumiy tenglamasi ($a^2+b^2\neq0$ – to‘g‘ri chiziqning mavjudlik sharti)	$a\cdot x+b\cdot y+c=0$
Oxy tekisligida to‘g‘ri chiziqning burchak koeffitsiyentli tenglamasi	$y=k\cdot x+b$
Oxy tekisligida $y=k1\cdot x+b1$ va $y=k2\cdot x+b2$ to‘g‘ri chiziqlarning **perpendikulyarlik** sharti	$k1\cdot k2=-1$
Oxy tekisligida $a1\cdot x+b1\cdot y+c1=0$ va $a2\cdot x+b2\cdot y+c2=0$ to‘g‘ri chiziqlarning **perpendikulyarlik** sharti	$a1\cdot a2+b1\cdot b2=0$
Oxy tekisligida $a1\cdot x+b1\cdot y+c1=0$ va $a2\cdot x+b2\cdot y+c2=0$ to‘g‘ri chiziqlarning **perpendikulyarlik** sharti	$\begin{vmatrix} a1 & -b1 \\ b2 & a2 \end{vmatrix}=0$
Oxy tekisligida $y=k1\cdot x+b1$ va $y=k2\cdot x+b2$ to‘g‘ri chiziqlarning **parallellik** sharti	$k1=k2$
Oxy tekisligida $a1\cdot x+b1\cdot y+c1=0$ va $a2\cdot x+b2\cdot y+c2=0$ to‘g‘ri chiziqlarning **parallellik** sharti	$\frac{a1}{a2}=\frac{b1}{b2}$
Oxy tekisligida $a1\cdot x+b1\cdot y+c1=0$ va $a2\cdot x+b2\cdot y+c2=0$ to‘g‘ri chiziqlarning **parallellik** sharti	$\begin{vmatrix} a1 & b1 \\ a2 & b2 \end{vmatrix}=0$
Oxy tekisligida parallel bo‘lmagan $a1\cdot x+b1\cdot y+c1=0$ va $a2\cdot x+b2\cdot y+c2=0$ to‘g‘ri chiziqlarning kesishish nuqtasi **M(x0; y0)** koordinatalari	$x0=\frac{\begin{vmatrix} -c1 & b1 \\ -c2 & b2 \end{vmatrix}}{\begin{vmatrix} a1 & b1 \\ a2 & b2 \end{vmatrix}}$ $y0=\frac{\begin{vmatrix} a1 & -c1 \\ a2 & -c2 \end{vmatrix}}{\begin{vmatrix} a1 & b1 \\ a2 & b2 \end{vmatrix}}$
A(x0; y0) nuqtadan $a\cdot x+b\cdot y+c=0$ to‘g‘ri chiziqqacha bo‘lgan **dA** masofa	$dA=\frac{\lvert a\cdot x0+b\cdot y0+c\rvert}{\sqrt{a^2+b^2}}$

Quyidagi qiziqarli ayniyatlar turli usullar yordamida isbotlanadi.

1.	$1+2+3+\ldots+N\equiv\frac{N\cdot(N+1)}{2}$ **Isboti**. Arifmetik progressiyada birinchi hadi $a_1=1$ va N-hadi $a_N=N$ bo‘lsa, u holda arifmetik progressiyaning birinchi n ta hadi yig‘indisi formulasi yuqoridagi ayniyatni beradi.
2.	$1^2+2^2+3^2+\ldots+N^2\equiv\frac{N\cdot(N+1)\cdot(2\cdot N+1)}{6}$
3.	$1^3+2^3+3^3+\ldots+N^3=(1+2+3+\ldots+N)^2\equiv\left(\frac{N\cdot(N+1)}{2}\right)^2$

4. $$1^5+2^5+3^5+\ldots+N^5 \equiv \frac{N^2\cdot(N+1)^2\cdot(2\cdot N^2+2\cdot N-1)}{12}$$

5. $$1+3+5+\ldots+(2\cdot N-1) \equiv N^2$$
Isboti. Matematik induksiya usuli yordamida isbotlaymiz.
1-qadam. N=1 da tenglik o'rinli: $1^2 \equiv 1^2$.
2-qadam. N=k da tenglik o'rinli bo'lsin: $1+3+5+\ldots+(2\cdot K-1) \equiv K^2$.
3-qadam. N=k+1 da quyidagi tenglik o'rinli bo'lishini isbotlaymiz:
$$1+3+5+\ldots+(2\cdot(K+1)-1) \equiv (K+1)^2.$$
Bunda 2-qadam natijasidan foydalanamiz:
$1+3+5+\ldots+(2\cdot K-1)+(2\cdot(K+1)-1) = K^2+(2\cdot(K+1)-1) =$
$= K^2+2\cdot K+2-1 = K^2+2\cdot K+1 = (K+1)^2$.
Demak, ayniyat o'rinli.

6. $$1^2+3^2+5^2+\ldots+(2\cdot N-1)^2 \equiv \frac{N\cdot(2\cdot N-1)\cdot(2\cdot N+1)}{3}$$

7. $$1^3+3^3+5^3+\ldots+(2\cdot N-1)^3 \equiv N^2\cdot(2\cdot N^2-1)$$

8. $$1\cdot 2+2\cdot 3+3\cdot 4+\ldots+N\cdot(N+1) \equiv \frac{1}{3}N\cdot(N+1)\cdot(N+2)$$

9. $$1\cdot 2\cdot 3+2\cdot 3\cdot 4+\ldots+N\cdot(N+1)\cdot(N+2) \equiv \frac{1}{4}N\cdot(N+1)\cdot(N+2)\cdot(N+3)$$

10. $$\frac{1}{A\cdot(A+k)}+\frac{1}{(A+k)\cdot(A+2\cdot k)}+\cdots+\frac{1}{(A+(N-1)\cdot k)\cdot(A+N\cdot k)} \equiv$$
$$\equiv \frac{N}{A\cdot(A+k\cdot N)}$$
Isboti.
$\frac{1}{A\cdot(A+k)} = \frac{1}{k}\left(\frac{1}{A}-\frac{1}{A+k}\right)$,
$\frac{1}{(A+k)\cdot(A+2\cdot k)} = \frac{1}{k}\left(\frac{1}{A+k}-\frac{1}{A+2\cdot k}\right)$,
…
$\frac{1}{(A+(N-1)\cdot k)\cdot(A+N\cdot k)} = \frac{1}{k}\left(\frac{1}{A+(N-1)\cdot k}-\frac{1}{A+N\cdot k}\right)$.
Demak:
$$\frac{1}{A\cdot(A+k)}+\frac{1}{(A+k)\cdot(A+2\cdot k)}+\cdots+\frac{1}{(A+(N-1)\cdot k)\cdot(A+N\cdot k)} =$$
$$=\frac{1}{k}\left(\frac{1}{A}-\frac{1}{A+k}\right)+\frac{1}{k}\left(\frac{1}{A+k}-\frac{1}{A+2\cdot k}\right)+\ldots+\frac{1}{k}\left(\frac{1}{A+(N-1)\cdot k}-\frac{1}{A+N\cdot k}\right)=$$
$$=\frac{1}{k}\left(\frac{1}{A}-\frac{1}{A+N\cdot k}\right)=\frac{1}{k}\left(\frac{A+N\cdot k-A}{A\cdot(A+N\cdot k)}\right)=\frac{1}{k}\cdot\frac{N\cdot k}{A\cdot(A+N\cdot k)}=\frac{N}{A\cdot(A+N\cdot k)}$$

11. $$\frac{1^2}{1\cdot 3}+\frac{2^2}{3\cdot 5}+\frac{3^2}{5\cdot 7}+\ldots+\frac{N^2}{(2\cdot N-1)\cdot(2\cdot N+1)} \equiv \frac{N\cdot(N+1)}{2\cdot(2\cdot N+1)}$$

12. $$\left(1-\frac{1}{4}\right)\cdot\left(1-\frac{1}{9}\right)\cdot\left(1-\frac{1}{16}\right)\cdot\ldots\cdot\left(1-\frac{1}{N^2}\right) \equiv \frac{N+1}{2\cdot N}$$

13. $$\frac{1}{Ln2\cdot Ln4}+\frac{1}{Ln4\cdot Ln8}+\frac{1}{Ln8\cdot Ln16}+\ldots+\frac{1}{Ln2^{N-1}\cdot Ln2^N} \equiv \left(1-\frac{1}{N}\right)\cdot\frac{1}{Ln^2 2}$$

14.	$\sqrt{2+\sqrt{2+\sqrt{2+\ldots+\sqrt{2}}}} \equiv 2\cdot\cos\frac{\pi}{2^{N+1}}$ (N ta ildiz)
15.	$\sin\frac{\pi}{3}+\sin\frac{2\cdot\pi}{3}+\ldots+\sin\frac{N\cdot\pi}{3} \equiv 2\cdot\sin\frac{N\cdot\pi}{6}\cdot\sin\frac{(N+1)\cdot\pi}{3}$
16.	$\operatorname{arctg}\frac{1}{2}+\operatorname{arctg}\frac{1}{8}+\ldots+\operatorname{arctg}\frac{1}{2\cdot N^2} \equiv \operatorname{arctg}\frac{N}{N+1}$

1-§. C++ TILIDA MAKROS JOYLASHTIRISH

PREPROTSESSORNING #define DIREKTIVASI

Preprotsessorning #define direktivasi identifikator nomini va dasturda shu identifikator uchraganda uning o'rniga almashtiriladigan belgilar ketma-ketligini aniqlaydi. Shu nuqtayi nazardan identifikator **makros** nomi deb, belgilar bilan almashtirish jarayoni **makrosni joylashtirish** deb ataladi. Bu direktivaning ko'rinishi quyidagicha:

```
#define makros_nomi belgilar_ketma-ketligi
```

Bu yerda nuqtali vergul qo'yilmaydi, identifikator va belgilar ketma-ketligi orasida ixtiyoriy sondagi orachiq bo'lishi mumkin. Makros yozilishi yangi satrga o'tish bilangina tugaydi. Agar makros yozilishi keyingi satrga ham o'tadigan bo'lsa, u holda birinchi satr oxirida \ belgisi yoziladi. Masalan:

```
#define Bir 1
#define Pi 3.1415
#define E 2.71
#define Nom "Kitob"
```

Avval aniqlangan makros keyingi makroslarni aniqlashda ishlatilishi mumkin. Masalan:

```
#define BIR 1
#define IKKI BIR + BIR
#define UCH BIR + IKKI
```

Yuqoridagi misollar shuni ko'rsatadiki, makroslarni bunday qo'llash o'zgarmaslarni tavsiflash bilan bir xil ma'no berar ekan.

Lekin keyingi qo'llash usuli umuman boshqa imkoniyatlarni ochib beradi. Bu imkoniyat makroslarga argument kiritib funksiya kabi qo'llashdir. Masalan:

Dastur	Natijasi

`#include <iostream>`
`#define Disk(k1,k2,k3) k2*k2-4*k1*k3`
`using namespace std;`
`int main(){`
	Diskriminant qiymati:
	-12
	0
	8

```
  int a=1, b=2, c=4;
  cout<<"Diskriminant qiymati: "<<endl;
  cout<<Disk(a,b,c)<<endl;
  cout<<Disk(a,b,a)<<endl;
  cout<<Disk(2,4,1)<<endl;
  return 0;
}
```

Dastur kompilyatsiyasi vaqtida Disk(k1,k2,k3) o‘rniga makrosga mos ifoda formal k1,k2,k3 argumentlar o‘rniga aniq argumentlar bilan joylashtiriladi, ya’ni:

```
cout<<b*b-4*a*c<<endl;
cout<<b*b-4*a*a<<endl;
cout<<4*4-4*2*1<<endl;
```

Ba’zan **#define** orqali funksiya hosil qilish oddiy joylashtirish bajarilayotganligi uchun xatolikka ham sabab bo‘lishi mumkin. Masalan, yuqoridagi dasturda

```
cout<<Disk(1+1,4,1)<<endl;
```

kabi yozilsa, u holda joylashtirish natijasida quyidagicha xatolik hosil bo‘ladi:

```
cout<<4*4-4*1+1*1<<endl;
```

PREPROTSESSORNING # undef DIREKTIVASI

Preprotsessorning #undef direktivasi makrosni qayta aniqlash imkoniyatini beradi.

```
#undef makros_nomi
```

Bu yerda ham nuqtali vergul qo‘yilmaydi. Bu direktiva ishlagandan so‘ng identifikator unga bog‘langan belgilar ketma-ketligidan ozod bo‘ladi. Endi uni boshqa makros nomi sifatida ishlatish mumkin. Masalan:

Dastur	Natijasi

`#include <iostream>`
`#define Satr "Tong"`
`using namespace std;`
`int main(){`
`  cout<<Satr<<endl;`
`  #undef Satr`
`  #define Satr "Peshin"`
`  cout<<Satr<<endl;`
`  #undef Satr`
`  #define Satr "Shom"`
`  cout<<Satr<<endl;`
`  return 0;`
`}`
	Tong
	Peshin
	Shom

2-§. C++ TILIDA MATEMATIK FUNKSIYALAR

C++ dasturlash tilining **cmath (math.h)** kutubxonasi o'zida ko'p qo'llanadigan matematik funksiyalarni saqlaydi. Bu funksiyalarni qo'llash uchun **cmath** yoki **math.h** kutubxonalaridan birini dasturga qo'shish kerak.

ABSOLYUT QIYMAT, BUTUN QISM, YAXLITLASH VA QOLDIQ HISOBLASH

Eng sodda va ko'p masalalarda qo'llanadigan funksiya bu sonning absolyut qiymatini hisoblash funksiyasi bo'lib, C++ tilida quyidagi ko'rinishlarda qo'llanadi:

`float fabs(float x)`
`double fabs(double x)`
`long double fabs(long double x)`
	x sonining absolyut qiymatini qaytaradi

`int abs(int x)`
`long int abs(long int x)`
`long long int abs(long long int x)`
	x sonining absolyut qiymatini qaytaradi (bu funksiya stdlib.h kutubxonasida ham bor bo'lib, bu kutubxonadagi funksiya faqat int turi uchundir)

Absolyut qiymatni hisoblash funksiyalari ishini quyidagi dastur orqali ko'rsatamiz:

Dastur	Natijasi

`#include <iostream>`
`#include <cmath>`
`using namespace std;`
`int main(){`
`  cout<<"abs va fabs funksiyalari: "<<endl;`
`  cout<<"abs(-3.4) = "<< abs(-3.4)<<endl;`
`  cout<<"abs(3.4) = "<< abs(3.4)<<endl<<endl;`
`  cout<<"fabs(-3.4) = "<<fabs(-3.4)<<endl;`
`  cout<<"fabs(3.4) = "<<fabs(3.4)<<endl;`
`  return 0;`
`}`
	abs va fabs funksiyalari:
	abs(-3.4) = 3.4
	abs(3.4) = 3.4
	
	fabs(-3.4) = 3.4
	fabs(3.4) = 3.4

Sonning butun qismini ajratish va yaxlitlash funksiyalari ko'p qo'llanadigan funksiyalardan bo'lib, ularga quyidagilarni misol qilish mumkin:

`float trunc(float x)`
`double trunc(double x)`
`long double trunc(long double x)`
	x sonining kasr qismini tashlab yuboradi, ya'ni: trunc(–1.3) = –1; trunc(2.3) = 2; trunc(2.8) = 2

`float round(float x)`
`double round(double x)`
`long double round(long double x)`
	x sonini matematik eng yaqin butun songacha yaxlitlaydi: round(2.49) = 2; round(2.5) = 3; round(–1.49) = –1; round(–1.5) = –2

`float ceil(float x)`
`double ceil(double x)`
`long double ceil(long double x)`
	x sonini yuqoriga yaxlitlaydi, ya'ni **ceil** x dan kichik bo'lmagan eng kichik butun sonni javob sifatida qaytaradi, ya'ni [x]

`float floor(float x)`
`double floor(double x)`
`long double floor(long double x)`
	x sonini quyiga yaxlitlaydi, ya'ni **floor** x dan katta bo'lmagan eng katta butun sonni javob sifatida qaytaradi, ya'ni [x]

Quyidagi dastur orqali **trunc** va **round**, **ceil**, **floor** funksiyalari ishini taqqoslaymiz:

Dastur

```
#include <iostream>
#include <cmath>
using namespace std;
int main(){
 cout<<"|x soni"<<"|ceil"<<"|floor";
 cout<<"|round"<<"|trunc|"<<endl;
 cout<<"  "<<2.5<<"   "<<ceil(2.5)<<"   "<<floor(2.5);
 cout<<"  "<<round(2.5)<<"    "<<trunc(2.5)<<endl;
 cout<<"  "<<1.3<<"   "<<ceil(1.3)<<"   "<<floor(1.3);
 cout<<"  "<<round(1.3)<<"    "<<trunc(1.3)<<endl;
 cout<<"  "<<0.8<<"   "<<ceil(0.8)<<"   "<<floor(0.8);
 cout<<"  "<<round(0.8)<<"    "<<trunc(0.8)<<endl;
 cout<<"  "<<-2.4<<"  "<<ceil(-2.4)<<"  "<<floor(-2.4);
 cout<<"  "<<round(-2.4)<<"   "<<trunc(-2.4)<<endl;
 cout<<"  "<<-4.5<<"  "<<ceil(-4.5)<<"  "<<floor(-4.5);
 cout<<"  "<<round(-4.5)<<"   "<<trunc(-4.5)<<endl;
 cout<<"  "<<-1.9<<"  "<<ceil(-1.9)<<"  "<<floor(-1.9);
 cout<<"   "<<round(-1.9)<<"   "<<trunc(-1.9)<<endl;
 return 0;
}
```

Natijasi

```
|x soni|ceil|floor|round|trunc|
  2.5    3     2     3     2
  1.3    2     1     1     1
  0.8    1     0     1     0
 -2.4   -2    -3    -2    -2
 -4.5   -4    -5    -5    -4
 -1.9   -1    -2    -2    -1
```

Sonni yaxlitlash funksiyalariga quyidagilarni ham misol qilish mumkin:

`long int lround(float x)`
`long int lround(double x)`
`long int lround(long double x)`
	x sonini eng yaqin butun songa yaxlitlaydi (**round** funksiyasi kabi) va natijani **long int** turida qaytaradi

`long long int llround(float x)`
`long long int llround(double x)`
`long long int llround(long double x)`
	x sonini eng yaqin butun songa yaxlitlaydi (**round** funksiyasi kabi) va natijani **long long int** turida qaytaradi

Quyidagi dastur natijasidan **lround** va **llround** funksiyalarining farqini bilib olish mumkin. Dasturda **lround** funksiyasi qiymati **int** turining chegarasiga sig'maganligi sababli -2147483648 qiymatini qaytaradi, **llround** funksiyasi esa to'g'ri javob qaytaradi (**10000000000** soni **long long int** chegarasiga sig'adi).

Dastur	Natijasi

`#include <iostream>`
`#include <cmath>`
`using namespace std;`
`int main(){`
`  cout<<"lround(3.4)="<<lround(3.4)<<endl;`
`  cout<<"llround(3.4)="<<llround(3.4)<<endl;`
`  cout<<"lround(-7.8)="<<lround(-7.8)<<endl;`
`  cout<<"llround(-7.8)="<<llround(-7.8)<<endl;`
`  cout<<"lround(10000000000.23)="<<endl;`
`  cout<<lround(10000000000.23)<<endl;`
`  cout<<"llround(10000000000.23)="<<endl;`
`  cout<<llround(10000000000.23)<<endl;`
`  return 0;`
`}`
	lround(3.4)=3
	llround(3.4)=3
	lround(-7.8)=-8
	llround(-7.8)=-8
	lround(10000000000.23)=
	-2147483648
	llround(10000000000.23)=
	10000000000

Quyidagi funksiyalar bo‘lishda qoldiq hisoblash uchun qo‘llaniladi:

`float fmod(float a, float b)`
`double fmod(double a, double b)`
`long double fmod(long double a, long double b)`
	a sonni b songa bo‘lgandagi qoldiqni hisoblaydi, bu yerda qoldiq quyidagicha hisoblanadi:
	qoldiq = a – q * b; (q = trunc(a/b))

`float remainder(float a, float b)`
`double remainder(double a, double b)`
`long double remainder(long double a, long double b)`
	a sonni b songa bo‘lgandagi qoldiqni hisoblaydi, bu yerda qoldiq quyidagicha hisoblanadi:
	qoldiq = a – q * b; (q = round(a/b))

Quyidagi dastur **fmod** va **remainder** funksiyalari ishini farqlab beradi:

Dastur	Natijasi

`#include <iostream>`
`#include <cmath>`
`using namespace std;`
`int main(){`
` const float a=5, b=2;`
` cout<<"fmod funksiyasi: "<<endl;`
` cout<<"fmod(5.1,2.0)="<<fmod(a+0.1,b)<<endl;`
` cout<<"fmod(5.0,2.0)="<<fmod(a,b)<<endl;`
` cout<<"fmod(4.9,2.0)="<<fmod(a-0.1,b)<<endl;`
` cout <<endl<<"remainder funksiyasi:"<<endl;`
` cout<<"remainder(5.1,2.0)="<<remainder(a+0.1,b)<<endl;`
` cout<<"remainder(5.0,2.0)="<<remainder(a,b)<<endl;`
` cout<<"remainder(4.9,2.0)="<<remainder(a-0.1,b)<<endl;`
` return 0;`
`}`
	fmod funksiyasi:
	fmod(5.1,2.0)=1.1
	fmod(5.0,2.0)=1
	fmod(4.9,2.0)=0.9
	
	remainder funksiyasi:
	remainder(5.1,2.0)=-0.9
	remainder(5.0,2.0)=1
	remainder(4.9,2.0)=0.9

Ma’lumki, a sonni b songa bo‘lgandagi qoldiq q=a–k·b bo‘lib, bunda k=[a/b] ga tengdir. Demak, funksiyalardan **fmod** xuddi shu qoldiqni hisoblaydi, **remainder** funksiyasi esa q

qoldiq sifatida a–k·b va a–(k+1)·b sonlardan absolyut qiymati bo'yicha kichigiga mos ifodani hisoblaydi.

TRIGONOMETRIK FUNKSIYALAR

C++ tili dasturda trigonometrik funksiyalar va teskari trigonometrik funksiyalarni qo'llash uchun ko'p qulayliklar beradi.

`float sin(float x)`
`double sin(double x)`
`long double sin(long double x)`
	x burchak (radianda) sinusini qaytaradi

`float cos(float x)`
`double cos(double x)`
`long double cos(long double x)`
	x burchak (radianda) kosinusini qaytaradi

`float tan(float x)`
`double tan(double x)`
`long double tan(long double x)`
	x burchak (radianda) tangensini qaytaradi

`float acos(float x)`
`double acos(double x)`
`long double acos(long double x)`
	x son arkkosinusini (radianda) [0, π] oraliqda qaytaradi. Bunda x soni [–1; 1] oraliqqa tegishli bo'lishi shart

`float asin(float x)`
`double asin(double x)`
`long double asin(long double x)`
	x son arksinusini (radianda) $[-\frac{\pi}{2}, \frac{\pi}{2}]$ oraliqda qaytaradi. Bunda x soni [-1; 1] oraliqqa tegishli bo'lishi shart

`float atan(float x)`
`double atan(double x)`
`long double atan(long double x)`
	x son arktangensini (radianda) $[-\frac{\pi}{2}, \frac{\pi}{2}]$ oraliqda qaytaradi

Trigonometrik funksiyalardan foydalanishga misol tariqasida ushbu dasturni keltiramiz:

Dastur	Natijasi

`#include <iostream>`
`#include <cmath>`
`using namespace std;`
`int main(){`
`  double pi = 3.141592653589;`
`  double pi2=pi/2.0, pi4=pi/ 4.0;`
`  cout<<"sin(pi/2)="<<sin(pi2)<<endl;`
`  cout<<"cos(pi)="<<cos(pi)<<endl;`
`  cout<<"tan(pi/4)="<<tan(pi4)<<endl;`
`  cout<<"acos(1)="<<acos(1.0)<<endl;`
`  cout<<"asin(1)="<<asin(1.0)<<endl;`
`  cout<<"atan(1)="<<atan(1.0)<<endl;`
`  return 0;`
`}`
	sin(pi/2)=1
	cos(pi)=-1
	tan(pi/4)=1
	acos(1)=0
	asin(1)=1.5708
	atan(1)=0.785398

EKSPONENSIAL VA LOGARIFMIK FUNKSIYALAR

Ma'lumki, eksponensial va natural logarifm funksiyalari bir-biriga teskari funksiyalardir. C++ tilida bu funksiyalar bilan bir qatorda o'nlik va ikkilik logarifm hamda ko'rsatkichli funksiyalarni ham hisoblash imkoniyati bor.

`float exp(float x)`
`double exp(double x)`
`long double exp(long double x)`
	x sonining eksponentasi qiymatini qaytaradi, ya'ni exp(x) = e^x

`float log(float x)`
`double log(double x)`
`long double log(long double x)`
	Musbat x sonining natural logarifmi qiymatini qaytaradi, ya'ni log(x) = *ln x*

`float log10(float x)`
`double log10(double x)`
`long double log10(long double x)`
	Musbat x sonining 10 asosli logarifmi qiymatini qaytaradi, ya'ni log10(x) = $log_{10}{}^x$

`float log2(float x)`
`double log2(double x)`
`long double log2(long double x)`
	Musbat x sonining 2 asosli logarifmi qiymatini qaytaradi, ya'ni log2(x) = $log_2{}^x$

`float exp2(float x)`
`double exp2(double x)`
`long double exp2(long double x)`
	x sonining 2 asosli eksponentasini qiymatini qaytaradi, ya'ni exp2(x) = 2^x

`float expm1(float x)`
`double expm1(double x)`
`long double expm1(long double x)`
	x soni uchun $e^x - 1$ qiymatini qaytaradi

`float log1p(float x)`
`double log1p(double x)`
`long double log1p(long double x)`
	x soni uchun *ln* $(x+1)$ qiymatini qaytaradi

Eksponensial va logarifmik funksiyalarni dasturda qo'llashga misol:

Dastur	Natijasi

`#include <iostream>`
`#include <cmath>`
`using namespace std;`
`int main(){`
`  double eksp = 1.0, eksp10 = 1000.0;`
`  double eksp2 = 1024.0;`
`  cout<<"exp("<<eksp<<")="<<exp(eksp)<<endl;`
`  cout<<"log("<<eksp<<")="<<log(eksp)<<endl;`
`  cout<<"log10("<<eksp10<<")="<<log10(eksp10)<<endl;`
`  cout<<"exp2("<<eksp<<")="<<exp2(eksp)<<endl;`
`  cout<<"expm1("<<eksp<<")="<<expm1(eksp)<<endl;`
`  cout<<"log1p("<<eksp<<")="<<log1p(eksp)<<endl;`
`  cout<<"log2("<<eksp2<<")="<<log2(eksp2)<<endl;`
`    return 0;`
`}`
	exp(1)=2.71828
	log(1)=0
	log10(1000)=3
	exp2(1)=2
	expm1(1)=1.71828
	log1p(1)=0.693147
	log2(1024)=10

SONNING DARAJASINI HISOBLASH FUNKSIYALARI

Sonni darajaga ko'tarish yoki ildiz chiqarish uchun C++ tilida maxsus funksiyalar kiritilgan.

`float pow(float a, float x)`
`double pow(double a, double x)`
`long double pow(long double a, long double x)`
	a va x sonlari uchun a^x qiymatini qaytaradi

`float sqrt(float x)`
`double sqrt(double x)`
`long double sqrt(long double x)`
	Nomanfiy x sonining kvadrat ildizi qiymatini qaytaradi

`float cbrt(float x)`
`double cbrt(double x)`
`long double cbrt(long double x)`
	x sonining kub ildizi qiymatini qaytaradi

`float hypot(float a, float b)`
`double hypot(double a, double b)`
`long double hypot(long double a, long double b)`
	a va b katetli to'g'ri burchakli uchburchak gipotenuzasi qiymatini qaytaradi

Bu funksiyalar beradigan qulayliklarni quyidagi dastur orqali ko'rish mumkin:

Dastur

```
#include <iostream>
#include <cmath>
using namespace std;
int main(){
  cout<<"pow(2.0,10.0)="<<pow(2.0,10.0)<<endl;
  cout<<"sqrt(4.0)="<<sqrt(4.0)<<endl;
  cout<<"cbrt(8.0)="<<cbrt(8.0)<<endl;
  cout<<"hypot(3.0,4.0)="<<hypot(3.0,4.0)<<endl;
  cout<<"pow yordamida sqrt va cbrt ni hisoblash:"<<endl;
  cout<<"pow(4.0,0.5)="<<pow(4.0,0.5)<<endl;
  cout<<"pow(8.0,1/3)="<<pow(8.0,1.0/3.0)<<endl;
  cout<<"pow va sqrt yordamida hypot ni hisoblash:"<<endl;
  cout<<"hypot(3.0,4.0)=sqrt(3^2+4^2)=";
  cout<<sqrt(pow(3.0,2.0)+pow(4.0, 2.0))<<endl;
  return 0;
}
```

Natijasi

```
pow(2.0,10.0)=1024
sqrt(4.0)=2
cbrt(8.0)=2
```

```
hypot(3.0,4.0)=5
pow yordamida sqrt va cbrt ni hisoblash:
pow(4.0,0.5)=2
pow(8.0,1/3)=2
pow va sqrt yordamida hypot ni hisoblash:
hypot(3.0,4.0)=sqrt(3^2+4^2)=5
```

C++ tilining hisoblashlarni osonlashtiruvchi yana ko‘pgina qiziqarli funksiyalari bor bo‘lib, misol sifatida ulardan birini keltiramiz:

`float fma(float a, float b, float c)`
`double fma(double a, double b, double c)`
`long double fma(long double a, long double b, long double c)`
	a, b, c sonlari uchun **a*b+c** ifoda qiymatini qaytaradi. Bu usulda oraliq hisoblashlar jarayonida kasr qismni yo‘qotish holatlariga yo‘l qo‘yilmaydi

Quyidagi dastur shu funksiyani qo‘llashni ifodalaydi:

Dastur	Natijasi

`#include <iostream>`
`#include <cmath>`
`using namespace std;`
`int main(){`
`  cout<<"fma funksiyasi:"<<endl;`
`  cout<<"fma(1,2,3)="<<fma(1,2,3)<<endl;`
`  return 0;`
`}`
	fma funksiyasi:
	fma(1,2,3)=5

MAXSUS QIYMATLARDAN FOYDALANISH IMKONIYATI

Ba’zi dasturlash tillarida bo‘lgani kabi ko‘p qo‘llaniladigan matematik o‘zgarmaslar C++ tilining global yoki standart nomlar fazosida e’lon qilinmagan. Faqatgina quyidagi usul bilan ba’zi bir o‘zgarmaslardan foydalana olishimiz mumkin:

1) Dastur boshida `#define _USE_MATH_DEFINES` direktivasini yozish va

2) dasturga cmath kutubxonasini qo‘shish kerak.

Shundan so‘ng quyidagi jadvalda ko‘rsatilgan o‘zgarmaslardan foydalanish mumkin bo‘ladi:

Matematik ko‘rinishi	C++ tilidagi o‘zgarmas nomi	Qiymati (taqribiy)
π	`M_PI`	`3.14159265358979323846`
π/2	`M_PI_2`	`1.57079632679489661923`
π/4	`M_PI_4`	`0.78539816339744830962`

$1/\pi$	M_1_PI	0.31830988618379067153 8
$2/\pi$	M_2_PI	0.63661977236758134307 6
$2/\sqrt{\pi}$	M_2_SQRTPI	1.12837916709551257390
$\sqrt{2}$	M_SQRT2	1.41421356237309504880
$1/\sqrt{2}$	M_SQRT1_2	0.70710678118654752440 1
e	M_E	2.71828182845904523536
$Log_2(e)$	M_LOG2E	1.44269504088896340736
$Log_{10}(e)$	M_LOG10E	0.43429448190325182765 1
$Log_e(2)$	M_LN2	0.69314718055994530941 7
$Log_e(10)$	M_LN10	2.30258509299404568402

Bu usul C++ kompilyatorlarining barchasida ham ishlayvermaydi. GNU GCC kompilyatorida -std=c++11 yoki shu kabi parametrlarni o‘chirib qo‘yish kerak bo‘ladi.

3-§. CHIZIQLI ALGORITMLI MASALALAR

Berilgan misol va masalalar chiziqli algoritmli dastur (qisqacha chiziqli dastur) tuzishga mo‘ljallangan. Ya’ni berilgan vazifalardagi ifoda yoki funksiyalar kiritilayotgan qiymatlar uchun aniqlangan va shuning uchun ham biror-bir qo‘shimcha shartsiz hisoblash mumkin.

= A =

Chiziqli 1. Berilgan a=10, b=20 va c=30 natural sonlar uchun xotirani tejagan holda quyidagi ifodalarni hisoblang:

a) $x = \frac{a}{b+c}$ CH: 0.2	b) $y = \frac{a-b}{a+c}$ CH: –0.25	c) $z = \frac{a^2}{b^2+c}$ CH: 0.232558	d) $w = \frac{a \cdot b \cdot c}{a^2+2 \cdot b^2+3 \cdot c^2}$ CH: 1.66667

Yechim a). Bu misol juda sodda bo‘lgani bilan ba’zan xato natijalar olinadi. Shu kabi misollarda yo‘l qo‘yiladigan eng ko‘p kamchilik – bu o‘zgaruvchi turini tanlashdagi xatolikdir.

Masala shartida berilgan a, b va c sonlarning natural, ya’ni musbat hamda butun son, ikki xonali bo‘lganligi uchun bu o‘zgaruvchilarga xotiradan (eng kam) 1 bayt joy egallaydigan **char** turini tanlash mumkin. O‘zgaruvchi x esa nisbatning, ya’ni haqiqiy sonni, qiymatini o‘zlashtirgani uchun bu o‘zgaruvchiga xotiradan (eng kam) 4 bayt joy egallaydigan **float** turning tanlanishi tabiiy. Demak, barcha o‘zgaruvchilar egallaydigan xotira umumiy hisoblanganda 7 baytga teng bo‘ladi.

Ko‘pchilik, odatda, dasturni quyidagi ko‘rinishda tashkil etadi va xato natija oladi:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  char a=10,b=20,c=30;`
`  float x;`
`  x=a/(b+c);`
`  cout<<x<<endl;`
`  return 0;`
`}`
	0

Lekin natija sifatida 0,2 chiqishi kerak edi-ku. Xatolik nimada?

Xatolik bor-yo‘g‘i C++ tili butun tur sonlari uchun / amalida butun bo‘lishni bajarishi e’tiborga olinmaganidadir. Bu kamchilikni tuzatishning turli usullari bo‘lib, ularni tahlil etamiz:

1-usul. Barcha o‘zgaruvchilarni **float** turda, ya’ni `float x,a=10,b=20,c=30;` kabi tavsiflash mumkin. Lekin bu holda xotiradan 16 bayt joy egallanadi;

2-usul. Yuqoridagi dasturdagi nisbatning suratida **a*1.0** kabi yozish. Bu holda bo‘lish amali haqiqiy turdagi kabi bajariladi;

3-usul. Yuqoridagi dasturdagi nisbatning suratida **(float) a** kabi yozish. Bu holda ham bo‘lish amali haqiqiy turdagi kabi bajariladi

Chiziqli 2. Berilgan a, b va c natural sonlar uchun quyidagi ifodalarni hisoblang:

a) $x=\frac{\sqrt{a+b}}{c}$	b) $y=\frac{c}{\sqrt[3]{a+b}}$	c) $z=\frac{\sqrt{a+1}-\sqrt[3]{b-2}}{2\cdot\sqrt[4]{c+7}}$	d) $w=a+\frac{\sqrt{1+c}}{\sqrt[3]{b+9}}$
K1: 1 2 3 CH1: 0.57735 K2: 45 55 100 CH2: 0.1	K1: 10 20 30 CH1: 9.65489 K2: 100 101 202 CH2: 34.4841	K1: 3 5 7 CH1: 0.144171 K2: 41 53 79 CH2: 0.455184	K1: 11 13 17 CH1: 12.5141 K2: 5 18 99 CH2: 8.33333

Chiziqli 3. Berilgan a, b va c natural sonlar uchun quyidagi ifodalar qiymatini 10^{-3} aniqlikda hisoblang:

a) $x=\frac{1}{c}+\frac{b}{a}$	b) $y=\frac{a\cdot c+b\cdot a}{a\cdot b\cdot c}$	c) $z=\frac{b^3+a^4}{21+c^2}$	d) $w=a^3+\frac{63-a}{19+b\cdot c}$
K1: 7 4 3 CH1: 0.905 K2: 17 151 13 CH2: 8.959	K1: 1 1 1 CH1: 2.000 K2: 100 77 1 CH2: 1.013	K1: 3 4 5 CH1: 3.152 K2: 5 5 5 CH2: 16.304	K1: 11 21 31 CH1: 1331.078 K2: 5 1 2 CH2: 127.762

Chiziqli 4. Berilgan A va B sonlarning o‘rta arifmetigini hisoblang.

Chiziqli 5. Berilgan A va B sonlar kublarining o‘rta arifmetigini hisoblang.

Chiziqli 6. Berilgan A, B, C, D va E sonlarning o‘rta arifmetigini hisoblang.

Chiziqli 7. Berilgan A va B sonlar modullarining o'rta geometrigini hisoblang.

Chiziqli 8. Berilgan A, B, C va D sonlar kvadratlarining o'rta geometrigini hisoblang.

Chiziqli 9. Bir tomoni A va perimetri P berilgan to'g'ri to'rtburchakning ikkinchi tomonini toping.

Chiziqli 10. Tomoni A va perimetri P bo'lgan to'g'ri to'rtburchakning yuzini toping.

Chiziqli 11. Tomoni A va perimetri P bo'lgan to'g'ri to'rtburchak diagonalini toping.

Chiziqli 12. Perimetri P bo'lgan kvadratning yuzini toping.

Chiziqli 13. Perimetri P bo'lgan kvadratning diagonalini toping.

Chiziqli 14. Tomoni A, B va C bo'lgan uchburchakning yuzini hisoblang.

Chiziqli 15. Katetlari A va B bo'lgan uchburchakning yuzini hisoblang.

Chiziqli 16. Uchburchakning o'rta chiziqlari hosil qilgan kichik uchburchak perimetri A va yuzasi B berilgan. Katta uchburchakning perimetri va yuzini toping.

Chiziqli 17. Uchburchakka tashqi chizilgan doira yuzi S va uchburchakning perimetri P berilgan bo'lsa, uchburchakning yuzini hisoblang.

Chiziqli 18. Teng tomonli uchburchakka ichki chizilgan doira yuzi S berilgan bo'lsa, uchburchak medianasi va balandligini toping.

Chiziqli 19. Parallelogrammning tomonlari A va B bo'lsa, uning bitta diagonali orqali bo'linishidan hosil bo'lgan uchburchaklar yuzalarini toping.

Chiziqli 20. Doiraning yuzi S berilgan. Uni chegaralab turgan aylana uzunligini toping.

Chiziqli 21. Markazlari bir nuqtada bo'lgan R1 va R2 radiusli doiralar hosil qilgan halqa yuzini hisoblang.

Chiziqli 22. Parallelepipedning a, b va c qirralari berilgan bo'lsa, uning hajmining yarmini va to'liq sirti yuzining choragini aniqlang.

Chiziqli 23. Ox o'qida A(x1), B(x2) va C(x3) nuqtalar berilgan. AB va BC kesmalar uzunliklari yig'indisini aniqlang.

Chiziqli 24. Noldan farqli a son va b son berilgan. $ax+b=0$ tenglamani yeching.

Chiziqli 25. Berilgan a, b va c sonlar $b^2-4ac=0$ shartni qanoatlantiradi. $ax^2+bx+c=0$ tenglamani yeching.

Chiziqli 26. Berilgan a, b va c sonlar $b^2-4ac>0$ shartni qanoatlantiradi. $ax^2+bx+c=0$ tenglamani yeching.

Chiziqli 27. Tezligi V km/soat bo'lgan samolyotning T vaqtda uchib o'tgan S yo'lini aniqlang (yo'llanma: $S=V\cdot T$).

Chiziqli 28. T ($T>0$) soatda bosib o'tgan yo'li S km bo'lgan avtomobilning V tezligini aniqlang.

Chiziqli 29. T1 ($T1>0$) soatda tezligi V1 km/soat, T2 soatda tezligi V2 km/soat bo'lgan poyezdning o'rtacha V tezligini aniqlang (yo'llanma: $V=(V1\cdot T1+V2\cdot T2)/(T1+T2)$).

Chiziqli 30. Kompyuterning narxi A so‘m edi. Unga bo‘lgan talab oshgandan so‘ng uning narxi B % ga ko‘tarildi. Kompyuterning yangi bahosini aniqlang.

= B =

Chiziqli 31. Berilgan a, b va c natural sonlar uchun quyidagi ifodalar qiymatini 10^{-7} aniqlikda hisoblang:

a) $x = \frac{\sqrt{a}+\sqrt{b}}{\sqrt{c}+7}$	b) $y = \sqrt{\frac{1+a}{b+c}}$	c) $z = \frac{a^3}{\sqrt[4]{b^2+5\cdot c}}$	d) $w = \frac{-b-\sqrt{\lvert b^2-4\cdot a\cdot c\rvert}}{2\cdot a}$
K1: 1 1 1 CH1: 0.2500000 K2: 97 2 3 CH2: 1.2898540	K1: 2 2 2 CH1: 0.8660254 K2: 25 3 7 CH2: 1.6124516	K1: 3 2 1 CH1: 15.5884571 K2: 10 11 12 CH2: 272.6342163	K1: 1 2 1 CH1: –1.0000000 K2: 1 1 1 CH2: –1.3660254

Yechim d). Kiritilayotgan qiymatlar natural, ya’ni musbat son bo‘lgani uchun **unsigned int** turda tavsiflash yetarlidek va dastur quyidagicha bo‘lishi mumkindek ko‘rinadi:

Dastur	Natijasi

`#include <iostream>`
`#include <cmath>`
`#include <iomanip>`
`using namespace std;`
`int main(){`
`  unsigned int a, b, c;`
`  float w;`
`  cin>>a>>b>>c;`
`  w=(-b-pow(fabs(b*b-4*a*c),0.5))/(2*a);`
`  cout<<setprecision(7)<<fixed<<w;`
`  return 0;`
`}`
	1 2 1
	2147483648.0000000

Natija esa, –1.0000000 o‘rniga, ko‘rib turganingizdek, qandaydir tushunib bo‘lmaydigan songa teng bo‘ldi. Bu yerda xatolik **unsigned int** turida C++ tili manfiymas sonlarni xotirada ishora razryadisiz tasvirlaganida bo‘lib, oraliq hisoblashlarda ham ishora razryadining yo‘qligi bilan bog‘liqdir. Ya’ni C++ tili ifodani tavsiflangan miqdorlarning umumiy turi bo‘yicha hisoblaydi, so‘ng tegishli o‘zgaruvchi turiga mos o‘zlashtiradi. Agar yuqoridagi dasturda a, b va c o‘zgaruvchilar **int** turda tavsiflansa, hech qanday muammo paydo bo‘lmaydi.

Yana bir holatga e’tibor qaratish zarur. Agar dasturdagi **fixed** manipulyatori olib tashlansa, u holda natija –1.0000000 kabi emas, balki natijaning kasr qismi 0 bo‘lgani uchun –1 kabi chiqariladi. Bu esa misol shartiga mos emas.

Chiziqli 32. Berilgan ixtiyoriy a qiymat uchun quyidagi ifodalarni hisoblang:

a) $x = \frac{a^7 + 21}{\sqrt{a^2 + 1}}$	b) $y = \sqrt{\frac{\lvert 1 + a \rvert}{1 + \sqrt{\lvert a \rvert}}}$	c) $z = \frac{\sin a}{1 + \cos^2 a}$	d) $w = \frac{\sqrt{e^{a-2} + sin^2 a}}{2 + \lvert a + 1 \rvert}$
K1: –1 CH1: 14.1421 K2: 2.55 CH2: 263.63	K1: –7 CH1: 1.28287 K2: 2.1 CH2: 1.12506	K1: –0.1 CH1: –0.0333887 K2: 100 CH2: –0.184563	K1: 1 CH1: 0.25932 K2: –0.5 CH2: 0.223404

Chiziqli 33. Tomoni A va perimetri P bo‘lgan to‘g‘ri to‘rtburchakka tashqi chizilgan aylananing yuzini toping.

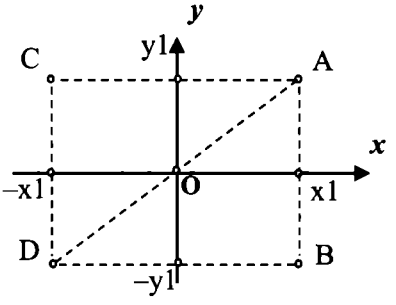
Chiziqli 34. Perimetri P bo‘lgan kvadratga tashqi chizilgan aylana yuzini va ichki chizilgan aylana uzunligini toping.

Chiziqli 35. Kvadratga ichki chizilgan doiraning yuzi S berilgan bo‘lsa, kvadratning yuzini toping.

Chiziqli 36. Kubning hajmi V berilgan. Kubning yon sirti yuzini aniqlang.

Chiziqli 37. Bankka yiliga B foizli daromad olish uchun qo‘yilgan A so‘m pulning M yildan keyingi qiymatini aniqlang.

Chiziqli 38. Koordinatasi (x0; y0) bo‘lgan A nuqtaga nisbatan Ox o‘qiga simmetrik B nuqtani, Oy o‘qiga simmetrik C nuqtani va koordinata boshiga simmetrik D nuqtaning koordinatalarini aniqlang.



Yo‘llanma. A(x0; y0) nuqta Ox o‘qidan |y0| masofada, Oy o‘qidan |x0| masofada yotadi. Shuning uchun (rasmga qarang) A nuqtaga Ox o‘qiga nisbatan simmetrik B nuqta koordinatasi (x0; –y0), A nuqtaga Oy o‘qiga nisbatan simmetrik D nuqta koordinatasi (–x0; y0), A nuqtaga koordinata boshiga O nuqtaga nisbatan simmetrik E nuqta koordinatasi (–x0; –y0) bo‘ladi.

Chiziqli 39. Uchlari (x1; y1), (x2; y2) va (x3; y3) nuqtalarda bo‘lgan uchburchak perimetri va yuzini toping.

Chiziqli 40. Markazlari (x1; y1) va (x2; y2) nuqtalarda bo‘lgan teng radiusli aylanalar bir-biriga uringan. Aylana diametrini aniqlang.

Chiziqli 41. Markazlari (x1; y1) va (x2; y2) nuqtalarda bo‘lgan teng radiusli aylanalar bir-biriga uringan. Ikkala aylana chegaralab turgan soha yuzini aniqlang.

Chiziqli 42. A soni berilgan. Funksiyalar qo‘llamasdan faqat yordamchi o‘zgaruvchilar va 3 ta ko‘paytirish amalidan foydalanib yozib A^8 ni hisoblang. A ning hisoblangan barcha darajalarini chiqaring.

Chiziqli 43. A soni berilgan. Funksiyalar qo'llamasdan faqat yordamchi o'zgaruvchilar va 5 ta ko'paytirish amalidan foydalanib yozib A^{15} ni hisoblang. Buning uchun avval A^2, A^3, A^5 va A^{10} qiymatlarini hisoblang va chiqaring.

Chiziqli 44. A soni berilgan. Funksiyalar qo'llamasdan faqat yordamchi o'zgaruvchilar va 6 ta ko'paytirish amalidan foydalanib yozib A^{21} ni hisoblang.

Chiziqli 45. A soni berilgan. Funksiyalar qo'llamasdan faqat yordamchi o'zgaruvchilar va 6 ta ko'paytirish amalidan foydalanib yozib A^{28} ni hisoblang.

Chiziqli 46. A soni berilgan. Funksiyalar qo'llamasdan faqat yordamchi o'zgaruvchilar va 6 ta ko'paytirish amalidan foydalanib yozib A^{64} ni hisoblang.

Chiziqli 47. X kilogramm shokolad A so'm turishi ma'lum. T kilogramm shokolad necha so'm turishini aniqlang.

Chiziqli 48. Bir nuqtadan chiqqan avtomobillardan birinchisining tezligi A km/soat, ikkinchisining tezligi B km/soat bo'lsin. T vaqtdan keyingi ular orasidagi masofani aniqlang.

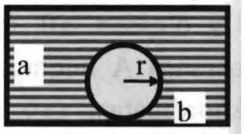
Chiziqli 49. Orasidagi masofa S bo'lgan avtomobillardan birinchisining tezligi A km/soat, ikkinchisining tezligi B km/soat bo'lsin. Ulat T1 vaqt bir-biriga qarab, T2 vaqt orqasiga qarab yurgandan keyingi ular orasidagi masofani aniqlang.

Chiziqli 50. Boshlang'ich tezligi v_0 bo'lib, a tezlanish bilan tekis harakat qilayotgan moddiy nuqtaning t vaqt ichida bosib o'tadigan yo'lini aniqlang (yo'llanma: $s=v_0 \cdot t+a \cdot t^2/2$).

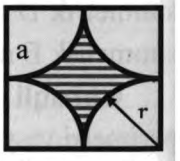
Chiziqli 51. Yer sirtiga nisbatan α burchak ostida v_0 boshlang'ich tezlik bilan otilgan jismning uchish masofasini aniqlang (yo'llanma: $L=2 \cdot v_0^2 \cdot \cos\alpha \cdot \sin\alpha/g$, $g=9,81$).

Chiziqli 52. Mos ravishda R_1, R_2, R_3, R_4 qarshilikka ega bo'lgan o'tkazgichlar parallel ulanganda hosil bo'ladigan **R** qarshilikni hisoblang (yo'llanma: $1/R=1/R_1+1/R_2+1/R_3+1/R_4$).

Chiziqli 53. Berilgan musbat a, b va r sonlari uchun shaklning shtrixlangan qismi yuzasini hisoblang (yo'llanma: bir shakl yuzasidan ikkinchi shakl yuzasini ayirish).



Chiziqli 54. Berilgan musbat a va r sonlari uchun shaklning shtrixlangan qismi yuzasini hisoblang (yo'llanma: bir shakl yuzasidan ikkinchi shakl yuzasini ayirish).



Chiziqli 55. Berilgan a, b va c sonlari $b^2-4 \cdot a \cdot c>0$ shartni qanoatlantiradi. $y=ax^2+bx+c$ funksiyaning absissalar o'qi bilan kesishish nuqtalari orasidagi masofani aniqlang.

= C =

Chiziqli 56. Berilgan x va y uchun quyidagi ifodalarni hisoblang:

a) $f = \dfrac{x + \dfrac{2+y}{\lvert x-1\rvert+1}}{y + \dfrac{7}{\sqrt{x^2+y^4+19}}}$ K1: 2 1 CH1: 1.441 K2: –1.9 –2.1 CH2: 1.88673	b) $g = \dfrac{\dfrac{19}{x^2+2}+50}{\sqrt{\lvert x-23\rvert} + \dfrac{63}{\lvert y\rvert+1}}$ K1: 0 1 CH1: 1.63931 K2: –2.3 –1 CH2: 1.44009	c) $h = \dfrac{\dfrac{\sin y}{4-\cos^2 x}}{\dfrac{\sqrt[3]{1+\mathrm{tg}^2 90}}{5+\lvert x+y\rvert}} + 19$ K1: 10.5 –2.3 CH1: 17.4727 K2: –0.5 –0.5 CH2: 18.4785

Yechim c). Aniqlik yuqori bo'lishi uchun barcha o'zgaruvchilarni **long double** turda tavsiflash mumkin. Ifodada kasrli ifoda va funksiyalarning ko'pligi C++ tilida ifodalarni chiziqli shaklda yozish uchun qiyinchilik keltirib chiqaradi. Shuning uchun ifodani bo'laklarga ajratib hisoblash maqsadga muvofiq:

Dastur	Natijasi

`#include <iostream>`
`#include <cmath>`
`using namespace std;`
`int main(){`
`  double x,y,a,b,c,g;`
`  cin>>x>>y;`
`  a=4-pow(cos(x),2);`
`  a=sin(y)/a;`
`  b=1+pow(tan(90),2);`
`  b=pow(b,1.0/3);`
`  c=5+fabs(x+y);`
`  b=b/c;`
`  g=a/b+19;`
`  cout<<g;`
`  return 0;`
`}`
	0 0
	19

Chiziqli 57. Berilgan x va y uchun quyidagi ifodalar qiymatini 10^{-4} aniqlikda hisoblang:

a) $f = \dfrac{2\cdot\sin\left(x-\frac{\pi}{3}\right)}{1+\cos^2(x+y)}e^{-x^2}$	b) $g = \dfrac{\sqrt{81+\lvert x-y\rvert+\lvert x\rvert}}{e^{\lvert x-y\rvert}+x^2+25}$	c) $h = \dfrac{\sin^3\lvert x+y\rvert}{\left\lvert\dfrac{\sqrt[3]{1+x^2\cdot\cos^2 y}}{19+(x-y)^4}\right\rvert}$

Chiziqli 58. Berilgan x va y uchun quyidagi ifodalar qiymatini 10^{-1} aniqlikda hisoblang:

a) $f = \dfrac{\sqrt{\lvert x-1\rvert} - \sqrt[3]{y}}{1 + x^2 + \ln\left(1 + \frac{y^2}{4}\right)}$	b) $g = \dfrac{(3 + e^{y-x}) \cdot \sqrt{x^2 + \sqrt{\lvert y+1\rvert}}}{1 + x^2 \cdot \lvert y - \cos(x-3)\rvert}$	c) $h = x + \dfrac{\frac{y}{\sqrt{x^2+4}}}{\sqrt[3]{\sin^2\frac{x}{2} + 1}}$

Chiziqli 59. Berilgan x va y uchun quyidagi ifodalar qiymatini 10^{-5} aniqlikda hisoblang:

a) $f = \dfrac{\sqrt[3]{y + \sqrt[4]{\lvert x-1\rvert}}}{\lvert x-y\rvert \cdot \sin^2(x \cdot y) + 1}$	b) $g = \dfrac{1 + \lvert\cos(x-y) \cdot x^5\rvert}{\sin^2\left(x + \frac{\pi}{6}\right) + 2 \cdot e^{x \cdot y}}$	c) $h = \dfrac{1 + \frac{2 \cdot \sin x}{3 + \cos^3 y}}{4 + \lvert e^{x \cdot y - 7} - 99\rvert}$

Chiziqli 60. Berilgan x va y uchun quyidagi ifodalarni hisoblang:

a) $f = \dfrac{\operatorname{arctg}(x+y)}{2 + \frac{e^{x \cdot y - 5}}{\sqrt{\sin^4 y + 19}}}$	b) $g = \dfrac{e^{\frac{19}{x^2+2}} + e^{\frac{50}{y^2+2}}}{2 + \frac{\sin(x+\pi)+1}{\lvert\lvert y\rvert + 19\rvert}}$	c) $h = \dfrac{\frac{\sin y + cosx}{4 - \operatorname{arctg}^2 x}}{\sin\left\lvert x + \frac{\pi}{3}\right\rvert + 7}$

Chiziqli 61. To‘g‘ri to‘rtburchakka tashqi chizilgan doiraning yuzi S berilgan bo‘lsa, to‘g‘ri to‘rtburchakning yuzini toping.

Chiziqli 62. Aylanaga teng tomonli balandligi h bo‘lgan uchburchak ichki chizilgan. Aylana uzunligini toping.

Chiziqli 63. Qayiqning turg‘un suvdagi tezligi V_t km/soat, daryo oqimi tezligi V_d km/soat ($V_d<V_t$) bo‘lsin. Qayiq daryo oqimi bo‘ylab T_1 soat, oqimga qarshi T_2 soat yurgan bo‘lsa, qayiq bosib o‘tgan masofani toping.

Chiziqli 64. Turg‘un suvdagi tezligi V_t km/soat bo‘lgan qayiqning daryo oqimi bo‘ylab T_1 soatda bosib o‘tgan masofasi oqimga qarshi T_2 soatda bosib o‘tgan masofasiga teng bo‘lsa, daryo oqimining tezligini toping.

Chiziqli 65. Berilgan a1, b1, c1, a2, b2 va c2 sonlar $a1 \cdot b2 - a2 \cdot b1 \neq 0$ shartni qanoatlantiradi. Quyidagi tenglamalar sistemasini yeching (yo‘llanma: o‘rniga qo‘yish usuli yoki qo‘shish usuli yoki Kramer formulalari):

$$\begin{cases} a1 \cdot x + b1 \cdot y = c1 \\ a2 \cdot x + b2 \cdot y = c2 \end{cases}$$

Chiziqli 66. Oxy tekisligida I kvadrant ichida A(x1; y1), III kvadrant ichida B(x2; y2), II kvadrant ichida C(x3; y3) va IV kvadrant ichida D(x4; y4) nuqtalari berilgan. AB va CD nuqtalardan o‘tuvchi to‘g‘ri chiziqlar kesishish nuqtasini aniqlang.

Chiziqli 67. Oxy tekisligida I kvadrant ichida A(x1; y1), II kvadrant ichida B(x2; y2), III kvadrant ichida C(x3; y3) va IV kvadrant ichida D(x4; y4) nuqtalari berilgan. Markazi koordinatalar boshida bo‘lgan va ABCD nuqtalar hosil qilgan to‘rtburchakni o‘z ichiga olgan eng kichik radiusli doira yuzini aniqlang.

Chiziqli 68. Oxy tekisligida I kvadrant ichida A(x1; y1), II kvadrant ichida B(x2; y2), III kvadrant ichida C(x3; y3) va IV kvadrant ichida D(x4; y4) nuqtalari berilgan. ABCD nuqtalar hosil qilgan to'rtburchak yuzini aniqlang (yo'llanma: to'rtburchak diagonali uni 2 ta uchburchakka ajratadi).

Chiziqli 69. Ox o'qida koordinata boshidan turli uzoqlikda yotgan A(x1), B(x2) va C(x3) nuqtalar berilgan. Koordinata boshiga eng yaqin nuqta bilan eng uzoq nuqtalar orasidagi masofani aniqlang.

Chiziqli 70. Oxy tekislikda koordinata boshidan turli uzoqlikda yotgan A(x1; y1), B(x2; y2) va C(x3; y3) nuqtalar berilgan. Barcha nuqtalar tegishli bo'lgan eng kichik radiusli doiraning yuzini aniqlang (yo'llanma: pi=acos(–1) yoki pi=4·arctan(1)).

K: 1 1 2 2 3 3 CH: 56.5487	K: 0 1 0 2 0.5 0.5 CH: 12.5664	K: 1.5 1.5 0 7 –2.5 –3.5 CH: 153.938

Chiziqli 71. Uchlari bir nuqtada bo'lgan a(x1; y1; z1) va b(x2; y2; z2) vektorlar hosil qilgan parallelogrammning yuzini aniqlang.

Chiziqli 72. Uchlari bir nuqtada bo'lgan a(x1; y1; z1), b(x2; y2; z2) va c(x3; y3; z3) vektorlar hosil qilgan parallelepipedning hajmini aniqlang.

Chiziqli 73. Natural N soni berilgan. 1 dan N sonigacha bo'lgan barcha sonlar yig'indisini aniqlang.

Chiziqli 74. ABCD to'g'ri to'rtburchakning 3 ta A(x1; y1), B(x2; y2) va C(x3; y3) uchlari koordinatalari berilgan. Uning D uchi koordinatalarini toping.

Chiziqli 75. A(x1; y1) va B(x2; y2) nuqtalardan o'tuvchi ax+by+c=0 to'g'ri chiziq tenglamasining a, b va c koeffitsiyentlarini toping.

Chiziqli 76. A(x1; y1) va B(x2; y2) nuqtalardan o'tuvchi to'g'ri chiziq bilan C(x0; y0) nuqta orasidagi masofani toping.

Chiziqli 77. A(x1; y1), B(x2; y2) va C(x0; y0) nuqtalar hosil qilgan uchburchak yuzini Geron formulasini qo'llamasdan aniqlang.

Chiziqli 78. Markazlari (x1; y1) va (x2; y2) nuqtalarda bo'lgan teng radiusli aylanalar bir-biriga uringan. Ikkala aylanani chegaralab turgan eng kichik qabariq soha yuzini aniqlang (yo'llanma: aylanani chegaralab turgan eng kichik qabariq soha chegarasi aylanalarga urinmalar va aylana chegaralaridan tashkil topadi).

Chiziqli 79. Markazlari (x1; y1), (x2; y2) va (x3; y3) nuqtalarda bo'lgan teng radiusli aylanalar bir-biriga uringan. Uchala aylanani chegaralab turgan eng kichik qabariq soha yuzini aniqlang.

Chiziqli 80. Markazlari (x1; y1) va (x2; y2) nuqtalarda bo'lgan teng radiusli aylanalar bir-biriga uringan. Ikkala aylanani chegaralab turgan eng kichik qabariq soha perimetrini aniqlang.

Chiziqli 81. Berilgan x soni uchun $A=1-2\cdot x+3\cdot x^2-4\cdot x^3$ ko'phad qiymatini eng kam amal bajargan holda hisoblang.

Yo'llanma:

Ko'phad qiymatini kompyuterda bajargandagi amallar sonini hisoblab ko'raylik: $A1=2\cdot x$; $A2=x\cdot x(=x^2)$; $A3=3\cdot A2(=3\cdot x^2)$; $A4=x\cdot x(=x^2)$; $A5=x\cdot A4(=x^3)$; $A6=4\cdot A5(=4\cdot x^3)$; $A7=1-A1(=1-2\cdot x)$; $A8=A7+A3(=1-2\cdot x+3\cdot x^2)$; $A9=A8-A6(=1-2\cdot x+3\cdot x^2-4\cdot x^3)$. Demak, 9 ta amal.

Bajariladigan amallarni 1-usulda guruhlaymiz: $A1=2\cdot x$; $A2=x^2$; $A3=1-A1(=1-2\cdot x)$; $A4=3\cdot A2(=3\cdot x^2)$; $A5=A3+A4(=1-2\cdot x+3\cdot x^2)$; $A6=A1\cdot A1(=4\cdot x^2)$; $A7=A6\cdot x(=4\cdot x^3)$; $A8=A5-A7$ $(=1-2\cdot x+3\cdot x^2+4\cdot x^3)$. Amallar soni bittaga kamaydi.

Bajariladigan amallarni 2-usulda guruhlash uchun ifodani $1-2\cdot x+x^2+2\cdot x^2\cdot(1-2\cdot x)$ kabi yozib olamiz: $A1=2\cdot x$; $A2=1-A1(=1-2\cdot x)$; $A3=x\cdot x(=x^2)$; $A4=A1+A2(=1-2\cdot x+x^2)$; $A5=2\cdot A3(=2\cdot x^2)$; $A6=A5\cdot A2(=2\cdot x^2\cdot(1-2\cdot x))$; $A7=A4+A6(=1-2\cdot x+x^2+2\cdot x^2\cdot(1-2\cdot x))$. Amallar soni ikkitaga kamaydi.

Bajariladigan amallarni 3-usulda guruhlash uchun ifodani $1+x\cdot(-2+x\cdot(3-4\cdot x))$ (**Gorner usuli**) kabi yozib olamiz: $A1=4\cdot x$; $A2=3-A1(=3-4\cdot x)$; $A3=x\cdot A2(=x\cdot(3-4\cdot x))$; $A4=A3-2$ $(=-2+x\cdot(3-4\cdot x))$; $A5=x\cdot A4(=x\cdot(-2+x\cdot(3-4\cdot x)))$; $A6=1+A5(=1+x\cdot(-2+x\cdot(3-4\cdot x)))$. Amallar soni uchtaga kamaydi. Ko'phad qiymatini hisoblashda Gorner usuli eng samarador bo'ladi.

Chiziqli 82. Berilgan x soni uchun $A=1+2\cdot x+3\cdot x^2+4\cdot x^3$ ko'phad qiymatini eng kam amal bajargan holda hisoblang.

Chiziqli 83. Berilgan x soni uchun $A=2\cdot x^4-3\cdot x^3+4\cdot x^2-5\cdot x+6$ ko'phad qiymatini eng kam amal bajargan holda hisoblang.

Chiziqli 84. Berilgan natural N soni uchun $S=3+6+9+\ldots+3\cdot N$ yig'indi qiymatini hisoblang.

Chiziqli 85. Berilgan natural N soni uchun $S=2^2+4^2+6^2+\ldots+(2\cdot N)^2$ yig'indi qiymatini hisoblang.

Chiziqli 86. Berilgan natural N soni uchun $S=5^3+10^3+15^3+\ldots+(5\cdot N)^3$ yig'indi qiymatini hisoblang.

Chiziqli 87. Berilgan natural N soni uchun $S=(1/3)^5+(2/3)^5+(3/3)^5+\ldots+(N/3)^5$ yig'indi qiymatini hisoblang.

Chiziqli 88. Berilgan natural N soni uchun $S=7+21+35+\ldots+(14\cdot N-7)$ yig'indi qiymatini hisoblang.

Chiziqli 89. Berilgan natural N soni uchun $S=1^2+3^2+5^2+\ldots+(2\cdot N-1)^2$ yig'indi qiymatini hisoblang.

Chiziqli 90. Natural N soni berilgan. $S=1^3+3^3+5^3+\ldots+(2\cdot N-1)^3$ yig'indi qiymatini hisoblang.

Chiziqli 91. Natural N soni berilgan. S=1·2+2·3+3·4+...+N·(N+1) yig'indi qiymatini hisoblang.

Chiziqli 92. Natural N soni berilgan. S=1·2·3+2·3·4+...+N·(N+1)·(N+2) yig'indi qiymatini hisoblang.

Chiziqli 93. Natural N soni berilgan. $S=\frac{1}{1\cdot 2}+\frac{1}{2\cdot 3}+\frac{1}{3\cdot 4}+...+\frac{1}{N\cdot(N+1)}$ yig'indi qiymatini hisoblang.

Chiziqli 94. Natural N soni berilgan. $S=\frac{1}{1\cdot 5}+\frac{1}{5\cdot 9}+\frac{1}{9\cdot 13}+...+\frac{1}{(4\cdot N-3)\cdot(4\cdot N+1)}$ yig'indi qiymatini hisoblang.

Chiziqli 95. Natural N soni berilgan. $S=\frac{1}{4\cdot 5}+\frac{1}{5\cdot 6}+\frac{1}{6\cdot 7}+...+\frac{1}{(N+3)\cdot(N+4)}$ yig'indi qiymatini hisoblang.

Chiziqli 96. Natural N soni berilgan. $S=\frac{1}{1\cdot 4}+\frac{1}{4\cdot 7}+\frac{1}{7\cdot 10}+...+\frac{1}{(3\cdot N-2)\cdot(3\cdot N+1)}$ yig'indi qiymatini hisoblang.

Chiziqli 97. Natural N soni berilgan. $S=\frac{7}{1\cdot 8}+\frac{7}{8\cdot 15}+\frac{7}{15\cdot 22}+...+\frac{7}{(7\cdot N-6)\cdot(7\cdot N+1)}$ yig'indi qiymatini hisoblang.

Chiziqli 98. Natural N soni berilgan. $S=\frac{1}{4\cdot 8}+\frac{1}{8\cdot 12}+\frac{1}{12\cdot 16}+...+\frac{1}{4\cdot N\cdot(4\cdot N+4)}$ yig'indi qiymatini hisoblang.

Chiziqli 99. Natural N soni berilgan. $S=\frac{1^2}{1\cdot 3}+\frac{2^2}{3\cdot 5}+\frac{3^2}{5\cdot 7}+...+\frac{N^2}{(2\cdot N-1)\cdot(2\cdot N+1)}$ yig'indi qiymatini hisoblang.

Chiziqli 100. Natural N soni berilgan. $S=\left(1-\frac{1}{4}\right)\cdot\left(1-\frac{1}{9}\right)\cdot\left(1-\frac{1}{16}\right)\cdot....\cdot\left(1-\frac{1}{N^2}\right)$ ko'paytma qiymatini hisoblang.

Chiziqli 101. Natural N soni berilgan. $S=\frac{1}{Ln2\cdot Ln4}+\frac{1}{Ln4\cdot Ln8}+\frac{1}{Ln8\cdot Ln16}+...+\frac{1}{Ln2^{N-1}\cdot Ln2^N}$ yig'indi qiymatini hisoblang.

Chiziqli 102. Natural N soni berilgan. N ta ildiz qatnashgan $S=\sqrt{2+\sqrt{2+\sqrt{2+...+\sqrt{2}}}}$ yig'indi qiymatini hisoblang.

Chiziqli 103. Natural N soni berilgan. $S=\sin\frac{\pi}{3}+\sin\frac{2\cdot\pi}{3}+...+\sin\frac{N\cdot\pi}{3}$ yig'indi qiymatini hisoblang.

Chiziqli 104. Natural N soni berilgan. $S=\operatorname{arctg}\frac{1}{2}+\operatorname{arctg}\frac{1}{8}+...+\operatorname{arctg}\frac{1}{2\cdot N^2}$ yig'indi qiymatini hisoblang.

= D =

Bu guruh vazifalarini bajarish uchun mantiqiy va matematik tahlildan foydalanish kifoya. Dastur tuzish uchun olingan natijalar realizatsiya qilinadi.

Chiziqli 105. Alida 2 dan 7 gacha raqamlar yozilgan kubiki bor edi. Ali kubikni har bir otganda yerga tushgan kubikni ustki tomonidagi qiymatlar yig'indisini hisoblab boradi. Ali shunchalik mohir ediki, kubikni otganda kerakli raqamni tushira olardi. N sonini hosil qilish uchun ($2\le N\le 1000$) Ali kubikni eng kamida necha marta otishi kerak bo'lishini aniqlang.

K:	2	5	7	15	36	55	10	998	999	1000
CH:	1	1	1	3	6	8	2	143	143	143

Chiziqli 106. Butun N va K ($1 \le N, K \le 10^8$) sonlari berilgan. Ali N ta origami (qog'ozni buklab shakl yasash) tayyorlamoqchi edi. Har bir origamiga 2 ta qizil, 5 ta yashil va 8 ta sariq rangli varaq kerak bo'ladi. Do'konda rangli varaqlar bloknot kabi birlashtirilgan bo'lib, har bir bloknotda K ta bir xil rangli varaq bor edi. Ali sotib olishi kerak bo'lgan eng kam bloknotlar sonini aniqlang.

K:	3 5	15 6	1 1	100000000 1	1 100000000	96865066 63740710
CH:	10	38	15	1500000000	3	25

Chiziqli 107. Butun a, b, k ($1\le$ a, b, k $\le 10^9$) sonlari berilgan. Baqa Ox o'qining 0 koordinatali nuqtasida turibdi. U quyidagi tartibda sakraydi: birinchi sakrash a birlik o'ngga, ikkinchi sakrash b birlik chapga, uchinchi sakrash a birlik o'ngga, to'rtinchi sakrash b birlik chapga, va hokazo. Baqa k marta sakragandan keyin turgan koordinatasini aniqlang.

K:	5 2 3	100 1 4	1 10 5	1000000000 1 6	602436426 877914575 158260522
CH:	8	198	-17	2999999997	-21798657830166889

Chiziqli 108. Butun N ($1\le$ N $\le 10^9$) soni berilgan. Berilgan N dollarni 1, 5, 10, 20 va 100 dollarlik yaxlit kupyuralar yordamida maydalash kerak. Maydalash uchun kamida nechta kupyura kerak bo'lishini aniqlang.

K:	125	43	1000000000	4	5	74	31	7	719	847	4704	365173
CH:	3	5	10000000	4	1	8	3	3	13	13	51	3658

Chiziqli 109. Alining uyi Ox o'qining 0 nuqtasida, Valining uyi esa X ($1 \le X \le 1000000$) nuqtasida joylashgan. Ali har bir qadamda 1 yoki 2 yoki 3 yoki 4 yoki 5 birlikka surilishi mumkin. Ali Valining uyiga borgunicha eng kamida necha qadam tashlashi keraklıgını aniqlang.

K:	999999	5	12	1000000	1	534204	53	942212	716
CH:	200000	1	3	200000	1	106841	11	188443	144

Chiziqli 110. Berilgan n ($1 \le n \le 10^{15}$) natural son uchun quyidagi funksiyani hisoblang:

$$f(n) = -1 + 2 - 3 + 4 - \cdots + (-1)^n \cdot n.$$

K:	1	100	1953	208170109961052	288565475053	100000017040846
CH:	-1	50	-977	104085054980526	-144282737527	50000008520423

Chiziqli 111. Butun N va B ($1 \le N, B \le 500$) sonlari berilgan. Ali homiyligida tashkil etilgan tennis turnirida N ta ishtirokchi qatnashmoqda. Ali musobaqani olimpiada sistemasida o'tkazgani uchun yutqazgan o'yinchi turnirdan chiqib ketadi. Har bir turda eng ko'p sondagi juftliklar aniqlanib, har bir juftlikdagi o'yinchi orasida musobaqa o'tkaziladi va yutganlar keyingi turga yo'l oladi. Juftliklardan ortganlar esa keyingi turga to'g'ridan to'g'ri o'tadi. Har bir turda har bir ishtirokchi uchun B ta va hakam uchun bir shisha mineral suv kerak bo'ladi. Turnir yakunigacha yetadigan eng kam sondagi mineral suv shishalari sonini aniqlang.

K:	5 2	8 2	20 500	500 500	500 237	7 12	81 477	41 41	100 11
CH:	20	35	19019	499499	237025	150	76400	3320	2277

Chiziqli 112. Butun N va K ($1 \le N, K \le 10^9$) sonlari berilgan. N sonidan katta hamda K ga karrali butun sonlarning eng kichigi X bo'lsin. Shu X sonini chiqaring.

K:	5 3	26 13	197 894	97259 41764	76770926 13350712	1234567 123456
CH:	6	39	894	125292	80104272	1358016

Chiziqli 113. Butun N ($2 \le N \le 100000$) soni berilgan. Har qanday 2 dan kichik bo'lmagan butun N sonini tub sonlar yig'indisi ko'rinishida tasvirlash mumkin, masalan 2=2, 3=3, 5=2+3, 7=2+3+2. N soni eng ko'p sondagi tub sonlar yig'indisi ko'rinishida tasvirlangan bo'lsa, shu tub sonlar sonini chiqaring.

K:	5	6	99999	7	13	99993	57	774	7166
CH:	2	3	49999	3	6	49996	28	387	3583

Chiziqli 114. Butun N ($1 \le N \le 10^5$) soni berilgan. Vali bir necha yil avval boshqa maktabga o'tib ketgan bo'lib, uning qaysi maktabda o'qiyotganini Ali bilmas edi. Shahardagi maktablar soni N ta va ular 1 dan N gacha tartiblangan. Do'stini qidirib topish uchun Ali yo'lga chiqmoqchi bo'ldi. Bilet narxi m-maktab bilan k-maktab oralig'i uchun (m+k) mod (N+1) so'mga teng va shu bilet bilan ikkala maktab oralig'ida istalgancha yurish mumkin. Ali izlashni istalgan maktabdan boshlashi mumkin. Ali sarflaydigan eng kam yo'l haqi qancha ekanini chiqaring.

K:	2	10	43670	4217	17879	100000	5188	12802
CH:	0	4	21834	2108	8939	49999	2593	6400

Chiziqli 115. Butun N va K ($1 \le N,\ K \le 10^{12}$) sonlari berilgan. Dasturlash bo'yicha olimpiadada N kishi qatnashmoqda. G'oliblarga diplom va faxriy yorliq berishga kelishildi. Hakamlar tomonidan taqdirlanganlar soni qatnashuvchilarning yarmidan oshmasligi, faxriy yorliq bilan taqdirlanuvchilar soni diplom bilan taqdirlanuvchilar sonidan K marta ko'p bo'lishi e'lon qilindi. Birorta ham qatnashuvchi taqdirlanmasligi ham mumkin ekan. Agar taqdirlanuvchilar bo'lsa, u holda diplom olganlar sonini, faxriy yorliq olganlar sonini va hech nima olmaganlar sonini chiqaring.

K:	18 2	1000000000000 5	1000000000000 499999999999
CH:	3 6 9	83333333333 416666666665 500000000002	1 499999999999 500000000000

Chiziqli 116. Butun N ($1 \le N \le 10^9$) soni berilgan. Ali oxirgi raqami 0 bilan tugagan sonlarni yoqtiradi. Berilgan N sonini Ali yoqtirgan va N sonidan kichik bo'lmagan eng yaqin songa aylantiring.

K:	3	9	113	1000000000	5432359	5432359	99999994	10002
CH:	10	10	120	1000000000	5432360	5432360	100000000	10010

Chiziqli 117. Butun K, N, S va P ($1 \le$ K, N, S, P $\le 10^4$) sonlari berilgan. K ta o'quvchining har biri N tadan to'g'ri to'rtburchakli oq varaqdan laylak yasamoqchi edi. Har bir varaqdan S ta laylak yasash mumkin. Bir taxlamda P ta oq varaq bor bo'lib, ular taxlamlardagi varaqlarni o'zaro taqsimlab olishgach, laylak yasashni boshlashadi. Ularga kerak bo'ladigan taxlamlar sonini aniqlang.

K:	5 3 2 3	5 3 100 1	10000 10000 1 1	300 300 21 23	6246 8489 1227 9
CH:	4	5	100000000	196	4858

Chiziqli 118. Butun N va K ($1 \le N,\ K \le 10^{18}$) sonlari berilgan. Oxy tekisligida 0 dan 2N gacha tartiblangan (2N+1) ta nuqtani quyidagicha aniqlaymiz: i-nuqtaning x-koordinatasi i ga, y-koordinatasi 0 ga teng, ya'ni A_i=(i; 0). Hosil bo'lgan A_i-nuqta A_{i+1}-nuqta bilan kesma yordamida tutashtirilganda biror f(x) funksiyaning grafigi bo'ladi, i=0,…,2N+1. Shunday amal bajarish mumkin bo'lsin: bitta qadamda ixtiyoriy toq tartib raqamli, ya'ni A_1, A_3, …, A_{2N-1} nuqtalardan birortasining y-koordinatasini 1 ga oshirish mumkin. Shunda quyidan Ox o'qi bilan, yuqoridan f(x) funksiya bilan chegaralangan yuzani K bilan belgilaymiz. Berilgan K uchun aytib o'tilgan amalni bajarib funksiyaning eng baland qiymati eng kamida qanday bo'lishini aniqlang.

K:	4 12	12839719 1294012934	5 13	4 15	10 45	7 1000000000000000000
CH:	3	101	3	4	5	142857142857142858

Chiziqli 119. Butun N ($1 \le N \le 10^{10}$) soni berilgan. Quyidagi ketma-ketlikning N-hadini aniqlang:

1, 2, 2, 3, 3, 3, 4, 4, 4, 4, 5, 5, 5, 5, 5, …

K:	3	55	100000	274	1999	10000000000	999999999	1000006282
CH:	2	10	447	23	63	141421	44721	44722

Chiziqli 120. Butun N ($1 \le N \le 10^9$) soni berilgan. Ali hokkey ko'rishni yoqtiradi. U darvozaga har bir shayba kiritilganda hisobni yozib bordi: 1:0, 1:1, 1:2, 1:3, 2:3. Shundan keyin barcha sonlarni qo'shib yig'indida 15 hosil qildi:

$$1+0+1+1+1+2+1+3+2+3=15.$$

Agar shu kabi yig'indi qiymati N berilgan bo'lsa, o'yinda darvozaga nechta shayba kiritilganini aniqlang.

K:	15	6	1275	153	499500	1038961	61732716	999961560
CH:	5	3	50	17	999	1441	11111	44720

Chiziqli 121. Butun N ($1 \le N \le 10^9$) soni berilgan. Quyidagicha atama kiritamiz: N sonining **bo'laklamasi** deb natural sonlardan iborat o'smaydigan shunday ketma-ketlikka aytiladiki, ketma-ketlik hadlari yig'indisi N ga teng bo'ladi. Masalan, 8 uchun bo'laklamaga misollar: {4; 4}, {4; 2; 2}, {3; 3; 2}, {2; 2; 2; 2} va hokazo. Birinchi hadga teng sonlar sonini bo'laklamaning **quvvati** deb ataymiz. Masalan: {4; 4} ning quvvati 2, {4; 2; 2} ning quvvati 1, {3; 3; 2} ning quvvati 2, {2; 2; 2; 2} ning quvvati 4. Berilgan N soni bo'laklamasining quvvatini aniqlang.

K:	15	30	1275	153	999999999	1234567	61732716	999961560
CH:	8	16	638	77	500000000	617294	30866359	499980781

4-BOB. C++ TILIDA TARMOQLANISH ALGORITMLARI

Ma'lumki, dasturlash tillarida turli xil usullar yordamida dastur boshqaruvini bir joydan ikkinchi joyga o'tkazish yoki biror shartga (talabga) asosan kerakli dastur parchasining bajarilishini ta'minlash mumkin, ya'ni tarmoqlanuvchi algoritmli dasturlar (qisqacha tarmoqlanuvchi dasturlar) tuzish mumkin. Tarmoqlanuvchi dasturlarning asosiy mohiyati shundaki, har bir tarmoqlanish shartining rost natijasiga ko'ra dasturdagi birinchi ko'rsatmalar guruhi bajariladi, ikkinchi ko'rsatmalar guruhi esa bajarilmaydi, va aksincha, shartning yolg'on natijasiga ko'ra dasturdagi ikkinchi ko'rsatmalar guruhi bajariladi, birinchi ko'rsatmalar guruhi esa bajarilmaydi.

Tarmoqlanuvchi dasturlardagi shartlar masala mazmunida oshkora yoki tagma'noda aks etgan bo'lishi mumkin. Masalada berilgan ma'lumotlar va shartlar aniqlangach, bu ikki ma'lumot orasidagi bog'lanish (formulalar, munosabatlar, ifodalar, tuzilmalar, va hokazo) aniqlanadi. Tarmoqlanuvchi dasturlarning asosini tashkil etadigan shartlar mazmuni va ularni ifodalash usullari avvalgi boblarda yoritilgan edi.

Bu bobda C++ tilining tarmoqlanuvchi dasturlar tuzish uchun beradigan asosiy imkoniyatlari ko'rib chiqiladi. Tarmoqlanuvchi dasturlar tuzishda qo'llanadigan **o'tish**, **tarmoqlash**, **shart** va **tanlash** operatorlarining sintaksisi hamda ishi misollar yordamida izohlanadi.

Bobning ba'zi masalalarini hal etishda tarmoqlanuvchi dasturlar tuzishda qo'llanadigan operatorlar o'rniga yoki birgalikda C++ tili funksiyalaridan ham foydalanish mumkin. Misol sifatida quyidagi funksiyalarni ko'rsatish mumkin:

Funksiya	Vazifasi
min(a, b)	a va b sonlardan kichigini qaytaradi
max(a, b)	a va b sonlardan kattasini qaytaradi
swap(a, b)	a va b ning qiymatlarini almashtiradi

1-§. O'TISH, TARMOQLASH, SHART VA TANLASH OPERATORLARI

O'TISH OPERATORI

Ko'pchilik shartsiz o'tish operatori deb ataydigan **goto** operatori dastur ishlashi davomida boshqaruvni bir joydan (shartli ravishda "uchish" joyi deb ataymiz) ko'rsatilgan boshqa joyga (shartli ravishda "qo'nish" joyi deb ataymiz) hech qanday shart tekshirmasdan o'tkazish imkoniyatini beradi. Demak, o'tish operatori dasturda doimo jufti bilan qatnashar ekan.

O‘tish operatori quyidagi ko‘rinishga ega:

```
goto nishon;
{ }//ishlanmay tashlab ketilayotgan dastur qismi
nishon:{}
```

Uchish joyida **goto** yozuvidan keyin qo‘nish joyini ko‘rsatuvchi **nishon** belgisi yoziladi. **Nishon** ham identifikatordir. Dasturda qo‘nish joyini ko‘rsatuvchi **nishon** belgisidan keyin ikki nuqta (:) qo‘yiladi. Dasturda uchish va qo‘nish joyi bitta funksiya tanasida joylashgan bo‘lishi shart. Bitta nishonga bir nechta goto operatori yordamida murojaat qilish mumkin, ammo bitta **main** funksiyasining ichida **ikkita bir xil nishon bo‘lishi mumkin emas**.

Bu operator quyidagicha ishlaydi: ishga tushirilgan dasturda **goto** operatorini ishlash navbati kelgach, kompilyator nishonni aniqlab qo‘nish joyini izlaydi va uni aniqlagach, ishlash navbatini nishondan keyin yozilgan : belgisidan keyinga uzatadi. Masalan:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  double a = 1, b;`
`  goto nish;`
`  b = 5 * a;`
`  nish: b = a + 1;`
`  cout << "b=" << b;`
`  return 0;`
`}`
	b=2

Bu dasturda goto operatori ishlagandan so‘ng `b = 5 * a;` operatori ishlanmasdan tashlab ketildi va ishlash navbati `b = a + 1;` operatoriga berildi.

Dasturda qo‘nish joyi uchish joyidan oldin ham yozilishi mumkin.

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  double a = 1, b;`
`  nish: b = 5 * a;`
`  goto nish;`
`  b = a + 1;`
`  cout << "b=" << b;`
`  return 0;`
`}`	

Bu dasturda avval `b = 5 * a;` operatori ishlaydi, so‘ng **goto** operatori ishlaganda ishlash navbati yana `b = 5 * a;` operatoriga beriladi, keyin ishlash navbati yana **goto** operatoriga beriladi, va bu jarayon takrorlanaveradi. Demak, bu dastur ishi hech qachon to‘xtamaydi, ya’ni dasturchilar tili bilan aytganda “**siklga tushib qoladi**”.

Goto operatori bajarilish natijasida o‘zgaruvchi yoki o‘zgarmas tavsifi qabul qilinmay qolib ketmasligi kerak. Ya’ni agar goto operatori o‘zidan keyin joylashgan nishonga murojaat etsa va shu goto operatori hamda nishon orasida biror o‘zgaruvchi yoki o‘zgarmas tavsifi uchragan bo‘lsa, kompilyator xatolik haqida xabar chiqaradi. Masalan, quyidagicha yozish mumkin emas:

```
goto nishon;
int a = 10;
nishon: b = 5;
```

Odatda, goto operatorini ishlatish professional dasturchilar tomonidan tavsiya etilmaydi. Buning sababi, goto operatoridan foydalanish ba’zan dastur matnini tushunarsiz holatga olib kelishi va debug (sozlash) jarayonini qiyinlashtirishi mumkin.

TARMOQLASH OPERATORI

Ko‘pchilik shart bo‘yicha o‘tish operatori deb ataydigan **`if`** tarmoqlash operatori dastur ishlashi davomida boshqaruvni u yoki bu operatorga uzatish uchun xizmat qiladi. Dasturda tarmoqlash operatorining to‘liq yoki qisqa ko‘rinishlaridan foydalanish mumkin. Tarmoqlash operatorining to‘liq ko‘rinishi quyidagicha:

```
if (shart) operator1; else operator2;
```

Bu yerda `shart` qiymati rost yoki yolg‘on bo‘lgan ifodadir. Bunda qo‘yilgan `shart` qiymati rost (ya’ni true yoki 0 dan farqli) bo‘lsa, `operator1` ishlanadi, aks holda, ya’ni `shart` qiymati yolg‘on (ya’ni false yoki 0) bo‘lsa, `operator2` ishlanadi. Masalan:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  int a, b;`
`  cout << "a="; cin >> a;`
`  if (a>0) b = 5 * a; else b = a + 1;`
`  cout << "b=" << b;`
`  return 0;`
`}`
	a=1
	b=5

Bu dastur quyidagi misol yechimini beradi:

Misol. Berilgan a soni uchun quyidagi funksiya qiymatini aniqlang:

$$b = \begin{cases} 5 \cdot a, & \text{agar } a > 0 \\ a + 1, & \text{aks holda} \end{cases}$$

Demak, tarmoqlash operatori bir-birini inkor etuvchi ikki holatning biriga mos amallar ketma-ketligini bajarar ekan.

Tarmoqlash operatori tarkibi bir satrda yozilishi shart emas. Masalan:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  int a, b;`
`  cout << "a="; cin >> a;`
`  if (a>0) b = 5 * a;`
`  else b = a + 1;`
`  cout << "b=" << b;`
`  return 0;`
`}`
	a=0
	b=1

Agar `operator1;` yoki `operator2;` o'rniga operatorlar guruhi yozilishi kerak bo'lsa, u holda operatorlar guruhi { } blok orasida yoziladi. Masalan:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  int a = 1, b;`
`  if (a>0) {b = 5 * a;`
`  b = b + 2;}`
`  else {b = a + 1; b = b * b;}`
`  cout << "b=" << b;`
`  return 0;`
`}`
	b=7

Tarmoqlash operatorining qisqa ko'rinishi quyidagicha:

```
if (shart) operator1;
```

Bunda qo'yilgan `shart` qiymati rost (ya'ni true yoki mantiqiy shartda 1 soniga ekvivalent bo'lgan 0 dan farqli son) bo'lsa, `operator1` ishlanadi va so'ngra boshqaruv `if` operatoridan keyin yozilgan operatorga uzatiladi, aks holda, ya'ni `shart` qiymati yolg'on (ya'ni false yoki 0) bo'lsa, u holda boshqaruv to'g'ridan to'g'ri `if` operatoridan keyin

yozilgan operatorga uzatiladi. Masalan, quyidagi dasturda **a>0** shart qiymati **false** bo‘lgani uchun `{b = 5 * a; b = b + 2;}` blok bajarilmasdan boshqaruv `b = a + 1;` operatoriga uzatiladi:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  int a = -3, b;`
`  if (a>0)`
`  {b = 5 * a; b = b + 2;}`
`  b = a + 1; b = b * b;`
`  cout << "b=" << b;`
`  return 0;`
`}`
	b=4

SHART BO‘YICHA O‘ZLASHTIRISH OPERATORI

C++ tilida tarmoqlash operatoridan farqli **shart bo‘yicha o‘zlashtirish** (yoki shartli o‘zlashtirish) operatori ham bor bo‘lib, uning ko‘rinishi quyidagicha:

```
o‘zgaruvchi = (shart) ? ifoda1: ifoda2;
```

Bu yerda `shart` qiymati rost yoki yolg‘on bo‘lgan ifodadir. Bunda qo‘yilgan `shart` qiymati rost (ya’ni true yoki 0 dan farqli son) bo‘lsa, `o‘zgaruvchi` ga `ifoda1` ning qiymati o‘zlashtiriladi, aks holda, ya’ni `shart` qiymati yolg‘on (ya’ni false yoki 0) bo‘lsa, `o‘zgaruvchi` ga `ifoda2` ning qiymati o‘zlashtiriladi. Masalan:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  int a, b;`
`  cout << "a="; cin >> a;`
`  b = (a>0) ? 5 * a: a + 1;`
`  cout << "b=" << b;`
`  return 0;`
`}`
	a=1
	b=5

Bu dastur ham yuqoridagi misol yechimini beradi va, ko‘rib turganingizdek, misoldagi shart faqat bitta o‘zgaruvchining qiymatini tanlash uchun qo‘llanganda shartli o‘zlashtirish operatoridan foydalanish juda ham qulay.

Demak, shartli o'zlashtirish operatorida ham bir-birini inkor etuvchi ikki holatdan biri qaralar ekan.

E'tiborli tomonlaridan biri, shartli o'zlashtirish operatori chiqarish operatori tarkibida ham yozilishi mumkin, faqat qavslar ichida yozish kerak bo'ladi:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  int a, b;`
`  cout << "a="; cin >> a;`
`  cout << "b=" << ((a>0) ? 5 * a: a + 1);`
`  return 0;`
`}`
	a=–1
	b=0

TANLASH OPERATORI

C++ tilida tanlash operatori berilgan ifoda qiymatiga mos kelgan ko'rsatmalar ketma-ketligiga o'tish uchun xizmat qiladi. Umumiy ko'rinishi quyidagicha:

```
switch (ifoda) {
      case konstanta_1: ko'rsatmalar ketma-ketligi; break;
      case konstanta_2: ko'rsatmalar ketma-ketligi; break;
      ...
      case konstanta_N: ko'rsatmalar ketma-ketligi; break;
      default: ko'rsatmalar ketma-ketligi;
}
```

Bu yerda **ifoda qiymati butun qiymatli** son (int), **belgi** (char) yoki **bool** turida bo'lishi shart. Ya'ni ifoda qiymati **double** yoki **float** turida bo'lishi mumkin emas. Ifodaning o'rnida o'zgaruvchi ham bo'lishi mumkin. Konstantalar sifatida qiymati konstanta bo'ladigan ifoda yozilishi mumkin. Tanlash operatoridan chiqish `break` (uzilish) operatori orqali bajariladi. Tanlash operatorining `default` qismi esa ifoda qiymati `case` yozuvidan keyingi birorta ham konstanta qiymatiga to'g'ri kelmaganida ishlaydi. Umuman olganda, mantiqan to'g'ri tashkil etilgan tanlash operatorida `default` qismini yozish shart emas.

Masala. Berilgan N ($1 \leq N \leq 7$) butun songa mos hafta kunini chiqaring.

Yechim. Bu masalani yechish uchun tanlash operatoridan foydalanish qulaydir.

Dastur	Natijasi

#include <iostream>
using namespace std;
int main(){
int n;
cout << "N="; cin >> n;
switch (n){
case 1: cout << "Dushanba"; break;
case 2: cout << "Seshanba"; break;
case 3: cout << "Chorshanba"; break;
case 4: cout << "Payshanba"; break;
case 5: cout << "Juma"; break;
case 6: cout << "Shanba"; break;
case 7: cout << "Yakshanba"; break;
default: cout << "Adashdingiz!";
}
return 0;
}
	N=5
	Juma

Unutmang: har bir **case** ga mos ko‘rsatmalar ketma-ketligi oxirida yozilgan **break** operatori shu ko‘rsatmalar ketma-ketligi bajarilganidan so‘ng tanlash operatoridan chiqishni ta’minlaydi. Bu operatorni yozish shart emas, lekin **break** operatori yozilmasa, quyidagi kabi xato holatga olib kelishi ham mumkin:

Dastur	Natijasi

#include <iostream>
using namespace std;
int main(){
int n;
cout << "N="; cin >> n;
switch (n){
case 1: cout << "Dushanba" <<endl;
case 2: cout << "Seshanba" <<endl;
case 3: cout << "Chorshanba" <<endl;
case 4: cout << "Payshanba" <<endl;
case 5: cout << "Juma" <<endl;
case 6: cout << "Shanba" <<endl;
case 7: cout << "Yakshanba" <<endl;
default: cout << "Adashdingiz!" <<endl;
}
cout << "Xato bo'ldi!" <<endl;
return 0;
}
	N=5
	Juma
	Shanba
	Yakshanba
	Adashdingiz!
	Xato bo'ldi!

Dasturga kiritilgan qiymat 5 bo'lgani uchun tanlash operatori boshqaruvni `case 5` ga mos amallar ketma-ketligiga o'tkazadi, ya'ni ekranga Juma matni chiqariladi. Lekin tanlash operatori ishi "uzilmagani" uchun keyingi qatorlar ham `case 5` ga mos amallar ketma-ketligi sifatida ishlanaveradi.

2-§. TARMOQLANUVCHI ALGORITMLI DASTUR TUZISH NAMUNALARI

Aytib o'tilganidek, misol yoki masala yechishda biror shartning bajarilishini tekshirish talabi qo'yilgan vazifaning mohiyatidan kelib chiqadi. Masalan, yuqoridagi **misol** va **masala**da shartlar oshkora berilgan edi. Agar "Berilgan A sonni B songa bo'lish natijasini aniqlang" vazifasi qo'yilgan bo'lsa, u holda bu masalada "0 ga bo'lish mumkin emas!" qoidasini qo'llash tagma'noda qatnashadi. Yoki agar biror funksiya qiymatini hisoblash talab qilinayotgan bo'lsa, u holda kiritilayotgan qiymatlar funksiya aniqlanish sohasiga tegishliligi ham shart tekshirishni talab etadi.

Avvalgi boblarda berilgan masalalarda yuqoridagi kabi shart tekshirish talablari hosil bo'lmasligi uchun masala shartiga qo'shimcha shartlar kiritilgan edi. Bu bobda esa bunday shartlar kelib chiqishining sabablari ham ko'rib o'tiladi.

Namuna 1. Berilgan x va y sonlarda quyidagi z funksiya qiymatini hisoblang:

$$z = \begin{cases} x^3 + 3 \cdot y, & \text{agar } x \le y \\ x \cdot y^2, & \text{aks holda} \end{cases}$$

Yechim. Bu misolda kiritilayotgan x va y sonlari turi aytilmagan, shu sababli barcha o'zgaruvchilarni **float** (**double** yoki **long double**) turlaridan birortasida tavsiflaymiz. Misolda "aks holda" holiga mos keluvchi shart x>y ko'rinishida yoziladi. U holda kerakli dasturni shartli o'zlashtirish yoki tarmoqlash operatorlari yordamida quyidagicha usullarda yozish mumkin.

Dastur 1	Natijasi

`#include <iostream>`
`#include <cmath>`
`using namespace std;`
`int main(){`
`  float x, y, z;`
`  cout << "x= "; cin >> x;`
`  cout << "y= "; cin >> y;`
`  z = (x<=y) ? pow(x,3)+3*y: x*y*y;`
	x= 1.5
	y= –5
	z= 37.5

```
      cout << "z= " << z;
      return 0;
}
```

Dastur 2	Natijasi

```
#include <iostream>
#include <cmath>
using namespace std;
int main(){
      float x, y, z;
      cout << "x= "; cin >> x;
      cout << "y= "; cin >> y;
      z = (x>y) ? x*y*y : pow(x,3)+3*y;
      cout << "z= " << z;
      return 0;
}
```

x= 1
y= 1
z= 4

Dastur 3	Natijasi

```
#include <iostream>
#include <cmath>
using namespace std;
int main(){
      float x, y, z;
      cout << "x= "; cin >> x;
      cout << "y= "; cin >> y;
      if (x<=y) z=pow(x,3)+3*y;
      else z=x*y*y;
      cout << "z= " << z;
      return 0;
}
```

x= –5
y= 1.5
z= –120.5

Dastur 4	Natijasi

```
#include <iostream>
#include <cmath>
using namespace std;
int main(){
      float x, y, z;
      cout << "x= "; cin >> x;
      cout << "y= "; cin >> y;
      if (x>y) z=x*y*y;
      else z=pow(x,3)+3*y;
      cout << "z= " << z;
      return 0;
}
```

x= 1
y= –2
z= 4

Namuna 2. Berilgan A sonining ishorasini aniqlovchi dastur tuzing.

Yechim. Kiritilayotgan A sonining ishorasi manfiy (ya'ni A<0), aks holda, ishorasiz (ya'ni A=0) yoki musbat (ya'ni, A>0) bo'lishi mumkin.

Masaladan ko'rinadiki, shart inkori (masalan, EMAS (A<0)) hamma vaqt ham yagona holat bo'lmasligi mumkin ekan. Haqiqatan ham, uch xil rangli svetoforning "qizil chirog'i yondi" holatining aks holati "sariq chirog'i yondi" yoki "yashil chirog'i yondi" holatlaridan, "yer osti transporti" (metro) holatining aks holati "yer usti transporti" yoki "suv osti transporti", "suv usti transporti" yoki "havo transporti" holatlaridan, "modda suyuq" holatining aks holati "gaz" yoki "qattiq" holatlaridan biri bo'lishi mumkin. Ma'lumki, bitta shartli o'zlashtirish yoki tarmoqlash operatori bunday holatlarni qamrab ololmaydi.

Masalani hal etish uchun **ichma-ich joylashtirilgan operatorlar**dan foydalanamiz:

Dastur 1	**Natijasi**

`#include <iostream>`
`using namespace std;`
`int main(){`
`  float a; string s;`
`  cout << "A= "; cin >> a;`
`  s = (a<0) ? "manfiy":`
`  (a==0) ?"ishorasiz":"musbat";`
`  cout << a << " soni " << s;`
`  return 0;`
`}`
	A= 21.7
	21.7 soni musbat

Dastur 2	**Natijasi**

`#include <iostream>`
`using namespace std;`
`int main(){`
`  float a;`
`  cout << "A= "; cin >> a;`
`  if (a<0) cout << a << " soni manfiy";`
`  else if (a==0)`
`  cout << a << " soni ishorasiz";`
`  else cout << a << " soni musbat";`
`  return 0;`
`}`
	A= –23.1
	–23.1 soni manfiy

Masala yechimining mantig'i quyidagidan iborat:

a) agar a<0 shart Rost bo'lsa, u holda ekranga **a soni manfiy** ekanligi haqida xabar chiqariladi va dastur ishi yakunlanadi;

b) agar a<0 shart Yolg'on bo'lsa (ya'ni a>0 yoki a=0), u holda keyingi shartni tekshirishga o'tiladi;

c) agar a==0 shart Rost bo'lsa, u holda ekranga **a soni ishorasiz** ekanligi haqida xabar chiqariladi va dastur ishi yakunlanadi;

d) agar a==0 shart Yolg'on bo'lsa (demak, a<0 va a==0 shartlar bajarilmadi va shuning uchun a>0 bo'ladi), u holda ekranga **a soni musbat** ekanligi haqida xabar chiqadi va dastur ishi yakunlanadi.

Masalani **tarmoqlash operatorlari**ni ichma-ich joylashtirmasdan ham hal etish mumkin, masalan, **o'tish operatori**dan quyidagicha foydalanib:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  float a;`
`  cout << "A= "; cin >> a;`
`  if (a<0) {cout << a << " soni manfiy";`
`  goto oxiri; }`
`  if (a==0) cout << a << " soni ishorasiz";`
`  else cout << a << " soni musbat";`
`  oxiri: return 0;`
`}`
	A= 0
	0 soni ishorasiz

Namuna 3. Berilgan x va y sonlarida quyidagi z funksiya qiymatini hisoblang:

$$z = \begin{cases} \sqrt{y-x}, & \text{agar } x < y \\ x+y, & \text{agar } x = y \\ \sqrt{x-y}, & \text{aks holda} \end{cases}$$

Yechim. Dastur tuzishda 2 ta shart qatnashishi kerak bo'lgani uchun, zaruratga qarab, uchinchi shartni ham yozib olish mumkin. Ko'rinib turibdiki, aks holda holatiga mos keluvchi shart x>y ko'rinishida yoziladi.

Bu misolda ham 3 ta holat qaralishi kerak, shuning uchun **ichma-ich joylashtirilgan operatorlar**dan foydalanamiz:

Dastur 1	Natijasi

`#include <iostream>`
`#include <cmath>`
`using namespace std;`
`int main(){`
`  float x, y, z;`
`  cout << "x= "; cin >> x;`
	x= 1.5
	y= 2.5
	z= 1

```
      cout << "y= "; cin >> y;
      z = (x<y) ? pow(y - x,0.5):
          (x==y) ? x + y : pow(x - y,0.5);
      cout << "z= " << z;
      return 0;
}
```

Dastur 2	Natijasi

```
#include <iostream>
#include <cmath>
using namespace std;
int main(){
      float x, y, z;
      cout << "x= "; cin >> x;
      cout << "y= "; cin >> y;
      if (x<y) z = pow(y - x,0.5);
      else if (x==y) z = x + y ;
           else z = pow(x - y,0.5);
      cout << "z= " << z;
      return 0;
}
```

x= –5.7
y= –5.7
z= –11.4

Bu masalani ham **tarmoqlash operatorlari**ni ichma-ich joylashtirmasdan hal etish mumkin.

Namuna 4. Berilgan x soni uchun quyidagi funksiyani hisoblang:

$$A = \sqrt{\frac{x^2 - 1}{||x + 10| - 5|}}$$

Yechim. Bu kabi misollarda funksiyaning aniqlanish sohasi o‘rganiladi. Hisoblanayotgan funksiya, o‘z navbatida, bir nechta funksiyadan iborat bo‘lishi ham mumkin.

Berilgan ifoda kasrli funksiya va kasr darajali funksiyalardan iborat. Ma’lumki, kasrli funksiyada maxraj 0 ga teng bo‘lganda nisbatni hisoblab bo‘lmaydi. Kasr darajali funksiya bir ko‘rayotgan holatda juft darajali ildiz chiqarish bilan bog‘liq bo‘lib, ildiz ostidagi son manfiy bo‘lsa, hisoblash mumkin bo‘lmaydi. Maxrajdagi funksiya bir yoki bir necha argumentda 0 ga aylanishi mumkin. Masalan, yuqoridagi holatda maxrajdagi ||x + 10| – 5| funksiya x= –5 va x= –15 bo‘lganda 0 ga aylanadi.

Ba’zan funksiyani 0 ga aylantiradigan argument qiymatlarini osonlikcha aniqlab bo‘lmaydi. Shu sababli masalani hal etish uchun dasturda maxrajning 0 ga tengligini tekshirish kifoya!

Ildiz ostidagi funksiyaning manfiy bo‘lishiga sabab bo‘ladigan argument qiymatlarini aniqlash tengsizliklar bilan bog‘liq bo‘lib, ko‘pincha argument qiymatlarini umuman aniqlash imkoniyati bo‘lmasligi ham mumkin. Bu holda ham, qulaylik uchun, ildiz ostidagi funksiyaning manfiyligini tekshirib qo‘yish qulay. Misolimizda maxrajdagi funksiya manfiy emas, shu sababli suratdagi funksiyaning manfiyligini tekshirish kifoya.

Dastur sodda bo‘lishi uchun ikkala (0 ga aylanish `abs(x+10)-5==0` va manfiylik `x*x-1<0`) shartni YOKI mantiqiy amali yordamida bitta shart sifatida birlashtiramiz.

Dastur	Natijasi

`#include <iostream>`
`#include <cmath>`
`using namespace std;`
`int main(){`
`  long double x, a;`
`  cout << "x= "; cin >> x;`
`  if (abs(x+10)-5==0 || x*x-1<0)`
`  cout << "Hisoblash mumkin emas!";`
`  else {a = pow((x*x-1)/abs(abs(x+10)-5),0.5);`
`  cout << "A= " << a;}`
`  return 0;`
`}`
	x= –1
	A= 0

Namuna 5. Berilgan a son musbat bo‘lsa, u holda uning kvadratini va kvadrat ildizini chiqaring, aks holda hech qanday natija chiqarmang.

Yechim. Masalada bitta holat, ya’ni a sonining musbatligi qaralishi yetarli bo‘lib, faqat musbat bo‘lsagina amallar bajariladi. Shuning uchun shartli o‘zlashtirish operatoridan foydalanib bo‘lmaydi, chunki uning barcha qismlari to‘liq bo‘lishi shart (C++ tilining oxirgi naqllarida bu kamchilik bartaraf etilgan). Bu holda dasturni **qisqa tarmoqlash operatori** yordamida yozish qulay.

Dastur	Natijasi

`#include <iostream>`
`#include <cmath>`
`using namespace std;`
`int main(){`
`  float a, kv;`
`  cout << "a= "; cin >> a;`
`  if (a>0){`
`  kv=pow(a,2);`
	a= 7
	7^2= 49
	7^(1/2)= 2.64575

```
        cout << a <<"^2= " << kv << endl;
        kv=pow(a,0.5);
        cout << a <<"^(1/2)= " << kv << endl;}
        return 0;
}
```

Namuna 6. Uchta a, b va c sonlari berilgan. Ular ichida manfiy bo'lmagan sonlarning kvadrat ildizini hisoblang.

Yechim. Bu masalada ham bitta holat uchun amallar bajariladi. Lekin tarmoqlash operatori 3 ta o'zgaruvchining har biri uchun alohida bajariladi. Demak, bu holda ham dasturni 3 ta **qisqa tarmoqlash operatori** yordamida yozish qulay.

Dastur	Natijasi

`#include <iostream>`
`#include <cmath>`
`using namespace std;`
`int main(){`
`  float a, b, c, kvi;`
`  cout << "a= "; cin >> a;`
`  cout << "b= "; cin >> b;`
`  cout << "c= "; cin >> c;`
`  if (a>0){kvi=pow(a,0.5);`
`  cout << a <<"^(1/2)= " << kvi << endl;}`
`  if (b>0){kvi=pow(b,0.5);`
`  cout << b <<"^(1/2)= " << kvi << endl;}`
`  if (c>0){kvi=pow(c,0.5);`
`  cout << c <<"^(1/2)= " << kvi << endl;}`
`  return 0;`
`}`
	a= 5
	b= 5.5
	c= 6
	5^(1/2)= 2.23607
	5.5^(1/2)= 2.34521
	6^(1/2)= 2.44949

Namuna 7. Berilgan a va b sonlardan kattasini aniqlang.

Yechim. C++ tilida ikki sondan kattasini aniqlash **max(a,b)** funksiyasi borligini aytib o'tdik. Undan foydalanib dasturni quyidagicha yozish mumkin:

Dastur 1	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  float a, b, kattasi;`
`  cout << "a= "; cin >> a;`
	a= –5
	b= 0
	–5 va 0 dan kattasi 0

```
      cout << "b= "; cin >> b;
      kattasi = max(a,b);
      cout << a <<" va " << b;
      cout << " dan kattasi " << kattasi;
      return 0;
}
```

Lekin berilgan vazifani shart va tarmoqlash operatori yordamida ham bajarish mumkin.

Dastur 2	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  float a, b, kattasi;`
`  cout << "a= "; cin >> a;`
`  cout << "b= "; cin >> b;`
`  kattasi = (a>b) ? a: b;`
`  cout << a <<" va " << b;`
`  cout << " dan kattasi " << kattasi;`
`  return 0;`
`}`
	a= 5.25
	b= 5.3
	5.25 va 5.3 dan kattasi 5.3

Dastur 3	**Natijasi**

`#include <iostream>`
`using namespace std;`
`int main(){`
`  float a, b, kattasi;`
`  cout << "a= "; cin >> a;`
`  cout << "b= "; cin >> b;`
`  if (a>b) kattasi = a;`
`  else kattasi = b;`
`  cout << a <<" va " << b;`
`  cout << " dan kattasi " << kattasi;`
`  return 0;`
`}`
	a= –5.25
	b= –5.3
	–5.25 va –5.3 dan kattasi –5.25

Namuna 8. Berilgan A sonning B songa nisbatini hisoblang.

Yechim. Bu masala yechimi chiziqli tuzilmali bo‘ladi deb o‘ylash xatodir. Chunki masala mazmuni tagma’nosida shart bo‘lib, u bo‘luvchi B ning 0 ga tengligi bilan bog‘liq. Agar B=0 bo‘lsa, u holda bo‘lish amalini bajarish mumkin emas. Shu sababli B=0 sharti tekshiriladi.

Dastur	Natijasi

#include <iostream>
using namespace std;
int main(){
float a, b, nisbat;
cout << "A= "; cin >> a;
cout << "B= "; cin >> b;
if (b==0)cout<<"0 ga bo'lish mumkin emas";
else {nisbat = a/b; cout << a <<"/" << b;
cout << "= " << nisbat;}
return 0;
}
	a= –5
	b= 0
	0 ga bo'lish mumkin emas

Chiqarilayotgan natija ikki xil turda (sonli va matnli) bo‘lgani uchun bu vazifani faqatgina shartli o‘zlashtirish operatori yordamida bajarib bo‘lmaydi.

Namuna 9. Berilgan a ($a\neq0$), b va c koeffitsiyentlariga ko‘ra $a\cdot x^2 + b\cdot x + c = 0$ kvadrat tenglamani yeching.

Yechim. Kvadrat tenglamaning yechilishi 3 xil holat bilan bog‘liqligi, ya’ni $D=B^2–4\cdot A\cdot C$ diskriminantning ishorasiga bog‘liq bo‘lishi avvalgi boblarda ko‘rilgan edi. Demak, diskriminantning ishorasiga bog‘lab tarmoqlanuvchi dastur tuzilar ekan. Dasturni tarmoqlash operatori ichma-ich joylashgan va joylashmagan hollariga mos tuzamiz.

Dastur 1	Natijasi

#include <iostream>
#include <cmath>
using namespace std;
int main(){
float a, b, c, d;
cout << "a= "; cin >> a;
cout << "b= "; cin >> b;
cout << "c= "; cin >> c;
d = b*b-4*a*c;
if (d<0) cout << "Yechim yo‘q";
else if (d==0){cout<<"Yechim yagona x1=x2= ";
cout << -b/(2*a);}
else {cout<<"Yechim ikkita:"<<endl;
cout<<"x1= "<<-b-pow(d,0.5) << endl;
cout<<"x2= "<<-b+pow(d,0.5) << endl;}
return 0;
}
	a= 1
	b= 2
	c= 3
	Yechim yo‘q

Dastur 2	Natijasi

`#include <iostream>`
`#include <cmath>`
`using namespace std;`
`int main(){`
`  float a, b, c, d;`
`  cout << "a= "; cin >> a;`
`  cout << "b= "; cin >> b;`
`  cout << "c= "; cin >> c;`
`  d = b*b-4*a*c;`
`  if (d<0){cout<<"Yechim yo'q"; goto oxiri;}`
`  if (d==0){cout<<"Yechim yagona x1=x2= ";`
`  cout << -b/(2*a);}`
`  else {cout<<"Yechim ikkita:"<<endl;`
`  cout<<"x1= "<<-b-pow(d,0.5) << endl;`
`  cout<<"x2= "<<-b+pow(d,0.5) << endl;}`
`  oxiri: return 0;`
`}`
	a= 1
	b= –5
	c= 4
	Yechim ikkita:
	x1= 2
	x2= 8

Namuna 10. Berilgan butun A ($0 \le A \le 9999$) sonining xonalari (razryadlari) sonini aniqlang.

Yechim. Masalani bir necha usulda yechamiz.

1-usul. Masalani yechishning eng sodda usuli tarmoqlash operatoridan bir necha marta foydalanish bilan bog'liq. Bunda sonning qaysi oraliqda yotishi tekshiriladi.

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  unsigned int a;`
`  cout << "A= "; cin >> a;`
`  if(0<=a&&a<=9)cout <<"1 xonali";`
`  if(10<=a&&a<=99)cout <<"2 xonali";`
`  if(100<=a&&a<=999)cout <<"3 xonali";`
`  if(1000<=a&&a<=9999)cout <<"4 xonali";`
`  if(a>9999)cout <<"Adashdingiz!";`
`  return 0;`
`}`
	a= 1963
	4 xonali

2-usul. Masalani mantiqiy va arifmetik ifoda orqali echamiz. Quyida ifoda qiymatini N o‘zgaruvchiga o‘zlashtiramiz:

N=(A>=0)+(A>9)+(A>99)+(A>999)+(A>9999).

Ma’lumki, C++ tilida mantiqiy mulohaza qiymati 0 yoki 1 bo‘ladi. Shu sababli A soniga qo‘yilgan shartlar qiymatini hisoblaymiz:

A soni 1 xonali: (A>=0)=1, (A>9)=0, (A>99)=0, (A>999)=0, (A>9999)=0, ya’ni N=1;

A soni 2 xonali: (A>=0)=1, (A>9)=1, (A>99)=0, (A>999)=0, (A>9999)=0, ya’ni N=2;

A soni 3 xonali: (A>=0)=1, (A>9)=1, (A>99)=1, (A>999)=0, (A>9999)=0, ya’ni N=3;

A soni 4 xonali: (A>=0)=1, (A>9)=1, (A>99)=1, (A>999)=1, (A>9999)=0, ya’ni N=4;

A soni 9999 dan katta: (A>=0)=1, (A>9)=1, (A>99)=1, (A>999)=1, (A>9999)=1, ya’ni N=5.

U holda tanlash operatoridan foydalanib tuzilgan quyidagi dastur o‘rinli:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  unsigned int a, n;`
`  cout << "A= "; cin >> a;`
`  n=(a>=0)+(a>9)+(a>99)+(a>999)+(a>9999);`
`  switch (n){`
`case 1: cout << " 1 xonali " <<endl; break;`
`case 2: cout << " 2 xonali " <<endl; break;`
`case 3: cout << " 3 xonali " <<endl; break;`
`case 4: cout << " 4 xonali " <<endl; break;`
`default: cout << "Adashdingiz!" <<endl;`
`  }`
`  return 0;`
`}`
	a= 721
	3 xonali

Masalani faqat tarmoqlash operatorlari yordamida ham yechish mumkin edi. Bu usulda N o‘zgaruvchi ifodasidagi shartlar yana qo‘shimcha shartlar bilan qatnashadi.

N o‘zgaruvchi uchun boshqa ko‘rinishdagi ifodalarni hosil qilish ham mumkin, masalan:

N=(A/1>=0)+(A/10>0)+(A/100>0)+(A/1000>0)+(A/10000>0).

3-usul. Masalani yechishda butun bo‘lish amali va tarmoqlash operatoridan foydalanamiz:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  unsigned int a, n=0;`
`  cout << "A= "; cin >> a;`
`  if(a>=0)n++; a=a/10;`
`  if(a>0)n++;  a=a/10;`
`  if(a>0)n++;  a=a/10;`
`  if(a>0)n++;  a=a/10;`
`  if(a<0||a>0)cout << "Adashdingiz!";`
`  else cout << n <<" xonali";`
`  return 0;`
`}`
	a= 2107
	4 xonali

Dasturda n soni inkrementi qo'llanildi. Bu usulda takrorlangan bir xil amallarning bajarilishi umumlashtirish imkonini beradi.

Namuna 11. Berilgan haqiqiy x soni uchun quyidagi butun qiymatli A funksiyani hisoblang.

$$A = \begin{cases} 21, \text{agar x soni } (-\infty; 0) \text{ oraliqda} \\ 7, \text{agar x soni } [0; 1), [2; 3), [4; 5), \ldots \text{ oraliqlarning birortasida} \\ 63, \text{agar x soni } [1; 2), [3; 4), [5; 6), \ldots \text{ oraliqlarning birortasida} \end{cases}$$

Yechim. Kerakli shartlarni yozish murakkab emas, masalan, (–∞; 0) oraliqni x<0 kabi, [0; 1) oraliqni 0≤x<1 kabi, [1; 2) oraliqni 1≤x<2 kabi yoza olamiz.

Lekin ikkinchi va uchinchi qiymatlarni aniqlashda sohalar soni chekli bo'lmagani uchun barcha shartlarni yozib chiqishning iloji yo'q. Agar sohalar soni chekli 10000 tagina oraliq bo'lsa ham, yozib chiqishni tasavvur qilib bo'lmaydi.

Demak, boshqa yo'l tutish kerak. Ishni tahlildan boshlaymiz.

x soni	x ning butun qismi	x soni	x ning butun qismi
[0; 1) oraliqda	0	[1; 2) oraliqda	1
[2; 3) oraliqda	2	[3; 4) oraliqda	3
[4; 5) oraliqda	4	[5; 6) oraliqda	5
…	…	…	…

Qonuniyatni aniqladik: ikkinchi shartga mos oraliqlardagi barcha x sonlarining butun qismi juft, uchinchi shartga mos oraliqlardagi barcha x sonlarining butun qismi toq ekan.

O'zgaruvchi x soni uchun `long double` turni tanlab, aniqlangan qonuniyat va olingan xulosa asosida quyidagi dasturni tuzamiz:

Dastur	Natijasi

`#include <iostream>`
`#include <cmath>`
`using namespace std;`
`int main(){`
`long double x;`
`int a;`
`long long b;`
`cout << "x= "; cin >> x;`
`b=trunc(x)%2;`
`if (x<0)a=21;`
`else if (b==0)a=7;`
`else a=63;`
`cout << "A= " << a;`
`return 0;`
`}`
	x= 1234567890.98765
	A= 7

3-§. TARMOQLANUVCHI ALGORITMLI MASALALAR

Quyidagi misol va masalalar tarmoqlanuvchi algoritmli dastur (qisqa aytganda, tarmoqlanuvchi dastur) tuzishga mo'ljallangan. Eslatib o'tamiz, tarmoqlanish shartlari misol yoki masala mazmunida oshkora yoki tagma'noda aks etishi mumkin. Misol yoki masala yechimi yetarli aniqlikda chiqishi uchun haqiqiy sonlarni **long double** turida tavsiflashni tavsiya etamiz.

= A =

Tarmoqlanuvchi 1. Berilgan x uchun quyidagi funksiyani hisoblang:

$$y = \begin{cases} -x + 5, & x < 1 \\ 0, & \text{aks holda} \end{cases}$$

Tarmoqlanuvchi 2. Berilgan x uchun quyidagi funksiyani hisoblang:

$$y = \begin{cases} \sqrt{x + 2}, & x \geq -2 \\ x^2 + 5 \cdot x - 7, & \text{aks holda} \end{cases}$$

Tarmoqlanuvchi 3. Berilgan x uchun quyidagi funksiyani hisoblang:

$$y = \begin{cases} \dfrac{x + 5}{x}, & x \geq 3 \\ |x^3|, & \text{aks holda} \end{cases}$$

Tarmoqlanuvchi 4. Berilgan x uchun quyidagi funksiyani hisoblang:

$$y = \begin{cases} \sin x - \sqrt{-x}, & x < 0 \\ 5 - \cos x, & \text{aks holda} \end{cases}$$

Tarmoqlanuvchi 5. Berilgan x uchun quyidagi funksiyani hisoblang:

$$y = \begin{cases} 1 - \dfrac{5+x}{x^2}, & x \geq 1 \\ 3 \cdot \operatorname{arctg} x - 7 \cdot \cos \pi x, & \text{aks holda} \end{cases}$$

Tarmoqlanuvchi 6. Berilgan x uchun quyidagi funksiyani hisoblang:

$$y = \begin{cases} x \cdot \sin(x^2 + 1) + 5, & x < 0 \\ \sqrt[4]{x^3 + \sqrt{\pi \cdot x}}, & x \geq 0 \end{cases}$$

Tarmoqlanuvchi 7. Berilgan x uchun quyidagi funksiyani hisoblang:

$$y = \begin{cases} \sqrt{x - \cos(\pi + x) + 11}, & x \geq -10 \\ \sqrt[4]{1 - x^5}, & x < -10 \end{cases}$$

Tarmoqlanuvchi 8. Berilgan x uchun quyidagi funksiyani hisoblang:

$$y = \begin{cases} 2 \cdot \sqrt{e^x - 1}, & x \geq 0 \\ \dfrac{x}{1 - x^3}, & x < 0 \end{cases}$$

Tarmoqlanuvchi 9. Berilgan x uchun quyidagi funksiyani hisoblang:

$$y = \begin{cases} \sqrt{\ln(x+1)} + 2, & x > 5 \\ e^{3 \cdot x} + \sqrt{5 - x}, & x \leq 5 \end{cases}$$

Tarmoqlanuvchi 10. Berilgan x uchun quyidagi funksiyani hisoblang:

$$y = \begin{cases} \sqrt{1 + |1 - x|}, & x < 100 \\ \sin(2 \cdot x + \pi) + \cos^3 x^5, & x \geq 100 \end{cases}$$

Tarmoqlanuvchi 11. Berilgan x uchun quyidagi funksiyalarni hisoblang:

a) $f = \dfrac{1}{1-x}$	b) $g = \dfrac{1-x}{25+x}$	c) $h = \dfrac{x^2}{x^2-25}$

Tarmoqlanuvchi 12. Berilgan x uchun quyidagi funksiyalarni hisoblang:

a) $f = \dfrac{5 \cdot x}{\lvert 100 - x \rvert}$	b) $g = \dfrac{x}{\sqrt{\lvert 25 + x \rvert}}$	c) $h = \dfrac{x^2 - 2{,}5}{x+5}$

Tarmoqlanuvchi 13. Berilgan x uchun quyidagi funksiyalarni hisoblang:

a) $f = \frac{\sqrt{1+x}}{9}$	b) $g = \frac{9}{\sqrt[3]{1+x}}$	c) $h = \frac{7}{\sqrt[4]{x+7}}$

Tarmoqlanuvchi 14. Berilgan x uchun quyidagi funksiyalarni hisoblang:

a) $f = \frac{\sqrt{x-7}}{\sqrt{x^2+9}}$	b) $g = \frac{19}{\lvert 21-x\rvert - 2}$	c) $h = \frac{x}{\sqrt[10]{\lvert x^2-9\rvert}}$

Tarmoqlanuvchi 15. Berilgan x uchun quyidagi funksiyalarni hisoblang:

a) $f = \frac{1}{x} + \frac{x}{7}$	b) $g = \frac{2}{\sqrt[3]{x^4+x}}$	c) $h = \frac{20}{\sqrt[3]{x^2-4}}$

Tarmoqlanuvchi 16. Berilgan x uchun quyidagi funksiyalarni hisoblang:

a) $f = \frac{7}{x} + \frac{\sqrt{\lvert x\rvert}}{21}$	b) $g = \frac{2}{\sqrt[3]{x^5-32}}$	c) $h = \frac{\sqrt{1+x}}{\sqrt[4]{x^2+20}}$

Tarmoqlanuvchi 17. Berilgan x uchun quyidagi funksiyalarni hisoblang:

a) $f = \sqrt{\left\lvert \frac{1+x}{x+21} \right\rvert}$	b) $g = \frac{1}{\lvert x\rvert} + \frac{\sqrt{\lvert x\rvert}}{7}$	c) $h = \frac{1}{x^2+5} + \frac{\sqrt{x}}{19}$

Tarmoqlanuvchi 18. Berilgan x uchun quyidagi funksiyalarni hisoblang:

a) $f = \sqrt{\frac{-2-x}{19}}$	b) $g = \sqrt{\frac{21+x}{1+\lvert x\rvert}}$	c) $h = \frac{x}{\sqrt{\lvert x\rvert} - 5}$

Tarmoqlanuvchi 19. Berilgan x va y uchun quyidagi funksiyalarni hisoblang:

a) $f = \sqrt{\frac{\lvert 2+y\rvert}{\lvert x^2+5x+4\rvert}}$	b) $g = \sqrt{\frac{19-7\cdot y}{x^2+2x+3}}$	c) $h = \frac{-23\cdot y}{\sqrt{\lvert (x+4)\cdot(x-5)\rvert}}$

Tarmoqlanuvchi 20. Berilgan x va y uchun quyidagi funksiyalarni hisoblang:

a) $f = \frac{x-y}{1-x^2\cdot y^4}$	b) $g = \frac{-23}{x^2+y^2} + \frac{\sqrt{\lvert x-y\rvert}}{2}$	c) $h = \frac{9\cdot x^2 + 6\cdot y^3}{\sqrt{\lvert 1+3\cdot x\cdot y\rvert}}$

Tarmoqlanuvchi 21. A soni berilgan. Agar A soni musbat bo‘lsa, u holda A soniga 21 sonini qo‘shib, aks holda A sonining o‘zini chiqaring.

Tarmoqlanuvchi 22. A soni berilgan. Agar A soni manfiy bo‘lsa, u holda A soniga 22 sonini qo‘shib, aks holda A sonidan 22 sonini ayirib chiqaring.

Tarmoqlanuvchi 23. A soni berilgan. Agar A soni 21 ga teng bo‘lsa, u holda A sonidan 23 sonini ayirib, aks holda A soniga 23 sonini ko‘paytirib chiqaring.

Tarmoqlanuvchi 24. A butun son berilgan. Agar A soni juft bo‘lsa, u holda A sonining 2 soniga nisbatini, aks holda A sonini 3 ga ko‘paytmasini chiqaring.

Tarmoqlanuvchi 25. A va B sonlari berilgan. Ulardan kattasini tartib raqamini chiqaring. Masalan, A=5 va B=7 bo‘lsa, javob 2.

Tarmoqlanuvchi 26. A va B sonlari berilgan. Avval ulardan kichigini, keyin ulardan kattasini chiqaring.

Tarmoqlanuvchi 27. A va B sonlari berilgan. Agar sonlar teng bo‘lmasa ularga 21 sonini qo‘shib, agar teng bo‘lsa ulardan 7 sonini ayirib chiqaring.

Tarmoqlanuvchi 28. A va B sonlari berilgan. Agar sonlar qiymati 100 taga farq qilsa, ularni ko‘paytmasini, aks holda yig‘indisini chiqaring.

Tarmoqlanuvchi 29. A va B sonlari berilgan. Agar B son A sondan kichik bo‘lmasa, u holda B sonni nol bilan almashtirib, aks holda sonlarni o‘zgarishsiz chiqaring.

Tarmoqlanuvchi 30. Noldan farqli A va B butun sonlar berilgan. Agar B son A songa bo‘linsa “B:A”, aks holda “B..A” chiqaring.

Tarmoqlanuvchi 31. Noldan farqli A va B butun sonlar berilgan. Agar B son A songa bo‘linsa bo‘linmani, aks holda bo‘linma va qoldiqni chiqaring.

Tarmoqlanuvchi 32. A va B sonlari berilgan. Agar B sonni A songa bo‘lish mumkin bo‘lsa bo‘linmani, aks holda ularning yig‘indisini chiqaring.

Tarmoqlanuvchi 33. A, B va d sonlari berilgan. Agar d soni $A \cdot x+B=0$ tenglamaning yechimi bo‘lsa “+1”, aks holda “–1” javobni chiqaring.

Tarmoqlanuvchi 34. A, B, C va d sonlari berilgan. Agar d soni $A \cdot x^2+B \cdot x+C=0$ tenglamaning yechimi bo‘lsa “Ha”, aks holda “Yo‘q” so‘zini chiqaring.

Tarmoqlanuvchi 35. A, B, C, D va d sonlari berilgan. Agar d soni $A \cdot x^3+B \cdot x^2+C \cdot x+D=0$ tenglamaning yechimi bo‘lsa “Yechim”, aks holda “Yechim emas” jumlani chiqaring.

Tarmoqlanuvchi 36. A, B, x0 va y0 sonlari berilgan. Agar (x0; y0) nuqta $y=A \cdot x+B$ funksiya grafigiga tegishli bo‘lsa “Tegishli”, aks holda “Tegishli emas” jumlani chiqaring.

Tarmoqlanuvchi 37. A, B, C, x0 va y0 sonlari berilgan. Agar (x0; y0) nuqta $y=A \cdot x^2+B \cdot x+C$ funksiya grafigiga tegishli bo‘lsa “YES”, aks holda “NO” so‘zini chiqaring.

Tarmoqlanuvchi 38. A, B, C, D, x0 va y0 sonlari berilgan. Agar (x0; y0) nuqta $y=A \cdot x^3+B \cdot x^2+C \cdot x+D$ funksiya grafigiga tegishli bo‘lsa “OK”, aks holda “NOT” so‘zini chiqaring.

Tarmoqlanuvchi 39. Radiusi r va markazi (x0; y0) nuqtada bo‘lgan doira berilgan. Agar (x1; y1) nuqta doiraga tegishli bo‘lsa “YES”, aks holda “NO” so‘zini chiqaring.

Tarmoqlanuvchi 40. Ikki xonali natural son berilgan. Son raqamlaridan kattasini chiqaring.

= B =

Tarmoqlanuvchi 41. Berilgan x uchun quyidagi funksiyani hisoblang:

$$y = \begin{cases} \sqrt{x-5}, & x > 5 \\ \sqrt{5-x}, & 0 < x \le 5 \\ x^2, & x < 0 \end{cases}$$

Tarmoqlanuvchi 42. Berilgan x uchun quyidagi funksiyani hisoblang:

$$y = \begin{cases} 63 \cdot (x+21), & |x| < 5 \\ -\sin^3 x + 19, & x \ge 5 \\ -\sqrt[5]{x+7}, & x \le -5 \end{cases}$$

Tarmoqlanuvchi 43. Berilgan x uchun quyidagi funksiyani hisoblang:

$$y = \begin{cases} 1, & x > 0 \\ 0, & x = 0 \\ -1, & \text{aks holda} \end{cases}$$

Tarmoqlanuvchi 44. Berilgan x uchun quyidagi funksiyani hisoblang:

$$y = \begin{cases} x, & x > 0 \\ 0, & x = 0 \\ -x, & \text{aks holda} \end{cases}$$

Tarmoqlanuvchi 45. Berilgan x uchun quyidagi funksiyani hisoblang:

$$y = \begin{cases} 10 \cdot \sin x, & |x| > 1 \\ \dfrac{1-x}{|x|}, & 0 < x \le 1 \\ -5, & \text{aks holda} \end{cases}$$

Tarmoqlanuvchi 46. Berilgan x uchun quyidagi funksiyani hisoblang:

$$y = \begin{cases} \sqrt{x}, & x > 0 \\ \sqrt{3-x}, & x \le -3 \\ \sqrt{x+3}, & \text{aks holda} \end{cases}$$

Tarmoqlanuvchi 47. Berilgan x uchun quyidagi funksiyani hisoblang:

$$y = \begin{cases} \sqrt{|x-21|}, & x < -3 \\ \cos(\pi + x), & -7 \le x \le -3 \\ \text{arctg}x, & \text{aks holda} \end{cases}$$

Tarmoqlanuvchi 48. Berilgan x uchun quyidagi funksiyani hisoblang:

$$y = \begin{cases} \frac{x}{|x|}, & x > 0 \\ \frac{x}{|x|}, & x < 0 \\ 0, & \text{aks holda} \end{cases}$$

Tarmoqlanuvchi 49. Berilgan x uchun quyidagi funksiyani hisoblang:

$$y = \begin{cases} x^2 + e^x, & -2 \le x < 5 \\ x^2 + 4 \cdot x + 5, & x < -2 \\ 2 - x, & \text{aks holda} \end{cases}$$

Tarmoqlanuvchi 50. Berilgan x va y uchun quyidagi funksiyani hisoblang:

$$y = \begin{cases} x^2 - y, & x < y \\ x^2 + y^2, & x = y \\ x \cdot y, & \text{aks holda} \end{cases}$$

Tarmoqlanuvchi 51. Berilgan x uchun quyidagi funksiyani hisoblang:

$$y = \begin{cases} \frac{\sin 5x + 47}{x \cdot (x - 11)}, & 0 < x < 11 \\ \frac{\cos(3 + x)}{x - 10}, & x \ge 11 \\ \sqrt{-x + 1}, & \text{aks holda} \end{cases}$$

Tarmoqlanuvchi 52. Berilgan x uchun quyidagi funksiyani hisoblang:

$$y = \begin{cases} \sqrt{1 - x^2}, & |x| < 1 \\ \sqrt{x^2 + 2 \cdot |x| + 1}, & |x| = 1 \\ \frac{50}{\sqrt{x^2 - 1}}, & \text{aks holda} \end{cases}$$

Tarmoqlanuvchi 53. Berilgan x va y uchun quyidagi funksiyani hisoblang:

$$y = \begin{cases} \sqrt{y - x}, & x < y \\ \sqrt{x - y}, & x > y \\ \sqrt{x \cdot y}, & \text{aks holda} \end{cases}$$

Tarmoqlanuvchi 54. Berilgan x va y uchun quyidagi funksiyani hisoblang:

$$y = \begin{cases} \sqrt{1 + y^2 - x^2}, & x^2 < y^2 \\ \sqrt{x^2 + 3 \cdot y^2}, & x = y \\ \sin x^2 + \cos y^2, & \text{aks holda} \end{cases}$$

Tarmoqlanuvchi 55. Berilgan x va y uchun quyidagi funksiyani hisoblang:

$$y = \begin{cases} \dfrac{\sqrt{-x-y}}{1-x-y}, & x + y < 0 \\ \sqrt{x^3 + y^3}, & x + y > 0 \\ x + y, & \text{aks holda} \end{cases}$$

Tarmoqlanuvchi 56. Berilgan x uchun quyidagi funksiyani hisoblang:

$$y = \begin{cases} \sqrt{5 - x - y}, & |x + y| < 3 \\ \dfrac{x \cdot y}{x + y}, & |x + y| > 3 \\ 0, & \text{aks holda} \end{cases}$$

Tarmoqlanuvchi 57. Berilgan x uchun quyidagi funksiyani hisoblang:

$$y = \begin{cases} x^2 + 1, & x \cdot y < 0 \\ \sqrt{1 - x^4 \cdot y^3}, & x \cdot y = 0 \\ \operatorname{arctg}(x + y), & \text{aks holda} \end{cases}$$

Tarmoqlanuvchi 58. Berilgan x uchun quyidagi funksiyani hisoblang, bunda kiruvchi K1: –5; K2: 7; K3: –9; K4: 11; K5: –13 qiymatlardan xatolikka olib keladigan qiymatni aniqlab, xatolikni bartaraf etish shartini dasturga kiriting:

$$y = \begin{cases} \dfrac{x + 11}{x^2}, & x > 6 \\ \dfrac{x^2 + 5}{|x + 9|}, & x \leq 6 \end{cases}$$

Tarmoqlanuvchi 59. Berilgan x uchun quyidagi funksiyani hisoblang:

$$y = \begin{cases} \dfrac{e^x}{x^2 + 1}, & x < 0 \\ \dfrac{1 + \sin x}{\sqrt{x}}, & \text{aks holda} \end{cases}$$

Xatolikka olib keladigan qiymatni aniqlab, xatolikni bartaraf etish shartini dasturga kiriting.

Tarmoqlanuvchi 60. Berilgan x uchun quyidagi funksiyani hisoblang:

$$y = \begin{cases} \sqrt{1+x}, & x < 1 \\ \sqrt{|x|-1}, & x \geq 1 \end{cases}$$

Xatolikka olib keladigan qiymatni aniqlab, xatolikni bartaraf etish shartini dasturga kiriting.

Tarmoqlanuvchi 61. Berilgan x uchun quyidagi funksiyalarni hisoblang:

a) $f = \frac{x}{x-1} + \frac{2+x}{\lvert x+1\rvert}$	b) $g = \sqrt{x-1} + \frac{63}{\lvert x\rvert - 1}$	c) $h = \frac{x}{\lvert 1-x\rvert} + 19 \cdot \sqrt{x}$

Tarmoqlanuvchi 62. Berilgan x uchun quyidagi funksiyalarni hisoblang:

a) $f = \frac{21}{\frac{\lvert x-1\rvert}{x+21}}$	b) $g = \frac{2}{x^2-2} + \frac{1}{\sqrt{\lvert x-23\rvert}}$	c) $h = \sqrt{x-4} + \frac{20}{4-x}$

Tarmoqlanuvchi 63. Berilgan x uchun quyidagi funksiyalarni hisoblang:

a) $f = \frac{41 - \sqrt[4]{x}}{1 - e^{-x^2}}$	b) $g = \frac{\sqrt{\lvert x+1\rvert - \lvert x\rvert}}{x^2 - 25}$	c) $h = \frac{x^3}{19 - (x+1)^4} - \frac{1}{x}$

Tarmoqlanuvchi 64. Berilgan x uchun quyidagi funksiyalarni hisoblang:

a) $f = \frac{\sqrt{x} - \sqrt[3]{x}}{1 - x^2}$	b) $g = \frac{\sqrt{x^2 - 25}}{125 - x^2 \cdot \lvert x\rvert}$	c) $h = \frac{3}{\sqrt{\lvert 4 - x^2\rvert}} + \frac{7}{\sqrt[3]{x}}$

Tarmoqlanuvchi 65. Berilgan x uchun quyidagi funksiyalarni hisoblang:

a) $f = \frac{\lvert x\rvert}{\lvert x\rvert - 1} + \frac{11}{\lvert x^2 - 1\rvert}$	b) $g = \sqrt{x+1} + \frac{89}{\sqrt{x}}$	c) $h = \frac{x}{\sin^2 \frac{x}{2} - 1} + \sqrt{x}$

Tarmoqlanuvchi 66. Berilgan x uchun quyidagi funksiyalarni hisoblang:

a) $f = \frac{\sqrt[4]{\sqrt[3]{x} - 2}}{\lvert x+8\rvert}$	b) $g = \frac{1 + \sqrt{\cos(x - \pi)}}{\sin\left(x + \frac{\pi}{6}\right) + 1}$	c) $h = \frac{\frac{2 \cdot \sin x}{1 + 2 \cdot \cos^3 x}}{4 - e^x}$

Tarmoqlanuvchi 67. Berilgan x uchun quyidagi funksiyalarni hisoblang:

a) $f = \frac{\lvert x-1\rvert}{x} + \frac{\sqrt[4]{2+x}}{\lvert x\rvert + 1}$	b) $g = \sqrt{5 \cdot x^2 - x} + \frac{5}{\lvert x\rvert}$	c) $h = \frac{\sqrt{\lvert x\rvert - 5}}{\lvert 25 + x\rvert}$

Tarmoqlanuvchi 68. Berilgan x uchun quyidagi funksiyalarni hisoblang:

a) $f = \frac{\operatorname{tg}(x+1)}{2-\sqrt{x^2+1}}$	b) $g = \frac{\sqrt{e^x-1}}{2+4\cdot \sin x}$	c) $h = \frac{1111}{\sin\lvert x\rvert \cdot \operatorname{arctg}^2 x}$

Tarmoqlanuvchi 69. Berilgan x uchun quyidagi funksiyalarni hisoblang:

a) $f = \frac{x}{e^x-1} + \frac{3}{\sqrt{\lvert x+1\rvert}}$	b) $g = \frac{\sqrt{x^2-25}}{\lvert\lvert x\rvert-19\rvert}$	c) $h = \sqrt{1-x^2} + \frac{4}{\sqrt{-x}}$

Tarmoqlanuvchi 70. Berilgan x uchun quyidagi funksiyalarni hisoblang:

a) $f = \frac{1}{x^2+x} + \frac{5}{2\cdot x+1}$	b) $g = \sqrt{1-\lvert x\rvert} + \operatorname{tg} x$	c) $h = \sqrt{x-\sqrt{x}}$

Tarmoqlanuvchi 71. Berilgan A sonini o‘zining kvadrati bilan taqqoslama ko‘rinishida chiqaring. Masalan, agar A= –0,5 bo‘lsa, javob: –0.5<0.25, agar A= 1 bo‘lsa, javob: 1=1.

Tarmoqlanuvchi 72. Berilgan A, B va C sonlardan kattasini chiqaring (bunda **max** va **min** funksiyalaridan foydalanish mumkin emas).

Tarmoqlanuvchi 73. Berilgan A, B va C sonlardan o‘shish tartibida o‘rtada joylashganini chiqaring (bunda **max** va **min** funksiyalaridan foydalanish mumkin emas).

Tarmoqlanuvchi 74. Berilgan A, B va C sonlardan avval eng kichigini, keyin eng kattasini chiqaring (bunda **max** va **min** funksiyalaridan foydalanish mumkin emas).

Tarmoqlanuvchi 75. Berilgan A, B va C sonlardan ikkita eng kattasi yig‘indisini chiqaring (bunda **max** va **min** funksiyalaridan foydalanish mumkin emas).

Tarmoqlanuvchi 76. Berilgan A, B va C sonlar o‘sish tartibida kiritilgan bo‘lsa, u holda ularning har birini kvadrati bilan, aks holda har birini ishorasini teskarilab chiqaring (bunda **max** va **min** funksiyalaridan foydalanish mumkin emas).

Tarmoqlanuvchi 77. Avtomobil shahardan S kilometr uzoqlikda joylashgan. U V kilometr/soat tezlikda yurib, T soatda shaharga yetib kelolsa, “Shaharda”, aks holda “Yo‘lda” so‘zini chiqaring.

Tarmoqlanuvchi 78. $1 < A \leq 5$ shartni qanoatlantiruvchi A raqam berilgan. Shu raqamni so‘z bilan chiqaring.

Tarmoqlanuvchi 79. Berilgan butun A soni musbat va juft bo‘lsa, u holda shu sonning kvadrat ildizini hisoblang.

Tarmoqlanuvchi 80. Berilgan butun A soni ikki xonali va toq bo‘lsa, u holda shu sonning kubini hisoblang.

Tarmoqlanuvchi 81. Berilgan A soni [B, M] kesmaga tegishli bo‘lsa B sonining kvadratini, aks holda M sonining kvadratini chiqaring.

Tarmoqlanuvchi 82. A va B natural sonlar berilgan. Bu sonlarning biri ikkinchisining kvadrati bo‘lishi yoki bo‘lmasligini aniqlang.

Tarmoqlanuvchi 83. A son berilgan. A son bilan bir qatorda agar A musbat bo'lsa 1 qo'shilgan, manfiy bo'lsa absolyut qiymatiga 2 qo'shilgan, aks holda 100 qo'shilgan qiymatni chiqaring.

Tarmoqlanuvchi 84. A va B sonlar berilgan. Agar ularning ko'paytmasi musbat va kattasidan kichigini ayirmasi 21 dan katta bo'lsa ularning har birining kvadratini, aks holda har birini 100 marta oshirib chiqaring.

Tarmoqlanuvchi 85. A, B va C sonlar berilgan. A < B < C shart bajarilsa A+B–C qiymatini, aks holda C–|A+B| qiymatini chiqaring.

Tarmoqlanuvchi 86. A, B va C sonlar berilgan. Shu sonlardan musbatlari sonini aniqlang.

Tarmoqlanuvchi 87. A, B va C sonlar berilgan. Shu sonlardan musbatlari va manfiylari sonini aniqlang.

Tarmoqlanuvchi 88. Kiritilgan 1 dan 12 gacha bo'lgan butun songa mos oy nomini ekranga chiqaring.

Tarmoqlanuvchi 89. Berilgan A butun sonni 2 ga va 3 ga karraliligini tekshiring. Karralilik xossasiga asosan javobni quyidagi ko'rinishda chiqaring: "2", "3", "2-3", "NO".

Tarmoqlanuvchi 90. Ox sonlar o'qida x1, x2 va x3 nuqtalar berilgan. Koordinatalar boshiga eng yaqin nuqtani va shu nuqtadan koordinata boshigacha bo'lgan masofani aniqlang.

Tarmoqlanuvchi 91. Oxy tekislikda A nuqta (x0, y0) koordinatalari orqali berilgan. Shu nuqta joylashgan kvadrantning tartib raqamini aniqlang.

Tarmoqlanuvchi 92. Oxy tekislikda A nuqta (x0, y0) koordinatalari orqali berilgan. Shu nuqta joylashgan kvadrantning tartib raqamini aniqlang.

Tarmoqlanuvchi 93. $a \cdot x+b \cdot y+c=0$ ($a^2+b^2 \neq 0$) to'g'ri chiziq markazi A(x0,y0) nuqtada bo'lgan R radiusli doira bilan kesishishi yoki kesishmasligini aniqlang.

Tarmoqlanuvchi 94. Ikkilikda 6 xonali A son va o'nlikdagi natural B son berilgan. Shu sonlar teng bo'lsa "TRUE", aks holda ularning farqini chiqaring.

Tarmoqlanuvchi 95. A, B va M tangalarning og'irligi berilgan. Ulardan ikkitasi haqiqiy, ya'ni og'irligi teng, uchunchisining esa og'irligi haqiqiy tangalar o'g'irligidan farqlanadi. Qalbaki tangani aniqlang.

Tarmoqlanuvchi 96. O'nlik sanoq sistemasidagi 300 dan kichik A natural son berilgan. Shu son 2 ning darajasi bo'lishi yoki bo'lmasligini aniqlang.

Tarmoqlanuvchi 97. 7 xonali N son 2 lik sanoq sistemasida berilgan. Shu sonni o'nlikdagi 15 ga bo'linishi yoki bo'linmasligini aniqlang.

Tarmoqlanuvchi 98. Berilgan natural N va x uchun $S=\cos x+\cos 2x+\ldots+\cos Nx$ trigonometrik funksiyalar yig'indisini hisoblang (yo'llanma:

$$\cos x + \cos 2x + \ldots + \cos Nx = \frac{\sin\left(N + \frac{1}{2}\right)x}{2 \cdot \sin\frac{1}{2}x} - \frac{1}{2}$$

Tarmoqlanuvchi 99. Berilgan natural N va x uchun $S = \sin x + \sin 2x + \ldots + \sin Nx$ trigonometrik funksiyalar yig'indisini hisoblang. Bunda yo'llanma:

$$\sin x + \sin 2x + \ldots + \sin Nx = \frac{\sin\left(N + \frac{1}{2}\right)x \cdot \sin\frac{N}{2}x}{\sin\frac{1}{2}x}$$

Tarmoqlanuvchi 100. Berilgan natural N va x uchun $P = \cos x \cdot \cos 2x \cdot \cos 4x \cdot \ldots \cdot \cos 2^N x$ trigonometrik funksiyalar ko'paytmasini hisoblang. Bunda yo'llanma:

$$\cos x \cdot \cos 2x \cdot \cos 4x \cdot \ldots \cdot \cos 2^N x = \frac{\sin 2^{N+1}x}{2^{N+1} \cdot \sin x}$$

= C =

Tarmoqlanuvchi 101. Berilgan x ning qiymatlarida quyidagi A funksiya qiymatini 10^{-1} aniqlikda hisoblang.

$$A = \frac{\frac{9 + 5x}{19 - 2x^2}}{\sqrt{2 - x^2} + \sqrt{5 - x^4}}$$

Tarmoqlanuvchi 102. Berilgan x ning qiymatlarida quyidagi B funksiya qiymatini 10^{-2} aniqlikda hisoblang.

$$B = \frac{\sqrt[4]{21 + 23\sqrt{7x^3 - x}}}{\sqrt{\sin x + \cos^2 x} + \sqrt{\cos x + \sin^2 x}}$$

Tarmoqlanuvchi 103. Berilgan x ning qiymatlarida quyidagi R funksiya qiymatini 10^{-3} aniqlikda hisoblang.

$$R = \frac{9x - 23\sqrt{x^3 + 5x + 1}}{\sqrt{\frac{1}{x} + \mathrm{tg}^2 x} + \sqrt{\mathrm{ctg}^2 x + \frac{5}{x + 4}}}$$

Tarmoqlanuvchi 104. Berilgan x soni uchun quyidagi A funksiyani hisoblang.

$$A = \begin{cases} 50, & \text{agar x soni } (-\infty; 0) \text{ oraliqda} \\ 2, & \text{agar x soni } [0; 3), [6; 9), [12; 15), \ldots \text{ oraliqlarning birortasida} \\ 19, & \text{agar x soni } [3; 6), [9; 12), [15; 18), \ldots \text{ oraliqlarning birortasida} \end{cases}$$

Tarmoqlanuvchi 105. Berilgan x soni uchun quyidagi A funksiyani hisoblang.

$$A = \begin{cases} 0, & \text{agar x soni } (-\infty; 0) \text{ oraliqda} \\ 5, & \text{agar x soni } [0; 4), [8; 12), [16; 20), \ldots \text{ oraliqlarning birortasida} \\ 9, & \text{agar x soni } [4; 8), [12; 16), [20; 24), \ldots \text{ oraliqlarning birortasida} \end{cases}$$

Tarmoqlanuvchi 106. A, B va C sonlar berilgan. Agar sonlardan biror ikkitasi teng, bittasi farqli bo'lsa "YES", aks holda "NO" so'zlarini chiqaring.

Tarmoqlanuvchi 107. A, B va C sonlar berilgan. Agar sonlardan biror ikkitasi teng, bittasi farqli bo'lsa, u holda farqli sonning tartib raqamini, aks holda "–1" chiqaring.

Tarmoqlanuvchi 108. A, B, C va D sonlar berilgan. Agar sonlardan biror uchtasi teng, bittasi farqli bo'lsa, u holda farqli sonning tartib raqamini, aks holda "–1" chiqaring.

Tarmoqlanuvchi 109. 4 xonali A natural son berilgan. Shu son raqamlarining o'rnini almashtirib hosil qilinadigan eng katta sonni aniqlang.

Tarmoqlanuvchi 110. Berilgan 999 dan katta bo'lmagan natural A soni palindrom (chapdan ham, o'ngdan ham bir hil o'qiladigan) son bo'lsa, u holda "A=P", aks holda "AN" yozuvini chiqaring.

Tarmoqlanuvchi 111. Butun A ($1 \le A \le 9999$) son berilgan. A soni xususiyatini quyidagi kabi izohlab chiqaring: "bir xonali juft", "ikki xonali toq" va hokazo.

Tarmoqlanuvchi 112. Berilgan A butun sonni 2 ga, 3 ga va 5 ga karraliligini tekshiring. Karralilik xossasiga asosan javobni quyidagi ko'rinishda chiqaring: "2-3-5", "2-3", "2-5", "3-5", "2", "3", "5", "NO".

Tarmoqlanuvchi 113. Rim raqamlari orqali berilgan 5 xonali A sonni 10 lik sanoq sistemasidagi qiymatini aniqlang.

Tarmoqlanuvchi 114. N ($1 \le N \le 1000$) natural son berilgan. Agar N=A+B, bu yerda A va B juft natural sonlar, ko'rinishida tasvirlash mumkin bo'lsa, "Ha", aks holda "Yo'q" javobini chiqaring.

Tarmoqlanuvchi 115. N tiyinni eng kam sondagi 3 tiyinlik va 2 tiyinlik yordamida aniq maydalash mumkin bo'lsa 3 va 2 tiyinliklar sonini, aks holda "Mumkin emas" iborasini chiqaring.

Tarmoqlanuvchi 116. A, B, C va N sonlari berilgan. A, B, C sonlarining joyini almashtirmasdan qo'shish va ayirish amallarini ular orasiga shunday qo'yingki, natija N soniga teng bo'lsin. Buning iloji bo'lmasa, u holda "Impossible" yozuvi chiqsin. Masalan: N=15, A=10, B=7, C=2 bo'lsa, javob: 15=10+7–2.

Tarmoqlanuvchi 117. Yil tartib raqami bo'lgan A son berilgan. Shu yilni oddiy (365 kunlik) yoki kabisa (366 kunlik) yil ekanligini aniqlang. Kabisa yil tartib raqami 100 ga bo'linib, 400 ga bo'linmaydiganlardan tashqari, barcha 4 ga bo'linadigan tartib raqamli yillardir. Masalan, 300, 1300 va 1900 yillar oddiy, 1200 va 2000 yillar kabisa yillardir.

Tarmoqlanuvchi 118. Yuqoriga A metr/sekund tezlik bilan tik otilgan jism B balandlikka chiqa olishi yoki chiqa olmasligini aniqlang (yo'llanma: v tezlik bilan yuqoriga tik otilgan jism chiqa oladigan balandlik h $\approx \frac{v^2}{2 \cdot g}$, bu yerda g≈9,8 metr/sekund^2).

Tarmoqlanuvchi 119. Gorizontga nisbatan A gradus burchak ostida B metr/sekund tezlik bilan otilgan jism G m balandlikda necha marta bo'lishini aniqlang (yo'llanma: α burchak ostida v tezlik bilan otilgan jism chiqa oladigan balandlik h $\approx \frac{v^2 \cdot sin^2 \alpha}{2 \cdot g}$, bu yerda g ≈9,8 metr/sekund^2).

Tarmoqlanuvchi 120. Berilgan butun a va b sonlar kvadrat tenglamaning yechimlari bo'lsa, mos kvadrat tenglamaning uchhad ko'rinishini chiqaring. Masalan, a=b=1 bo'lsa, u holda javob: x^2-2x+1=0.

Tarmoqlanuvchi 121. Berilgan a, b va c koeffitsiyentga asosan $a \cdot x^4 + b \cdot x^2 + c = 0$ tenglamani yeching.

Tarmoqlanuvchi 122. Kvadrat va doiraninig yuzasi berilgan. Doira kvadrat ichida to'liq yotishi yoki yotmasligini aniqlang.

Tarmoqlanuvchi 123. Kvadrat va doiraninig yuzasi berilgan. Kvadrat doira ichida to'liq yotishi yoki yotmasligini aniqlang.

Tarmoqlanuvchi 124. Musbat qiymatli a, b, c, d uzunlikdagi kesmalar berilgan. Ulardan uchburchak hosil qiladigan har uchtasining uzunligini va hosil bo'ladigan uchburchaklarning yuzalarini chiqaring.

Tarmoqlanuvchi 125. Berilgan A, B va C sonlari uchun tomonlari A, B va C bo'lgan uchburchak mavjud bo'lsa, u holda bu uchburchak perimetri va yuzini, aks holda "Mavjud emas" iborasini chiqaring.

Tarmoqlanuvchi 126. Berilgan A, B va C sonlari uchun tomonlari A, B va C bo'lgan uchburchak mavjud bo'lsa, u holda bu uchburchakni teng yonli, teng tomonli yoki turli tomonliligi haqida ma'lumot, aks holda "Mavjud emas" iborasini chiqaring.

Tarmoqlanuvchi 127. Birinchi o'yinchida n1 ta konfet bo'lib, bir yurishda 1 donadan to k1 donagacha, ikkinchi o'yinchida n2 ta konfet bo'lib, bir yurishda 1 donadan to k2 donagacha konfet yeyishi mumkin. O'yinchilar navbat bilan yurishadi, yurishni birinchi o'yinchi boshlaydi. Yurish navbati kelganda konfeti qolmagan o'yinchi yutqazadi. Agar ikkala o'yinchi samarador o'ynasa, u holda g'olib o'yinchini aniqlang.

Tarmoqlanuvchi 128. Oxy tekisligida A nuqta (x0, y0) koordinatalari orqali berilgan. Shu nuqta koordinata boshida yotsa 0, Ox o‘qida yotsa 1, Oy o‘qida yotsa 2, boshqa hollarda 3 sonini chiqaring.

Tarmoqlanuvchi 129. Oxy tekisligida tomonlari koordinata o‘qlariga parallel bo‘lgan to‘g‘ri to‘rtburchakning uchta uchi koordinatalari orqali berilgan. Shu to‘g‘ri to‘rtburchakning to‘rtinchi uchi koordinatalarini aniqlang.

Tarmoqlanuvchi 130. Uchlari A(x1,y1), B(x2,y2), C(x0,y0) nuqtalarda bo‘lgan uchburchakning AB asosidagi burchaklari o‘tkir, to‘g‘ri yoki o‘tmas burchakligini aniqlang.

Tarmoqlanuvchi 131. A va B sonlar berilgan. Agar y=Ax+B to‘g‘ri chiziq Oxy tekisligida koordinatalar o‘qi bilan birgalikda chegaralangan uchburchak hosil qilsa, uchburchak yotgan kvadrant tartib raqamini, aks holda “No” chiqaring.

Tarmoqlanuvchi 132. Tekislikda koordinatalari orqali berilgan A(x1, y1), B(x2, y2), C(x3, y3) va D(x4, y4) nuqtalar shu ketma-ketlikda qabariq to‘rtburchakning uchlari bo‘ladi. Bu to‘rtburchakka ichki aylana chizish mumkin yoki mumkin emasligini aniqlang.

Tarmoqlanuvchi 133. Uchlari (x1,y1), (x2,y2), (x3,y3) va (x4,y4) nuqtalar ketma-ketligi orqali aniqlangan qabariq to‘rtburchak trapetsiya yoki to‘g‘ri to‘rtburchak yoki kvadrat yoki parallelogramm yoki rombdan birortasi bo‘lishini aniqlang, aks holda “0” chiqaring.

Tarmoqlanuvchi 134. Berilgan a, b, va c uchun $y=a\cdot x^2+b\cdot x+c$ parabolaning uchi yotadigan kvadrantni aniqlang (yo‘llanma: parabolaning uchi (x_0, y_0): $x_0= -\frac{b}{2\cdot a}$, $y0= a\cdot x_0^2+b\cdot x_0+c$).

Tarmoqlanuvchi 135. $a\cdot x+b\cdot y+c=0$ ($a^2+b^2\neq 0$) tenglama bilan aniqlanuvchi to‘g‘ri chiziq yotadigan kvadrantlarni aniqlang.

Tarmoqlanuvchi 136. Berilgan a ($a\neq 0$), b, va c uchun $y=a\cdot x^2+b\cdot x+c$ parabolaning grafigi yotadigan kvadrantlarni aniqlang.

Yo‘llanma: Masaladagi $a\neq 0$ sharti parabola hosil bo‘lish sharti hisoblanadi, aks holda $y=b\cdot x+c$ to‘g‘ri chiziq hosil bo‘ladi.

Matematikadan ma’lumki, agar $a>0$ bo‘lsa, parabolaning shoxchasi yuqoriga yo‘nalgan, $a<0$ bo‘lsa, parabolaning shoxchasi quyiga yo‘nalgan bo‘ladi. Parabolaning grafigi Oy o‘qini albatta kesib o‘tadi. Avval ko‘rilgani kabi parabola holatini $a\cdot x^2+b\cdot x+c=0$ tenglama bilan bog‘laymiz:

1) $D = b^2 - 4\cdot a\cdot c<0$ bo‘lsa, tenglama yechimga ega emas, ya’ni parabola grafigi Ox o‘qi bilan umumiy nuqtaga ega emas. Bu holda agar $a>0$ bo‘lsa, parabola grafigi Ox o‘qidan yuqorida joylashgan (demak, I va II kvadrantda), $a<0$ bo‘lsa, parabola grafigi Ox o‘qidan quyida joylashgan (demak, III va IV kvadrantda);

2) $D = b^2 - 4\cdot a\cdot c=0$ bo‘lsa, tenglama yagona yechimga ega, ya’ni parabola grafigi Ox o‘qi bilan bitta nuqtada kesishadi va u parabola uchining absissasi bo‘ladi. U holda agar $a>0$

bo‘lsa, parabola grafigi Ox o‘qidan yuqorida joylashgan (ya’ni I va II kvadrantda), $a<0$ bo‘lsa, parabola grafigi Ox o‘qidan quyida joylashgan (ya’ni III va IV kvadrantda), faqat parabola uchi Ox o‘qiga tegib turadi.

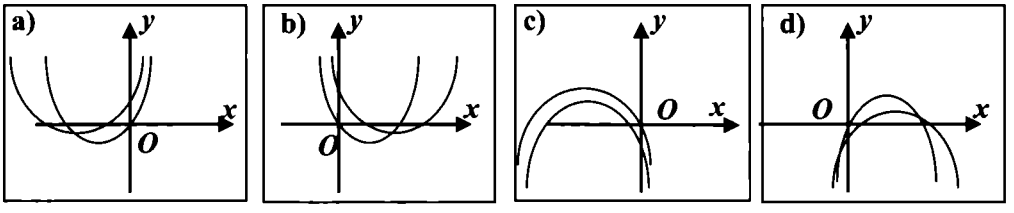
Demak, agar $D = b^2 - 4\cdot a\cdot c \le 0$ va $a>0$ bo‘lsa, u holda parabola grafigi I va II kvadrantda, agar $D = b^2 - 4\cdot a\cdot c \le 0$ va $a<0$ bo‘lsa, u holda parabola grafigi III va IV kvadrantda joylashadi.

3) $D = b^2 - 4\cdot a\cdot c>0$ bo‘lsa, tenglama ikkita turli yechimga ega, ya’ni parabola grafigi Ox o‘qini 2 ta turli x_1 va x_2 nuqtalarda kesib o‘tadi. Quyidagi hollarni tahlil etishda Viyet formulalaridan foydalanamiz:

a) $x_1 \cdot x_2 = \frac{c}{a} < 0$ bo‘lsin. Bu holda x_1 va x_2 sonlarining ishoralari turli bo‘lgani uchun ikkala kesishish nuqtasi Oy o‘qining turli tomonlarida yotadi, demak, a ning ishorasidan qat’iy nazar, parabola grafigi I, II, III va IV kvadrantlarda yotadi;

b) $x_1 \cdot x_2 = \frac{c}{a} \ge 0$ bo‘lsin. Bu holda x_1 va x_2 sonlarining ishoralari bir xil yoki bittasi 0 va ikkinchisi 0 dan farqli bo‘lgani uchun Ox o‘qi bilan kesishish nuqtalari Oy o‘qining bir tomonida yotadi. Shu bilan birga, x_1 va x_2 sonlarining yig‘indisi 0 bo‘la olmaydi. Agar x_1 va x_2 sonlari Oy o‘qining chap tomonida yotsa $x_1 + x_2 = -\frac{b}{a} < 0$ tengsizlik o‘rinli, agar o‘ng tomonida yotsa $x_1 + x_2 = -\frac{b}{a} > 0$ tengsizlik o‘rinli bo‘ladi.

Avval $a>0$ holni qaraymiz. Agar $-\frac{b}{a} < 0$ bo‘lsa, u holda parabola grafigi I, II va III kvadrantlarda (a rasm), aks holda, ya’ni $-\frac{b}{a} > 0$ bo‘lsa, grafik I, II va IV kvadrantlarda (b rasm) yotadi.



Endi $a<0$ holni qaraymiz. Agar $-\frac{b}{a} < 0$ bo‘lsa, u holda parabola grafigi II, III va IV kvadrantlarda (c rasm), aks holda, ya’ni $-\frac{b}{a} > 0$ bo‘lsa, parabola grafigi I, III va IV kvadrantlarda (d rasm) yotadi.

Dastur tuzish uchun yuqoridagi tahlildan faqatgina parabola koeffitsiyentlaridan iborat bo‘lgan va bir-birini inkor etuvchi shartlarni ajratib olish yetarli.

Tarmoqlanuvchi 137. Berilgan (x0; y0) nuqta uchlari (x1; y1) va (x2; y2) nuqtalarda bo‘lgan kesmada yotishi yoki yotmasligini aniqlang.

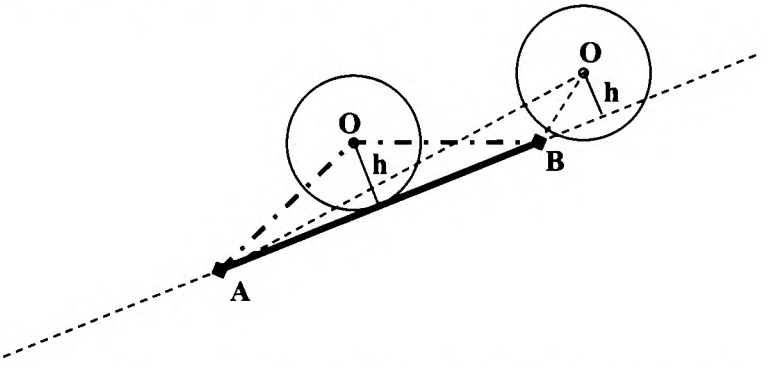
Tarmoqlanuvchi 138. O(x0, y0), A(x1,y1), B(x2,y2), C(x3,y3) nuqtalar berilgan. Agar uchlari A, B va C nuqtalarda bo'lgan uchburchak mavjud bo'lsa, u holda O nuqtaning uchburchakda yotishi yoki yotmasligini aniqlang.

Tarmoqlanuvchi 139. Uchlari A(x1,y1) va B(x2,y2) nuqtalarda bo'lgan kesma bilan markazi O(x0,y0) nuqtada bo'lgan R radiusli doiraning kesishishi yoki kesishmasligini aniqlang.

Yo'llanma. Kesmani o'z ichiga olgan to'g'ri chiziq tenglamasini ikki nuqtadan o'tuvchi to'g'ri chiziq tenglamasi yordamida quyidagicha tuzish mumkin:

$$\frac{y-y_1}{y_2-y_1}=\frac{x-x_1}{x_2-x_1} \text{ yoki bundan } (y_1-y_2)\cdot x+(x_2-x_1)\cdot y+(x_1-x_2)\cdot y_1+(y_2-y_1)\cdot x_1=0.$$

Lekin bu masala to'g'ri chiziq bilan doira kesishishi masalasidan tubdan farq qiladi. Chunki doira markazidan to'g'ri chiziqqacha bo'lgan masofa d (chizmaga qarang: ABO uchburchak balandligi h=d) va radius R orasidagi $d \le R$ munosabat o'rinli bo'lgani bilan doira to'g'ri chiziqning kesmaga tegishli bo'lmagan nuqtasida ham kesishishi mumkin.



ABO uchburchakka e'tibor beramiz. Agar h balandlik uchun h > R tengsizlik o'rinli bo'lsa, kesma va doira kesishmaydi. Chizmadan ko'rinadiki, agar ABO uchburchakning AB asosidagi burchaklari o'tkir yoki to'g'ri bo'lsa, u holda kesma va doiraning kesishishi uchun $h \le R$ tengsizlik o'rinli bo'lishi yetarli. Agar ABO uchburchakning AB asosidagi burchaklaridan biri o'tmas bo'lsa, u holda kesma va doira kesishishi uchun ($d_{AO} \le R$) YOKI ($d_{BO} \le R$) bajarilishi shart.

Balandlik h ni hisoblash uchun quyidagi formulalardan foydalanish mumkin:

$$p=\frac{d_{AB}+d_{AO}+d_{BO}}{2},\ S=\sqrt{p\cdot(p-d_{AB})\cdot(p-d_{AO})\cdot(p-d_{BO})},\ h=\frac{2\cdot S}{d_{AB}}$$

Dastur tuzish uchun avvalgi masalalar natijalaridan foydalaniladi.

Tarmoqlanuvchi 140. Berilgan a, b, c, n, m, p qiymatlar asosida $a\cdot x+b\cdot y+c=0$ va $n\cdot x+m\cdot y+p=0$ tenglamalar to'g'ri chiziqlar hosil qilsa, u holda bu to'g'ri chiziqlarni kesishishi, ustma-ust tushishi yoki kesishmasligini aniqlang, aks holda "NO" yozuvini chiqaring.

Yo'llanma. 1) Ma'lumki, a, b, c, n, m, k qiymatlar asosida to'g'ri chiziqlar hosil bo'lishi uchun $a^2+b^2\neq 0$ va $n^2+m^2\neq 0$ shartlar bajarilishi zarur va yetarli.

Ikki to'g'ri chiziq bir-biriga nisbatan quyidagi holatlarda joylashishi mumkin:

2) To'g'ri chiziqlar ustma-ust tushadi (boshqacha aytganda cheksiz ko'p nuqtada keshishadi). U holda biror $k\neq 0$ uchun $a=k\cdot n$, $b=k\cdot m$, $c=k\cdot p$ shartlar bajariladi. Bundan $k=\frac{a}{n}=\frac{b}{m}=\frac{c}{p}$ ekanligi, ya'ni $a\cdot m=b\cdot n$, $a\cdot p=c\cdot n$, $b\cdot p=c\cdot m$ ayniyatlar to'g'riligi kelib chiqadi. Koeffitsiyent k ning ifodasidan $a\cdot m=b\cdot n$ va $a\cdot p=c\cdot n$ tenglik $b\cdot p=c\cdot m$ tenglikning, $a\cdot m=b\cdot n$ va $b\cdot p=c\cdot m$ tenglik $a\cdot p=c\cdot n$ tenglikning bajarilishini taqozo etishi kelib chiqadi.

3) To'g'ri chiziqlar kesishmaydi, ya'ni ular parallel bo'ladi. U holda biror $k\neq 0$ uchun $a=k\cdot n$, $b=k\cdot m$, $c\neq k\cdot p$ shartlar bajariladi. Bundan $k=\frac{a}{n}=\frac{b}{m}\neq\frac{c}{p}$ ekanligi, natijada $a\cdot m=b\cdot n$, $a\cdot p\neq c\cdot n$, $b\cdot p\neq c\cdot m$ ekanligi kelib chiqadi.

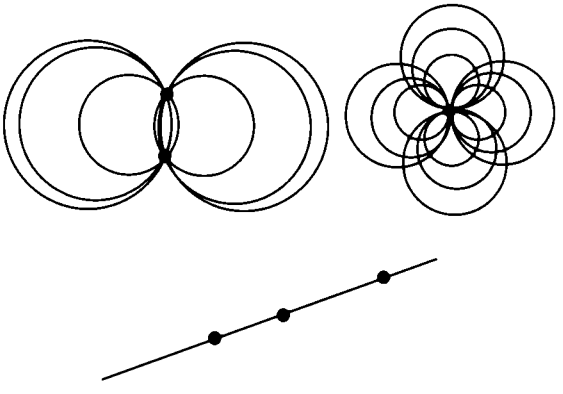
4) To'g'ri chiziqlar bitta nuqtada kesishadi. U holda har qanday $k\neq 0$ uchun $a\neq k\cdot n$, $b\neq k\cdot m$ shartlar, ya'ni $\frac{a}{b}\neq\frac{b}{m}$ shart bajariladi. Bundan $a\cdot m\neq b\cdot n$ ekanligi kelib chiqadi.

Yuqoridagi 4 ta holat a, b, c, n, m, k ning barcha qiymatlariga javob beradi.

Tarmoqlanuvchi 141. Berilgan a, b, c, n, m, p ($a\cdot m\neq b\cdot n$) qiymatlar asosida $a\cdot x+b\cdot y+c=0$ va $n\cdot x+m\cdot y+p=0$ to'g'ri chiziqlar kesishish nuqtasini aniqlang (yo'llanma: kesishish nuqtasi $x_0=\frac{c\cdot m-b\cdot p}{a\cdot m-b\cdot n}$, $y_0=\frac{a\cdot p-c\cdot n}{a\cdot m-b\cdot n}$).

Tarmoqlanuvchi 142. Berilgan 3 ta A(x1,y1), B(x2,y2), C(x3,y3) nuqtalardan o'tuvchi aylana mavjudligini, mavjud bo'lsa, uning yagonaligini, yagona bo'lsa, uning radiusini aniqlang.

Yo'llanma. 1) Nuqtalardan uchtasi yoki biror ikkitasi ustma-ust tushsin. O'ng tomondagi chiz-malardan ko'rinadiki, aylana mavjud, lekin yagona bo'lmas ekan. Bu holda (x1=x2) VA (y1=y2) YOKI (x1=x3) VA (y1=y3) YOKI (x2=x3) VA (y2=y3) murakkab shart yoki teng kuchli $(|x1-x2|+|y1-y2|)\cdot(|x1-x3|+|y1-y3|)\cdot(|x2-x3|+|y2-y3|)=0$ shart Rost qiymat qabul qiladi.



2) Berilgan 3 ta nuqtadan birortasi ham ustma-ust tushmasin, lekin bitta to'g'ri chiziqda yotsin. Ma'lumki, aylana va to'g'ri chiziq ko'pi bilan ikki nuqtada kesishadi. Demak, bu holda berilgan 3 ta nuqtadan o'tuvchi aylana mavjud emas. Uchta nuqta bir to'g'ri chiziqda

yotsa, u holda quyidagi tenglik (ikki nuqtadan o‘tuvchi to‘g‘ri chiziq tenglamasiga ko‘ra) o‘rinli bo‘ladi:

$$(y1-y2)\cdot x3+(x2-x1)\cdot y3+(x1-x2)\cdot y1+(y2-y1)\cdot x1=0$$

yoki $(y2-y1)\cdot(x1-x3)+(y3-y1)\cdot(x2-x1)=0$.

3) Xulosa qilib shuni aytish mumkinki, 3 ta nuqtadan o‘tuvchi aylana mavjud va yagona bo‘lishi uchun ular bir to‘g‘ri chiziqda yotmasligi va ularning biror ikkitasi ustma-ust tushmasligi shart. Bu holda berilgan uchta nuqta biror uchburchakning uchlari bo‘ladi. Uchburchakka tashqi chizilgan aylana esa biz izlayotgan aylana bo‘ladi.

Geometriyadan quyidagi formulalar ma’lum:

a) Uchburchak yuzasini uchlarining koordinatalari orqali ifodasi

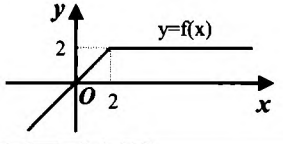
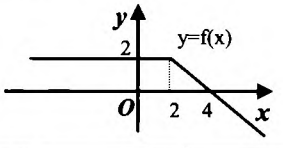
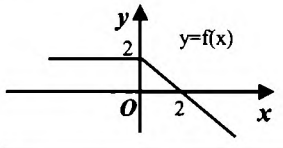
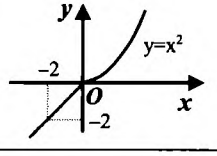
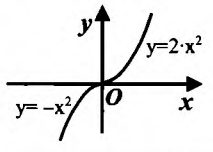
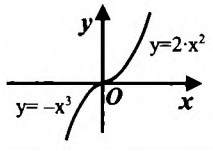
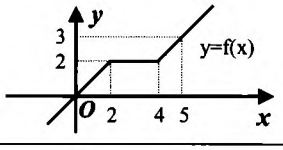
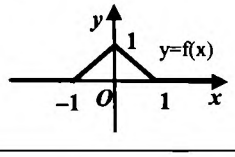
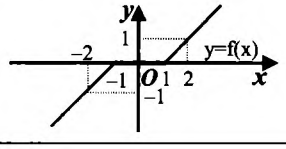
$$S=((y2-y3)\cdot x1+(y3-y1)\cdot x2+(y1-y2)\cdot x3))/2.$$

b) Uchburchakka tashqi chizilgan aylana radiusi:

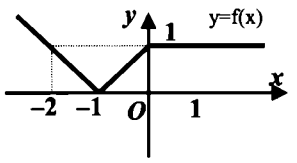
$$Rt=\frac{\sqrt{(x1-x2)^2+(y1-y2)^2}\cdot\sqrt{(x1-x3)^2+(y1-y3)^2}\cdot\sqrt{(x2-x3)^2+(y2-y3)^2}}{4\cdot S}$$

Tarmoqlanuvchi 143. A nuqta (x0, y0) koordinatasi orqali berilgan. Agar A nuqta $B=\{(x,y)|\ x^2+y^2<21,\ y>x,\ x,\ y\in R\}$ yoki $C=\{(x,y)|\ |x+y|\le 7,\ y<|x|,\ x,\ y\in R\}$ to‘plamlardan, hech bo‘lmasa, biriga tegishli bo‘lsa “A+BC” javobini, aks holda “A–BC” javobini chiqaring.

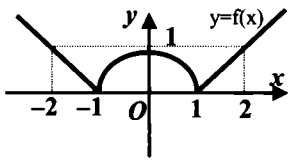
Tarmoqlanuvchi xx. Berilgan A(x0; y0) nuqta chizmada tasvirlangan funksiya grafigiga tegishli bo‘lsa “YES”, aks holda “NO” javobini chiqaring (yo‘llanma: funksiyani tengsizliklar sistemasi yordamida ifodalash).

<table>
<tr>
<td>144.
 
</td>
<td>145.
 
</td>
<td>146.
 
</td>
</tr>
<tr>
<td>147.
 
</td>
<td>148.
 
</td>
<td>149.
 
</td>
</tr>
<tr>
<td>150.
 
</td>
<td>151.
 
</td>
<td>152.
 
</td>
</tr>
</table>

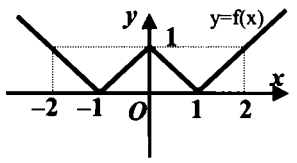
153.



154.



155.



= D =

Tarmoqlanuvchi 156. A va N ($1 \le A, N \le 10000$) natural sonlar berilgan. A^N sonning oxirgi raqamini aniqlang.

K:	10 1963	1021 2018	9999 6789	6798 9999	518 10000	123 1963
CH:	0	1	9	8	6	7

Tarmoqlanuvchi 157. N, M va A ($1 \le N, M, A \le 10^9$) natural sonlar berilgan. O'lchami N×M bo'lgan to'g'ri to'rtburchakni eng kam sondagi A×A o'lchamdagi kvadratlar bilan to'liq qoplash zarur. Qoplashda kvadrat qismi to'g'ri to'rtburchakdan chiqib ketishi ham mumkin. Kvadratlar tomonlari to'g'ri to'rtburchak tomonlariga parallel.

K:	6 6 4	2 2 1	12 13 4	222 332 5	1001 1000 10	1000000000 1000000000 192
CH:	4	4	12	3015	10100	27126743055556

Tarmoqlanuvchi 158. A, B, C va D ($1 \le N \le 10^2$) natural sonlar berilgan. Agar shu sonlar uzunligidagi kesmalarning barchasidan foydalanib musbat yuzali uchburchak yasash mumkin bo'lsa "Uchburchak", musbat yuzali uchburchak yasash mumkin emas, lekin 0 yuzali uchburchak yasash mumkin bo'lsa "Kesma", aks holda "Imkonsiz" javobini chiqaring.

K:	4 2 1 3	7 2 2 4	3 5 9 1	1 1 1 1	3 3 3 10	13 25 12 1
CH:	Uchburchak	Kesma	Imkonsiz	Uchburchak	Uchburchak	Kesma

Tarmoqlanuvchi 159. $A(x_1; y_1), B(x_2; y_2)$ va $C(x_3; y_3)$ ($|x_k| \le 10^2$, $|y_k| \le 10^2$) butun koordinatali nuqtalar berilgan. Agar uchlari shu nuqtalarda bo'lgan to'g'ri burchakli uchburchak hosil bo'lsa "To'g'ri", agar to'g'ri burchakli uchburchak hosil bo'lmasa, lekin nuqtalardan birortasining koordinatasini vertikal yoki gorizontal yo'nalishda surib to'g'ri burchakli uchburchak hosil qilish mumkin bo'lsa "Deyarli", aks holda "Imkonsiz" javobini chiqaring.

K:	0 0 0 1 1 0	0 -3 -3 -10 4 -7	28 -15 86 32 -19 43	60 4 90 -53 32 -12	52 -34 -37 -63 23 54
CH:	To'g'ri	Imkonsiz	To'g'ri	Deyarli	Deyarli

Tarmoqlanuvchi 160. 1, 2 va 3 tartibda joylashtirilgan 3 ta to'ntarilgan piyola bor. Aniq tartib raqamli piyola tagiga sharcha qo'yiladi va 2 ta piyolaning o'rni almashtiriladi. Har safar piyolalarning o'rni almashganda yangidan tartib raqami beriladi. Shu qoida bo'yicha piyo-

lalarning o‘rni 3 marta almashtirilganda sharcha o‘tib qoladigan piyolaning tartib raqamini chiqaring.

K:	1 1 2 2 1 2 1	2 2 3 1 3 1 3	3 3 2 3 1 3 1	3 3 1 2 3 3 2	2 1 3 1 2 2 1	1 2 3 1 3 1 2	1 2 1 3 1 1 3	1 3 2 1 3 3 1	1 3 2 3 1 1 2
CH:	2	3	2	1	2	3	2	1	3

Tarmoqlanuvchi 161. Tosh, qaychi va qog‘oz o‘yinida: tosh qaychini sindiradi, qaychi qog‘ozni kesadi, qog‘oz toshni o‘rab oladi, ya’ni yutadi. Tosh=1, qaychi=2 va qog‘oz=3 kabi belgilangan bo‘lsin. Ali A, Vali V va Soli S ($1 \le A, V, S \le 3$) natural sonlarga mos predmet tanlashdi. Agar bitta o‘yinchi qolgan ikkita o‘yinchini yutsa, o‘sha g‘olib bo‘ladi, aks holda g‘olib aniqlanmaydi. O‘yinda g‘olib bo‘lganning ismidagi birinchi harfni, g‘olib aniqlanmasa “N” harfini chiqaring.

K:	1 3 1	3 1 1	2 1 1	2 3 3	3 1 2	3 2 3	1 1 3	2 2 3	1 2 2
CH:	V	A	N	A	N	V	S	N	A

Tarmoqlanuvchi 162. N ($1 \le N \le 1000$) natural son berilgan. Aylana bo‘ylab joylashtirilgan N ta likopchada bir donadan olma bor. Ali boshlang‘ich (nolinchi) minutda birorta likopchadagi olmani yeydi. Ali soat mili yo‘nalishida likopchalar bo‘ylab harakatlana oladi. Bunda k-minutda o‘z oldida turgan likopchadan keyingi (k–1) ta likopchani tashlab o‘tib, ulardan keyingi likopchadagi olmani yeydi. Agar Ali qachondir barcha likopchadagi olmalarni yeb tugata oladigan bo‘lsa “HA”, aks holda “YO‘Q” so‘zini chiqaring.

K:	16	3	2	4	5	14	1	580	256
CH:	HA	YO‘Q	HA	HA	YO‘Q	YO‘Q	HA	YO‘Q	HA

Tarmoqlanuvchi 163. N va x_1, y_1, x_2, y_2 ($1 \le N \le 1000, 0 \le x_1, y_1, x_2, y_2 \le N$) butun sonlar berilgan. Oxy tekisligida uchlari (0; 0), (0; N), (N; N) va (N; 0) nuqtalarda bo‘lgan kvadrat chizilgan. Kvadratning chegarasida yotgan $A(x_1; y_1)$ nuqtadan $B(x_2; y_2)$ nuqtaga faqat kvadratchalarning chegarasi bo‘ylab yurib borish mumkin. A nuqtadan B nuqtagacha bo‘lgan eng qisqa masofani aniqlang.

K:	4 0 3 1 4	1000 564 0 436 1000	99 12 0 35 99	11 0 2 11 4	341 158 0 0 190	967 967 607 279 0
CH:	2	2000	146	17	348	1295

Tarmoqlanuvchi 164. N va K ($1 \le N \le 50, 1 \le K \le 250$) sonlari berilgan. Agar K sonini faqat 2, 3, 4 va 5 raqamlarini qatnashtirib N tasining yig‘indisi ko‘rinishida tasvirlash mumkin bo‘lsa, u holda 2 raqami eng kamida necha marta qatnashishini aniqlang.

K:	4 8	1 3	4 9	29 116	27 75	49 219	7 20	47 128	45 104
CH:	4	0	3	0	6	0	1	13	31

Tarmoqlanuvchi 165. N ($1 \le N \le 1000$) butun son berilgan. Tomoni N×N metr bo'lgan kvadratning quyi chap uchiga sharcha qo'yildi. Keyingi sharchalar chegara bo'ylab soat mili yo'nalishida har (N+1) metrga qo'yib boriladi. Kvadratning quyi chap uchiga ikkinchi sharcha qo'yilganda hammasi bo'lib nechta sharcha qo'yilganini aniqlang.

K:	4	67	2	100	1	38	112	28	59
CH:	17	68	9	401	3	153	449	113	60

Tarmoqlanuvchi 166. Ali har doim yonida qizil, yashil, ko'k va oq rangli qog'ozlarni olib yuradi. Shu bilan birga, rangli qog'ozlarga turli tartib raqamlari berishni yaxshi ko'radi. Ranglarning tartib raqamlari r1, r2, r3, r4 ($1 \le$ r1, r2, r3, r4 ≤ 100000) bo'lsin. Har kuni Ali qog'ozlarni qarab, yetmayotganini sotib oladi. Berilgan ranglarning tartib raqamiga ko'ra Ali eng kamida necha xil rangdagi qog'oz sotib olishi kerakligini aniqlang.

K:	5 5 5 5	100 10 2 99	1 2 2 3	1 99 50 2	100000 1 1 1	19 63 21 7
CH:	3	0	1	0	2	0

Tarmoqlanuvchi 167. Butun xa, ya, xb, yb, xc, yc (|xa|, |ya|, |xb|, |yb|, |xc|, |yc| ≤ 1000000000) sonlar berilgan. Bo'g'irsoq koordinata tekisligida A(xa; ya) nuqtadan chiqib, B(xb; yb) nuqtaga yetib keldi va kelgan yo'nalishi bo'yicha qarab turibdi. U C(xc; yc) nuqtaga borishi kerak bo'lsa, u holda yurishi kerak bo'lgan yo'nalishni aniqlang. Agar C nuqta bo'g'irsoq qarab turgan yo'nalish bo'ylab joylashgan bo'lsa "To'g'riga", qarab turgan yo'nalishga nisbatan chapda bo'lsa "Chapga", qarab turgan yo'nalishga nisbatan o'ngda bo'lsa "O'ngga" so'zlarini chiqaring.

K:	71 43 96 -15 171 -189	-22 -84 -117 8 -25 103	-61 -24 -61 35 -120 35	-19 27 -115 -63 -25 -159	47 16 -13 -52 -253 -324	3609 -639705 294730 -1024276 -89841 -1315397	-702371 875896 -1445450 1767452 -2337006 1024373
CH:	To'g'riga	O'ngga	Chapga	Chapga	To'g'riga	O'ngga	Chapga

Tarmoqlanuvchi 168. Natural N, M va A sonlari berilgan ($1 \le$ N, M ≤ 1000000, $0 \le$ A ≤ 180). Oxy tekisligida koordinata o'qlariga parallel bo'lgan tomonlari N va M ga teng, simmetriya markazi koordinata boshida joylashgan to'g'ri to'rtburchak chizilgan. Mazkur to'g'ri to'rtburchakning nusxasi soat miliga qarshi yo'nalishda A gradusga burildi. Ikkala to'g'ri to'rtburchak kesishishidan hosil bo'lgan soha yuzasini toping. Javob 10^{-7} aniqlikda bo'lsin.

K:	6 4 30	100 100 30	303304 904227 3	606243 819219 106	1 1 45
CH:	19.6683849	8452.9946162	262706079399.4969020	382341849885.3648987	0.8284271

Tarmoqlanuvchi 169. Butun N ($10 \le |N| \le 10^9$) soni berilgan. Son ustida quyidagicha amallardan birini bajarish mumkin: a) birlik raqamni o'chirish; b) o'nlik raqamni o'chirish; b) birorta ham raqamni o'chirmaslik. Shu amallardan birini qo'llab N sonini kattalashtirish mumkin bo'lsa, u holda kattalashtirilgan N sonini, aks holda N sonining o'zini chiqaring.

K:	-10	2230	-10003	-847251738	-623563697	-5112	123456	-777	-87
CH:	0	2230	-1000	-84725173	-62356367	-511	123456	-77	-7

Tarmoqlanuvchi 170. Butun x va y ($0 < |x|, |y| \le 10^6$) sonlar berilgan. Oxy tekisligida tomonlari koordinata o‘qlariga parallel, chap quyi uchi A(0; 0) nuqtada va o‘ng yuqori uchi N(x; y) nuqtada bo‘lgan to‘g‘ri to‘rtburchak chizilgan. Shunday B(x1; y1) va C(x2; y2) nuqtalarni topingki, quyidagi talablar bajarilsin:

a) x1, y1, x2, y2 sonlar butun va x1<x2;

b) ABC uchburchak teng yonli va A – to‘g‘ri burchak;

c) chizilgan to‘g‘ri to‘rtburchak nuqtalari ABC uchburchakka tegishli;

d) ABC uchburchak yuzasi iloji boricha kichik.

K:	10 5	1321 -23131	-10 5	1 1	20 -10	-1 1	2 -435
CH:	0 15 15 0	0 -24452 24452 0	-15 0 0 15	0 2 2 0	0 -30 30 0	-2 0 0 2	0 -437 437 0

Tarmoqlanuvchi 171. Agar natural sonning 2 tadan ortiq bo‘luvchisi bo‘lsa, u holda u murakkab son deyiladi. Har qanday 12 sonidan kichik bo‘lmagan natural N sonini ikkita murakkab natural son yig‘indisi ko‘rinishida tasvirlash mumkin, ya’ni N=x+y, bu yerda x va y murakkab natural sonlar. Berilgan N ($12 \le N \le 1000000$) natural soni uchun x va y murakkab natural sonlarni aniqlang. Agar bunday sonlar ko‘p bo‘lsa, ulardan ixtiyoriy bittasini chiqaring. Masalan:

K:	12	23	1000000	41	111	12	15	23	12345
CH:	6 6	8 15	500000 500000	20 21	12 99	4 8	6 9	9 14	100 12245

Tarmoqlanuvchi 172. Berilgan butun a, b, c, d ($1 \le a, b, c, d \le 1000$) sonlar uzunligidagi 4 ta tayoqcha bor. Bu tayoqchalardan tovuq yoki laylak shaklini yasash mumkinligini qaraymiz. Tayoqchalardan parranda yoki qushlar shakli quyidagicha yasaladi:

a) ikkita tayoqcha parranda yoki qushlarning oyoqlarini tasvirlaydi, ular teng bo‘lishi shart;

b) qolgan ikkita tayoqcha parranda yoki qushlarning tanasi va bo‘ynini tasvirlaydi. Tovuqda tana-tayoqcha bo‘yin-tayoqchadan uzun bo‘ladi. Laylakda esa tana-tayoqcha va bo‘yin-tayoqcha bir xil uzunlikda bo‘ladi. Tayoqchalar faqat butun holatda ishlatiladi. Agar tayoqchalardan laylak yasash mumkin bo‘lsa “Laylak”, agar laylak yasab bo‘lmasada tovuq yasab bo‘lsa “Tovuq”, aks holda “Boshqa” javobini chiqaring.

K:	1 1 1 1	2 2 2 1	1 2 3 4	10 11 12 10	99 76 99 11	1000 909 1000 909	101 201 202 102
CH:	Laylak	Tovuq	Boshqa	Tovuq	Tovuq	Laylak	Boshqa

Tarmoqlanuvchi 173. Butun N, M, A va B ($1 \le N, M, A, B \le 1000$) sonlari berilgan. Ali metroda 1 marta yurishi uchun A so‘m to‘laydi. U metroda M marta yurish imkoniyatini beruvchi B so‘m turadigan imtiyozli bilet borligini bilib qoldi. Ali metroda N marta yurishi uchun eng kamida necha so‘m sarflashi kerakligini aniqlang.

K:	1 1 1 1	556 2 16 15	995 1 2 1	477 2 16 14	101 110 1 100	10 3 1 2	5 2 2 3
CH:	1	4170	995	3346	100	7	8

Tarmoqlanuvchi 174. Butun N ($3 \le N \le 10^9$) soni berilgan. N soni uchun N=A+B+C ko'rinishida tasvirlash mumkin bo'lgan shunday 3 ta natural A, B va C sonlarni aniqlangki, bu sonlar 3 ga bo'linmasin. Agar to'g'ri javoblar ko'p bo'lsa ixtiyoriy bittasini chiqaring. Masalan:

K:	3	4	233	1000000000	1000000000	175744383
CH:	1 1 1	1 1 2	1 2 230	1 2 999999997	1 1 999999998	2 2 175744379

Tarmoqlanuvchi 175. Butun A, B va C ($1 \le A, B, C \le 10$) sonlar berilgan. A, B, C sonlarning o'rnini almashtirmasdan oralariga "+" va "*" amallarini joylashtirib, zarur bo'lsa, qavslar ham joylashtirib hosil qilish mumkin bo'lgan eng katta sonni aniqlang. Masalan, hosil qilish mumkin bo'lgan ifodalardan ba'zilari: A+B+C, (A+B)*C, A*(B+C), A*B*C.

K:	1 2 3	2 10 3	10 10 10	3 1 5	1 8 3	9 7 2	1 1 10	8 7 9	1 10 1
CH:	9	60	1000	20	27	126	20	504	12

Tarmoqlanuvchi 176. x0, y0 va S ($-10^9 \le x0, y0 \le 10^9$, $1 \le S \le 2 \cdot 10^9$) butun sonlar berilgan. Oxy Dekart koordinatalar sistemasida yurish qoidasi shunday: bir qadamda (x; y) nuqtadan faqat (x+1; y), (x–1; y), (x; y+1) yoki (x; y–1) nuqtalarga o'tish, ya'ni ixtiyoriy vertikal yoki gorizontal yo'nalishda bir birlik yurish mumkin. Agar O(0; 0) nuqtadan A(x0; y0) nuqtaga aniq S qadamda borish mumkin bo'lsa "YES", aks holda "NO" chiqaring.

K:	5 5 11	10 15 25	0 0 2	999999999 999999999 2000000000	-467780354 -721273539 1369030008
CH:	NO	YES	YES	YES	NO

Tarmoqlanuvchi 177. Biror r raqamni akslantirish deganda (9–r) ga almashtirish tushuniladi, masalan, 3 raqami akslantirilganda (9–3)=6 hosil bo'ladi. Berilgan 5 xonali N natural sonning raqamlarini akslantirib, 5 xonali eng kichik natural sonni hosil qiling.

K:	12327	45454	54371	87729	33110	39518	80478	19999
CH:	12322	44444	44321	12220	33110	30411	10421	10000

Tarmoqlanuvchi 178. Biror r raqamni akslantirish deganda (9–r) ga almashtirish tushuniladi, masalan, 3 raqami akslantirilganda (9–3)=6 hosil bo'ladi. Berilgan N ($1 \le N \le 99999$) natural sonning raqamlarini akslantirib, xonalari soni N sonining xonalari soniga teng eng kichik natural sonni hosil qiling.

K:	1	8	27	89	139	7908	5004	87654	39999
CH:	1	1	22	10	130	2001	4004	12344	30000

Tarmoqlanuvchi 179. Butun N va K ($1 \le K \le N \le 10^{18}$) sonlari berilgan. Ali va Vali quyidagicha o'yin o'ynashmoqda: Ali N ta toshni yonma-yon joylashtiradi va ketma-ket turgan K ta toshni olib tashlaydi. Keyin Vali ketma-ket turgan K ta toshni olib tashlaydi.

Toshlarni chap tomondan yoki o‘ng tomondan olib tashlash mumkin. Shu tariqa kımgadir ketma-ket turgan K ta tosh qolmasa o‘sha yutqazadi. Agar ikkala o‘yinchi samarador o‘ynasa, u holda g‘olib o‘yinchini aniqlang.

Yo‘llanma. Vaziyatlar tahlili, qonuniyat.

K:	1 1	2 1	6 2	10 4	251656215122324104 164397544865601257	100000 3
CH:	Ali	Vali	Ali	Vali	Ali	Ali

Tarmoqlanuvchi 180. Butun N ($1 \le N \le 1000$) soni berilgan. 1 dan N gacha bo‘lgan sonlar ketma-ket yozib chiqildi. Qatordagi barcha raqam sonini aniqlang.

K:	7	13	100	99	999	51	1000	313	35
CH:	7	17	192	189	2889	45	2893	831	61

Tarmoqlanuvchi 181. Butun k, n, w ($1 \le k, w \le 1000$, $0 \le n \le 10^9$) sonlari berilgan. Ali n so‘m puliga banan sotib olmoqchi edi. Banan importi chegaralangani uchun narxlari ham o‘z-gacha belgilangan. Ali 1-bananga k so‘m, 2-bananga 2·k so‘m, 3-bananga 3·k so‘m, va hoka-zo pul to‘lashi kerak. Aliga w dona banan sotib olishi uchun yetishmayotgan pulni aniqlang.

K:	3 17 4	1000 0 1000	859 453892 543	1000 500500000 1000	432 10000 241	20 43 3
CH:	13	500500000	126416972	0	12587552	77

Tarmoqlanuvchi 182. B1, I1, U1, T1, B2, I2, U2, T2 raqamlar berilgan. Aylanma qulf 4 xonali bo‘lib, uning har bir xonasida 0 dan 9 gacha raqamlar aylanadi. Bitta harakatda bir xonadagi raqamni o‘zidan bitta oldingi yoki bitta keyingi raqamga o‘tkazish mumkin. Masalan, bitta harakatda biror xonadagi 9 raqamini 0 raqamiga yoki 8 raqamiga o‘tkazish mumkin. Qulfning xonalarida, mos ravishda, B1, I1, U1, T1 raqamlar aks etib turibdi. Agar xonalariga mos ravishda B2, I2, U2, T2 raqamlari terilgach qulf ochilsa, u holda qulfni ochishga sarflanadigan eng kam harakatlar sonini aniqlang.

K:	0809 0636	1111 1111	0000 5555	9876 3333	9999 7777	7777 9999	2301 9898	9999 0000	2345 7654
CH:	8	0	20	16	8	8	12	4	10

Tarmoqlanuvchi 183. Butun N ($1 \le N \le 5000$) soni berilgan. Ali N ta o‘quvchidan imti-hon olishi kerak. Xona uzun bo‘lgani uchun o‘quvchilarni bir qatorga yonma-yon o‘tqaziladi. O‘qish vaqtida o‘quvchilarga 1, 2, ..., N tartib raqamlari berilgan va shu tartibda o‘tirishgan edi. Ali o‘qish davrida har bir qo‘shni bo‘lib o‘tirgan o‘quvchilar do‘stlashib qolganini biladi, ya’ni 1 va 2, 2 va 3, ..., k va (k+1), ..., N-1 va N tartib raqamli o‘quvchilar do‘stlashib qolgan. U do‘stlashib qolgan o‘quvchilarni imtihon vaqtida yonma-yon o‘tirmasligini xohlaydi. Ali xohlagandek qilib joylashtirish mumkin bo‘lgan o‘quvchilarning eng ko‘p sonini aniqlang. Masalan, N=3 bo‘lsa, u holda javob 2, ya’ni bu holda 1 va 3 tartib raqamli o‘quvchilarni yonma-yon joylashtirish mumkin. Chunki 1 va 2 hamda 2 va 3 tartib raqamli o‘quvchilar do‘stlashib qolgan.

K:	1	13	6	3	128	5000	2	1349	2501
CH:	1	13	6	2	128	5000	1	1349	2501

Tarmoqlanuvchi 184. Butun d1, d2 va d3 ($1 \le d1, d2, d3 \le 10^8$) sonlar berilgan. Alining uyiga do'sti Vali mehmonga keladigan bo'ldi. Shu sababli Ali 2 ta do'kondan mahsulot xarid qilishi zarur. Alining uyidan birinchi do'kongacha d1, ikkinchi do'kongacha d2, ikkala do'kon orasi esa d3 metr edi. Uyidan chiqqan Ali ikkita do'kondan mahsulot xarid qilib yana uyiga kelganda bosib o'tadigan eng qisqa masofani aniqlang.

K:	1 1 5	10 20 30	100 33 34	777 777 777	2 2 8	12 34 56	27485716 99999999 35182
CH:	4	60	134	2331	8	92	55041796

Tarmoqlanuvchi 185. Butun k, a va b ($1 \le k \le 10^{18}$, $-10^{18} \le a \le b \le 10^{18}$) sonlar berilgan. [a; b] kesmadagi k ga karrali barcha sonlar sonini aniqlang.

K:	1 1 10	2 -4 4	2 1 2	2 0 0	5 -1000000000000000000 -402710171917	3 0 29102
CH:	10	5	1	1	199999919457965617	9701

Tarmoqlanuvchi 186. Ali uyiga basseyn qurish uchun Dekart koordinatalar sistemasida tomonlarini Ox va Oy o'qlariga parallel joylashtirib musbat yuzali to'g'ri to'rtburchak shaklini chizdi. Shundan keyin to'g'ri to'rtburchakning uchlari koordinatalarini ($|x_k|$, $|y_k| \le 1000$) alohida varaqqa yozib oldi. Do'sti Vali kelib bexosdan varaqdagi to'g'ri to'rtburchakning ba'zi uchlari koordinatalarini o'chirib yubordi. O'chmay qolgan N ta ($1 \le N \le 4$) kiritiladigan to'g'ri to'rtburchak uchlariga mos koordinatalarga asosan basseynning yuzasini hisoblang, agar hisoblashning iloji bo'lmasa, "-1" chiqaring.

K:	1 71 -740	4 -56 -858 -56 -174 778 -858 778 -174	2 14 153 566 -13	2 -337 451 32 -395	2 -1000 -1000 0 -1000	3 -890 778 -418 296 -890 296	3 0 0 0 1 1 0	4 0 0 1 0 1 1 0 1	4 -474 -894 -474 -833 -446 -894 -446 -833
CH:	-1	570456	91632	312174	-1	227504	1	1	1708

Tarmoqlanuvchi 187. Butun N va D ($1 \le N \le 19$, $2 \le D \le 10$) sonlari berilgan. D ga bo'linadigan N xonali sonni aniqlang. Agar javoblar ko'p bo'lsa ixtiyoriy bittasini chiqaring. Masalan:

K:	3 5	3 2	9 4	1 7	2 8	2 10	5 10	5 2
CH:	125	712	123456700	7	80	10	11110	22222

Tarmoqlanuvchi 188. Butun A va B ($1 \le A, B \le 1000$) sonlar berilgan. Alini A ta qizil va B ta yashil rangli paypog'i bor edi. U har kuni ertalab bir oyog'iga qizil, boshqa oyog'iga yashil rangli paypoq kiyib do'stlarini kuldirib yuradi. Turli rangli paypoqlari tugasa, bir xil rangli paypoq kiyib ular ham tugaguncha jiddiy yuradi. Har kuni kechqurun esa kiyilgan

paypoqlarni butunlay tashlab yuboradi. Ali necha kun do'stlarini kuldirib va necha kun jiddiy yurishini aniqlang.

K:	3 1	2 3	11 56	100 23	1000 999	68 59	59 12	100 11	1 1
CH:	1 1	2 0	11 22	23 38	999 0	59 4	12 23	11 44	1 0

Tarmoqlanuvchi 189. Butun N, M, A va B ($1 \le N$, $K \le 10^{12}$, $1 \le A$, $B \le 100$) sonlari berilgan. Futbol bo'yicha Osiyo chempionatiga mezbonlik qilayotgan davlatda M ta kommentatorlar delegatsiyasi uchun N ta xona tayyorlashdi. Delegatsiya a'zolari har bir delegatsiya uchun bir xil sondagi xona berilishini talab qilishdi, ya'ni N soni M ga qoldiqsiz bo'linishi shart edi. Mezbonlar uchun 2 yo'ldan biri qoldi: ortiqcha xonalarni buzib tashlash yoki yangilarini qurish. Ma'lumki, bitta xona qurishga A, bitta xonani buzishga esa B miqdorda pul xarajat qilinardi. Mezbonlar sarflashi mumkin bo'lgan eng kam miqdordagi pulni aniqlang.

K:	9 7 3 8	2 7 3 7	30 6 17 19	500000000001 1000000000000 100 100	7 2 3 7
CH:	15	14	0	49999999999900	3

Tarmoqlanuvchi 190. Alining uyi Ox o'qining 0 nuqtasida, Valining uyi esa X ($2 \le X \le 1000000$) nuqtasida joylashgan. Ali har bir qadamda oldinga 2 yoki 5 birlikka surilishi mumkin. Ali Valining uyiga borgunicha eng kamida necha qadam tashlashi kerakligini aniqlang.

K:	2	5	7	99	1000000	999999	999998	86	89
CH:	1	1	2	21	200000	200001	200002	19	19

Tarmoqlanuvchi 191. Ali yasagan Robot Oxy koordinatalar sistemasida (x1; y1) nuqtadan (x2; y2) nuqtaga borishi kerak ($-10^9 \le x1, y1, x2, y2 \le 10^9$). Robot har bir qadamda koordinatalarini bir birlikka o'zgartirishi, ya'ni birinchi yoki ikkinchisini yoki ikkalasini oshirishi yoki kamaytirishi mumkin. Shunday qilib Robot 8 ta yo'nalishda harakatlanishi mumkin ekan. Kerakli manzilga borish uchun Robotga kerak bo'ladigan eng kam qadamlar sonini aniqlang.

K:	0 0 4 5	3 4 6 1	-1 -1 -10 100	-1000000000 -1000000000 0 999999999	0 0 2 1	1 1 -3 4	12 16 12 1
CH:	5	3	101	1999999999	2	4	15

Tarmoqlanuvchi 192. Alida rangli sharchalarning qizilidan A ta, sarig'idan B ta, yashilidan C ta bor ($0 \le A, B, C \le 1000000$). U 2 ta bir xil rangli sharchadan 1 ta boshqa rangli sharcha hosil qila oladi. Unga X ta qizil, Y ta sariq va Z ta yashil ($0 \le X, Y, Z \le 1000000$) sharcha kerak. Agar Ali kerakli sharchalarni hosil qila olsa "Yes", aks holda "No" chiqaring.

K:	4 4 0 2 1 2	5 6 1 2 7 2	0 0 0 0 0 0	500000 1000000 500000 750001 0 750000	191789 291147 691092 324321 416045 176232	1 2 4 2 1 3	4 0 0 0 1 1
CH:	Yes	No	Yes	No	Yes	No	Yes

Tarmoqlanuvchi 193. Butun N ($1 \le N \le 2 \cdot 10^9$) soni berilgan. Uzunligi N bo'lgan tekis uzun yog'ochning 3 ta joyidan shunday arralanadiki, natijada uzunligi butun son bo'lgan 4 ta tayoqcha hosil bo'ladi. Turli xil shu kabi arralash yordamida tayyorlangan tayoqchalardan yasash mumkin bo'lgan to'g'ri to'rtburchaklar sonini aniqlang. Bunda kvadrat bo'lgan hollar hisobga olinmasligi kerak.

K:	6	20	2	2000000000	1924704072	3	854000	1967345604
CH:	1	4	0	499999999	481176017	0	213499	491836400

Tarmoqlanuvchi 194. Butun N va A ($1 \le A \le N \le 100000$, N juft) sonlari berilgan. Ali yashaydigan ko'chada N ta uy bor. Ko'chada chap tomondagi uylarga ko'cha boshidan oxiriga qarab 1, 3, ... kabi toq tartib raqami, o'ng tomonidagi uylarga ko'cha boshidan oxiriga qarab N, N-2, ... kabi juft tartib raqami berilgan. Ali ko'cha boshida turibdi. U velosipedda ko'cha boshidan chap yoki o'ng tomondagi 1-uylarga, ya'ni chap tomondagi 1 tartib raqamli yoki o'ng tomondagi N tartib raqamli uylarga 1 minutda keladi. Ikki uy orasidagi masofani ham velosipedda 1 minutda bosib o'tadi. Ali ko'cha boshidan A tartib raqamli uyga borishga sarflaydigan vaqtni minutlarda aniqlang.

K:	4 2	8 5	100000 3	100000 2	96998 8992	3000 34	30518 286	24842 1038
CH:	2	3	2	50000	44004	1484	15117	11903

Tarmoqlanuvchi 195. Butun d, L, v1, v2 ($1 \le d, L, v1, v2 \le 10000, d < L$) va musbat t0 ($0 < t0 \le 100$) sonlari berilgan. Orasidagi masofa L metr bo'lgan ikki velosipedchi bir-biriga qarab yo'lga tushdi. Birinchi velosipedchining tezligi v1 metr/sekund, ikkinchisiniki esa v2 metr/sekund bo'lib, ular orasidagi masofa d metr qolguncha yaqinlashib kelishi kerak edi. Agar ular bu vazifani t0 vaqt ichida bajara olishsa "Ha", aks holda "Yo'q" javobini chiqaring. Hisoblashning aniqlik darajasi 10^{-6} bo'lsin.

K:	2 6 2 2 1	1 9 1 2 2.5	1 9 1 2 2.66667	9999 10000 10000 10000 0.000049	1 2 1 1 0.5
CH:	Ha	Yo'q	Ha	Yo'q	Ha

Tarmoqlanuvchi 196. Butun A, B, C va N ($0 \le A, B, C, N \le 100$) sonlari berilgan. Ali o'qigan sinfda N ta o'quvchi bo'lib, ba'zi sinfdoshlari oliy o'quv yurtiga kira olishmadi. O'qishga kirgan sinfdoshlari 2 ta choyxonada nishonlash marosimini o'tkazishdi. O'qishga kirgan sinfdoshlaridan ba'zilari choyxonaning birinchisida, ba'zilari choyxonaning ikkinchisida, ba'zilari esa ikkalasida nishonlash marosimida qatnashishga ulgurishdi. Nishonlash marosimiga qatnashganlar rasmga tushib telegramdagi "Neutrino" guruhiga yuklab turishdi. Ali o'quv yurtiga kira olmagan sinfdoshlari sonini bilmasdi, lekin o'zi kira olmagani aniq edi. Shu sababli rasmlarga qarab A ta sinfdoshi birinchi choyxonada, B ta sinfdoshi ikkinchi choyxonada, C ta sinfdoshi esa ikkala choyxonada ham qatnashgan, deb hisobladi. Agar Ali hisobda adashmagan bo'lsa, nechta sinfdoshi o'quv yurtiga kira olmaganini aniqlang, aks holda "-1" chiqaring.

K:	10 10 5 20	2 2 0 4	1 1 0 1	98 98 97 100	1 5 2 10	8 45 2 67	36 36 18 65
CH:	5	-1	-1	1	-1	16	11

Tarmoqlanuvchi 197. Butun N ($1 \le N \le 2 \cdot 10^{18}$) soni berilgan. Besh sonining N darajasining oxirgi 3 ta raqamini aniqlang. Agar besh sonining N darajasida raqamlar soni 3 tadan kam bo'lsa, u holda oldiga kerakli sonda 0 raqamlarini chiqaring.

K:	1	2	111111111111111	3	1999999999999999999	4
CH:	005	025	125	125	125	625

Tarmoqlanuvchi 198. Butun A va B ($1 \le A, B \le 10^{18}$) sonlari berilgan. A dan B gacha bo'lgan barcha sonlarning eng katta umumiy bo'luvchisini, ya'ni EKUB(A, A+1, ..., B) ni hisoblang.

K:	1 1000000000000000000	18033988749894848 18033988749894848
CH:	1	18033988749894848

Tarmoqlanuvchi 199. Butun d, h, v, e ($1 \le d, h, v, e \le 10^4$) sonlar berilgan. Ali "Fanta" ichimligini yaxshi ko'rgani uchun "Fanta" ishlab chiqaruvchi kompaniyaga ishga kirdi. U diametri d bo'lgan silindrsimon idishga h santimetr balandlikda "Fanta" ichimligidan quyib oldi. Shundan so'ng yoqtirgan ichimligi tugab qolmasligi uchun idish ustida quyish jo'mragini ochib idishdagi ichimlikni sekundiga v millilitr tezlik bilan icha boshladi. Ali idishdan ichmay turganda jo'mrakdan oqayotgan ichimlik hisobiga idishdagi ichimlik sathi sekundiga e santimetr ko'tarilgan bo'lardi. Ma'lumki, 1 millilitr 1 kub santimetrga teng. Agar Alining idishi hech bo'sh qolmasa "Mazza", aks holda "Tugadi" va idish birinchi marta bo'shagan sekundni chiqaring. Hisoblash aniqligi 10^{-4} bo'lsin.

K:	1 2 3 100	1 1 1 1	48 7946 7992 72	20 287 3845 5	20 2961 9852 15	2 5000 12 3
CH:	Mazza	Tugadi 3.65979	Mazza	Tugadi 39.6463	Tugadi 180.9914	Tugadi 6099.6539

Tarmoqlanuvchi 200. Butun N ($1 \le N \le 10^9$) soni berilgan. N sonini turli butun sonlarga bo'laklash mumkin, masalan, N=3 bo'lsa 1, 1, 1 yoki 1, 2 kabi. Ketma-ket bir xil son uchramaydigan bo'laklash usulida N sonini ajratish mumkin bo'lgan eng ko'p bo'laklar sonini aniqlang. Masalan, N=3 bo'lsa 1, 2 yoki N=4 bo'lsa 1, 2, 1.

K:	1	2	3	4	6	102	1000000000	521	522
CH:	1	1	2	3	4	68	666666667	347	348

Tarmoqlanuvchi 201. Butun N ($1 \le N \le 10^6$) soni berilgan. Neytrino sayyorasida bir yil N kundan iborat. Hafta kuni esa Yer sayyorasidagi kabi bo'lib, 5 ish kuni va 2 (shanba, yakshanba) dam olish kunidan iborat. Neytrino sayyorasining bir yilidagi dam olish kunlarining eng kam va eng ko'p sonini aniqlang. Masalan, Neytrino sayyorasining bir yili 14

kun bo'lsa, u holda yil haftaning qaysi kunidan boshlanishining ahamiyati yo'q, javob 4 4, bir yili 2 kun bo'lsa, u holda haftaning kuniga bog'liq bu kunlar ish kuni ham, dam olish kuni ham bo'lishi mumkin, javob 0 2.

K:	1	2	9	1000000	30	95	365	1963
CH:	0 1	0 2	2 4	285714 285715	8 10	26 28	104 105	560 562

Tarmoqlanuvchi 202. Butun A, B va C ($-10^9 \le A, B, C \le 10^9$) sonlari berilgan. Shunday ketma-ketlik tuzamiz: birinchi hadi A bo'lsin, ketma-ket kelgan ikkita hadda keyingi hadidan oldingi hadini ayirsak C ga teng bo'lsin. Agar B soni shu ketma-ketlikning birorta hadiga teng bo'lsa "Ha", aks holda "Yo'q" javobini chiqaring.

K:	1 -4 5	1 7 3	0 60 50	10 10 0	-1000000000 1000000000 5	115078364 -899474523 -1
CH:	Yo'q	Ha	Yo'q	Ha	Ha	Ha

Tarmoqlanuvchi 203. Butun N, A va B ($1 \le N \le 100$, $1 \le A \le N$, $-100 \le B \le 100$) sonlari berilgan. Alining mahallasida uylar aylana bo'ylab qurishgan bo'lib, barchasining eshigi aylananing tashqarisiga qaragan. Uylar 1 dan N gacha tartiblangan, ya'ni 1 va N tartib raqamli uylar yonma-yon joylashgan. Alining uyiga A tartib raqami berilgan. Ali uyidan chiqib B ta uy oldidan soat mili yo'nalishida (B musbat bo'lsa) yoki soat miliga qarshi yo'nalishda (B manfiy bo'lsa) sayr qilib o'tadi yoki o'z uyi oldida (B=0 bo'lsa) sayr qiladi. Agar Ali sayrining oxirida o'z uyi oldiga kelsa "Ha", aks holda qaysi uy oldiga kelganini chiqaring.

K:	6 2 -5	5 1 3	3 2 7	99 23 15	87 50 0	97 37 -92	61 43 -82	51 43 -51	21 7 63
CH:	3	4	3	38	Uy	42	22	Uy	Uy

Tarmoqlanuvchi 204. Butun N, K va A ($1 \le N, K, A \le 10^9$) sonlari berilgan. N sonidan katta K ga karrali butun sonlarning eng kichigi X bo'lsin. Agar A soni [N; X] kesmaga tegishli bo'lsa "Ha", aks holda "Yo'q" javobini chiqaring.

K:	5 3 4	26 13 35	197 894 895	97259 41764 125292	76770926 13350712 80104251
CH:	Yo'q	Ha	Yo'q	Ha	Ha

Tarmoqlanuvchi 205. Shaxmat doskasidagi harflarni raqamlar bilan almashtirsak, u holda u 1..8×1..8 kataklardan iborat. Kiritilgan k sonining x va y ($k = \overline{xy}$, $1 \le x, y \le 8$) raqamlariga asosan Shoh yurishi mumkin bo'lgan kataklar sonini aniqlang. Masalan, Shoh k=34, ya'ni x=3 va y=4 katakda turgan bo'lsa, u holda 8 ta katakka yura oladi: 23, 24, 25, 33, 43, 44, 45, 35.

K:	13	34	81	18	43	31	88	11	85	58
CH:	5	8	3	3	8	5	3	3	5	5

Tarmoqlanuvchi 206. Butun N ($0 \le N \le 10^9$) soni berilgan. 1378^N sonining oxirgi raqamini aniqlang.

Yo'llanma. Matematika, tahlil, realizatsiya.

K:	1	1000	999999999	1378	51202278	999999998	999999997	989898989
CH:	8	6	2	4	4	4	8	8

Tarmoqlanuvchi 207. Butun N, A, B va C ($0 \le N$, A, B, $C \le 10^9$) sonlari berilgan. Alining N ta daftari bor edi. Alining maxsus maktabida faqat 4 ta fan o'qitilgani uchun har bir fandan bir xil sondagi daftar tutishga harakat qiladi. Do'konda 1 ta daftar A so'mga, birdaniga 2 ta daftar sotib olinsa, ikkitasi B so'mga, birdaniga 3 ta daftar sotib olinsa, uchtasi C so'mga sotiladi. Daftarlari 4 ga bo'linadigan son bo'lishi uchun yetishmayotgan daftarlarni sotib olishga Ali sarflaydigan eng kam pul miqdorini aniqlang. Boshqacha aytganda, N+K soni 4 ga qoldiqsiz bo'linishi uchun zarur bo'lgan K ta daftar sotib olishga Ali sarflaydigan eng kam pul miqdorini aniqlang.

Yo'llanma. Matematika, tahlil, realizatsiya.

K:	1 1 3 4	6 2 1 1	4 4 4 4	188683934 254114048 48014511 170254369	3 12 3 4
CH:	3	1	0	48014511	7

Tarmoqlanuvchi 208. Butun x1, x2 va x3 ($1 \le x1, x2, x3 \le 100$) sonlari berilgan. Ali, Vali va Soli Ox o'qida yashaydi. Alining uyi x1 nuqtada, Valining uyi x2 nuqtada va Solining uyi x3 nuqtada joylashgan. Ular birgalikda eng kam masofa bosib o'tib bir nuqtada uchrashishi zarur. Shu masofani aniqlang.

Yo'llanma. Matematika, tahlil, realizatsiya.

K:	7 1 4	30 20 10	71 85 88	30 38 99	9 94 77	1 53 51	25 97 93	21 5 93
CH:	6	20	17	69	85	52	72	88

Tarmoqlanuvchi 209. Butun N ($1 \le N \le 1000$) soni berilgan. Shunday M ($1 \le M \le 1000$) sonini aniqlangki, N·M+1 soni murakkab son bo'lsin. Ma'lumki, murakkab son kamida 3 ta bo'luvchiga ega. Shu xususiyatga ega bo'lgan M soni ko'p bo'lsa ixtiyoriy biitasini chiqaring. Masalan:

Yo'llanma. Matematika, tahlil, realizatsiya.

K:	3	4	2	11	11	998	998	210	270
CH:	1	2	4	9	1	996	1	4	4

Tarmoqlanuvchi 210. Butun N ($1 \le N \le 10^9$) soni berilgan. Ali va Vali quyidagicha o'yin o'ynashmoqda. Ali varaqqa biror natural N sonini yozadi va har bir yurishda:

- Vali varaqdagi N sonidan biror juft A ($1 \le A \le N$) sonni ayiradi va yangi N sonini hosil qiladi;
- Ali varaqdagi N sonidan biror toq B ($1 \le B \le N$) sonni ayiradi va yangi N sonini hosil qiladi;
- Ali va Vali navbatma-navbat yuradi.

Agar birorta o‘yinchi yura olmasa, yutqazgan hisoblanadi. Agar ikkala o‘yinchi samarador o‘ynasa, u holda g‘olib o‘yinchini aniqlang.

K:	1	2	10000	33333	1000000000	123123123
CH:	Ali	Vali	Vali	Ali	Vali	Ali

Tarmoqlanuvchi 211. Butun A, B va C ($1 \le A, B, C \le 1000$) sonlari berilgan. Kompot tayyorlashda limon, olma va nok 1:2:4 nisbatda solinadi. Bozorda limon A ta, olma B ta va nok C ta bo‘lsa, kompot tayyorlash uchun sotib olish mumkin bo‘lgan mevalarning birgalikdagi eng ko‘p sonini aniqlang. Masalan, bozorda mevalar A=4 ta, B=7 ta va C=13 ta bo‘lsa, u holda kompot uchun $3 \cdot 1:3 \cdot 2:3 \cdot 4$ nisbat o‘rinli, demak, 3+6+12=21 ta meva sotib olinadi.

K:	2 5 7	2 3 2	1000 1000 1000	1000 2 1000	1000 1000 3	100 200 399
CH:	7	0	1750	7	0	693

Tarmoqlanuvchi 212. Butun x, y, p va q ($0 \le x \le y \le 10^9$, $0 \le p \le q \le 10^9$, $y>0$, $p>0$) sonlari berilgan. (x+m)/(y+n+m) kasrni p/q ko‘rinishidagi qisqarmas kasr ko‘rinishiga keltirish uchun zarur bo‘lgan n va m sonlari yig‘indisini aniqlang. Agar buning imkoni bo‘lmasa -1 chiqaring.

Yo‘llanma. Matematika, tahlil, realizatsiya.

K:	3 10 1 2	7 14 3 8	20 70 2 7	5 6 1 1	127371699 575154303 513866599 558504245
CH:	4	10	0	-1	5568392392

Tarmoqlanuvchi 213. Butun A va B ($1 \le A, B \le 10^9$) sonlari berilgan. Ali va Valiga do‘stlari, mos ravishda, A ta va B ta konfet sovg‘a qilishdi. Ali Valiga 1 ta konfet berdi. Ali bergan konfetni yegan Vali o‘zini ko‘rsatish uchun Aliga 2 ta konfet berdi. Vali bergan konfetlarni yegan Ali endi Valiga 3 ta konfet berdi. Konfetlarni yeb bo‘lgan Vali Aliga 4 ta konfet berdi. Shu qonuniyatda do‘sti bergan konfetlarni yeb do‘stiga o‘ziga berilgan konfetdan 1 ta ortiq konfet beraverishdi. Do‘stlarning qaysi biri shu qonuniyatda do‘stiga konfet bera olmasligini aniqlang.

Yo‘llanma. Matematika, tahlil, realizatsiya.

K:	1 1	7 6	25 38	8311 2468	250708 857756	999963734 999994456
CH:	Vali	Ali	Ali	Vali	Ali	Ali

Tarmoqlanuvchi 214. Butun N va M ($1 \le N \le M \le 10^9$) sonlari berilgan. $N \le x \le M$ shartni bajaruvchi barcha butun sonlarning 1 dan farqli barcha bo‘luvchilarini vergul bilan ajratib yozib chiqsak, shu ketma-ketlikda eng ko‘p uchraydigan sonni aniqlang. Agar to‘g‘ri javoblar bir nechta bo‘lsa, u holda ixtiyoriy bittasini chiqaring. Masalan:

K:	19 29	3 6	9 12	9 12	96 99	760632747 760632751	908580371 908580373
CH:	2	3	10	11	3	3	908580372

Tarmoqlanuvchi 215. Butun N ($1 \le N \le 10^5$) soni berilgan. Agar sonning o'nlik yozuvida faqat bitta 0 dan farqli raqam bo'lsa, u holda son go'zal deyiladi. Joriy N-yildan keyingi go'zal yilgacha qolgan yilni aniqlang.

Yo'llanma. Matematika, realizatsiya.

K:	4	909	10	201	4000	3450	70813	454	99999
CH:	1	91	10	99	1000	550	9187	46	1

Tarmoqlanuvchi 216. Butun N ($0 \le N \le 10^{18}$) soni berilgan. Tortni kesmalar yordamida bir xil o'lchamdagi va shakldagi N+1 ta teng bo'lakka bo'lish kerak. Kesmalar sonini aniqlang.

K:	3	4	100	1099999999999999	798456000458874899	701	123456
CH:	2	5	101	550000000000000	399228000229437450	351	123457

Tarmoqlanuvchi 217. Butun A va B ($1 \le A \le B \le 20$) sonlari berilgan. Alining og'irligi A kilogramm, Valining og'irligi esa B kilogramm edi. Ali va Vali ko'p ovqat yeyishardi, shu sababli Alining og'irligi yiliga avvalgi yilgi og'irligiga nisbatan 3 marta, Valining og'irligi esa yiliga avvalgi yilgi og'irligiga nisbatan 2 marta oshardi. Necha yildan keyin Alining og'irligi Valining og'irligidan ortiq bo'lishini aniqlang.

Yo'llanma. Matematika, tahlil, realizatsiya.

K:	4 7	4 9	1 1	1 10	1 15	2 8	4 10	1 2	1 20	1 17
CH:	2	3	1	6	7	4	3	2	8	7

Tarmoqlanuvchi 218. Butun $x1, y1, x2, y2$ ($-10^5 \le x1, y1, x2, y2 \le 10^5$) va a, b ($1 \le a, b \le 10^5$) sonlari berilgan. Alining roboti (x; y) nuqtadan faqat quyidagi 4 ta nuqtadan biriga yura oladi:

(x; y) →(x+a; y+b) yoki (x; y) →(x-a; y-b) yoki (x; y) →(x-a; y+b) yoki (x; y) →(x+a; y-b). Agar robot yuqoridagi mumkin bo'lgan yurishlar yordamida (x1; y1) nuqtadan (x2; y2) nuqtaga bora olsa "HA", aks holda "YO'Q" javobini chiqaring.

Yo'llanma. Vaziyatlar tahlili, qonuniyat.

K:	0 0 0 6 2 3	1 1 3 6 1 5	70 -81 -17 80 87 23	-58533 -50999 -1007 -59169 8972 23345	14 88 14 88 100 500	-4 -8 -6 -1 1 3
CH:	HA	YO'Q	HA	YO'Q	HA	YO'Q

Tarmoqlanuvchi 219. Butun L, R va U ($0 \le L, R, U \le 100$) sonlari berilgan. Bouling o'yiniga L ta chap qo'lda, R ta o'ng qo'lda va U ta ikkala qo'lida ham o'ynay oladiganlar borardi. Komanda rahbari musobaqa uchun tuzilayotgan komanda a'zolarini shunday juftliklarga ajratishni maqsad qilib qo'ydiki, har bir juftlikdagi o'yinchining biri chap qo'lda va ikkinchisi o'ng qo'lda o'ynay olishi shart. Shu usulda juftlikka ajratilsa, musobaqaga nechta o'yinchi borishini aniqlang.

K:	1 4 2	5 5 5	0 2 0	30 70 34	89 32 24	89 44 77	100 100 100	1 1 1
CH:	6	14	0	128	112	210	300	2

Tarmoqlanuvchi 220. Butun A va B ($1 \le A, B \le 10^9$) sonlari berilgan. O'yin boshida Ali va Valida 1 balldan bor edi. O'yin juda sodda: konvertdan biror k natural soni yozilgan varaq olinadi va birinchi bo'lib shu sonni so'z bilan aytgan o'yinchi yutadi. Shunda o'yinda yutgan o'yinchining ballari k^2 marta, yutqazgan o'yinchining ballari k marta oshiriladi. O'yin bir necha raund davom etishi ham mumkin. Ertasi kuni Ali o'yin natijasini eslashga urinib ko'rdi, so'ng o'zi va Valining ballarini yozdi. Agar Ali yozgan o'yin natijasi to'g'ri bo'lishi mumkin bo'lsa "Ha", aks holda "Yo'q" javobini chiqaring. Masalan, Ali "2 4" yozgan bo'lsa, k=2 bo'lganda Ali yutqazgan va Vali yutgan holda ballar $1 \cdot 2 = 2$ va $1 \cdot 2^2 = 4$ bo'ladi, demak, javob "Ha".

Yo'llanma. Masala yechimi quyidagi ko'rinishdagi sistemani yechishga keladi:

$$\begin{cases} S^2 \cdot P = A \\ S \cdot P^2 = B \end{cases}$$

Sistema butun yechimga ega bo'lsa javob "HA".

K:	75 45	8 8	16 16	247 994	1000000000 1000000	12345 67890
CH:	HA	HA	YO'Q	YO'Q	HA	YO'Q

Tarmoqlanuvchi 221. Butun n, k, t ($1 \le n \le 10^9$, $1 \le k \le n$, $1 \le t < n + k$) sonlari berilgan. Futbol ishqibozlari "meksika to'lqini" nimaligini yaxshi bilishadi. Stadionda 1 dan n gacha tartiblangan ishqibozlar o'tirishgan bo'lsin. "Meksika to'lqini"ni 0-minutdan boshlab quyidagicha davom ettirishadi:

1-minut: 1-ishqiboz turadi;

2-minut: 2-ishqiboz turadi;

...

k-minut: k-ishqiboz turadi;

k+1-minut: (k+1)-ishqiboz turadi, 1-ishqiboz o'tiradi;

k+2-minut: (k+2)-ishqiboz turadi, 2-ishqiboz o'tiradi;

...

n-minut: n-ishqiboz turadi, (n-k)-ishqiboz o'tiradi;

(n+1)-minut: (n+1-k)-ishqiboz o'tiradi;

...

n+k-minut: n-ishqiboz o'tiradi.

"Meksika to'lqini"ning t-minutida turgan ishqibozlar sonini aniqlang.

K:	10 5 3	10 5 7	10 5 12	840585600 770678331 788528791	100 100 99
CH:	3	5	3	770678331	99

Tarmoqlanuvchi 222. Butun N, A, B va C ($1 \le N, A, B, C \le 100$) sonlari berilgan. Har kuni Ali N marta mehmonga boradi. Uning 3 ta do'sti bor: Vali, Soli va Dali. Vali va Soli uyi orasi A metr, Vali va Dali uyi orasi B metr, Soli va Dali uyi orasi C metr edi. Buning uchun u bir do'stining uyidan chiqib boshqa ikki do'stidan birining uyiga boraveradi. Hozir Ali Valining uyida mehmonda o'tiribdi. U yana N-1 marta mehmonga borishi kerak bo'lsa, u holda Ali bosib o'tadigan eng qisqa masofani aniqlang.

K:	3 2 3 1	1 2 3 5	7 10 5 6	9 9 7 5	76 46 77 11	65 5 8 7	73 71 69 66	100 100 100 100
CH:	3	0	30	42	860	320	4755	9900

Tarmoqlanuvchi 223. Butun V1, V2, V3, Vm ($1 \le V1, V2, V3, Vm \le 100$, $V1 > V2 > V3$) sonlari berilgan. A hajmli obyekt B hajmli avtomobilga sig'ishi uchun $A \le B$ shart bajarilishi zarur va yetarli bo'lsin. A hajmli obyektga B hajmli avtomobil yoqishi uchun $2 \cdot A \ge B$ shart bajarilishi zarur va yetarli bo'lsin. Ota ayiq, ona ayiq, bola ayiqlarning oilasida 3 ta avtomobil bor. Eng katta avtomobilga ota ayiq sig'adi va bu avtomobil unga yoqadi, o'rtacha avtomobilga ona ayiq sig'adi va bu avtomobil unga yoqadi, eng kichik avtomobilga bola ayiq sig'adi va bu avtomobil unga yoqadi. Mehmonga kelgan Masha hamma avtomobillarga sig'di, lekin unga eng kichik avtomobil yoqib qoldi. Ota ayiq V1, ona ayiq V2, bola ayiq V3, Masha Vm hajmli bo'lsin. Avtomobillarning shunday natural qiymatlarini topingki, yuqoridagi shartlar bajarilsin, aks holda -1 chiqaring. Agar yechimlar bir nechta bo'lsa, ixtiyoriy bittasini chiqaring. Masalan:

K:	50 30 10 10	50 30 10 10	100 50 10 21	97 95 3 2	3 2 1 1	100 50 19 25
CH:	50 30 10	100 60 10	-1	97 95 3	4 3 1	100 51 25

Tarmoqlanuvchi 224. Butun x_i va y_i ($|x_i|, |y_i| \le 10^9$, $x_i \ne 0$, i=1,2,3,4,5) sonlari berilgan. Berilgan (xi; yi) juftliklar Oxy koordinata tekisligida nuqtalarni aks ettiradi, shartga muvofiq birortasi ham Oy o'qida yotmaydi. Agar shu nuqtalardan biror bittasini olib tashlangach qolgan barcha nuqtalar Oy o'qining bir tomonida yotadigan bo'lsa "Ha", aks holda "Yo'q" chiqaring.

K:	6 6	1 1	-1 1	1 1	724209108 -936840259	-1 -1	-2 3	2 0	1 0
	7 7	2 2	-1 2	2 2	964531145 -108229816	-2 2	-3 3	3 0	3 0
	8 8	3 3	-2 4	-1 1	408514704 9410504	2 2	4 2	4 0	-1 0
	9 9	4 4	-7 -8	-2 2	168170846 258776371	2 -2	3 2	-5 0	-6 0
	-1 -1	5 5	-3 3	5 -1	172237060 -41787470	3 2	1 2	5 0	-4 1
CH:	Ha	Ha	Ha	Yo'q	Ha	Yo'q	Yo'q	Ha	Yo'q

Tarmoqlanuvchi 225. Butun N va M ($1 \le N, M \le 10^8$) sonlari berilgan. M mod 2^N sonini chiqaring (a mod b – bu a ni b ga bo'lgandagi qoldiq).

K:	4 42	1 58	98765432 23456789	8 88127381	32 92831989	25 33554432	26 92831989
CH:	10	0	23456789	149	92831989	0	25723125

Tarmoqlanuvchi 226. Butun X va Y ($0 \le X,\ Y \le 10^9$) sonlari berilgan. Alida 1 ta o'yinchoq bor edi. U nusxalovchi qurilma topib oldi. Qurilmaning ishlashi g'aroyib edi: agar asosiy o'yinchoq nusxalansa, 1 ta asosiy va 1 ta nusxa o'yinchoq hosil qilardi, agar nusxa o'yinchoq nusxalansa, 2 ta nusxa o'yinchoq hosil qilardi. Ali shunday savol berdi: nusxalash qurilmasini qo'llab X ta nusxa va Y ta asosiy o'yinchoq hosil qilish mumkinmi? Agar mumkin bo'lsa "Yes", aks holda "No" javobini chiqaring.

K:	6 3	4 2	1000000000 999999999	81452244 81452247	779351061 773124120
CH:	Yes	No	Yes	No	Yes

Tarmoqlanuvchi 227. Butun A, B, x, y va z ($0 \le A, B, x, y, z \le 10^9$) sonlari berilgan. Ali 2 ta sariq kristaldan bitta sariq sharcha, 1 ta sariq va 1 ta ko'k kristaldan bitta yashil sharcha, 3 ta ko'k kristaldan bitta ko'k sharcha hosil qilish usulini o'rgandi. Alining A ta sariq va B ta ko'k kristali bor. Unga x ta sariq, y ta yashil va z ta ko'k sharcha kerak edi. Kerakli sondagi sharchalarni hosil qilish uchun Aliga yana nechta kristal kerak bo'lishini aniqlang.

K:	4 3 2 1 1	3 9 1 1 3	12345678 87654321 43043751 1000000000 53798715	12 12 3 5 2	770 1390 170 442 311	100 10 2 3 4
CH:	2	1	2147483648	0	12	5

Tarmoqlanuvchi 228. Butun a, b va c ($1 \le a, b, c \le 100$) sonlari berilgan. Alida uzunligi a, b va c santimetr bo'lgan to'g'ri tayoqchalar bor. U 1 minutda tayoqchalardan ixtiyoriy bittasining uzunligini 1 santimetrga uzaytira oladi. Shu tayoqchalardan musbat yuzali uchburchak yasash mumkin bo'lishi uchun Ali sarflaydigan eng kam vaqtni aniqlang.

Yo'llanma. Agar tayoqchalar $a \le c$ va $b \le c$ shartlar bajarilganda edi, u holda masala yechimini quyidagi ifodadan aniqlash mumkin bo'lardi: t=max(0, c – a – b + 1);

K:	3 4 5	2 5 3	100 10 10	100 1 1	12 34 56	23 56 33	98 12 23	14 21 76
CH:	0	1	81	99	11	1	64	42

Tarmoqlanuvchi 229. Butun s, v1, v2, t1, t2 ($1 \le s, v1, v2, t1, t2 \le 1000$) sonlari berilgan. Ali va Vali "Alifbo" sayti yordamida matn terish bo'yicha musobaqalashmoqchi. Musobaqada saytdan s ta belgili matn Aliga t1, Valiga t2 millisekundda yuboriladi. Ali v1 millisekundda 1 ta belgi, Vali esa v2 millisekundda 1 ta belgi tera oladi. Ali matnni to'liq terib bo'linganidan t1 millisekund, Vali matnni to'liq terib bo'linganidan t2 millisekund o'tgach saytga xabar yetib boradi. To'liq terib bo'lingan matn haqidagi habar birinchi bo'lib yetib borgan yutgan hisoblanadi. Agar xabar bir vaqtda yetib borsa, u holda durang

hisoblanadi. Agar musobaqada Ali yutib chiqsa “Ali”, Vali yutib chiqsa “Vali”, aks holda “Durang” matnini chiqaring.

K:	5 1 2 1 2	4 5 3 1 5	15 14 32 65 28	637 324 69 612 998	13 849 819 723 918	9 5 7 8 7
CH:	Ali	Durang	Ali	Vali	Durang	Ali

Tarmoqlanuvchi 230. Butun n, x, y, d ($1 \le n, d \le 10^9$, $1 \le x, y \le n$) sonlari berilgan. Ali sahifalari 1 dan n gacha tartiblangan n sahifali elektron kitobni o‘qimoqda. U hozir x-sahifada bo‘lib, y-sahifaga o‘tmoqchi. Ali bitta harakatda d ta oldingi yoki keyingi sahifaga o‘ta oladi, lekin kitob sahifalaridan chiqib ketolmaydi. Masalan, n=10 va d=3 bo‘lsa, u holda Ali x=2 bo‘lganda oldinga o‘tmoqchi bo‘lsa 1-sahifaga, keyinga o‘tmoqchi bo‘lsa 5-sahifaga, x=1 bo‘lganda oldinga o‘tmoqchi bo‘lsa 1-sahifaga, keyinga o‘tmoqchi bo‘lsa 4-sahifaga, x=8 bo‘lganda oldinga o‘tmoqchi bo‘lsa 5-sahifaga, keyinga o‘tmoqchi bo‘lsa 10-sahifaga o‘ta oladi. Ali y-sahifaga o‘tishi uchun qilinadigan eng kam harakatlar sonini aniqlang.

K:	10	5	20	575154303	341436697	80	59	758
	4	1	4	518933616	330232197	49	58	235
	5	3	19	187312755	168141277	23	15	411
	2	4	3	484983724	11553028	19	1	2
CH:	4	-1	5	-1	16	5	43	88

Tarmoqlanuvchi 231. Butun N va R ($3 \le N \le 100$, $1 \le R \le 100$) sonlari berilgan. Radiusi R bo‘lgan doira atrofiga shu doiraga va bir-biriga faqat tashqi tomondan urinadigan N ta bir xil radiusli doira chizish mumkin. Tashqi doiralar radiusini aniqlang. Hisoblash aniqligi 10^{-8} bo‘lsin.

K:	3 1	6 1	100 100	5 6	78 87
CH:	6.464101615	1.000000000	3.242939086	8.555519989	3.650111980

5-BOB. C++ TILIDA TAKRORLANUVCHI ALGORITMLI DASTURLAR

Hozirgacha qo'llanmada dastur tuzish uchun berilgan vazifalarning chiziqli va tarmoqlanish algoritmli turlari ko'rib chiqildi. Umuman, biror masala uchun bir necha chiziqli algoritmli dastur birlashtirilsa, balki murakkabroq bo'lar, lekin chiziqli algoritmli dastur hosil bo'ladi. Mazmunan, chiziqli algoritmda bajariladigan amallar kelish tartibida ketma-ket va birma-bir bajarib boriladi.

Ma'lumki, tarmoqlanuvchi algoritmli dasturda biror shartning rost qiymatiga ko'ra amallarning birinchi chiziqli algoritmli qismigina bajariladi, ikkinchi qismi bajarilmaydi, va aksincha, shartning yolg'on qiymatiga ko'ra amallarning ikkinchi chiziqli algoritmli qismi bajariladi, birinchi qismi bajarilmaydi. Agar tarmoqlanuvchi algoritmli ikki va undan ortiq dastur birlashtirilsa, murakkabroq tarmoqlanuvchi algoritmli dastur hosil bo'ladi. Bu holatda har bir shartning rost yoki yolg'on qiymatiga ko'ra ma'lum bir amallar bajarilib, boshqalari bajarilmay qoladi.

Bu ikki turkumdagi vazifalarni MS Excel kabi amaliy dasturlar yordamida ham hal etish mumkin. Lekin vazifalarning shunday turlari borki, ularni dasturlash tilisiz hal etib bo'lmaydi. Bunday vazifalarning asosida takrorlash algoritmlari yotadi. Bunday vazifalarga misol sifatida sonlar ketma-ketligini tartiblash masalasini olish mumkin.

Takrorlanuvchi algoritmli dasturlarda aniq bir yoki bir necha amallar takror va takror bajarilish imkoniyati ko'zda tutilgan bo'ladi. Takrorlash amalga oshirilishi uchun dasturlash tilining takrorlashni ko'zda tutgan operatorlaridan foydalanish mumkin bo'ladi. Dasturlash tillarida bunday operatorlarni **takrorlash operatorlari** deb atashadi.

Takrorlanuvchi algoritmli dasturlarda boshlang'ich qiymati aniq bo'lgan butun a va b uchun **a=a+b;** (C++ tilida **a+=b;** kabi yozish ham mumkin!) ifoda ko'p qo'llaniladi. Bunda b soni o'zgarish qadami (qisqacha, **qadami**) deb aytiladi. Masalan, agar a=0 va b=1 bo'lsa, **a=a+1; (a++;)** ifoda bittalab "sanaydi":

a=a+1=0+1=1 (a ning avvalgi qiymati 0 edi, keyingi qiymati 1 bo'ldi);

a=a+1=1+1=2 (a ning avvalgi qiymati 1 edi, keyingi qiymati 2 bo'ldi);

a=a+1=2+1=3 (a ning avvalgi qiymati 2 edi, keyingi qiymati 3 bo'ldi); ...

Shu sababli takroriy bajarilayotgan a=a+1; (a++;) ifodani sanagich deb ham atashadi!

O'zgaruvchi a ning qiymati har safar 1 taga ortayotgani uchun qadami 1 bo'ladi.

Umumiyroq bo'lgan s=s+f(x_k); yoki p=p*f(x_k); (s+=f(x_k); yoki p*=f(x_k);), k=1, 2, ..., ifodalar orqali boshlang'ich qiymati aniq bo'lgan s o'zgaruvchisiga yig'indi yoki p o'zgaruvchisiga ko'paytma qiymati hisoblab boriladi:

k=1 da $s=s+f(x_1)$ (s ning boshlang'ich qiymati aniq edi, keyingi qiymati $s+f(x_1)$ bo'ldi);

k=2 da $s=s+f(x_2)=s+f(x_1)+f(x_2)$ (s ning avvalgi qiymati $s+f(x_1)$ edi, keyingi qiymati $s+f(x_1)+f(x_2)$ bo'ldi);

...

k=n da $s=s+f(x_n)=s+f(x_1)+f(x_2)+\ldots+f(x_n)$ (s ning avvalgi qiymati $s+f(x_1)+f(x_2)+\ldots+f(x_{n-1})$ edi, keyingi qiymati $s+f(x_1)+f(x_2)+\ldots+f(x_{n-1})+f(x_n)$ bo'ldi);

yoki

k=1 da $p=p*f(x_1)$ (p ning boshlang'ich qiymati aniq edi, keyingi qiymati $p*f(x_1)$ bo'ldi);

k=2 da $p=p*f(x_2)=p*f(x_1)*f(x_2)$ (p ning avvalgi qiymati $p*f(x_1)$ edi, keyingi qiymati $p*f(x_1)*f(x_2)$ bo'ldi);

...

k=n da $p=p*f(x_n)=p*f(x_1)*f(x_2)*\ldots*f(x_n)$ (p ning avvalgi qiymati $p*f(x_1)*f(x_2)*\ldots*f(x_{n-1})$ edi, keyingi qiymati $p*f(x_1)*f(x_2)*\ldots*f(x_n)$ bo'ldi).

Bu ifodalarda $s=s+f(x_k)$; yoki $p=p*f(x_k)$; ko'rinishdagi ifoda takroriy bajarilmoqda.

Yig'indi qiymatiga ta'sir etmasligi uchun s=0 boshlang'ich qiymat, ko'paytma qiymatiga ta'sir etmasligi uchun p=1 boshlang'ich qiymat olinadi.

1-§. DASTURLASHNING MATEMATIK ASOSLARIGA OID

Takrorlanuvchi algoritmli masalalarni yechishda matematik asos zarur bo'lib, bu bobda ham foydali ma'lumotlar va formulalar keltiriladi.

BO'LINISH, BO'LINISH ALOMATLARI

Bo'linish tushunchasi butun sonlar to'plamida kiritilgan bo'lib, bo'linish haqida gap borganda ishtirokchi sonlar butun sonlar ekanligi nazarda tutiladi.

Ta'rif-1. A soni B soniga **bo'linadi** deyiladi, agar B>0 va A sonining B songa bo'linmasi butun son bo'lsa.

A soni B soniga bo'linsa, A|B kabi belgilanadi. Albatta, **0 ga bo'lish mumkin emas.**

Avvalgi boblardan ma'lumki, A soni B soniga bo'linsa, u holda shunday bir butun k soni mavjudki, $A=k\cdot B$ o'rinli bo'ladi. Demak:

$A|B \Leftrightarrow B>0$ va biror butun k sonda: $A=k\cdot B$.

Ta'rif-2. A soni B soniga bo'linsa, B soni A sonining **bo'luvchisi** deb ataladi.

Ta'rif-3. A soni B soniga **karrali** deyiladi, agar biror butun k soni uchun $A=k\cdot B$ o'rinli bo'lsa.

Ta'riflardan ko'rinadiki, karrali va bo'linish turli tushunchalar ekan. Masalan, 0 soniga karrali 0 soni bor, lekin 0 soniga bo'linadigan son yo'q.

Sonlarning bo‘linish alomatlari juda muhim ahamiyatga ega. Bu alomatlar asosida sonlarning bo‘luvchilarini, bo‘linuvchilarini topish, ularninig xossalarini o‘rganish mumkin. Berilgan natural

$$a = \overline{a_n a_{n-1} \dots a_1 a_0} = a_n \cdot 10^n + a_{n-1} \cdot 10^{n-1} + \dots + a_1 \cdot 10^1 + +a_0$$

sonining berilgan b natural songa bo‘linish yoki bo‘linmasligini aniqlash kerak bo‘lsin. 10 ning darajalarini b ga qoldiqli bo‘lamiz: $10=b\cdot q_1+r_1$; $10^2=b\cdot q_2+r_2$; ... ; $10^n=b\cdot q_n+r_n$.

Bu tengliklarni yuqoridagi tenglikka qo‘yib, shakl almashtirsak:

$$\mathbf{a = A\cdot b + B}$$

hosil bo‘ladi. Bu yerda $A = a_nq_n+a_{n-1}q_{n-1}+ \dots + a_1q_1$, $B = a_0+a_1r_1+ \dots +a_nr_n$.

Hosil bo‘lgan tenglikdan ko‘rinib turibdiki, **B soni b soniga bo‘linganda va faqat shu holda a soni b soniga bo‘linadi**. Bu xulosadan sonlarning bo‘linish belgilarini topishda foydalaniladi.

2 ga bo‘linish alomati: 10^k (k=1, 2, ... , n) ni b=2 ga bo‘lishdan chiqadigan qoldiqlar nolga teng. Shuning uchun $B = a_0$ bo‘ladi. Bundan a sonning oxirgi raqami 2 ga qoldiqsiz bo‘linsa, a soni 2 ga qoldiqsiz bo‘linadi degan xulosaga kelamiz.

3 va 9 ga bo‘linish alomati: 10 ning darajalarini $10^n=(9+1)^n=9\cdot A_n+1$ ko‘rinishda ifodalasak (bu yerda $A_n\in N$), 10^n darajalarni b=9 (yoki b = 3) ga bo‘lishdan chiqadigan qoldiqlar 1 ga tengligi kelib chiqadi. Shuning uchun $B=a_0+a_1+ \dots + a_n$ hosil bo‘ladi. Bundan ushbu qoida kelib chiqadi: agar berilgan a sonining raqamlari yig‘indisi 9 ga (3 ga) qoldiqsiz bo‘linsa, u holda bu son 9 ga (3 ga) qoldiqsiz bo‘linadi.

4 va 25 ga bo‘linish alomati: b=4 bo‘lganda $10=2\cdot b+2$, $10^2=25\cdot b+0$, $10^3=250\cdot b+0$, ... , $r_1=2$, $r_2=r_3= \dots =r_n=0$ bo‘lib, $B=a_0+2\cdot a_1$ bo‘ladi, ya’ni sonning 4 ga bo‘linishi uchun, uning birlik raqami bilan o‘nlik raqami ikkilanganining yig‘indisi 4 ga bo‘linishi zarur va yetarlidir. $B=a_0+2\cdot a_1$ ifodani bunday yozamiz:

$B_1=a_0+2\cdot a_1+8\cdot a_1= B+8\cdot a_1=10\cdot a_1+a_0=\overline{a_1a_0}$. Yoki $B= \overline{a_1a_0} - 8\cdot a_1$ bo‘lgani uchun B son $\overline{a_1a_0}$ soni 4 ga bo‘linganda va faqat shu holdagina 4 ga qoldiqsiz bo‘linadi. Bundan, oxirgi ikkita raqamidan tuzilgan son 4 ga bo‘linadigan sonlar va faqat shunday sonlar 4 ga bo‘linishi kelib chiqadi.

Xuddi shunday, oxirgi ikki raqamidan tuzilgan son 25 ga bo‘linadigan sonlar va faqat shunday sonlar 25 ga bo‘linadi.

5 ga bo‘linish alomati: 10^k (k=1, 2, ... , n) ni b=5 ga bo‘lishdan chiqadigan qoldiqlar nolga teng. Shuning uchun $B = a_0$ bo‘ladi. Bundan a sonning oxirgi raqami 5 ga qoldiqsiz bo‘linsa, bu son 5 ga qoldiqsiz bo‘linadi, degan xulosaga kelamiz.

6 ga bo‘linish alomati: 2 va 3 ga bo‘linsa, 6 ga ham bo‘linadi.

7 ga bo‘linish alomati. Bizda b=7 va $10=7+3$, $r_1=3$; $10^2=7\cdot 14+2$, $r_2=2$; $10^3=7\cdot 142+6$, $r_3=6$; $10^4=7\cdot 1428+4$, $r_4=4$; $10^5=7\cdot 14285+5$, $r_5=5$; $10^6=7\cdot 142857+1$, $r_6=1$.

10^7 da $r_7=3=r_1$ qoldiqlar qaytadan takrorlanmoqda. Topilgan natijalarni yuqoridagi ifodaga qo'yamiz. U holda $a=A\cdot7+B$ da $B=a_0+3\cdot a_1+2\cdot a_2+6\cdot a_3+4\cdot a_4+5\cdot a_5+a_6+3\cdot a_7+a_8+$ ... yoki koeffitsiyentlarni 7 ga nisbatan yozsak:

$$B=a_0+3\cdot a_1+2\cdot a_2+(7\cdot a_3-a_3)+(7\cdot a_4-3\cdot a_4)+(7\cdot a_5-2\cdot a_5)+...=7\cdot(a_3+a_4+a_5+a_9+a_{10}+a_{11}+...)+$$
$$+(a_0+3\cdot a_1+2\cdot a_2+a_6+3\cdot a_7+2\cdot a_8+...)-(a_3+3\cdot a_4+2\cdot a_5+a_9+3\cdot a_{10}+2\cdot a_{11}+...)$$

ifodani hosil qilamiz. Oxirgi ifodada:

$$a_0+3\cdot a_1+2\cdot a_2+a_6+3\cdot a_7+2\cdot a_8+...=B_2 \text{ va } a_3+3\cdot a_4+2\cdot a_5+a_9+3\cdot a_{10}+2\cdot a_{11}+...=B_1$$

kabi belgilash kiritsak, $a=7\cdot A+B_2-B_1$ tenglikka ega bo'lamiz. Shunday qilib, B_2-B_1 ayirma 7 ga qoldiqsiz bo'linsa, berilgan a son ham 7 ga qoldiqsiz bo'linishi kelib chiqadi. Yuqoridagi ifodalardan har 3 ta qo'shiluvchi uchun koeffitsiyentlarda "132" qonuniyat ko'zga tashlanmoqda: $a_0+3\cdot a_1+2\cdot a_2$, $a_6+3\cdot a_7+2\cdot a_8$, $a_3+3\cdot a_4+2\cdot a_5$, $a_9+3\cdot a_{10}+2\cdot a_{11}$, ... Bundan quyidagicha qonuniyatni hosil qilamiz:

Har uch xona son 231 ga xonalar bo'yicha ko'paytirib qo'shiladi.

Masalan, 675056742 sonini 7 ga bo'linishi yoki bo'linmasligini aniqlash kerak bo'lsin. U holda

742	056	675
231	231	231
$14+12+2=28$	$0+15+6=21$	$12+21+5=38$

bo'lgani uchun 38+28–21=66–21=45 soni 7 ga bo'linmaydi. Demak, berilgan son 7 ga bo'linmaydi.

11 ga bo'linish alomati: berilgan sonning juft o'rinda turgan raqamlari yig'indisidan toq o'rinda turgan raqamlari yig'indisi ayirilganda hosil bo'ladigan ayirma 11 ga bo'linsa, son 11 ga bo'linadi.

8 ga bo'linish alomati: oxirgi uchta raqamdan tuzilgan son 8 ga bo'linadigan sonlar 8 ga bo'linadi.

10 ga bo'linish alomati: oxirgi raqami 0 bo'lsa son 10 ga bo'linadi.

NATURAL SONLAR, TUB VA MURAKKAB SONLAR

Natural sonlar to'plami N dagi 1 soni faqat 1 ta bo'luvchiga ega (1 ning o'zi), 1 dan boshqa barcha natural sonlar kamida ikkita bo'luvchiga ega (ya'ni 1 va sonning o'zi). Masalan, 20 sonining bo'luvchilari 1, 2, 4, 5, 10 va 20 bo'lsa, 11 sonining bo'luvchilari 1 va 11 bo'ladi.

Ta'rif-4. Bo'luvchisi faqat 1 va o'zidan iborat bo'lgan 1 dan katta natural son **tub son** deyiladi.

Demak, tub sonning aniq 2 ta bir-biridan farqli bo'luvchisi bo'ladi, bu bo'luvchilar 1 soni va sonning o'zidir, ya'ni 2 tadan ko'p ham, kam ham emas. Masalan, 2, 3, 5, 7, 11, 13, 17, 19, 23 sonlari tub sonlardir.

Ta'rif-5. Bo'luvchilari soni 2 tadan ortiq bo'lgan natural son **murakkab son** deyiladi.

Masalan, 4, 6, 8, 9, 10, 12, 14, 15, 16, 18, 20 sonlari murakkab sonlardir.

Biror M sonidan katta bo'lmagan tub sonlarni aniqlashda **Eratosfen g'alviri** deb ataladigan oddiy usuldan foydalanish mumkin. Uning mohiyati quyidagicha:

- 1, 2, 3, ..., M sonlar qatorini olamiz;
- Sonlar qatoridagi 1 soni tub bo'lmagani uchun o'chiramiz;
- Sonlar qatorida qolgan birinchi soni 2 va u faqat 1 ga hamda o'ziga bo'linadi, demak, 2 tub son. Qatorda 2 ni qoldirib, qatordagi unga karrali bo'lgan barcha sonlarni o'chiramiz;
- 2 dan keyin turuvchi o'chirilmay qolgan son 3 (chunki u 2 ga bo'linmaydi). Demak, 3 faqat 1 ga va o'ziga bo'linadi, shuning uchun u tub son. Qatorda 3 ni ham qoldirib, qatordagi unga karrali bo'lgan barcha sonlarni o'chiramiz;
- 3 dan keyin turuvchi o'chirilmay qolgan birinchi son 5 (chunki u 2 va 3 ga bo'linmaydi). Demak, 5 faqat 1 ga va o'ziga bo'linadi, shuning uchun u tub son. Qatorda 5 ni ham qoldirib, qatordagi unga karrali bo'lgan barcha sonlarni o'chiramiz;
- va h.k.

Sonlar nazariyasida juda ko'p qiziqarli xossalar o'rinli:

1-xossa. Agar A soni B soniga bo'linsa, u holda A soni $\frac{A}{B}$ soniga ham bo'linadi.

Masalan, 10 soni 2 ga bo'linadi, demak, 10 soni $\frac{10}{2} = 5$ soniga ham bo'linadi. Umuman, A sonining har bir bo'luvchisi B uchun A sonining shunday C bo'luvchisi mavjudki, $A=B\cdot C$ bo'ladi. Bu mulohaza quyidagi xossaga olib keladi:

2-xossa. A sonining $\sqrt{A}$ dan kichik B bo'luvchisi uchun $\sqrt{A}$ dan katta yagona C bo'luvchisi mavjudki, $A=B\cdot C$ bo'ladi.

Masalan, 20 soni uchun $\sqrt{20} \approx 4{,}47$ bo'ladi. Demak, 20 sonining bo'luvchilarini 4,47 dan kichik va katta juftliklarga ajratish mumkin: 1 va 20, 2 va 10, 4 va 5. Ildizi butun son 4 bo'lgan 16 uchun esa 4 dan kichik va katta juftliklar quyidagilar: 1 va 16, 2 va 8. Tub sonlar uchun esa bunday juftliklar 1 va sonning o'zi. Bu misollar yordamida quyidagi xulosaga kelamiz:

3-xossa. A sonining ildizi $\sqrt{A}$ butun son bo'lsa, u holda A sonining bo'luvchilari soni toq, aks holda, ya'ni A sonining ildizi ildizi $\sqrt{A}$ butun son bo'lmasa, u holda A sonining bo'luvchilari soni juft bo'ladi.

4-xossa. Birdan katta har qanday p natural sonining 1 dan katta bo'luvchilarining eng kichigi tub sondir.

5-xossa. Murakkab p sonining 1 dan katta eng kichik bo‘luvchisi $\sqrt{p}$ sonidan katta bo‘lmagan tub sondir.

Yuqoridagi xossalardan foydalanmay yoki foydalanib berilgan 1 dan katta M sonini tub son yoki murakkab son bo‘lishini aniqlash uchun quyidagi usullarni yozish mumkin:

1-usul. 1 dan boshlab M gacha bo‘lgan barcha sonlarga bo‘lib, bo‘luvchilar soni S ni hisoblaymiz. Agar S=2 bo‘lsa, M soni tub, aks holda, ya’ni S>2 bo‘lsa, M soni murakkab. Bu holda M marta qoldiqning 0 ga tengligi tekshiriladi.

2-usul. 2 dan boshlab M–1 gacha bo‘lgan barcha sonlarga bo‘lib, bo‘luvchilar soni S ni hisoblaymiz. Agar S=0 bo‘lsa, M soni tub, aks holda, ya’ni S>0 bo‘lsa, M soni murakkab. Bu holda M–2 marta qoldiqning 0 ga tengligi tekshiriladi.

3-usul. 2 dan boshlab $\sqrt{M}$ gacha bo‘lgan barcha sonlarga bo‘lib, bo‘luvchilar soni S ni hisoblaymiz. Agar S=0 bo‘lsa M soni tub, aks holda, ya’ni S>0 bo‘lsa, M soni murakkab. Bu holda, $\sqrt{M}$–1 marta qoldiqning 0 ga tengligi tekshiriladi.

4-usul. 2 ga va 3 dan boshlab $\sqrt{M}$ gacha bo‘lgan barcha toq sonlarga bo‘lib, bo‘luvchilar soni S ni hisoblaymiz. Agar S=0 bo‘lsa, M soni tub, aks holda, ya’ni S>0 bo‘lsa, M soni murakkab. Bu holda $\frac{\sqrt{M}+1}{2}$ marta qoldiqning 0 ga tengligi tekshiriladi.

5-usul. $\sqrt{M}$ gacha bo‘lgan barcha tub sonlarga bo‘lib, bo‘luvchilar soni S ni hisoblaymiz. Agar S=0 bo‘lsa, u holda M soni tub, aks holda, ya’ni S>0 bo‘lsa, M soni murakkab. Bu M soni yetarlicha katta bo‘lganda eng samarador usuldir.

Misol uchun 827 sonini tub son bo‘lishlikka tekshiramiz. Ma’lumki, $\sqrt{827} \approx 28{,}8$ dan kichik bo‘lgan tub sonlar (jami: 9 ta) 2, 3, 5, 7, 11, 13, 17, 19, 23 bo‘lgani va 827 soni bu tub sonlarning hech qaysisiga bo‘linmasligi sababli 827 sonining tub son ekanligi kelib chiqadi.

6-xossa. **Evklid teoremasi**. Tub sonlar cheksiz ko‘pdir.

Quyidagi teorema **arifmetikaning asosiy teoremasi** deb ataladi:

Teorema. Har qanday **murakkab son tub sonlar ko‘paytmasiga yoyiladi** va agar ko‘paytuvchilarning yozilish tartibi nazarga olinmasa, bu yoyilma yagonadir.

Bu teorema va natural sonning 0 darajasi 1 ekanligidan ixtiyoriy 1 dan katta natural A soni uchun quyidagi **kanonik yoyilma**ni yozish mumkin:

$$A = p_1^{\alpha_1} \cdot p_2^{\alpha_2} \cdot p_3^{\alpha_3} \cdot \ldots \cdot p_m^{\alpha_m},$$

bu yerda $p_1<p_2<p_3<\ldots<p_m$ tub sonlar, α_1, α_2, α_3, …, α_m mos tub sonlarning yoyilmadagi takrorlanishlari soni (takrorlanishlar soni musbatdir). **Bu ma’noda aniqlangan kanonik yoyilma yagonadir**. Masalan, $11=11^1$, $121=11^2$ va $363=3^1 \cdot 11^2$ kanonik yoyilmalardir.

Natural sonlarning kanonik yoyilmasidan foydalanib uning bo‘luvchilarini va bo‘luvchilar sonini topish mumkin.

Teorema-1. Natural A sonining kanonik yoyilmasi $A = p_1^{\alpha_1} \cdot p_2^{\alpha_2} \cdot \ldots \cdot p_m^{\alpha_m}$ bo'lsa, u holda A sonining har qanday bo'luvchisi $d = p_1^{\beta_1} \cdot p_2^{\beta_2} \cdot \ldots \cdot p_m^{\beta_m}$ ko'rinishida bo'ladi, bunda $0 \le \beta_k \le \alpha_k$, k=1, 2, 3, …, m.

Misol uchun 20 ning bo'luvchilarini aniqlasak: $20=2^2 \cdot 5^1$ bo'lganligidan, uning bo'luvchilari quyidagilardir: $2^0 \cdot 5^0=1$, $2^1 \cdot 5^0=2$, $2^2 \cdot 5^0=4$, $2^0 \cdot 5^1=5$, $2^1 \cdot 5^1=10$, $2^2 \cdot 5^1=20$.

A natural sonining natural bo'luvchilari soni $\tau(A)$ bilan belgilanadi.

Teorema-2. Agar A natural sonining kanonik yoyilmasi $A = p_1^{\alpha_1} \cdot p_2^{\alpha_2} \cdot \ldots \cdot p_m^{\alpha_m}$ bo'lsa, u holda $\tau(A) = (\alpha_1 + 1) \cdot (\alpha_2 + 1) \cdot \ldots \cdot (\alpha_m + 1)$ tenglik o'rinli bo'ladi.

Ba'zi hollarda natural A soni bo'luvchilarining yig'indisi $\delta(A)$ ni topish kerak bo'ladi.

Teorema-3. Agar A natural sonining kanonik yoyilmasi $A = p_1^{\alpha_1} \cdot p_2^{\alpha_2} \cdot \ldots \cdot p_m^{\alpha_m}$ bo'lsa, u holda $\delta(A) = \frac{p_1^{\alpha_1+1}-1}{p_1-1} \cdot \frac{p_2^{\alpha_2+1}-1}{p_2-1} \cdot \ldots \cdot \frac{p_m^{\alpha_m+1}-1}{p_m-1}$ tenglik o'rinli bo'ladi.

Masalan, 20 ning bo'luvchilari sonini va bo'luvchilari yig'indisini topamiz. $20=2^2 \cdot 5^1$ bo'lgani uchun 20 ning bo'luvchilari soni $\tau(20)=(2+1) \cdot (1+1)=6$, bo'luvchilarining yig'indisi esa quyidagicha:

$$\delta(20) = \frac{2^{2+1}-1}{2-1} \cdot \frac{5^{1+1}-1}{5-1} = 7 \cdot 6 = 42$$

Haqiqatan, 20 ning bo'luvchilari soni 6 ta, ularning yig'indisi esa 1+2+4+5+10+20=42 bo'lar edi.

EKUB VA EKUK

Ta'rif-6. Natural A va B sonlarning har biri bo'linadigan son shu sonlarning **umumiy bo'luvchisi** deyiladi.

Ta'rif-7. Natural A va B sonlarini umumiy bo'luvchilarining eng kattasi **eng katta umumiy bo'luvchi** deyiladi.

Uni qisqacha **EKUB** (ba'zan B) deb ham atashadi (ingl. **GCD**, yoyilmasi **greatest common divisor**, rus. НОД, yoyilmasi наибольший общий делитель).

Misol sifatida A=12 va B=18 sonlarini qaraymiz. Avval sonlarning **bo'luvchi**larini yozamiz:

A: 1, 2, 3, 4, 6, 12.

B: 1, 2, 3, 6, 9, 18.

Bu sonlarning **umumiy bo'luvchilari** 1, 2, 3 va 6 bo'ladi. Endi umumiy bo'luvchilar ichida eng kattasi 6 ekanligini bildik, va demak, EKUB(12, 18)=6 ekan.

Ikki sonning **eng katta umumiy bo'luvchi**sini topish usullarini ko'rib chiqamiz.

1-usul. EKUB ni 1 ga tenglaymiz. Sonlarni taqqoslaymiz. Agar sonlar teng bo'lsa, EKUB ulardan biriga teng. Aks holda sonlardan kichigi Kichik va kattasi Kattani aniqlaymiz.

2 dan boshlab bittalab Kichik songacha bo'lgan barcha sonlarga Kichik va Katta sonni bo'lib ko'ramiz. Har safar ikkala son bo'lingan sonni (ya'ni ikkala sonning bo'luvchisini) EKUB sifatida olamiz. Demak, oxirgi, ya'ni ikkala sonning eng katta bo'luvchisi EKUB da saqlanib qoladi. Bu usulda eng ko'pi bilan Kichik marta qoldiqning 0 ga tengligi tekshiriladi.

Misol uchun A=12 va B=18 bo'lsin. EKUB=1 deb olamiz. A va B teng emas, u holda Kichik=12 va Katta=18 bo'ladi. 2 dan Kichik songacha sonlarni birma-bir qarab, Kichik va Katta sonlar bir vaqtda avval 2 ga, keyin 3 ga, so'ngra 6 bo'linishini, EKUB ham, o'z navbatida, avval 2 ga, keyin 3 ga, so'ngra 6 ga teng bo'lishini aniqlaymiz. Demak, EKUB(12, 18)=6 ekan.

2-usul. EKUB ni 1 ga tenglaymiz. Sonlarni taqqoslaymiz. Agar ular teng bo'lsa, EKUB ulardan biriga teng. Aks holda sonlardan kichigi Kichik va kattasi Kattani aniqlaymiz. Kichik sonidan boshlab bittalab 1 gacha bo'lgan barcha sonlarga Kichik va Katta sonni bo'lib ko'ramiz. Birinchi uchragan Kichik va Katta son bo'luvchisini EKUB sifatida olamiz va jarayonni to'xtatamiz. Bu usulda ham eng ko'pi bilan Kichik marta qoldiqning 0 ga tengligi tekshiriladi.

Misol sifatida yana A=12 va B=18 sonlarini qaraymiz. EKUB ni 1 ga tenglaymiz. A va B teng emas, u holda Kichik=12 va Katta=18 bo'ladi. Kichik sondan 1 sonigacha sonlarni birma-bir qarab, Kichik va Katta sonlar bir vaqtda 6 ga bo'linishini, EKUB ham, o'z navbatida, 6 ga teng bo'lishini aniqlaymiz. Xulosa: EKUB(12, 18)=6 ekan.

3-usul (bu usul eramizdan 300 yil avval yashab o'tgan Evklid tomonidan ishlab chiqilgan). Ikkala son tenglashmaguncha quyidagi jarayonni davom ettiramiz: sonlardan kattasini kattasi bilan kichigi ayirmasiga tenglaymiz. Sonlar tenglashgach, ulardan birini EKUB sifatida olamiz. Bu usulda eng ko'pi bilan Katta marta ayirish bajariladi (masalan, A=100 va B=1 bo'lganda).

Yana A=12 va B=18 sonlarini qaraymiz. A va B sonlar teng emas, shu sababli jarayonni quyidagicha ifodalaymiz:

(12; 18) → (12; 18–12) → (12; 6) → (12–6; 6) → (6; 6) → EKUB=6.

Natijada EKUB(12, 18)=6 bo'ldi.

4-usul (bu usulni umumlashgan Evklid algoritmi deb ham atashadi). Sonlardan biri 0 ga tenglashmaguncha quyidagi jarayonni davom ettiramiz: sonlardan kattasini kattasini kichigiga bo'lgandagi qoldiqqa tenglaymiz. Sonlardan biri 0 ga tenglashgach, ikkinchisini EKUB sifatida olamiz. Bu usul EKUB hisoblashda eng samarador usul hisoblanadi. Masalan, A=100 va B=1 bo'lganda 1 marta qoldiq hisoblanadi, xolos.

Bu usulni ham A=12 va B=18 sonlari uchun bajaramiz. A va B sonlar 0 dan farqli, shu sababli jarayonni quyidagicha ifodalaymiz:

(12; 18) → (12; 18%12) → (12; 6) → (12%6; 6) → (0; 6) → EKUB=6.

Olingan natijamiz ham avvalgilari kabi EKUB(12, 18)=6 bo‘ldi.

5-usul. Matematikada EKUB hisoblash uchun quyidagi usul qo‘llanadi, lekin bu usuldan dasturlashda foydalanish yuqoridagi usullarga nisbatan ancha murakkab:

$$A = p_1^{\alpha_1} \cdot p_2^{\alpha_2} \cdot p_3^{\alpha_3} \cdot \ldots \cdot p_m^{\alpha_m} \text{ va } B = p_1^{\beta_1} \cdot p_2^{\beta_2} \cdot p_3^{\beta_3} \cdot \ldots \cdot p_m^{\beta_m}$$

yoyilma ko‘rinishida ifodalaymiz, bu yerda $p_1<p_2<p_3<\ldots<p_m$ sonlar ikkala son yoyilmasidagi tub sonlarni umumlashtirilgani, ya’ni α_1, α_2, α_3, …, α_m va β_1, β_2, β_3, …, β_m sonlarning ba’zilari 0 bo‘lishi ham mumkin. U holda $EKUB(A,B) = p_1^{t_1} \cdot p_2^{t_2} \cdot p_3^{t_3} \cdot \ldots \cdot p_m^{t_m}$ bo‘ladi, bu yerda $t_k=\min\{\alpha_k; \beta_k\}$, k=1, 2, …, m.

Misol sifatida A=60 va B=99 sonlarini qaraymiz. Kanonik yoyilmalar $60 = 2^2 \cdot 3^1 \cdot 5^1$ va $99 = 3^2 \cdot 11^1$ bo‘lgani uchun quyidagicha yozish mumkin:

$$60 = 2^2 \cdot 3^1 \cdot 5^1 \cdot 11^0 \text{ va } 99 = 2^0 \cdot 3^2 \cdot 5^0 \cdot 11^1$$

o‘rinli bo‘ladi. U holda $p_1=2$ uchun $t_1=\min\{2; 0\}=0$, $p_2=3$ uchun $t_2=\min\{1; 2\}=1$, $p_3=5$ uchun $t_3=\min\{1; 0\}=0$, $p_4=11$ uchun $t_4=\min\{0; 1\}=0$ bo‘ladi. Ya’ni $EKUB(60,99) = 2^0 \cdot 3^1 \cdot 5^0 \cdot 11^0=3$.

Ta’rif-8. Agar A va B natural sonlarning umumiy bo‘luvchisi faqat 1 bo‘lsa, u holda A va B sonlari **o‘zaro tub sonlar** deyiladi.

7-xossa. Ketma-ket kelgan 2 ta natural son o‘zaro tub bo‘ladi.

8-xossa. O‘zaro tub A va B sonlari uchun EKUB(A, B)=1.

Ta’rif-9. Natural A va B sonlarning **umumiy karralisi** deb, A soniga ham, B soniga ham bo‘linuvchi natural songa aytiladi.

Natural A va B sonlarning umumiy karralisi doimo mavjud va agar kichikrog‘i mavjud bo‘lmasa, u holda ko‘pi bilan A·B ga teng.

Ta’rif-10. Natural A va B sonlarning umumiy karralisining eng kichigi A va B sonlarining **eng kichik umumiy karralisi** deyiladi.

Uni qisqacha EKUK deb ham atashadi (ingl. **LCM**, yoyilmasi **least common multiple**, rus. НОК, yoyilmasi наименьшее общее кратное). Umuman, A soniga karrali sonlar k·A, k=0, 1, 2, …, ko‘rinishida ifodalanadi. Lekin EKUK ta’rifida natural son haqida gap borayotgani uchun k>0 qaralishi kerak.

Misol sifatida A=12 va B=18 sonlarini qaraymiz. Avval bu sonlarga **karrali** sonlarini yozamiz:

A: 12, 24, 36, 48, 60, 72, 84, 96, 108, …

B: 18, 36, 54, 72, 90, 108, 126, 144, …

Bu sonlarning **umumiy karralilari** 36, 72, 108 va hokazo bo‘ladi. Umumiy karralilar ichida eng kichigi 36 bo‘lgani uchun EKUB(12, 18)=36 ekan.

Quyida ikki sonning **eng katta umumiy bo‘luvchi**sining matematik usulini keltiramiz:

Berilgan A va B sonlarni

$$A = p_1^{\alpha_1} \cdot p_2^{\alpha_2} \cdot p_3^{\alpha_3} \cdot \ldots \cdot p_m^{\alpha_m} \text{ va } B = p_1^{\beta_1} \cdot p_2^{\beta_2} \cdot p_3^{\beta_3} \cdot \ldots \cdot p_m^{\beta_m}$$

yoyilma ko'rinishida ifodalaymiz, bu yerda $p_1<p_2<p_3<\ldots<p_m$ sonlar ikkala son yoyilmasidagi tub sonlarni umumlashtirilgani, ya'ni α_1, α_2, α_3, …, α_m va β_1, β_2, β_3, …, β_m sonlarning ba'zilari 0 bo'lishi ham mumkin. U holda $EKUK(A, B) = p_1^{t_1} \cdot p_2^{t_2} \cdot p_3^{t_3} \cdot \ldots \cdot p_m^{t_m}$ bo'ladi, bu yerda $t_k=\max\{\alpha_k; \beta_k\}$, k=1, 2, …, m.

Misol sifatida A=60 va B=99 sonlarini qaraymiz. Kanonik yoyilmalar $60 = 2^2 \cdot 3^1 \cdot 5^1$ va $99 = 3^2 \cdot 11^1$ bo'lgani uchun quyidagicha yozish mumkin:

$$60 = 2^2 \cdot 3^1 \cdot 5^1 \cdot 11^0 \text{ va } 99 = 2^0 \cdot 3^2 \cdot 5^0 \cdot 11^1$$

o'rinli bo'ladi. U holda $p_1=2$ uchun $t_1=\max\{2; 0\}=2$, $p_2=3$ uchun $t_2=\max\{1; 2\}=2$, $p_3=5$ uchun $t_3=\max\{1; 0\}=1$, $p_4=11$ uchun $t_4=\max\{0; 1\}=1$ bo'ladi. Demak, eng kichik umumiy karrali son:

$$EKUK(60, 99) = 2^2 \cdot 3^2 \cdot 5^1 \cdot 11^1 = 1980.$$

Dasturlashda EKUK masalasi EKUB yordamida quyidagi teoremaga asoslanib hal etiladi.

Teorema-4. Natural A va B sonlari uchun quyidagi tenglik o'rinli:

$$\mathbf{A \cdot B = EKUB(A, B) \cdot EKUK(A, B).}$$

A, B ∈ N va A ≥ B bo'lsin. Ma'lumki, bu holda A va B sonlari uchun $\mathbf{A = B \cdot q + r}$ ($0 \le r < b$) tenglik o'rinli bo'ladigan $q \in N$, $r \in Z$ sonlari mavjud hamda q va r sonlari bir qiymatli aniqlanadi.

Quyidagi teorema va uning natijalari ham dasturlashda muhim o'rin tutadi.

Teorema-5. A va B natural sonlar A ≥ B shartni qanoatlantirsin va $A = B \cdot q + r$ ($0 \le r < B$) bo'lsin. U holda A va B sonlarining barcha umumiy bo'luvchilari B va r sonlarining ham umumiy bo'luvchilari bo'ladi va, aksincha, B va r sonlarining barcha umumiy bo'luvchilari A va B sonlarining ham umumiy bo'luvchilari bo'ladi.

Natija-1. Agar $A = B \cdot q + r$ bo'lsa, EKUB(A, B) = EKUB(B, r) bo'ladi.

Natija-2. Agar EKUB(p, q)=1 va A soni ham p ga, ham q ga bo'linsa, u holda A soni p·q soniga ham bo'linadi.

ODDIY, ARALASH, O'NLI VA DAVRIY KASRLAR

Ta'rif-11. Birning butun miqdordagi ulushlaridan hosil qilingan son **oddiy kasr** deb ataladi.

Oddiy kasrlar $\frac{A}{B}$ kabi belgilanadi, bu yerda A butun va B natural son bo'lib, A surat va B maxraj deb ataladi. Bu kasrni hosil qilish uchun birni B ta teng bo'lakka bo'lib A ta bo'lak olinadi. Masalan, $\frac{3}{5}$ sonini ifodalash uchun birni 5 ta teng bo'lakka bo'lib 3 ta bo'lagi olinadi.

Ta'rif-12. Agar oddiy kasrda surat maxrajdan kichik bo'lsa **to'g'ri kasr**, aks holda **noto'g'ri kasr** deb ataladi.

Demak, agar $\frac{A}{B}$ noto'g'ri kasr bo'lsa, u holda $|A|>B$ bo'lar ekan, ya'ni shunday $m\in Z/\{0\}$ va $q\in Z$ sonlari mavjudki, $A=m\cdot B+q$ va $0\leq q <B$ o'rinli bo'ladi. Oxirgi ifoda asosida noto'g'ri kasrlarni quyidagicha ifodalash mumkin:

$$\frac{A}{B}=\frac{m\cdot B+q}{B}=m\frac{q}{B}$$

Ta'rif-13. $\mathbf{m\frac{q}{B}}$ ko'rinishidagi son **aralash kasr** deyiladi, bu yerda m – sonning butun qismi va $\frac{q}{B}$ – sonning kasr qismi deb ataladi.

Kasrlarning asosiy xossasi sifatida quyidagini e'tirof etishadi:

9-xossa. Surat va maxraj biror 0 dan farqli butun songa ko'paytirilsa kasr qiymati o'zgarmaydi, ya'ni uchun $\frac{A}{B}=\frac{m\cdot A}{m\cdot B}$, bu yerda $A\in Z$ va B, $m\in Z/\{0\}$.

Ta'rif-14. To'g'ri kasr **qisqarmas kasr** deyiladi, agar surat va maxraj umumiy bo'luvchiga ega bo'lmasa.

Asosiy xossaga asosan quyidagi xossalarga ega bo'linadi:

10-xossa. Ixtiyoriy 2 ta kasrning maxrajini bir xil maxrajga keltirish mumkin, ya'ni $\frac{A}{B}$ va $\frac{C}{D}$ kasrlar uchun $\frac{A\cdot D}{B\cdot D}$ va $\frac{B\cdot C}{B\cdot D}$ kasrlarning maxrajlari teng, bu yerda $A, C\in Z$ va $B, D\in N$.

11-xossa. Maxrajlari B va D bo'lgan ixtiyoriy 2 ta kasr uchun eng kichik bir xil maxraj EKUK(B, D) ga teng.

12-xossa. Agar $\frac{A}{B}$ to'g'ri kasrning surati va maxraji EKUB(A, B) ga qisqartirilsa, kasr qis-qarmas kasrga aylanadi, ya'ni $\frac{A:EKUB(A,B)}{B:EKUB(A,B)}$ kasr qisqarmas kasr bo'ladi.

13-xossa. Qisqarmas kasrning surat va maxraji o'zaro tub bo'ladi.

Ta'rif-15. Maxraji 10 sonining biror darajasiga teng bo'lgan kasr **o'nli kasr** deyiladi.

O'nli kasrlar maxraj va kasr chizig'isiz, surat vergul yordamida yoziladi. Masalan:

$\frac{7}{10}=0,7; \frac{7}{100}=0,07; \frac{21}{100}=0,21; \frac{2107}{100}=21\frac{7}{100}=21,07; \frac{1102}{10}=110\frac{2}{10}=110,2.$

Bu o'nli kasrlarda verguldan keyin chekli sonda raqam qatnashmoqda.

Ta'rif-16. O'nli kasrda verguldan keyin chekli sondagi raqam qatnashsa, bunday o'nli kasr **chekli o'nli kasr** (ba'zan qisqacha o'nli kasr) deb ataladi.

Shunday oddiy kasrlar borki, ularni o'nli kasr ko'rinishida tasvirlansa, verguldan keyin cheksiz sondagi raqam qatnashadi. Masalan, $\frac{1}{3}=0,33\ldots$ yoki $\frac{11}{3}=3,66\ldots$ yoki $\frac{8}{7}=1,142857142857\ldots$.

Ta'rif-17. O'nli kasrda verguldan keyin 0 dan farqli cheksiz raqam qatnashsa, bunday o'nli kasr **cheksiz o'nli kasr** deb ataladi.

14-xossa. Agar $\frac{A}{B}$ qisqarmas kasrning maxrajini $2^m \cdot 5^k$ (m, k$\in$N $\cup\{0\}$) ko'rinishda tasvirlash mumkin bo'lsa, u holda bu kasr **chekli o'nli kasr**ga aylanadi.

15-xossa. Agar qisqarmas kasr maxrajining biror bo'luvchisi 2 va 5 sonlaridan farqli bo'lsa, u holda bu kasr **cheksiz o'nli kasr**ga aylanadi.

Ta'rif-18. Agar cheksiz o'nli kasrning verguldan keyin biror raqamidan boshlab qandaydir raqamlar guruhi ma'lum bir tartibda cheksiz marta takrorlansa, bunday o'nli kasr **davriy o'nli kasr** deb, takrorlanuvchi raqamlar guruhi shu kasrning **davri**, verguldan keyingi davrdan oldingi raqamlar guruhi **davroldi** deb ataladi.

Odatda, davriy o'nli kasrning davri qavs ichiga olingan holda bir marta yoziladi: 3,666... = 3,(6); 0,131131131131... = 0,(131); 6,12777...7... = 6,12(7). Oxirgi 6,12(7) misolida davr 7 ga, davr uzunligi 1 ga, davroldi 12 ga, davroldi uzunligi 2 ga teng.

QISQARMAS KASRNING DAVRI VA DAVROLDI UZUNLIGI

16-xossa. Agar qisqarmas $\frac{A}{B}$ to'g'ri kasrning maxraji 2 va 5 sonlari darajalarining ko'paytmasidan iborat bo'lsa, ya'ni maxraj $B=2^m \cdot 5^k$ ko'rinishda ($m\geq 0$, $k\geq 0$) bo'lsa, u holda davr uzunligi **d=0**;

17-xossa. Agar qisqarmas $\frac{A}{B}$ to'g'ri kasr maxraji 10 soni bilan o'zaro tub bo'lsa, ya'ni **EKUB(B, 10)=1**, u holda mos o'nli kasr davri uzunligi **d** soni **(B–1)** sonidan katta bo'lmaydi;

18-xossa. Qisqarmas $\frac{A}{B}$ to'g'ri kasr maxraji 10 soni bilan o'zaro tub bo'lsa, ya'ni **EKUB(B, 10)=1**, u holda mos o'nli kasr davri uzunligi (10^d-1) soni **B** ga karrali bo'ladigan eng kichik natural **d** soniga teng;

19-xossa. Qisqarmas $\frac{A}{B}$ kasrga mos o'nli kasr davri uzunligi **d** soni **A** soniga bog'liq emas.

20-xossa. Qisqarmas $\frac{A}{B}$ to'g'ri kasrning maxraji $B=2^m \cdot 5^k \cdot C$ ko'rinishda (EKUB(C, 10)=1, $m\geq 0$, $k\geq 0$) bo'lsa, u holda mos o'nli kasr davroldi uzunligi **max{m, k}** ga teng.

21-xossa. Qisqarmas $\frac{A}{B}$ to'g'ri kasrning maxraji $B=2^m \cdot 5^k \cdot C$ ko'rinishda (EKUB(C, 10)=1, $m\geq 0$, $k\geq 0$) bo'lsa, u holda mos o'nli kasr davri uzunligi $\frac{1}{C}$ kasrga mos o'nli kasr davri uzunligiga teng.

KETMA-KETLIKLAR VA REKKURENTLIK

Ketma-ketliklar hosil qilinishi haqida ozgina to‘xtab o‘tamiz. M ta elementli ixtiyoriy X={a, b, c, d, …, xt} to‘plam berilgan bo‘lsin. Bu to‘plam elementlarini butun sonlar to‘plami Z ning 1 ta qadam bilan o‘suvchi M ta elementli Q qism to‘plami bilan o‘zaro mos qo‘yish mumkin (quyida k ixtiyoriy butun son):

Q to‘plam elementlari	k+1	k+2	k+3	k+4	…	k+M
X to‘plam elementlari	a	b	c	d	…	xt

Bu moslikni H orqali belgilasak, u holda quyidagi ketma-ketlik hosil bo‘ladi:

$$H_{k+1}=a,\ H_{k+2}=b,\ H_{k+3}=c,\ H_{k+4}=d,\ \ldots,\ H_{k+M}=xt.$$

X to‘plam elementlari ixtiyoriy yig‘ilgan yoki biror qonuniyat asosida tashkil etilgan bo‘lishi mumkin. Ixtiyoriy yig‘ilgan to‘plamga juda ko‘p misol keltirish mumkin. Biror qonuniyat asosida tashkil etilgan to‘plamlar ham yetarlicha ko‘p. Masalan, faqat bitta raqam chekli marta qatnashgan to‘plam, alifbo harflari to‘plami, matematika kursidagi arifmetik yoki geometrik progressiyalarning chekli sondagi hadlari to‘plami va boshqalar.

Dasturlash masalalarini hal etishda bir nechta yoki o‘nta yoki yigirmata o‘zgaruvchi bilan hech qiyinchiliksiz ishlash mumkin. Ba’zi masalalarda amalni bajarish uchun yuzlab yoki minglab o‘zgaruvchi bilan ishlash kerakdek tuyulgani bilan masalani besh yoki oltita o‘zgaruvchi yordamida hal etish mumkin bo‘ladi. Lekin shunday masalalar ham uchraydiki, yuzlab yoki minglab o‘zgaruvchi bilan ishlashga majbur bo‘lamiz. Misol sifatida berilgan 1000 ta nuqtadan bir-biriga eng yaqin joylashgan ikkita nuqtani topish masalasini olish mumkin.

Shuning uchun dasturlashga oid juda ko‘p masalalarda kiritilayotgan ma’lumotlar yoki hosil qilinadigan qiymatlar nom va tur bilan birlashtirilib, har bir alohida olingan ma’lumot yoki qiymat o‘z tartib raqamiga ega bo‘ladi. Birinchi misol sifatida yuqori registrdagi lotin alifbosi harflari olinsa, u holda bu ma’lumotlarni **A** nomda va **belgili** turda birlashtirish mumkin. U holda 26 ta lotin harfiga quyidagicha tartib raqamlarini belgilash mumkin:

A[1]:= ‘A‘, A[2]:= ‘B‘, …, A[26]:= ‘Z‘ yoki A[0]:= ‘A‘, A[1]:= ‘B‘, …, A[25]:= ‘Z‘ yoki A[-10]:= ‘A‘, A[-9]:= ‘B‘, …, A[15]:= ‘Z‘ yoki A[100]:= ‘A‘, A[101]:= ‘B‘, …, A[125]:= ‘Z‘.

Ikkinchi misol sifatida quyidagicha hosil qilinuvchi **Fibonachchi** sonlarini olish mumkin:

1) birinchi Fibonachchi soni 0 ga teng;

2) ikkinchi Fibonachchi soni 1 ga teng;

3) keyingi har bir Fibonachchi soni o‘zidan avvalgi ikkita Fibonachchi sonining yig‘indisiga teng.

Bu qonuniyat asosida quyidagi Fibonachchi sonlar ketma-ketligini hosil qilish mumkin:

0; **1**; (0+1=) **1**; (1+1=) **2**; (1+2=) **3**; (2+3=) **5**; (3+5=) **8**; (5+8=) **13**; (8+13=) **21**; …

Fibonachchi sonlarini **F** nomda va **butun** turda birlashtirib, quyidagi qonuniyat asosıda yozish mumkin: F[1]:=0; F[2]:=1; F[k]:=F[k–2]+F[k–1], k=3,4,5,….

Fibonachchi sonlari haqida so‘z yuritganda yana bir usulni aytib o‘tish joiz. Bu matrıtsalar usuli bo‘lib, u quyidagicha tavsiflanadi. Yordamchi matritsa kiritamiz:

$$A = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$$

U holda matritsalarni ko‘paytirish formulasiga ko‘ra:

$$\begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} F[2] \\ F[1] \end{pmatrix} = \begin{pmatrix} 1 \cdot F[2] + 1 \cdot F[1] \\ 1 \cdot F[2] + 0 \cdot F[1] \end{pmatrix} = \begin{pmatrix} F[2] + F[1] \\ F[2] \end{pmatrix} = \begin{pmatrix} F[3] \\ F[2] \end{pmatrix},$$

$$\begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} F[3] \\ F[2] \end{pmatrix} = \begin{pmatrix} 1 \cdot F[3] + 1 \cdot F[2] \\ 1 \cdot F[3] + 0 \cdot F[2] \end{pmatrix} = \begin{pmatrix} F[3] + F[2] \\ F[3] \end{pmatrix} = \begin{pmatrix} F[4] \\ F[3] \end{pmatrix}, \ldots,$$

$$\begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} F[N-1] \\ F[N-2] \end{pmatrix} = \begin{pmatrix} 1 \cdot F[N-1] + 1 \cdot F[N-2] \\ 1 \cdot F[N-1] + 0 \cdot F[N-2] \end{pmatrix} = \begin{pmatrix} F[N-1] + F[N-2] \\ F[N-1] \end{pmatrix} = \begin{pmatrix} F[N] \\ F[N-1] \end{pmatrix}$$

bo‘ladi. Ikkinchi tomondan, yuqoridagilarga asosan quyidagilarga ega bo‘lamiz:

$$\begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}^2 \cdot \begin{pmatrix} F[2] \\ F[1] \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} F[2] \\ F[1] \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} F[3] \\ F[2] \end{pmatrix} = \begin{pmatrix} F[4] \\ F[3] \end{pmatrix},$$

$$\begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}^3 \cdot \begin{pmatrix} F[2] \\ F[1] \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}^2 \cdot \begin{pmatrix} F[2] \\ F[1] \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} F[4] \\ F[3] \end{pmatrix} = \begin{pmatrix} F[5] \\ F[4] \end{pmatrix}, \ldots,$$

$$\begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}^{N-2} \cdot \begin{pmatrix} F[2] \\ F[1] \end{pmatrix} = \begin{pmatrix} F[N] \\ F[N-1] \end{pmatrix}.$$

Demak, bu usulda A matritsani (N–2) darajaga ko‘tarib, kerakli natija hosil qilinar ekan.

Shunday ketma-ketliklar ham uchraydiki, ularning hadlari o‘zidan oldingi bir nechta had orqali aniqlanadi. Bunday ketma-ketliklar rekkurent formula yordamida aniqlanuvchi ketma-ketlik deb ataladi.

Rekkurent formula hosil qilishda boshlang‘ich qiymatlar aniq bo‘lishi muhimdir. Misol sifatida Fibonachchi sonlari ketma-ketligini (boshlang‘ich qiymatlar: F[1]:=0, F[2]:=1, rekkurentlik: F[k]:=F[k–2]+F[k–1]), arifmetik va geometrik progressiya hadlarini (boshlang‘ich qiymatlar: a_1 va d, b_1 va q, rekkurentlik: $a_N=a_{N-1}+d$ va $b_N=b_{N-1}\cdot q$) olish mumkin.

Rekkurent formula hosil qilish jarayonini bir nechta misolda ko‘rib chiqamiz.

a) Berilgan a_1, a_2, a_3, … sonli ketma-ketlikni N ta hadi yig‘indisini aniqlash masalasini qaraymiz. S[k] orqali ketma-ketlikni k ta hadi yig‘indisi bo‘lsin. Boshlang‘ich qiymat: S[0]=0. U holda quyidagi rekkurent formulalarga ega bo‘lamiz: S[1]=S[0]+a_1; S[2]=S[1]+a_2; …; S[N]=S[N–1]+a_N.

Bu rekkurentlik yuqorida ko‘rilgan boshlang‘ich qiymati s=0; va takroriy bajariladigan ifoda s=s+a_k; bo‘lgan holga mos keladi.

b) Berilgan N ta hadli a_1, a_2, a_3, …, a_N sonli ketma-ketlikni o‘rta arifmetigini topish kerak bo‘lsin. Buning uchun sonlarni o‘rta arifmetigini hisoblash formulasini shu ketma-ketlikka nisbatan yozib olamiz va ko‘rinishini o‘zgartiramiz:

$$\text{O'rta arifmetiqi} = \frac{a_1+a_2+\cdots+a_N}{N} = a_1 \cdot \frac{1}{N} + a_2 \cdot \frac{1}{N} + \cdots + a_N \cdot \frac{1}{N}.$$

Boshlang‘ich qiymatni S[0]=0 va rekkurentlikni quyidagicha ifodalaymiz:

S[1]=S[0]+$a_1 \cdot \frac{1}{N}$; S[2]=S[1]+$a_2 \cdot \frac{1}{N}$; …; S[N]=S[N–1]+$a_N \cdot \frac{1}{N}$.

Bu rekkurentlik ham boshlang‘ich qiymati s=0; va takroriy bajariladigan ifoda s=s+$a_k \cdot \frac{1}{k}$; bo‘lgan holga mos keladi.

2-§. O‘TISH VA TARMOQLASH OPERATORLARI YORDAMIDA TAKRORLASH

Dasturda takrorlashni tashkil etish murakkab emas. Masalan, takrorlash operatorlarisiz ham bu ishni amalga oshirish mumkin. Buning uchun o‘tish va tarmoqlash operatorlaridan foydalanish mumkin (4-bob). Takrorlanuvchi algoritmli dasturlarni shu operatorlar yordamida tashkil etishga harakat qilamiz. Quyidagicha **yig‘indi hisoblash masalasi**ni qaraylik:

Masala-1. 1+2+3+…+11021999 yig‘indini hisoblang.

Yechim. Berilgan chegaraviy qiymatlarni qabul qila oladigan 1 dan 11021999 gacha bo‘lgan butun sonlarni “sanash” uchun **int** turidagi t o‘zgaruvchini kiritamiz.

Tahlil-1: t o‘zgaruvchisining boshlang‘ich qiymati 1 (quyi chegara), oxirgi qiymati 11021999 (yuqori chegara), ya’ni t o‘zgaruvchisining boshlang‘ich va oxirgi qiymatlari masala shartida aniq berilgan.

Tahlil-2: t o‘zgaruvchisining qiymati bittalab oshib boradi, demak qadam 1 ga teng, ya’ni bu o‘zgaruvchi uchun t=t+1 (yoki t++) ifodadan foydalanish mumkin.

Yig‘indini hisoblab borish uchun (yig‘indi katta son bo‘lishi mumkin ekanligini e’tiborga olib) **long long int** turidagi s o‘zgaruvchini kiritamiz.

Tahlil-3: s o‘zgaruvchining boshlang‘ich qiymati yig‘indi qiymatiga ta’sir etmasligi kerak, demak, s o‘zgaruvchisiga 0 qiymatni o‘zlashtiramiz.

Tahlil-4: t o‘zgaruvchisining har bir qiymati s o‘zgaruvchiga qo‘shib boriladi.

Yuqoridagilarga asosan quyidagicha yozish mumkin:

s=s+t (=0+1=1); t=t+1 (=1+1=2); agar t ≤ 11021999 bo‘lsa, yig‘indi hisoblashga o‘tilsin;

s=s+t (=1+2); t=t+1 (=1+1=3); agar t ≤ 11021999 bo‘lsa, yig‘indi hisoblashga o‘tilsin;

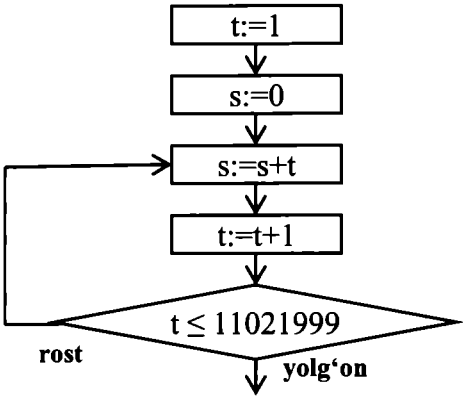
…

s=s+t (=1+2+3+...+11021998); t=t+1 (=11021998+1=11021999); agar t ≤ 11021999 bo'lsa, yig'indi hisoblashga o'tilsin;

s=s+t (=1+2+3+...+11021998+11021999); t=t+1 (=11021999+1=11022000); agar t ≤ 11021999 bo'lsa, yig'indi hisoblashga o'tilsin.

Tahlil-5: oxirgi t ≤ 11021999 tengsizlik ifodasi yolg'on qiymat qabul qiladi, chunki t o'zgaruvchisining 11022000 qiymatida 11022000 ≤ 11021999 tengsizlik o'rinli emas. **Demak, keyingi qadamda yig'indi hisoblashga o'tilmaydi.**

Takrorlanish algoritmli qismni to'liq tasavvur etish uchun blok-sxema shaklida ifodalaymiz:



Takrorlanayotgan "s=s+t; t=t+1; agar t ≤ 11021999 bo'lsa, yig'indi hisoblashga o'tilsin;" buyruqlarini avvalgi boblarda bayon etilgan o'zlashtirish, tarmoqlash va o'tish operatorlari yordamida quyidagicha yozish mumkin:

nishon: s=s+t; t=t+1;

if (t<11021999) goto nishon;

Endi bizga kerakli dasturni to'liq yoza olamiz:

Dastur 1	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  int t=1;`
`  long long int s=0;`
`  nishon: s=s+t; t=t+1;`
`  //yoki nishon:s+=t; t++;`
`  if(t<=11021999) goto nishon;`
`  cout <<"S= "<< s;`
`  return 0;`
`}`
	S= 60742236489000

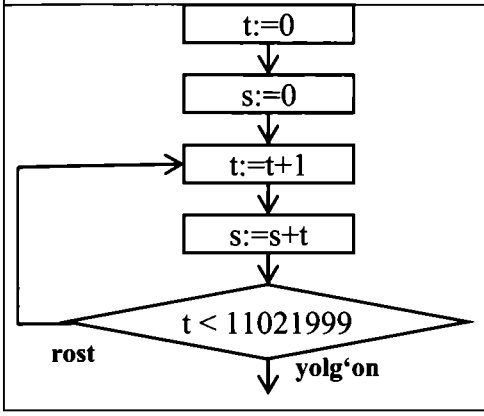
Albatta, bu masalani **chiziqli dastur** kabi ishlash imkoniyati ham bor (3-bob boshlanishidagi formulaga qarang).

Masalani yechishda s=s+t; t=t+1; ifodalarning o'rnini almashtirish ham mumkin, ya'ni t=t+1; s=s+t; kabi yozish mumkin. Bu holda:

- birinchidan, t o'zgaruvchisi avval hisoblanayotgani va t=t+1; ifodaning natijasi yig'indida birinchi qo'shiluvchi bo'lgan 1 ga teng bo'lishi uchun t ning boshlang'ich qiymati sifatida 0 olishga majburmiz, shunda t=t+1=0+1=1 bo'ladi;
- ikkinchidan, o'tish sharti sifatida t < 11021999 tengsizlik qaraladi, chunki yig'in-dining oxirgi qo'shiluvchisi 11021999 bo'lishi kerak edi.

Bu holda takrorlanish algoritmli qismni blok-sxemasi va masala to'liq dasturi quyidagicha:

Blok-sxema	**Dastur 1***

	`#include <iostream>`
	`using namespace std;`
	`int main(){`
	`  int t=0;`
	`  long long int s=0;`
	`  nishon: t=t+1;s=s+t;`
	`  if(t<11021999) goto nishon;`
	`  cout <<"S= "<< s;`
	`  return 0;`
	`}`

Masala-1 shartini quyidagi ko'rinishlarda umumlashtirish mumkin:

Masala-1A. 1+14+27+...+ 11021999 yig'indini hisoblang.

Tahlil-6: Masalada t o'zgaruvchisining qadami 1 emas 14–1=27–14=13 bo'lishini aniqlaymiz. Yuqori chegarani ham shu qadamga mosligini tekshiramiz. Buning uchun 1, 14 va 27 sonlarining 13 ga bo'lgandagi qoldiq 1 ekanligidan foydalanamiz, ya'ni qoldiq (1/13) = qoldiq (14/13) = = qoldiq (27/13) = 1 bo'lgani uchun yuqori chegarani bo'lgandagi qoldiqni tekshiramiz. Haqiqatan, qoldiq (11021999/13) = 1. Demak, bu masala yechimi **Dastur 1** yoki **Dastur 1*** da qadam sifatida 1 emas, 13 olinishi bilan hal bo'ladi, ya'ni: t=t+13.

Masala-1B. Berilgan N uchun 1+2+3+...+N yig'indini hisoblang ($1 \le N \le 1000000000$).

Tahlil-7: Masalada yuqori chegara o'zgaruvchan qilib berilgan. Demak, dasturda N o'zgaruvchisi tavsiflanishi va qiymati kiritilishi zarur. Endi masala yechimi **Dastur 1** yoki **Dastur 1*** da t o'zgaruvchisining yuqori chegarasi sifatida 11021999 o'rniga **N** olinishi bilan hal etiladi:

Dastur 1B	Dastur 1*B

#include <iostream>
using namespace std;
int main(){
int t=1, n;
long long int s=0;
cout<<"N= "; cin >> n;
nishon: s+=t; t++;
if(t<=n) goto nishon;
cout <<"S= "<< s;
return 0;
}
	#include <iostream>
	using namespace std;
	int main(){
	int t=0, n;
	long long int s=0;
	cout<<"N= "; cin >> n;
	nishon: t=t+1; s=s+t;
	if(t<n) goto nishon;
	cout <<"S= "<< s;
	return 0;
	}

Masala-1C. Berilgan M va N ($-10^9 \le M < N \le 10^9$) uchun M+(M+1)+(M+2)+…+N yig‘indini hisoblang.

Tahlil-8: Masalada quyi va yuqori chegara o‘zgaruvchan qilib berilgan. Demak, dasturda M va N o‘zgaruvchilari tavsiflanishi va qiymatlari kiritilishi zarur. Demak, masala yechimi **Dastur 1** yoki **Dastur 1*** o‘zgartrilib, quyidagicha hal etiladi:

Dastur 1C	Dastur 1*C

#include <iostream>
using namespace std;
int main(){
int t, m, n;
long long int s=0;
cout<<"M= "; cin >> m;
cout<<"N= "; cin >> n;
t=m;
nishon: s+=t; t++;
if(t<=n) goto nishon;
cout <<"S= "<< s;
return 0;
}
	#include <iostream>
	using namespace std;
	int main(){
	int t, m, n;
	long long int s=0;
	cout<<"M= "; cin >> m;
	cout<<"N= "; cin >> n;
	t=m-1;
	nishon: t=t+1; s=s+t;
	if(t<n) goto nishon;
	cout <<"S= "<< s;
	return 0;
	}

Masala-1D. Berilgan M, K va N ($-10^9 \le M, K \le 10^9$, $1 \le N \le 10^9$) uchun M, M+K, M+2·K, … ketma-ketlikning birinchi N ta hadi yig‘indisini hisoblang.

Tahlil-9: Masala shartiga ko‘ra quyidagi mazmundagi yig‘indini hisoblash talab qilinmoqda:

$$s=M+(M+K)+(M+2\cdot K)+\ldots+(M+(N-1)\cdot K)$$

Yuqoridagi umumlashmalarini e’tiborga olib masalaning yechimini quyidagicha hal etish mumkin:

- **Dastur 1** da t o‘zgaruvchisining boshlang‘ich qiymati sifatida M, oxirgi qiymati sifatida (N–1)·K, qadam sifatida K olinadi, ya’ni t=t+K;
- **Dastur 1*** da t o‘zgaruvchisining boshlang‘ich qiymati sifatida M–1, oxirgi qiymati sifatida (N–1)·K, qadam sifatida K olinadi, ya’ni t=t+K.

Masala-1 shartini yuqori chegara mavhum bo‘lgan (berilmagan) holga o‘zgartiramiz:

Masala-2. Berilgan N ($1 \le N \le 10^9$) uchun 1+2+3+... yig‘indining birinchi marta N sonidan katta bo‘lgandagi qiymatini aniqlang.

Yechim. Masala-1 dagi belgilashlarni saqlab qolamiz.

Tahlil-10. Kiritilgan t o‘zgaruvchisining boshlang‘ich qiymati 1 bo‘lib, oxirgi qiymati berilmagan, demak, keyingi qadamga o‘tish shartini o‘zgartirish kerak.

Tahlil-11. Masalada “yig‘indining birinchi marta N sonidan katta bo‘lgandagi qiymati” so‘ralmoqda, ya’ni birinchi marta s>N shart bajarilgandagi s o‘zgaruvchisining qiymati kerak. Bu “birinchi marta s>N shart bajarilganda” sharti dastur hisoblash jarayonini to‘xtatishi kerakligini bildirsa, bu shartga teskari bo‘lgan “s≤N shart bajarilganda” sharti dastur hisoblash jarayonini davom ettirishi kerakligini bildiradi. Shu sababli o‘tish shartini quyidagicha o‘zgartiramiz, ya’ni:

s=s+t (=0+1=1); t=t+1 (=1+1=2); agar $s \le N$ bo‘lsa, yig‘indi hisoblashga o‘tilsin;

s=s+t (=1+2); t=t+1 (=1+1=3); agar $s \le N$ bo‘lsa, yig‘indi hisoblashga o‘tilsin;

s=s+t (=1+2+3); t=t+1 (=1+3=4); agar $s \le N$ bo‘lsa, yig‘indi hisoblashga o‘tilsin;

s=s+t (=1+2+3+4); t=t+1 (=1+4=5); agar $s \le N$ bo‘lsa, yig‘indi hisoblashga o‘tilsin;

...

Masalaga mos dastur va namunaviy natija quyidagicha:

Dastur 2	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  int t=1, n;`
`  long long int s=0;`
`  cout<<"N= "; cin >> n;`
`  nishon: s+=t; t++;`
`  if(s<=n) goto nishon;`
`  cout <<"S= "<< s;`
`  return 0;`
`}`
	`N= 1902`
	`S= 1953`

Bu masalani ham **chiziqli dastur** kabi ishlash imkoniyati bor (mustaqil bajarib ko‘ring). Umuman, 3-bobdagi Chiziqli 84 – 104 oraliqdagi barcha masalalarni takrorlanuvchi algoritmli dastur kabi yechish ham mumkin.

Masala-2 shartini quyidagi ko‘rinishlarda umumlashtirish mumkin:

Masala-2A. Berilgan N ($1 \le N \le 10^9$) uchun 1+2+3+... yig‘indining qiymati birinchi marta N sonidan katta bo‘lgandagi qo‘shiluvchi qiymatini aniqlang.

Tahlil-12. Masala-2 da “yig‘indining birinchi marta N sonidan katta bo‘lgandagi qiymati” so‘ralgani uchun yig‘indi uchun kiritilgan s o‘zgaruvchisining qiymati ekranga chiqarilgan edi. Bu masalada “yig‘indining birinchi marta N sonidan katta bo‘lgandagi qo‘shiluvchisi qiymati” so‘ralgani uchun **Dastur 2** dagi ekranga chiqariladigan qiymat t o‘zgaruvchining qiymatiga almashtirilishi kerak ekan. Lekin bunda oxirgi marta yig‘indi qiymati hisoblangandan keyin s>N shart bajarilsa ham, t ning qiymati yana bittaga oshirilganini e’tiborga olib ekranga chiqarishdan avval bittaga kamaytirilishi shart, ya’ni dasturga t=t–1; (yoki t--;) ifoda qo‘shiladi. Demak:

Dastur 2A	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  int t=1, n;`
`  long long int s=0;`
`  cout<<"N= "; cin >> n;`
`  nishon: s+=t; t++;`
`  if(s<=n) goto nishon;`
`  t=t-1;`
`  cout <<"Qo'shiluvchi= "<< t;`
`  return 0;`
`}`
	N= 1902
	Qo'shiluvchi= 62

Masalani yechishda int t=1; ga mos s=s+t; t=t+1; ifodalarning o‘rnini almashtirilsa, ya’ni int t=0; ga mos t=t+1; s=s+t; kabi yozilsa (Dastur 1* ga qarang), u holda t=t–1; ifodani yozishga ehtiyoj qolmaydi.

Yanada umumiyroq hollarni qaraymiz:

Masala-2B. Berilgan M, K va N ($-10^9 \le M, K \le 10^9$, $1 \le N \le 10^9$) uchun M, M+K, M+2·K, ... ketma-ketlik yig‘indisining birinchi marta N sonidan katta bo‘lgandagi qiymatini hisoblang.

Masala-2C. Berilgan M, K va N ($-10^9 \le M, K \le 10^9$, $1 \le N \le 10^9$) uchun M, M+K, M+2·K, ... ketma-ketlik yig‘indisining birinchi marta N sonidan katta bo‘lgandagi hadi qiymatini aniqlang.

Bu ikkala masalaga mos dasturlar quyidagicha bo‘lishini tushunish qiyin emas:

Dastur 2B	Dastur 2C

#include <iostream>
using namespace std;
int main(){
int t, m, k, n;
long long int s=0;
cout<<"M= "; cin >> m;
cout<<"K= "; cin >> k;
cout<<"N= "; cin >> n;
t=m;
nishon: s+=t; t+=k;
if(s<=n) goto nishon;
cout <<"S= "<< s;
return 0;
}
	#include <iostream>
	using namespace std;
	int main(){
	int t, m, k, n;
	long long int s=0;
	cout<<"M= "; cin >> m;
	cout<<"K= "; cin >> k;
	cout<<"N= "; cin >> n;
	t=m;
	nishon: s+=t; t+=k;
	if(s<=n) goto nishon;
	t=t-k;
	cout <<"Qo'shilgan had= "<< t;
	return 0;
	}

C++ tilida takrorlanuvchi algoritmli dasturlar tuzish qulay bo'lishi uchun maxsus operatorlar kiritilgan. Adabiyotlarda bu operatorlarni **takrorlash operatorlari** yoki **sikl operatorlari** kabi nomlash yoki **xususiyati bilan bog'liq** boshqa bir nomlash qo'llanadi. Masalan:

- parametrli takrorlash operatori yoki **for** operatori
- shart bo'yicha takrorlash operatori yoki **while** operatori
- shart bo'yicha takrorlash operatori yoki **do while** operatori

3-§. PARAMETRLI TAKRORLASH OPERATORI

Parametrli takrorlash operatori parametrga qo'yilgan shartlar asosida tanasida joylashgan ko'rsatmalar ({ } qavslari ichiga olingan ko'rsatmalar ketma-ketligi) blokini takroriy ravishda ishlatishga xizmat qiladi. Parametrli takrorlash operatorining umumiy ko'rinishi quyidagicha:

```
for(<boshlang'ich qiymat berish>; <ifoda>; <ko'rsatma>){
    ko'rsatmalar ketma-ketligi
}
```

bu yerda (<...> – belgilari for operatori tarkibida bu qism yozilishi shart emasligini bildiradi):

- `boshlang'ich qiymat berish` – yozilishi shart bo'lmagan qism bo'lib, asosan **for** operatori parametri deb ataladigan o'zgaruvchini e'lon qilish yoki **for** operatori parametriga boshlang'ich qiymat berish kabilar uchun ishlatiladi. Bu qismda yozilgan ko'rsatma faqat bir marotaba bajariladi. Bu qismda e'lon qilingan o'zgaruvchi faqat shu takrorlash operatori tanasidagina mavjud bo'ladi. Albatta, bu qismni boshqa maqsadlarda ham ishlatish mumkin;

- `ifoda` – qiymati true yoki false ekvivalentlariga teng bo'ladigan ifoda bo'lib, takrorlashni bajarilish shartini ifodalaydi. Aksariyat hollarda **ifoda** takrorlash operatori parametriga bog'liq bo'ladi;
- `ko'rsatma` – maxsus ko'rsatma bo'lib, **for** operatorining har bir takrorlanishida bajariladi. Bu qismdan, asosan takrorlash operatori parametri qiymatini o'zgartirish maqsadida foydalaniladi;
- {ko'rsatmalar ketma-ketligi} – takroriy ravishda bajariladigan qism, blok.

Parametrli takrorlash operatorini qo'llashga doir eng sodda masalani qaraymiz:

Namuna 1. Ekranga 21 marta 7 raqamini chiqaring.

Yechim. Ekranga 21 marta 7 raqamini chiqarish uchun ekranga chiqarilayotgan 7 raqamlari sonini sanash kifoya. Bu vazifa **for** operatori yordamida juda oson bajariladi. Parametrli takrorlash operatori parametri sifatida 1 dan 21 gacha qiymat qabul qila oladigan ixtiyoriy butun turdagi o'zgaruvchini olish mumkin, masalan, **int a**. Parametr a ning boshlang'ich qiymati sifatida 1 ni olamiz. Parametrni **bitta**lab orttirib borib (ya'ni, a=a+1 asosida) yuqori chegarasi sifatida 21 ni ko'rsatamiz, ya'ni $a \leq 21$. U holda dasturni quyidagicha yozish mumkin:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  for(int a=1; a<=21; a++)`
`  cout <<"7 ";`
`  return 0;`
`}`
	7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7

Takrorlash tanasida faqat bittagina operator, ya'ni chiqarish operatori qatnashgani uchun blokka olish qavslarini yozish shart emas. Bu dasturda a parametri faqat sanagich sifatida qatnashdi. Keyingi masalada esa parametr takrorlash tanasida ham qatnashadi.

Namuna 2. Ekranga 1 dan 10 gacha bo'lgan sonlarni o'sish tartibida chiqaring.

Yechim. Bu masala avvalgi masaladan faqat son qiymatlar va chiqarish operatori tarkibi bilan farqlanadi. Bu masalada ham takrorlash tanasida faqat bittagina chiqarish operatori qatnashgani uchun blokka olish qavslarini yozish shart emas:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  for(int a = 1; a <= 10; a++)`
`  cout <<a <<" ";`
`  return 0;`
`}`
	1 2 3 4 5 6 7 8 9 10

Masalani **for** operatori tarkibida <boshlang'ich qiymat berish> va <ko'rsatma> qismlarini yozmasdan quyidagicha yechish mumkin:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  int a=1;`
`  for(; a <= 10;)`
`  {cout <<a <<" "; a++;}`
`  return 0;`
`}`
	1 2 3 4 5 6 7 8 9 10

Eng sodda cheksiz davom etuvchi siklni esa quyidagini yozib hosil qilish mumkin: `for(;;);`

Namuna 3. Ekranga 1 dan 10 gacha bo'lgan sonlarni kamayish tartibida chiqaring.

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  for(int a = 10; a >= 1; a--)`
`  cout <<a <<" ";`
`  return 0;`
`}`
	10 9 8 7 6 5 4 3 2 1

Namuna 4. 1 dan 1001 gacha bo'lgan toq sonlar yig'indisini hisoblash dasturini tuzing.

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  int s=0;`
`  for(int a=1; a<=1001; a=a+2) s=s+a;`
`  cout <<s;`
`  return 0;`
`}`
	251001

Namuna 5. Ushbu 0,2+0,7+1,2+1,7+…+100,2 yig‘indini hisoblash dasturini tuzing.

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  double s=0;`
`  for(double a=0.2; a<=100.2; a=a+0.5)`
`  s=s+a;`
`  cout <<s;`
`  return 0;`
`}`
	10090.2

Namuna 4 va Namuna 5 dan ko‘rinadiki, **ko‘rsatma**da takrorlash qadami istalgancha berilishi mumkin ekan.

Endi oldingi paragrafdagi **Masala-1** yechimini parametrli takrorlash operatori yordamıda quyidagicha yozamiz:

Dastur 1-for	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  long long int s=0;`
`  for(int t=1; t<=11021999; t++) s=s+t;`
`  cout <<"S= "<< s;`
`  return 0;`
`}`
	S= 60742236489000

Demak, for operatori o‘z tarkibida Masala-1 yechimidagi t o‘zgaruvchisiga boshlang‘ich qiymat berish va takrorlanayotgan “t=t+1; agar t ≤ 11021999 bo‘lsa, yig‘indi hisoblashga o‘tilsin;” ko‘rsatmalarni, ya’ni parametr uchun kiritilgan shartni va parametr qadamini saqlar ekan.

Oldingi paragrafdagi **Masala-1** uchun umumlashgan hollarning yechimlarida parametrli takrorlash operatori tarkibi va tanasini quyidagicha yozish mumkin:

s=1+14+27+…+ 11021999	`for(int t=1; t<=11021999; t=t+13) s+=t;`
s=1+14+27+…+ 11021999	`for(int t=0; t<=11021999/13; t++)` `  s+=1+t*13;`
s=1+2+3+…+N	`for(int t=1; t<=n; t++) s+=t;`
s=M+(M+1)+(M+2)+…+N	`for(int t=m; t<=n; t++) s+=t;`
s=M+(M+K)+(M+2·K)+…+ (M+(N–1)·K)	`for(int t=m; t<=m+(n-1)*k; t=t+k) s+=t;`
s=M+(M+K)+(M+2·K)+…+ (M+(N–1)·K)	`for(int t=0; t<=n-1; t++)` `  s+=m+t*k;`

Oldingi paragrafdagi **Masala-2** yechimini parametrli takrorlash operatori yordamida ham yozish mumkin va dasturi quyidagicha:

Dastur 2-for	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  int n;`
`  cout <<"N= "; cin>>n;`
`  long long int s=0;`
`  for(int t=1; s<=n; t++) s+=t;`
`  cout <<"S= "<< s;`
`  return 0;`
`}`
	S= 1950
	S= 1953

Xulosa: parametrli takrorlash operatorini parametrning boshlang'ich va oxirgi qiymati hamda o'zgarish qadami aniq bo'lganda qo'llash juda qulay.

4-§. SHART BO'YICHA TAKRORLASH OPERATORLARI

Shart bo'yicha takrorlash operatori, **while** parametrli takrorlash operatori **for** dan farqli ravishda, faqat berilgan **shart**ga asosan ish yuritadi. Bu operatorning umumiy ko'rinishi quyidagicha:

```
while(shart) {
        ko'rsatmalar ketma-ketligi
}
```

Bu yerda **shart** qiymati true yoki false ekvivalentlariga teng bo'ladigan ifoda bo'lib, takrorlashning bajarilish shartini ifodalaydi.

Bu operatorning ishlashi quyidagicha:

- avval **shart** tekshiriladi, agar shart qiymati **rost** (yoki rostning ekvivalentlariga teng) bo'lsa, u holda **while** operatori tanasidagi ko'rsatmalar ketma-ketligi bajariladi va yana shart tekshirishga qaytiladi;
- agar shart qiymati **yolg'on** (yoki yolg'onning ekvivalentlariga teng) bo'lsa, u holda boshqaruv **while** operatoridan keyingi operatorga uzatiladi.

Demak, **while** operatorida takrorlash tanasini tashkil etuvchi ko'rsatmalar ketma-ketligi bir marta ham bajarilmasligi mumkin ekan.

Parametrli takrorlash operatori va shart bo'yicha takrorlash operatorini taqqoslash maqsadida Namuna 2 masalasi yechimini **while** operatoridan foydalanib tuzamiz:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  int a=1;`
`  while(a<=10) {cout <<a <<" "; a++;}`
`  return 0;`
`}`
	1 2 3 4 5 6 7 8 9 10

Mazkur dasturni Namuna 2 ning **for** operatori yordamidagi ikkinchi yechimi bilan taqqoslang!

Namuna 6. Berilgan A va B ($1 \leq A, B \leq 10^9$) natural sonlarning qiymatlari teng bo'lmaguncha bu sonlarning kattasini kattasidan kichigini ayirmasi bilan almashtirib boruvchi dastur tuzing.

Yechim. Bu masalada A va B sonlarni kattasidan kichigini ayirib borish orqali tenglashtirish talab qilingan. Kattasidan kichigini ayirish jarayoni necha marta takrorlanishi avvaldan noma'lum. Shunga qaramay, taqqoslash qulay bo'lishi uchun, masala dasturini parametrli va shart bo'yicha takrorlash operatorlaridan foydalib tuzamiz.

Dastur for	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  int a, b;`
`  cout<<"A= ";cin>>a;`
`  cout<<"B= ";cin>>b;`
`  for(;a!=b;)`
`  if(a>b)a=a-b;else b=b-a;`
`  cout << "A= " <<a <<"\nB= "<< b;`
`  return 0;`
`}`
	A= 55
	B= 45
	A= 5
	B= 5

Dastur while	**Natijasi**

`#include <iostream>`
`using namespace std;`
`int main(){`
`  int a, b;`
`  cout<<"A= ";cin>>a;`
`  cout<<"B= ";cin>>b;`
`  while(a!=b)`
`  if(a>b)a=a-b;else b=b-a;`
`  cout << "A= " <<a <<"\nB= "<< b;`
`  return 0;`
`}`
	A= 55
	B= 45
	A= 5
	B= 5

Keltirilgan masala yechimi bo‘lgan ikkala dasturni taqqoslab shunday xulosa qilamiz:

Masalaning yechimida takrorlanish biror shartning natijasiga bog‘liq bo‘lganda dasturda shart bo‘yicha takrorlash operatoridan foydalanish qulaydir.

Oldingi paragrafdagi **Masala-2** ning yechimini **while** operatori yordamida ham ifodalaymiz:

Dastur 2-while	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  int t=1, n; long long int s=0;`
`  cout <<"N= "; cin>>n;`
`  while(s<=n) {s+=t; t++;}`
`  cout <<"S= "<< s;`
`  return 0;`
`}`
	S= 1950
	S= 1953

Ushbu masalaning umumlashgan hollari yechimlarini parametrli takrorlash operatori va shart bo‘yicha takrorlash operatori yordamida mustaqil bajaring.

Shart bo‘yicha takrorlash operatorining boshqa vakili **do-while** operatori bo‘lib, uning boshqa takrorlash operatorlaridan (for, while) farqi shundaki, bu operator shartni operator tanasidagi ko‘rsatmalar ketma-ketligi bajarilib bo‘lganidan so‘nggina tekshiradi. Bundan ko‘rinib turibdiki, **do-while** tanasida yozilgan ko‘rsatmalar ketma-ketligi eng kamida bir marta bajariladi. Operatorning umumiy ko‘rinishi quyidagicha:

```
do{ ko‘rsatmalar ketma-ketligi }
while(shart);
```

Namuna 2 masalasi bu operator yordamida quyidagicha yechiladi:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  int a=1;`
`  do { cout <<a <<" "; a++; }`
`  while(a<=10);`
`  return 0;`
`}`
	1 2 3 4 5 6 7 8 9 10

CONTINUE VA BREAK OPERATORLARI

Takrorlash operatorlari ishida ba'zan ma'lum bir qiymatlar uchun biror qism bajarilmasligi zarur bo'lib qoladi. Bunday hollarda **continue** operatori takrorlash operatorlari tanasida ishlatiladi va u keyingi takrorlanishga o'tish uchun xizmat qiladi.

Namuna 7. A, B va C ($-5 \le A, B, C \le 10^9$, B<C) butun sonlari berilgan. A sonini B sonidan C sonigacha bo'lgan butun sonlarga nisbati mavjud bo'lganlarini chop eting.

Yechim. Ikki sonning nisbatini hisoblashda maxraj 0 ga teng bo'lib qolishi mumkinligiga e'tibor berilishi kerak. Dasturda maxraj 0 bo'lganda uni o'tkazib yuborish uchun **continue** operatoridan foydalanamiz.

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  int a, b, c, t;`
`  cout<<"A= ";cin>>a;`
`  cout<<"B= ";cin>>b;`
`  cout<<"C= ";cin>>c;`
`  for(t=b; t<=c; t++) {`
`  if(t==0)continue;`
`  cout <<t<<" "<<a*1.0/t<<'\n';}`
`  return 0;`
`}`
	A= 10
	B= –2
	C= 3
	–2 –5
	–1 –10
	1 10
	2 5
	3 3.33333

Quyidagi dasturda esa **continue** operatori ekranga 5 soni chiqmasligini ta'minlaydi:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  for(int t=1; t<=10; t++) {`
`  if(t==5)continue;`
`  cout <<t<<" ";}`
`  return 0;`
`}`
	`1 2 3 4 6 7 8 9 10`

Shunday holatlar ham bo'ladiki, kerakli natija olingach yoki boshqa sababga ko'ra takrorlash operatorlari ishini tugatish kerak bo'lib qoladi. Bu holda **break** operatoridan foydalanish mumkin. Dasturda **break** operatori qaysi bir takrorlash operatori tanasida yozilgan bo'lsa, shu takrorlash operatorining ishi to'xtatiladi. Agar takrorlash operatorlari ichma-ich

joylashgan bo‘lsa, **break** operatori ularning qaysi biri tanasida yozilgan bo‘lsa, faqat shu takrorlash operatorining ishi to‘xtatiladi. Misol sifatida quyidagi dasturni keltirish mumkin:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  for(int t=1; t<=10; t++) {`
`  if(t==5) break;`
`  cout <<t<<" ";}`
`  return 0;`
`}`
	`1 2 3 4`

Ichma-ich joylashgan quyidagi takrorlash operatorlari ishida faqat 6 bilan tugaydigan qator chiqarilmasligi nazarda tutilgan:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  int k, t=10;`
`  while(t--){`
`  for(k=1; k<=t; k++) {`
`  if(t==6) break;`
`  cout<<k<<" ";}`
`  cout<<endl;}`
`  return 0;`
`}`
	`1 2 3 4 5 6 7 8 9`
	`1 2 3 4 5 6 7 8`
	`1 2 3 4 5 6 7`
	
	`1 2 3 4 5`
	`1 2 3 4`
	`1 2 3`
	`1 2`
	`1`

5-§. TAKRORLANUVCHI DASTURLARGA QO‘SHIMCHA NAMUNALAR

Namuna 8. N ($1 \le N \le 10^9$) natural sonining bo‘luvchilarini kamayish tartibida chop eting.

Yechim. Berilgan chegaralarga mos ravishda N soni va bo‘luvchilari uchun **int** turidagi n va b o‘zgaruvchilarni tavsiflaymiz. Agar b soni n sonining bo‘luvchisi bo‘lsa, u holda n soni b soniga qoldiqsiz bo‘linishi shart. Ma’lumki, sonni bo‘luvchilarining eng kichigi 1 va eng kattasi sonning o‘zi bo‘ladi. Bo‘luvchilarni kamayish tartibida chiqarish uchun b parametrga qiymatlarni kamayish tartibida o‘zlashtirishni tashkil etamiz. Shularni e’tiborga olgan holda

masala yechimiga mos dasturlarni parametrli va shart bo'yicha takrorlash operatorlaridan foydalanib quyidagicha tuzamiz:

Dastur for	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  int b, n;`
`  cout<<"N= "; cin>>n;`
`  for(b=n; b>=1; b--)`
`  if(n%b==0)cout<<b<<" ";`
`  return 0;`
`}`
	N= 15
	15 5 3 1

Dastur while	**Natijasi**

`#include <iostream>`
`using namespace std;`
`int main(){`
`  int b, n;`
`  cout<<"N= "; cin>>n;`
`  b=n;`
`  while(b>=1){`
`  if(n%b==0)cout<<b<<" ";`
`  b--;}`
`  return 0;`
`}`
	N= 11
	11 1

Namuna 9. N ($1 \le N \le 10^9$) natural sonining bo'luvchilari sonini aniqlang.

Yechim. Berilgan chegaralarga mos ravishda N soni, bo'luvchilar va sanoq uchun **int** turidagi n, b va s o'zgaruvchilarni tavsiflaymiz. Masalani Namuna 8 masalasi yechimini ozgina o'zgartirib (chiqarish o'rniga sanoqni kiritib) ham yechish mumkin. Lekin amallar sonini kamaytirish, ya'ni vaqtdan yutish maqsadida o'zgacha yo'l tutamiz. Bunda bo'luvchilarning xossalaridan foydalanamiz. Ma'lumki, 1-xossaga ko'ra, agar b soni N sonining bo'luvchisi bo'lsa, u holda N/b soni ham N sonining bo'luvchisi bo'ladi. Endi 2-xossaga ko'ra bo'luvchiga mos b parametr qiymatini 1 dan $\sqrt{N}$ gacha o'zgartirishimiz va har bir b bo'luvchi aniqlangach, unga mos n/b bo'luvchini ham sanab borishimiz yetarli. Buning uchun har bir b bo'luvchi aniqlanganda hisobni 2 taga oshirib boramiz. Lekin 3-xossaga ko'ra, agar $\sqrt{N}$ soni butun son bo'lsa, ya'ni "N soni ildizdan chiqsa", faqat bitta bo'luvchi qo'shilishi kerak. Shu sababli bu holni dasturda alohida qarashni tashkil etamiz.

Dastur for	Natijasi

#include <iostream>
#include <cmath>
using namespace std;
int main(){
int b, n, s=0;
cout<<"N= "; cin>>n;
for(b=1; b<sqrtl(n); b++)
if(n%b==0)s=s+2;
if(trunc(sqrtl(n))*trunc(sqrtl(n))==n)s++;
cout<<s;
return 0;
}
	N= 15
	4

Dasturda **cmath** yoki **math.h** kutubxonasiga tegishli **sqrtl** (ildiz chiqarish) funksiyasidan foydalanildi. Dasturga biroz o'zgartirish kiritib masala yechimini bu funksiyasiz ham tashkil etish mumkin. Ikkinchi usulda b soni $\sqrt{N}$ bilan emas, balki aksincha, b^2 soni N soni bilan taqqoslanadi. Bu usulda $N=10^9$ bo'lganda b soni kvadratining eng katta qiymati 10^9 dan ortishi bilan takrorlash operatori o'z ishini yakunlaydi:

Dastur for	Natijasi

#include <iostream>
using namespace std;
int main(){
int b, n, s=0;
cout<<"N= "; cin>>n;
for(b=1; b*b<n; b++)
if(n%b==0)s=s+2;
if(b*b==n)s++;
cout<<s;
return 0;
}
	N= 900000000
	243

Oxirgi qo'llangan usul ba'zi masalalar yechimini sodda dastur ko'rinishda ifodalash imkonini beradi. Masalan, quyidagi masalani qaraylik:

Namuna 10. Ildizi butun son bo'ladigan birinchi N ta ($1 \le N \le 10^9$) natural sonni o'sish tartibida ekranga chiqaring.

Yechim. Birinchi "xayolga keladigan" usul 1 dan boshlab har bir son ildizining butunligini tekshirish va agar ildizi butun son bo'lsa, uni sanab borishdir. O'zgaruvchilarni tavsiflashda $N^2 \le 10^{18}$ bo'lishini e'tiborga olib long long turni tanlaymiz. Masala dasturini **while** yordamida tuzamiz:

Dastur while	Natijasi

`#include <iostream>`
`#include <math.h>`
`using namespace std;`
`int main(){`
`  long long b=0, s=0, k, n;`
`  cout<<"N= "; cin>>n;`
`  while(s<n){b++; k=trunc(sqrtl(b));`
`  if(k*k==b){s++; cout<<b<<" ";}`
`  }`
`  return 0;`
`}`
	N= 9
	1 4 9 16 25 36 49 64 81

Lekin bu dastur N=10000 bo'lganda natija olishga taqriban 6 sekund vaqt sarflasa, masala yechimiga mos quyidagi dastur esa taqriban 2 sekund vaqt sarflaydi:

Dastur for	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  long long t, n;`
`  cout<<"N= "; cin>>n;`
`  for(t=1;t<=n;t++)`
`  cout<<t*t<<" ";`
`  return 0;`
`}`
	N= 9
	1 4 9 16 25 36 49 64 81

Namuna 11. Klaviaturadan butun sonlar ketma-ket kiritib boriladi. 0 soni kiritilgach sonlarni kiritish yakunlanadi. Shu sonlar o'rta arifmetigini aniqlang. Kiritilayotgan sonlarning moduli 10^6 dan oshmaydi.

Yechim. Kiritilayotgan butun sonlarni **x** o'zgaruvchi orqali belgilaymiz. Masalaga javob bo'ladigan kiritilgan sonlarning o'rta arifmetigi butun son bo'lishi shart emas, demak, unga mos **s** o'zgaruvchini (nechta son kiritilishi noma'lum bo'lgani uchun eng katta xotira hajmiga ega) **long double** turda tavsiflaymiz. Masala shartini tahlil qilib quyidagilarni aniqlaymiz:

1) kiritilayotgan x o'zgaruvchiga mos sonlar **int** turi chegarasidan chiqmaydi;

2) nechta son kiritilishi avvaldan ma'lum emas, hech bo'lmasa 0 kiritiladi;

3) 0 kiritilgach, ya'ni x o'zgaruvchiga 0 soni o'zlashtirilgach, sonlarni kiritish to'xtaydi;

4) biror k ta sonning o'rta arifmetigi $\frac{x1+x2+\cdots+xk}{k}$ bo'ladi.

2-3-tahlilga ko'ra kiritish jarayoni to'xtamasligi uchun **do{...}while(x!=0)** operatoridan foydalanish qulay ekanligi ma'lum bo'ldi. 4-tahlilga asosan nechta son kiritilganini sanab borish uchun **int** turidagi k o'zgaruvchisini kiritish zarur ekan. Oxirida kiritilgan 0 ni sanamaslik kerakligini unutmasdan dasturni quyidagi ko'rinishda tuzamiz:

Dastur do-while	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  int x, k=0;`
`  long double s=0;`
`  do { cin>>x; k++; s=s+x; }`
`  while(x!=0);`
`  cout<<"Javob: ";`
`  if(k==1) cout<<s;`
`  else cout<<s/(k-1);`
`  return 0;`
`}`
	2 8 5 0
	Javob 5

Namuna 12. a, m, x, s ($0< a, m, x, s <10^6$) butun sonlar berigan.

$$y_0 = a;\ y_k = \frac{m-1}{m}\cdot y_{k-1} + \frac{x}{m\cdot y_{k-1}^{m+1}},\ k=1, 2, 3, \ldots$$

qonuniyat asosida hosil qilingan y_1, y_2, ... ketma-ketlikning $|y_n - y_{n-1}| < s$ shartni qanoatlantiruvchi birinchi uchragan y_n-hadini topish dasturini tuzing.

Yechim. Masala shartidagi ketma-ketlikning har bir hadi rekkurent formula yordamida avvalgi hadi asosida hosil qilinayotganini ko'rish mumkin:

$$y[0]=a,\ y[1]=\frac{m-1}{m}\cdot y[0]+\frac{x}{m\cdot y[0]^{m+1}},\ \ldots,\ y[k]=\frac{m-1}{m}\cdot y[k-1]+\frac{x}{m\cdot y[k-1]^{m+1}},\ \ldots$$

Bu rekkurentlik boshlang'ich qiymati a=0; va takroriy bajariladigan ifoda $a = \frac{m-1}{m}\cdot a+\frac{x}{m\cdot a^{m+1}}$ bo'lgan holga mos keladi. Agar masalada faqat ketma-ketlikning N hadini aniqlash kerak bo'lganda dastur uchun a o'zgaruvchining o'zi yetarli bo'lardi. Lekin tekshirilayotgan shartda ketma-ketlikning ketma-ket kelgan ikkita hadi qatnashgani uchun avvalgi hadni dastur eslab qolishi kerak bo'ladi, ya'ni rekkurentlik ifodasini quyidagicha yozamiz:

$$b=a;\ a = \frac{m-1}{m}\cdot b+\frac{x}{m\cdot b^{m+1}}.$$

Shu sababli rekkurent formulaga asosan dasturni quyidagi tartibda tashkil etamiz:

1) a, m, x, s sonlari kiritiladi;

2) b=a va $a = \frac{m-1}{m}\cdot b+\frac{x}{m\cdot b^{m+1}}$;

3) agar $|a-b|<s$ bo'lsa, u holda a soni ekranga chiqariladi va takrorlash yakunlanadi;

4) agar $|a-b|\geq s$ bo'lsa, u holda 2)-bandga o'tiladi.

Dasturda m, x, s o'zgaruvchilarni **int** turda, a (rekkurent formula ifodasida nisbat qatnashgani uchun) va b o'zgaruvchilarni **long double** turda tavsiflaymiz. Sonning darajasini hisoblash funksiyasi **pow** qo'llanayotgani uchun **cmath** direktivasi qo'shilgan yechimni **do{...}while(abs(a-b)>=s)** yordamida tuzamiz:

Dastur do-while	Natijasi

#include <iostream>
#include <cmath>
using namespace std;
int main(){
int m, x, s;
long double a, b;
cin>>a>>m>>x>>s;
do { b=a;
a=(m-1)*1.0/m*b +x*1.0/(m*pow(b,m+1));}
while(abs(a-b)>=s);
cout<<"Javob:"<<a;
return 0;
}
	2 8 5 1
	Javob 1.75122

Namuna 13. Natural N ($1 \leq N \leq 10^{18}$) sonining 2 lik sanoq sistemasidagi ko'rinishida uchraydigan birlar sonini aniqlang.

Yechim. Masala shartida berilgan N soni uchun butun turdagi n10 (sonni o'nlik sanoq sistemasida berilganligi ta'kidlangandek) o'zgaruvchi kiritamiz. Bu masalani turli usullarda yechish mumkin. Quyida shunday usullardan bir nechtasini ko'rib chiqamiz.

1-usul. Sonni 2 lik sanoq sistemasiga o'tkazish masalasi 2-bob Namuna 7 da kichik chegaralar uchun qaralgan edi. Lekin qo'llangan quyidagi algoritm bu masala uchun ham to'g'ri keladi:

Berilgan sonni 2 ga qoldiqli bo'linmalaridan biri 2 dan kichik bo'lguncha 2 ga ketma-ket qoldiqli bo'linadi va qoldiqlar o'ngdan chapga qarab yozib olinadi.

Bu usulda biz faqat qoldiq 1 bo'lgan holni sanab borishimiz kerak, xolos.

Dastur while	Natijasi

#include <iostream>
using namespace std;
int main(){
long long n10; int s=0, q;
cout<<"N= "; cin>>n10;
while(n10>0){
q=n10%2;if(q==1)s++;
n10=n10/2;}
cout<<"Javob: "<<s;
return 0;}
	999999999
	Javob 21

2-usul. Bu usulda son 2 lik sanoq sistemasiga oshkor holda o'tkazilmasdan masala hal etiladi. Ya'ni C++ tili bajara oladigan bitga mos N&(N–1) mantiqiy ko'paytirish amali beradigan imkoniyatdan foydalanib yechim olinadi. Bu imkoniyatni n10=19 bo'lgan holda ko'rib chiqaylik.

C++ tili sonni xotirada 2 lik ko'rinishda saqlagani uchun bitli amallarni bajarishda sonning shu 2 lik ifodasidan foydalanadi. Berilgan n10=19 sonining 2 lik ifodasi n2=0..010011 ko'rinishda bo'ladi va u 3 ta 1 dan iborat ekan. Bu ko'rinishda 10011 sonining oldidagi 0 lar soni kompyuter xotirasi razryadi va o'zgaruvchi turi bilan bog'liq, masalan, 32 razryadli kompyuter xotirasida n10=19 soni 27 ta 0 va keyin 10011 orqali ifodalanadi. Bizga kerakli ikkinchi (n10–1)=18 sonining 2 lik ifodasi 0..010010 ko'rinishda bo'ladi va bitga mos mantiqiy ko'paytirish amali 1-qadamda:

n10&(n10–1)=0..010011&0..010010=0..010010

natijani, 2-qadamda esa:

n10&(n10–1)=0..010010&0..010001=0..010000

natijani va 3-qadamda:

n10&(n10–1)=0..010000&0..001000=0

natijani hosil qiladi.

Demak, N sonida nechta 1 raqami uchrasa, bitga mos mantiqiy ko'paytirish amali shuncha marta takrorlanar ekan. U holda dasturni quyidagicha yozish mumkin.

Dastur while	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  long long n10, s=0;`
`  cout<<"N= "; cin>>n10;`
`  while(n10>0){`
`  n10=n10&(n10-1);`
`  s++;}`
`  cout<<"Javob: "<<s;`
`  return 0;`
`}`
	17205
	Javob 7

Bu dastur, 1-usuldan farqli ravishda, N soni manfiy bo'lganda ham, birlar sonini kompyuter razryadiga mos ravishda hisoblay oladi.

Namuna 14. Natural N ($1 \le N \le 10^9$) sonini eng kam sondagi 2 sonining darajalari yig'indisi ko'rinishida tasvirlang.

Yechim. Masaladagi "eng kam sondagi" shartini tushunish uchun bir nechta misol ko'raylik:

$1=2^0$; $2=2^1$; $3=1+2=2^0+2^1$; $4=2^2$; $5=1+4=2^0+2^2$; $6=2+4=2^1+2^2$; $7=1+2+4=2^0+2^1+2^2$; ...

Bundan ko'rinadiki, masalani yechish uchun N sonini tanlov yordamida 2 lik sanoq sistemasiga o'tkazish usulini qo'llash kerak ekan. Tanlov usulida avval N sonidan katta bo'lmagan eng yaqin 2 sonining darajasi "tanlanadi" va N soni tanlangan 2 ning darajasi va qoldiq yig'indisi ko'rinishida tasvirlanadi. Keyingi qadamda qoldiq uchun ham qoldiqdan katta bo'lmagan eng yaqin 2 ning darajasi "tanlanadi" va qoldiq tanlangan 2 ning darajasi va yangi qoldiq yig'indisi ko'rinishida tasvirlanadi va hokazo. Masalan, 2107=2048+32+16+8+2+1 uchun ifoda quyidagicha hosil qilinadi:

$2107=2048+59=2^{11}+32+27=2^{11}+2^5+16+11=2^{11}+2^5+2^4+8+3=2^{11}+2^5+2^4+2^3+2^1+2^0$.

Tanlov usulidan quyidagi xossa kelib chiqadi:

22-xossa. Har qanday natural son eng kam sondagi 2 sonining darajalari o'sib borish tartibidagi yig'indisi ko'rinishida yagona usulda ifodalanadi.

Ma'lumki, $2^{30}=1073741824>1000000000=10^9$. Demak, N soni eng ko'pi bilan 30 ta son, ya'ni 2^0 darajasidan 2^{29} darajasigacha bo'lgan sonlar yig'indisiga teng bo'ladi. Endi masala yechimiga olib keladigan 2 sonining darajasini hosil qilish bilan bog'liq usullarni ko'rib chiqamiz.

1-usul. Tanlov uchun kerakli bo'lgan 2 sonining darajasini har safar hosil qilib boramiz. Hosil qilingan daraja sondan katta bo'lishi bilan 2 sonining darajasini yarmini ekranga chiqaramiz. Mos dastur quyidagicha:

Dastur while	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  int n10, d;`
`  cout<<"N= "; cin>>n10;`
`  cout<<"N=";`
`  while(n10>1){d=1;`
`  while(d<=n10)d=d*2;`
`  n10=n10%(d/2);`
`  cout<<d/2; if(n10>0)cout<<"+";}`
`  if(n10==1)cout<<1;`
`  return 0;`
`}`
	N= 2107
	N=2048+32+16+8+2+1

Bu usulda har safar 2 sonining darajasi qayta hisoblanayotgani amallar sonini ko'paytiradi.

2-usul. Bu usulda C++ tilidagi bajara oladigan bitga mos razryadli surish va bitga mos mantiqiy ko'paytirish N&(1<<t) amali beradigan imkoniyatdan foydalanib yechim olinadi.

Dastur for	Natijasi

#include <iostream>
using namespace std;
int main(){
int n10, t, b=0;
cout<<"N= "; cin>>n10;
cout<<"N=";
for (t=0;t<=29;t++){
if(n10&(1<<t)){cout<<(1<<t);b=1;}
if(b&&n10&(1<<t+1))cout<<"+";}
return 0;
}
	N= 1963
	N=1+2+8+32+128+256+512+1024

Ikkala usul dasturida ekranga chiqarilayotgan ma'lumotlar masala shartidagi kabi ko'rinishda bo'lishi uchun rejadan chetlanishlar qilindi.

Namuna 15. Fibonachchi ketma-ketligining birinchi hadi F[1]=0 va ikkinchi hadi F[2]=1 bo'lib, keyingi har bir hadi F[k]=F[k–2]+F[k–1], k=3,4,5,... qonuniyat bilan hosil qilinadi. Berilgan butun N ($0 \le N < 95$) soni uchun Fibonachchi ketma-ketligining F[N] hadni aniqlang.

Yechim. Rekkurent formuladan ko'rinadiki, Fibonachchi sonlarini eslab turish va hisoblash uchun 3 ta o'zgaruvchi yetarli bo'ladi. 3-Fibonachchi soni uchun A=F[1], B=F[2], F=F[3] kabi belgilansa F(=F[3]=F[1]+F[2])=A+B bo'ladi. Endi A=B(=F[2]), B=F(=F[3]) kabi almashtirish bajaramiz. U holda A=F[2], B=F[3] bo'lgani uchun 4-Fibonachchi soni F(=F[4]=F[2]+F[3])=A+B bo'ladi. Shu kabi davom ettirib kerakli hadni aniqlaymiz.

Dastur for	Natijasi

#include <iostream>
using namespace std;
int main(){
unsigned long long n,t,a=0,b=1,f;
cout<<"N= "; cin>>n;
if(n==0)cout<<a;
if(n==1)cout<<b;
for (t=2;t<=n;t++){
f=a+b;a=b; b=f;}
if(n>1)cout<<f;
return 0;
}
	N= 19
	2584

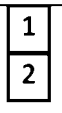
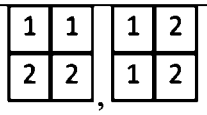
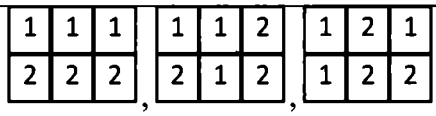
Dasturda bir necha marta shart tekshirilmoqda. Agar a= –1 (`unsigned long long` da a uchun xotirada 2^{64}–1 saqlanadi va shu sababli a+1=0 bo'ladi) va b=1 kabi olinsa, u holda

takrorlanishni t=1 dan boshlab bajarish mumkin bo‘ladi. Bu holda 1-Fibonachchi soni F=A+B=–1+1=0, 2-Fibonachchi soni A=B(=1), B=F(=0) va F=A+B=1+0=1 bo‘ladi:

Dastur for	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  unsigned long long n,t,a=-1,b=1,f;`
`  cout<<"N= "; cin>>n;`
`  for (t=1;t<=n;t++){`
`  f=a+b;a=b; b=f;}`
`  cout<<f;`
`  return 0;`
`}`
	N= 10
	34

Namuna 16. 2xN (0<N<95) o‘lchamli to‘rtburchakni 1x2 o‘lchamli to‘rtburchaklar bilan necha xil usulda qoplash mumkinligini aniqlovchi dastur tuzing. Masalan, Q – qoplash usullari bo‘lsa:

N=1 da Q[1]=1	N=2 da Q[2]=2	N=3 da Q[3]=3
		

Yechim. Har bir N uchun qoplash usullari soni Q[N] bo‘lsin. N sonini 1 dan 5 gacha olib, quyidagi jadvalni hosil qilish mumkin:

N	Q[N]
1	1
2	2
3	3
4	5
5	8

Jadvaldan ajoyib bir qonuniyatni aniqlash mumkin: N=3 dan boshlab har bir N uchun Q[N]= Q[N-2]+Q[N–1] (Fibonachchi sonlarini eslang). Jadvalni davom ettirib formulaning to‘g‘riligiga ishonch hosil qilish mumkin. Ammo dastur bunday kuzatishlarga tayanib tuzilmaydi. Ushbu qonuniyatning analitik isboti ko‘p adabiyotlarda keltirilgan. Masala dasturi esa Namuna 15 da bor.

Namuna 17. (E.Deykstr) Natural argumentli F funksiya quyidagicha aniqlangan: F(0)=0, F(1)=1, F(2k)=F(k), F(2k+1)=F(k)+F(k+1). Berilgan N ($0 \le N < 10^9$) sonida F(N) funksiya qiymatini hisoblash dasturini tuzing.

Yechim. Rekkurent formula asosida 3 ta a, b, d o'zgaruvchi yordamida F[N] funksiya qiymatini hosil qilamiz. k=N, a=1, b=0 va F[N]=a*F[k]+b*F[k+1] deb olamiz. U holda:

1) Agar k juft bo'lsa, ya'ni k=2*d bo'lsa, u holda: F[k]=F[2*d]=F[d],

F[k+1]=F[2*d+1]=F[d]+F[d+1].

Belgilashga asosan esa:

F[N]=a*F[k]+b*F[k+1]=a*F[d]+b*(F[d]+F[d+1])=(a+b)*F[d]+b*F[d+1].

Bundan keyingi qadamda a=a+b, k=d kabi olish mumkinligi kelib chiqadi.

2) Agar k toq bo'lsa, ya'ni k=2*d+1 bo'lsa, u holda: F[k]=F[2*d+1]=F[d]+F[d+1],

F[k+1]=F[2*d+2]=F[2*(d+1)]=F[d+1].

Belgilashga asosan esa:

F[N]=a*F[k]+b*F[k+1]=a*(F[d]+F[d+1])+b*F[d+1]=a*F[d]+(a+b)*F[d+1].

Demak, keyingi qadamda b=a+b, k=d kabi olish mumkin ekan. Nihoyat, k=0 bo'lganda F[N]=a*F[0]+b*F[1]=a*0+b*1=b ekanligi uchun javob sifatida b ni olamiz.

Dastur for	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  int n,k,a=1,b=0,d;`
`  cout<<"N= "; cin>>n;`
`  k=n;`
`  while(k!=0){d=k/2;`
`  if(k%2==0)a=a+b;`
`  else b=a+b;`
`  k=d;}`
`  cout<<"F["<<n<<"]="<<b;`
`  return 0;`
`}`
	N= 1002
	F[1002]=28

6-§. TAKRORLANUVCHI ALGORITMLI MASALALAR

Quyidagi misol va masalalar uchun, ba'zi misol va masalani chiziqli yoki tarmoqlanuvchi algoritmli dasturga keltirish mumkin bo'lsa ham, takrorlanuvchi algoritmli dastur tuzish talab etiladi. Agar shartda takrorlash operatori aniq ko'rsatilmagan bo'lsa, u holda takrorlashni ixtiyoriy operator yordamida yoki usulda amalga oshirish mumkin. Shart bo'yicha tekshirish operatori **while** ko'rsatilgan masalalarda **while** yoki **do...while** operatorlaridan ixtiyoriysini qo'llash mumkin. Masala shartida berilgan ma'lumotlarga e'tibor

qaratishni maslahat beramiz. Misol yoki masala yechimi yetarli aniqlikda chiqishi uchun haqiqiy sonlarni **long double** turida tavsiflashni tavsiya etamiz. Agar butun sonlar uchun yuqori chegara berilmagan bo'lsa, u holda sonlar **int** turi chegarasidan chiqmaydi. Natija uchun dasturchi tomonidan, masala mazmunidan kelib chiqib, mos tur tanlanishi kerak. Masalalarda uchraydigan "minimal" va "maksimal" atamalari "eng kichik" va "eng katta" tushunchalariga ekvivalentdir.

= A =

for 1. K haqiqiy son va natural N soni berilgan. Ekranga K sonini 10^{-2} aniqlikda N marta chiqaring.

for 2. Natural A va B sonlari berilgan (A<B). A dan B gacha bo'lgan (ularning o'zini ham) barcha sonlarni va chiqarilgan sonlar sonini chiqaring.

for 3. Natural A va B sonlari berilgan (A<B). A va B sonlar orasidagi (ularning o'zi kirmaydi) barcha sonlarni kamayish tartibida va chiqarilgan sonlar sonini chiqaring.

for 4. Bir dona tovar narxini belgilovchi haqiqiy son berilgan. Ikki xonali toq sonlar sonicha tovarlar narxini o'sish tartibida chiqaring. Natijalar eng kamida tiyinda ifodalanishi uchun 10^{-2} aniqlikda hisoblang.

for 5. Bir kilogramm un narxini belgilovchi haqiqiy son berilgan. 200 gramm, 400 gramm, ..., 3 kilo 800 gramm og'irlikkacha bo'lgan un narxini kamayish tartibida chiqaring. Natijalar eng kamida tiyinda ifodalanishi uchun 10^{-2} aniqlikda hisoblang.

for 6. Butun A va B sonlari berilgan (A<B). A dan B gacha bo'lgan (ularning o'zi kirmaydi) sonlar yig'indisini hisoblang.

for 7. Butun A va B sonlari berilgan. A va B sonlar oralig'idagi (ularning o'zi ham kiradi) sonlar yig'indisini hisoblang.

for 8. Butun A va B sonlari berilgan (A<B). A va B sonlar oralig'idagi (ularning o'zi ham kiradi) toq sonlar ko'paytmasini hisoblang.

for 9. Butun A va B sonlari berilgan (A<B). A va B sonlar oralig'idagi (ularning o'zi ham kiradi) juft sonlar kvadratlari yig'indisini hisoblang.

for 10. $1\cdot2+3\cdot4+5\cdot6+\ldots+101\cdot102$ yig'indini hisoblang.

for 11. Natural N soni berilgan. $(1\cdot2)^2+(2\cdot3)^2+(3\cdot4)^2+\ldots+((N-1)\cdot N)^2$ yig'indini hisoblang.

for 12. Natural N soni berilgan. $1\cdot2\cdot3$, $3\cdot4\cdot5$, $5\cdot6\cdot7$, . . . ketma-ketlikning dastlabki N ta hadi yig'indisini hisoblang (yo'llanma: ketma-ketlikning k-hadi: $(2\cdot k-1)\cdot2\cdot k\cdot(2\cdot k+1)$).

for 13. Natural N soni berilgan. 1 dan N gacha bo'lgan barcha natural sonlar ichida faqat 7 ga karrali sonlar yig'indisini toping

for 14. Natural N soni berilgan. 1 dan N gacha bo'lgan barcha natural sonlar ichida ham 3 ga, ham 11 ga karrali sonlar ko'paytmasini toping.

for 15. Natural N soni berilgan. $1+\frac{1}{2}+\frac{1}{3}+\cdots+\frac{1}{N}$ yig'indini hisoblang.

for 16. Natural N soni berilgan. $N^2+(N+1)^2+(N+2)^2+\ldots+(2\cdot N)^2$ yig'indini hisoblang.

for 17. Natural N soni berilgan. $1{,}1\cdot 1{,}2\cdot 1{,}3\cdot\ldots$ ifodadagi dastlabki N ta ko'paytuvchi ko'paytmasini hisoblang.

for 18. Natural N soni berilgan. $1{,}1-1{,}2+1{,}3-\ldots$ ishora almashtiruvchi ifodadagi dastlabki N ta had orasidagi amallar natijasini aniqlang. Shart tekshirish mumkin emas.

for 19. Haqiqiy A va natural N soni berilgan. $A^1, A^2, A^3, \ldots, A^N$ sonlarini chiqaring.

for 20. Haqiqiy A va natural N soni berilgan. Faqat bitta sikl yordamida $A+A^2+A^3+\ldots+A^N$ yig'indini hisoblang.

for 21. $1-\frac{1}{2}+\frac{1}{3}-\frac{1}{4}+\cdots+\frac{1}{9999}-\frac{1}{10000}$ yig'indini hisoblang (yo'llanma: ishora almashishini hisobga olish uchun ishora=–1; ishora=(–1)·ishora; ifodalardan foydalaning).

for 22. Haqiqiy A va natural N soni berilgan. Faqat bitta sikl yordamida $A-A^2+A^3-\ldots+(-1)^N A^N$ ishora almashtiruvchi ifoda natijasini hisoblang. Shart tekshirish mumkin emas.

for 23. Natural N soni berilgan. $N!=1\cdot 2\cdot 3\cdot\ldots\cdot(N-1)\cdot N$ faktorialni hisoblang. Har bir ko'paytuvchidan keyin ko'paytmani chop etib boring.

for 24. Natural N soni berilgan. Faqat bitta sikl yordamida $1!+2!+3!+\ldots+N!$ yig'indini hisoblang. Har bir qo'shiluvchidan keyin yig'indini chop etib boring.

for 25. Natural N soni berilgan. Faqat bitta sikl yordamida $\frac{1}{1!}+\frac{1}{2!}+\frac{1}{3!}+\cdots+\frac{1}{N!}$ yig'indini hisoblang. Har bir qo'shiluvchidan keyin alohida satrda 10^{-5} aniqlikda yig'indini chop etib boring.

for 26. Faqat bitta sikl yordamida $\frac{7}{11}+\frac{17}{21}+\frac{27}{31}+\cdots+\frac{2017}{2021}$ yig'indini hisoblang (yo'llanma: s:=s+k/(k+4) yoki s=s+(10*k+7)/(10*k+11) kabi ifodalardan foydalaning).

for 27. Natural N soni berilgan. $\frac{(-1)^1}{3}+\frac{(-1)^2}{10}+\frac{(-1)^3}{21}+\cdots+\frac{(-1)^N}{(2\cdot N+1)\cdot N}$ yig'indini hisoblang. Daraja hisoblash funksiyasi pow dan foydalanmang.

for 28. Natural N soni berilgan. $\frac{1}{2},\frac{3}{4},\frac{5}{6},\frac{7}{8},\ldots$ ketma-ketlikning dastlabki N ta hadi yig'indisini hisoblang.

for 29. Natural N soni berilgan. $\frac{1}{1},\frac{3}{2},\frac{5}{3},\frac{7}{4},\ldots$ ketma-ketlikning dastlabki N ta hadi yig'indisini hisoblang.

for 30. Natural N va k sonlari berilgan. Faqat bitta sikl yordamida $1+\frac{1}{2^k}+\frac{1}{3^k}+\cdots+\frac{1}{N^k}$ yig'indini hisoblang.

for 31. Berilgan a haqiqiy sonidan kichik barcha natural x larda $y=a\cdot x^2+20$ funksiyaning qiymatlarini chiqaring.

for 32. $y=2\cdot x+19$ funksiyaning qiymatini x ning [0; 10] oraliqdagi 0.25 qadami bilan hisoblang.

for 33. $y = x\cdot\sin x$ funksiyaning qiymatlarini $[-\pi, \pi]$ oraliqda 0,3 qadam bilan hisoblang.

for 34. Natural N sonining barcha toq bo'luvchilarini chiqaring. Agar bunday sonlar bo'lmasa, u holda bu haqida ma'lumot chiqaring.

for 35. Natural N soni berilgan. 1 dan N gacha bo'lgan natural sonlar ichida oxirgi raqami 3 ga karrali sonlarni chiqaring.

for 36. Ikki xonali natural sonlar ichidan raqamlari yig'indisi juft bo'lgan sonlarni chiqaring.

while 37. Natural N sonining raqamlari sonini chiqaring.

while 38. Natural N sonining raqamlari yig'indisini chiqaring.

while 39. Natural N sonining raqamlari ichida 2 raqami bor bo'lsa "2", yo'q bo'lsa "–1" chiqaring.

while 40. Natural N sonining raqamlari ichida toq qiymatli raqam bor bo'lsa "1", yo'q bo'lsa "–1" chiqaring.

while 41. Natural N soni berilgan ($1<N<100$). N soniga karrali biror uch xonali sonni aniqlang.

while 42. Natural N soni berilgan ($1000<N<10000$). N sonining raqamlari yig'indisiga karrali biror uch xonali sonni aniqlang.

while 43. Natural A va B sonlari berilgan ($A<B$). B uzunlikdagi kesma A uzunlikdagi kesmalarga bo'lindi. Ortib qolgan kesma uzunligini aniqlang. Ko'paytirish, bo'lish va qoldiqli bo'lish amallaridan foydalanish mumkin emas.

while 44. Natural A va B sonlari berilgan ($A<B$). B uzunlikdagi kesma A uzunlikdagi kesmalar kesib olindi. Ko'pi bilan nechta kesma kesib olinganini aniqlang. Ko'paytirish, bo'lish va qoldiqli bo'lish amallaridan foydalanish mumkin emas.

while 45. Natural N va M sonlari berilgan. Faqat qo'shish va ayirish amallaridan foydalanib N sonining M ga bo'lgandagi butun va qoldiq qismlarini aniqlang.

while 46. Natural N soni berilgan. Agar N soni 7 ning biror darajasi bo'lsa daraja ko'rsatkichini, aks holda –1 chiqaring.

while 47. Natural N soni berilgan. Kvadrati N sonidan katta bo'lmagan maksimal sonni aniqlang, ya'ni shunday eng katta M sonini aniqlangki: $M^2\le N$ bo'lsin.

while 48. Natural N soni berilgan. Kvadrati N sonidan kichik bo'lmagan minimal sonni aniqlang, ya'ni shunday eng kichik M sonini aniqlangki: $M^2\ge N$ bo'lsin.

while 49. Natural N soni berilgan. $1+\frac{1}{2}+\frac{1}{3}+\cdots+\frac{1}{K}+\cdots$ yig'indining qiymati N dan katta bo'lgan minimal K sonini va yig'indining qiymatini aniqlang.

while 50. Natural N soni berilgan. $1+\frac{1}{2}+\frac{1}{3}+\cdots+\frac{1}{K}+\cdots$ yig'indining qiymati N dan kichik bo'lgan maksimal K sonini va yig'indining qiymatini aniqlang.

while 51. 1 dan katta A soni berilgan. $7^K > A$ shart bajariladigan minimal butun K sonini aniqlang.

while 52. 1 dan katta A soni berilgan. $7^K \leq A$ shart bajariladigan maksimal butun K sonini aniqlang.

while 53. 1 dan katta A soni berilgan. Agar A tub son bo'lsa "0" raqamini, aks holda "–1" sonini chiqaring.

while 54. 1 dan katta A soni berilgan. Agar A tub son bo'lsa "0" raqamini, aks holda A sonining bo'luvchilari sonini chiqaring.

while 55. Natural A va B sonlari berilgan. EKUB(A, B) sonini aniqlang.

while 56. Natural A va B sonlari berilgan. EKUK(A, B) sonini aniqlang.

while 57. Natural N soni berilgan. N soni Fibonachchi soni bo'lishi yoki bo'lmasligini aniqlang.

while 57. Natural N soni berilgan. Qiymati N sonidan katta bo'lgan eng kichik Fibonachchi sonini aniqlang.

while 58. Natural N soni berilgan. Qiymati N sonidan kichik bo'lgan eng katta Fibonachchi sonining tartib raqamini aniqlang. Masalan, N=3 bo'lsa F[4]=2<N≤3=F[5], demak, javob: 4.

while 59. Klaviaturadan butun sonlar ketma-ket kiritib boriladi. Ketma-ketlikka tegishli bo'lmagan 0 soni kiritilgach sonlarni kiritish yakunlanadi. Kiritilgan sonlar ichida maksimal va minimal sonlarni aniqlang.

while 60. Klaviaturadan butun sonlar ketma-ket kiritib boriladi. Ketma-ketlikka tegishli bo'lmagan 0 soni kiritilgach sonlarni kiritish yakunlanadi. Kiritilgan sonlar ichida maksimal va minimal sonlarning ketma-ketlikdagi tartib raqamlarini aniqlang.

while 61. Klaviaturadan butun sonlar ketma-ket kiritib boriladi. Ketma-ketlikka tegishli bo'lmagan 0 soni kiritilgach sonlarni kiritish yakunlanadi. Kiritilgan sonlar ichida birinchi maksimal va oxirgi minimal sonlarning tartib raqamlarini aniqlang.

while 62. Klaviaturadan butun sonlar ketma-ket kiritib boriladi. Ketma-ketlikka tegishli bo'lmagan 0 soni kiritilgach sonlarni kiritish yakunlanadi. Kiritilgan sonlar ichida birinchi minimal va oxirgi maksimal sonlarning tartib raqamlarini aniqlang.

while 63. Klaviaturadan butun sonlar ketma-ket kiritib boriladi. Ketma-ketlikka tegishli bo'lmagan 0 soni kiritilgach sonlarni kiritish yakunlanadi. Kiritilgan sonlar ichida birinchi va oxirgi minimal sonlarning tartib raqamlarini aniqlang.

while 64. Klaviaturadan butun sonlar ketma-ket kiritib boriladi. Ketma-ketlikka tegishli bo'lmagan 0 soni kiritilgach sonlarni kiritish yakunlanadi. Kiritilgan sonlar ichida birinchi va oxirgi maksimal sonlarning tartib raqamlarini aniqlang.

while 65. Klaviaturadan natural sonlar juftligi ketma-ket kiritib boriladi. Ketma-ketlikka tegishli bo'lmagan 0 soni kiritilgach sonlarni kiritish yakunlanadi. Kiritilgan sonlar juftligi to'g'ri to'rtburchakning tomonlari bo'lib, minimal yuzani aniqlang.

while 66. Klaviaturadan natural sonlar juftligi ketma-ket kiritib boriladi. Ketma-ketlikka tegishli bo'lmagan 0 soni kiritilgach sonlarni kiritish yakunlanadi. Kiritilgan sonlar juftligi to'g'ri to'rtburchakning tomonlari bo'lib, maksimal perimetrni aniqlang.

while 67. Klaviaturadan haqiqiy sonlar ketma-ket kiritib boriladi. Ketma-ketlikka tegishli bo'lmagan 0 soni kiritilgach sonlarni kiritish yakunlanadi. Kiritilgan sonlar ichida minimal musbat sonni aniqlang. Agar musbat sonlar yo'q bo'lsa, u holda 0 sonini chiqaring.

while 68. Klaviaturadan haqiqiy sonlar ketma-ket kiritib boriladi. Ketma-ketlikka tegishli bo'lmagan 0 soni kiritilgach sonlarni kiritish yakunlanadi. Kiritilgan sonlar ichida maksimal manfiy sonni aniqlang. Agar manfiy sonlar yo'q bo'lsa, u holda 0 sonini chiqaring.

while 69. Klaviaturadan butun sonlar ketma-ket kiritib boriladi. Ketma-ketlikka tegishli bo'lmagan 0 soni kiritilgach sonlarni kiritish yakunlanadi. Kiritilgan sonlar ichida birinchi uchragan toq sonni aniqlang. Agar toq sonlar yo'q bo'lsa, u holda "–1" sonini chiqaring.

while 70. Klaviaturadan butun sonlar ketma-ket kiritib boriladi. Ketma-ketlikka tegishli bo'lmagan 0 soni kiritilgach sonlarni kiritish yakunlanadi. Kiritilgan sonlar ichida oxirgi uchragan juft sonni aniqlang. Agar juft sonlar yo'q bo'lsa, u holda "–1" sonini chiqaring.

while 71. Klaviaturadan avval haqiqiy musbat B soni va keyin butun sonlar ketma-ket kiritib boriladi. Ketma-ketlikka tegishli bo'lmagan 0 soni kiritilgach sonlarni kiritish yakunlanadi. Kiritilgan sonlardan B sonidan kichiklari ichidagi eng kattasini va shu sonning tartib raqamini aniqlang. Agar bunday son yo'q bo'lsa, u holda 0 sonini chiqaring.

while 72. Klaviaturadan avval haqiqiy B va C (0<B<C) sonlari va keyin butun sonlar ketma-ket kiritib boriladi. Ketma-ketlikka tegishli bo'lmagan 0 soni kiritilgach sonlarni kiritish yakunlanadi. Kiritilgan sonlardan (B; C) oraliqqa tegishlilari ichida maksimalini va shu sonning tartib raqamini aniqlang. Agar bunday sonlar yo'q bo'lsa, u holda 0 sonini chiqaring.

while 73. Klaviaturadan avval haqiqiy B va C (0<B<C) sonlari va keyin butun sonlar ketma-ket kiritib boriladi. Ketma-ketlikka tegishli bo'lmagan 0 soni kiritilgach sonlarni kiritish yakunlanadi. Kiritilgan sonlardan (B; C) oraliqqa tegishlilari sonini aniqlang. Agar bunday sonlar yo'q bo'lsa, u holda 0 sonini chiqaring.

while 74. Klaviaturadan avval haqiqiy B va C (0<B<C) sonlari va keyin butun sonlar ketma-ket kiritib boriladi. Ketma-ketlikka tegishli bo'lmagan 0 soni kiritilgach sonlarni

kiritish yakunlanadi. Kiritilgan sonlardan (B; C) oraliqqa tegishli bo'lmaganlari ichida maksimalini va shu sonning tartib raqamini aniqlang. Agar bunday sonlar yo'q bo'lsa, u holda 0 sonini chiqaring.

while 75. Klaviaturadan avval haqiqiy B va C (0<B<C) sonlari va keyin butun sonlar ketma-ket kiritib boriladi. Ketma-ketlikka tegishli bo'lmagan 0 soni kiritilgach sonlarni kiritish yakunlanadi. Kiritilgan sonlardan (B; C) oraliqqa tegishli bo'lmaganlari sonini aniqlang. Agar bunday sonlar yo'q bo'lsa, u holda 0 sonini chiqaring.

= B =

for 76. Natural N soni berilgan. $\frac{1}{1\cdot2\cdot3}, \frac{1}{2\cdot3\cdot5}, \frac{1}{3\cdot4\cdot7}, \frac{1}{4\cdot5\cdot9}$, ... ketma-ketlikning dastlabki N ta hadi yig'indisini hisoblang (yo'llanma: 1+2=3, 2+3=5, 3+4=7,...).

for 77. Ikkita A va B (A<B) haqiqiy son va natural N soni berilgan. [A; B] kesma N ta teng kesmaga bo'lingan. Kesma uzunligi H va [A; B] kesmani bo'luvchi quyidagi nuqtalar ketma-ketligi chiqarilsin: A, A+H, A+2·H, A+3·H, ..., B.

for 78. Ikkita A va B (A<B) haqiqiy son va natural N soni berilgan. [A; B] kesma N ta teng kesmaga bo'lingan. Kesma uzunligi H va kesmani bo'luvchi nuqtalarda f(x)=1–sinx funksiya qiymati chiqarilsin, ya'ni f(A), f(A+H), f(A+2·H), f(A+3·H), ..., f(B).

for 79. Natural N soni berilgan. Ketma-ketlik quyidagicha aniqlanadi:

$A[0]=2,\ A[k]=2+\frac{1}{A[k-1]},\ k=1, 2, \ldots.$

Ketma-ketlikning dastlabki N ta hadini chiqaring.

for 80. Natural N soni berilgan. Ketma-ketlik quyidagicha aniqlanadi:

$A[0]=1,\ A[k]=\frac{A[k-1]+1}{k},\ k=1, 2, \ldots.$

Ketma-ketlikning dastlabki N ta hadini chiqaring.

for 81. Natural N (N>1) soni berilgan. Ketma-ketlik quyidagicha aniqlanadi:

$A[1]=1,\ A[2]=2,\ A[k]=\frac{A[k-2]+2\cdot A[k-1]}{3},\ k=3, 4, \ldots.$

Ketma-ketlikning dastlabki N ta hadini chiqaring.

for 82. Natural N (N>1) soni berilgan. Ketma-ketlik quyidagicha aniqlanadi:

A[1]=1, A[2]=2, A[3]=3, A[k]=2·A[k–1]+A[k–2]–3·A[k–3], k=4, 5, 6, ...

Ketma-ketlikning dastlabki N ta hadini chiqaring.

for 83. Natural N va k sonlari berilgan. $1^k+2^k+3^k+\ldots+N^k$ yig'indini hisoblang. Daraja hisoblash funksiyasi pow dan foydalanish mumkin emas. Har bir qo'shiluvchidan keyin yig'indini chop etib boring (yo'llanma: ichma-ich joylashgan sikl).

for 84. Natural N soni berilgan. $1^N+2^{N-1}+3^{N-2}+\ldots+N^1$ yig'indini hisoblang. Daraja hisoblash funksiyasi pow dan foydalanish mumkin emas. Har bir qo'shiluvchidan keyin yig'indini chop etib boring (yo'llanma: ichma-ich joylashgan sikl).

for 85. Natural N soni berilgan. $1^1+2^2+3^3+\ldots+N^N$ yig'indini hisoblang. Daraja hisoblash funksiyasi pow dan foydalaninish mumkin emas. Har bir qo'shiluvchidan keyin yig'indini chop etib boring (yo'llanma: ichma-ich joylashgan sikl).

for 86. Natural M va N sonlari berilgan (M<N). M dan N gacha bo'lgan sonlarni chop eting (M va N soni ham kiradi). Bunda har bir son o'zining qiymaticha chiqarilsin, masalan, 3, 3, 3, 4, 4, 4, 4 kabi.

for 87. Natural M va N sonlari berilgan (M<N). M dan N gacha bo'lgan sonlarni chop eting (M va N sonlari ham kiradi). Bunda 1-son 1 marta, 2-son 2 marta, va hokazo, chiqarilsin, masalan, M, M+1, M+1, M+2, M+2, M+2 kabi.

for 88. Haqiqiy a, b, c va d sonlar berilgan. O'zgaruvchi x ning qiymati 0 dan 2 gacha 0,2 qadam bilan o'zgarganda $y=a\cdot x^3+b\cdot x^2+c\cdot x+d$ funksiyaning eng kichik qiymatini aniqlang.

for 89. Butun a, b, c, d va natural K sonlar berilgan. O'zgaruvchi x ning qiymati 0 dan 2 gacha 0,2 qadam bilan o'zgarganda $y=a\cdot x^3+b\cdot x^2+c\cdot x+d$ funksiyaning K sonidan oshmagan eng kichik qiymatini aniqlovchi dastur tuzing.

for 90. Natural N soni berilgan. $1\cdot2+2\cdot3\cdot4+3\cdot4\cdot5\cdot6+\ldots+N\cdot(N+1)\cdot\ldots\cdot2\cdot N$ yig'indi qiymatini aniqlang.

while 91. Natural N va m son berilgan. Raqamlari yig'indisining kvadrati m ga teng N sonidan katta bo'lmagan barcha natural sonlarni aniqlang. Bunday sonlar yo'q bo'lsa, u holda bu haqida axborot chiqaring (yo'llanma: ichma-ich joylashgan sikl).

while 92. Natural N va m son berilgan. Raqamlari kvadratining yig'indisi m ga teng N sonidan katta bo'lmagan barcha natural sonlarni aniqlang. Bunday sonlar yo'q bo'lsa, u holda bu haqida axborot chiqaring (yo'llanma: ichma-ich joylashgan sikl).

while 93. Klaviaturadan butun sonlar ketma-ket kiritib boriladi. Ketma-ketlikka tegishli bo'lmagan 0 soni kiritilgach sonlarni kiritish yakunlanadi. Kiritilgan sonlar ichida birinchi uchragan ekstremal (minimal yoki maksimal) sonning tartib raqamini aniqlang.

while 94. Klaviaturadan butun sonlar ketma-ket kiritib boriladi. Ketma-ketlikka tegishli bo'lmagan 0 soni kiritilgach sonlarni kiritish yakunlanadi. Kiritilgan sonlar ichida oxirgi uchragan ekstremal (minimal yoki maksimal) sonning tartib raqamini aniqlang.

while 95. Klaviaturadan haqiqiy sonlar ketma-ket kiritib boriladi. Ketma-ketlikka tegishli bo'lmagan 0 soni kiritilgach sonlarni kiritish yakunlanadi. Kiritilgan sonlar ichida birinchi minimal sondan oldin kiritilgan sonlar sonini aniqlang (yo'llanma: tartib raqami va kiritilganlar sonlar soni).

while 96. Klaviaturadan haqiqiy sonlar ketma-ket kiritib boriladi. Ketma-ketlikka tegishli bo'lmagan 0 soni kiritilgach sonlarni kiritish yakunlanadi. Kiritilgan sonlar ichida oxirgi

maksimal sondan keyin kiritilgan sonlar sonini aniqlang (yo'llanma: tartib raqami va kiritilgan sonlar soni).

while 97. Haqiqiy musbat E soni berilgan. A[0]=2, A[k]=2+$\frac{1}{A[k-1]}$, k=1, 2, ..., qonuniyat bilan aniqlangan ketma-ketlikning birinchi marta |A[N]–A[N–1]|<E shartni bajaruvchi A[N] hadini 10^{-5} aniqlikda chiqaring.

while 98. Haqiqiy musbat E soni berilgan. A[0]=1, A[k]=$\frac{A[k-1]+1}{k}$, k=1, 2, ..., qonuniyat bilan aniqlangan ketma-ketlikning birinchi marta |A[N]–A[N–1]|<E shartni bajaruvchi A[N] hadini 10^{-5} aniqlikda chiqaring.

while 99. Haqiqiy musbat E soni berilgan. A[1]=1, A[2]=2, A[k]=$\frac{A[k-2]+2\cdot A[k-1]}{3}$, k=3, 4, ..., qonuniyat bilan aniqlangan ketma-ketlikning birinchi marta |A[N]–A[N–1]|<E shartni bajaruvchi A[N] hadini 10^{-5} aniqlikda chiqaring.

while 100. Haqiqiy musbat E soni berilgan. Agar ketma-ketlik A[1]=1, A[2]=2, A[3]=3, A[k]=$\frac{2\cdot A[k-1]+A[k-2]-3\cdot A[k-3]}{k}$, k=4, 5, 6, ..., qonuniyat bilan aniqlangan bo'lsa, u holda birinchi marta |A[N]–A[N–1]|<E shartni bajaruvchi A[N] hadini 10^{-5} aniqlikda chiqaring.

= C =

for 101. Natural N va haqiqiy A sonlari berilgan. Tomoni A bo'lgan kvadrat ichiga aylana ichki chizilgan. Aylana ichiga kvadrat ichki chizilgan. Shu kabi ichki chizilgan figuralar ketma-ketligini davom ettirsak, N-kvadratning yuzini aniqlang.

for 102. Natural N va haqiqiy A sonlari berilgan. Tomoni A bo'lgan kvadrat ichiga aylana ichki chizilgan. Aylana ichiga kvadrat ichki chizilgan. Shu kabi ichki chizilgan figuralar ketma-ketligini davom ettirsak, N-aylananing radiusini aniqlang.

for 103. Natural N va haqiqiy A sonlari berilgan. Tomoni A bo'lgan teng tomonli uchburchak ichiga aylana ichki chizilgan. Aylana ichiga teng tomonli uchburchak ichki chizilgan. Shu kabi ichki chizilgan figuralar ketma-ketligini davom ettirsak, N-teng tomonli uchburchakning yuzini aniqlang.

for 104. Natural N va haqiqiy A sonlari berilgan. Tomoni A bo'lgan teng tomonli uchburchak ichiga aylana ichki chizilgan. Aylana ichiga teng tomonli uchburchak ichki chizilgan. Shu kabi ichki chizilgan figuralar ketma-ketligini davom ettirsak, N-aylananing radiusini aniqlang.

for 105. x·y+x+y=1000 tenglamani butun sonlarda yechish dasturini tuzing.

Yechim. Masala shartidan takrorlash tuzilmasidan foydalanish zarur bo'lgani bilan x va y sonlar uchun chegaraviy qiymatlar noma'lum. Shu sababli tenglama ustida quyidagi o'zgartirishlarni bajaramiz:

$x\cdot(y+1)+y=1000 \Leftrightarrow x\cdot(y+1)+y+1=1000+1 \Leftrightarrow (x+1)\cdot(y+1)=1001$.

Oxirgi tenglikdan hamda x va y sonlar butun bo'lishi kerakligidan yechimlar uchun chegaraviy qiymatlarni aniqlaymiz: $-1002\le x\le 1000$ va $-1002\le y\le 1000$.

1-usul. Endi ikkita parametrli takrorlash operatori yordamida x va y o'zgaruvchilarning berilgan tenglikni qanoatlantiruvchi yechimini aniqlashimiz mumkin. Bu holda takrorlanishlar soni tashqi sikl uchun $(1000-(-1002)+1)=2003$ ta, ichki sikl uchun $(1000-(-1002)+1)=2003$ ta, jami: $2003\cdot 2003=4012009$ ta bo'ladi.

2-usul. Oxirgi tenglik hamda x va y sonlar butun bo'lishi kerakligi (x+1) yoki (y+1) sonlar 1001 ni manfiy va musbat bo'luvchilari bo'lishi kerakligini ko'rsatmoqda. Agar z soni 1001 ning bo'luvchisi bo'lsa, u holda x=z–1 dan x ni va $y=\frac{1001}{z}-1$ dan y ni, y=z–1 dan y ni va $x=\frac{1001}{z}-1$ dan x ni aniqlaymiz. Bu holda takrorlanishlar soni bor-yo'g'i 1001 ta.

Dasturini mustaqil tuzing!

for 106. Natural N soni berilgan. $x\cdot y+x+y=N$ tenglamani butun sonlarda yeching.

for 107. Natural N soni berilgan. Arifmetik amallardan faqat qo'shish amalidan foydalanib A butun sonni N- darajaga ko'tarish dasturini tuzing (yo'llanma: |A| ta A ning yig'indisi A^2, |A| ta A^2 ning yig'indisi A^3).

for 108. Natural N soni berilgan. Ushbu sonni uchta natural sonning kvadratlari yig'indisi orqali ifoda etish mumkinligini aniqlang. Agar mumkin bo'lsa $N=x^2+y^2+z^2$ tenglik o'rinli bo'ladigan barcha x, y, z sonlar uchligini, aks holda, ya'ni bitta ham bunday uchlik bo'lmasa, –1 chiqaring (yo'llanma: ichma-ich joylashgan sikllar).

for 109. Natural N soni berilgan. $x^2+y^2=z^2$ tenglik o'rinli bo'ladigan x, y va z ning [1; N] oraliqdagi natural qiymatlari bor bo'lsa, u holda barcha (Pifagor) uchliklarini, aks holda "–1" chiqaruvchi dastur tuzing.

for 110. Berilgan natural N va M ($1 \le N, M \le 10000$) sonlar uchun N^M sonining oxirgi raqamini aniqlovchi dastur tuzing.

for 111. Butun X va Y berilgan ($1\le X<Y\le 100$). Agar qutqaruvchilar guruhidagi qizlar X% dan ko'p, lekin Y% dan kam bo'lsa, qutqaruvchilar guruhidagilarning eng kamida nechta bo'lishi mumkinligini aniqlang.

Yechim. Takrorlash tuzilmasi yordamida qutqaruv guruhidagi ishtirokchilar soni **ishs** ni 1 dan 101 gacha, qizlar soni **qs** ni 1 dan ishtirokchilar soni **ishs** gacha qaraymiz. Chegaralash foiz hisobida berilayotgani uchun qizlar soni **qs** ni foiz hisobi **qf** ga o'tkazib $X\% < qf < Y\%$ shartni tekshiramiz. Eng kam ishtirokchilar sonini aniqlash talab qilingani uchun birinchi marta shart bajarilgan zahoti javob chiqarib dastur ishini yakunlaymiz.

Dasturini mustaqil tuzing!

for 112. Natural N soni berilgan. Ushbu sondan katta bo'lmagan barcha mukammal sonlarni chiqaruvchi dastur tuzing. Natural son o'zidan kichik bo'luvchilari yig'indisiga teng bo'lsa **mukammal son** deyiladi. Masalan, 6 soni mukammal son, chunki uning o'zidan kichik bo'luvchilari 1, 2, 3 ga teng va 6=1+2+3, 8 soni esa mukammal son emas, chunki 8≠1+2+4.

for 113. Natural N soni berilgan. N sonidan kichik barcha tub sonlarni aniqlang.

for 114. Natural N son berilgan. N dan kichik barcha tub sonlar ichidan oxirgi raqami 3 ga tenglarining yig'indisini hisoblash dasturini tuzing.

while 115. Natural N soni berilgan. N sonini tub sonlar ko'paytmasi ko'rinishida tasvirlovchi dastur tuzing, agar tub bo'luvchilari bo'lmasa, bu haqida ma'lumot chiqaring. Masalan: N=384 bo'lsa, javob "384=2*2*2*2*2*2*2*3".

Yechim. Avval aytilganlarga asosan shunday xulosa chiqara olamiz: N soni 1 dan katta bo'lgani uchun hech bo'lmaganda bitta tub bo'luvchisi bor. Shuning uchun quyidagi loyiha asosida amallar bajaramiz.

1) Ekranga N soni va tenglik belgisi chiqariladi;

2) N sonining eng kichik tub bo'luvchisi Tb aniqlanadi va ekranga chiqariladi;

3) N soni Tb ga necha marta bo'linsa (ya'ni: N%Tb==0 bo'lsa), shuncha marta ekranga Tb va "*" belgisi chiqariladi va N=N/Tb hisoblanadi;

4) Agar N>1 bo'lsa, 2) bandga o'tiladi;

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  long long n,k,bs=1,tb=1;`
`  cout<<"N= ";cin>>n;`
`  if(n<2)cout<<"Tub bo'luvchilari yo'q!";`
`  else  {cout<<"N=";`
`  while(n>1){`
`  while(bs!=2||n%tb!=0) {`
`  tb++;bs=0;`
`  for(k=1;k<=tb;k++)`
`  if(tb%k==0)bs++; }`
`  while(n%tb==0){cout<<tb<<"*";n=n/tb; }`
`  }`
`  cout<<'\b'<<' ';}`
`  return 0;`
`}`
	N= 1002
	N=2*3*167

5) Oxiriga yozilgan "*" belgisi o'chiriladi.

while 116. Natural N soni berilgan. N!=1·2·3·…·N soni nechta nol bilan tugashini aniqlang.

Yechim. Masalani yechish uchun N! ni hisoblash va qoldiqli bo'lish yordamida 0 raqamlari sonini hisoblash mumkin. Lekin ko'paytmaning tez kattalashib ketishi sababli aniqlik nuqtayi nazaridan N soni sifatida (`unsigned long long` turda ham) 21 dan kichik sonlarni ko'rish mumkin, xolos. Lekin quyidagi usul eng samarador usul hisoblanadi.

Matematika kursidan ma'lumki, ko'paytmaning nechta 0 bilan tugashi undagi 2 va 5 sonlar soniga bog'liq. N! ko'paytuvchilari ichida 2 sonlari soni 5 sonlari soniga nisbatan ko'p, shu sababli faktorial tugaydigan 0 raqamlari soni shu ko'paytmada qatnashgan 5 sonlari soniga teng. Demak, faktorial tugaydigan nollar soni beshlar soniga teng ekan. Ko'paytmada qatnashgan 5 sonlari soni quyidagi formula bilan hisoblanadi:

$$\text{Beshlar\_soni} = \left[\frac{N}{5^1}\right] + \left[\frac{N}{5^2}\right] + \left[\frac{N}{5^3}\right] + \cdots$$

Bu yig'indidagi oxirgi hadlar har qanday N soni uchun 5 ning qaysidir darajasida 0 ga aylanadi.

Dastur while	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  long long n,nollar_soni=0,b=5;`
`  cout<<"N= ";cin>>n;`
`  while(n/b>0){`
`  nollar_soni+=n/b;`
`  b=b*5;  }`
`  cout<<nollar_soni<<" ta nol";`
`  return 0;`
`}`
	N= 1902
	474 ta nol

while 117. B, M, A natural son berilgan. Ketma-ketlik $Y_1=B$; $Y_k=\sqrt{M}+A\cdot Y_{k-1}$, k=2,3,… qonuniyat asosida hosil qilinadi. Ketma-ketlikning B·M·A sonidan kichik barcha hadlarini chop etuvchi dastur tuzing.

while 118. Natural N, M va K sonlarining eng katta umumiy bo'luvchisini (EKUB) aniqlash dasturini tuzing.

while 119. Natural N, M va K sonlarining eng kichik umumiy karralisini (EKUK) aniqlash dasturini tuzing.

while 120. Natural N soni berilgan. N sonining barcha tub bo'luvchilarini aniqlang.

while 121. Natural A va B sonlar berilgan (A≤B). A≤P≤B shartni qanoatlantiradigan barcha P tub sonlarni aniqlovchi dastur tuzing.

while 122. (1000, 9999) intervaldan shunday tub sonlarni topingki, ularning birinchi va ikkinchi raqamlari yig'indisi uchinchi va to'rtinchi raqamlari yig'indisiga teng bo'lsin.

while 123. Natural A va B sonlari berilgan. Tomonlari A va B ga teng to'g'ri to'rtburchakdan eng katta tomonli kvadrat kesib olindi. Hosil bo'lgan to'g'ri to'rtburchakdan yana eng katta tomonli kvadrat kesib olindi. Agar shu jarayon davom ettirilsa, u holda kesib olingan kvadratlar sonini aniqlang.

Yechim. Masala mazmunidan shart bo'yicha takrorlash operatorini qo'llash kerakligini tushunish qiyin emas. Masaladagi "eng katta tomonli kvadrat kesib olish" iborasining mazmuni agar A>B bo'lsa, tomoni B ga teng kvadrat kesib olish bilan (kesishdan qolgan tomon uzunligi A=A–B), aks holda tomoni A ga teng kvadrat kesib olish (kesishdan qolgan tomon uzunligi B=B–A) bilan teng kuchli. Agar A=B bo'lsa, u holda B=B–A=0 bo'lib, eng so'nggi kvadrat kesib olinadi. Demak, While operatorida A=B bo'lgandagina tugaydi.

Dastur while	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  int a, b, s=0;`
`  cout<<"A= ";cin>>a;`
`  cout<<"B= ";cin>>b;`
`  while(a!=b){s++;`
`  if(a>b)a=a-b;else b=b-a;}`
`  s++;`
`  cout<<"Javob "<<s;`
`  return 0;`
`}`
	A= 21
	B= 19
	Javob 12

Natijaning to'g'riligini tekshiramiz. A=21 va B=19 bo'lgani uchun 19x19 o'lchamli 1-kvadrat kesib olinganda A=2 va B=19 bo'ladi. Endi 2x2 o'lchamli kvadratlar soni 9 ta va A=2 va B=1 bo'ladi. Qolgan qismdan 2 ta 1x1 o'lchamli kvadrat kesib olamiz, xolos.

while 124. Savatda bir qancha olma bor edi. Savatdan 2 donalab, 3 donalab, 4 donalab, 5 donalab, 6 donalab olma olinsa, har safar 1 dona olma ortib qolaverdi. 7 donalab olinganda savatda olma qolmadi. Savatda dastlab eng kamida nechta olma bo'lganini aniqlash dasturini tuzing.

Yo'llanma. Dasturchilar tilida, masala talabiga ko'ra shunday eng kichik natural A soni topilishi kerakki, quyidagi shartlar bajarilishi zarur:

A%2=1, A%3=1, A%4=1, A%5=1, A%6=1, A%7=0.

while 125. Natural N soni berilgan. Agar $y=\sqrt{5+\sqrt{5+\sqrt{5+\sqrt{...\sqrt{5}}}}}$ ifodada ichma-ich N ta $\sqrt{5}$ qatnashgan bo'lsa, u holda y ning qiymatini aniqlang.

Yechim. K o'zgaruvchi ichma-ich joylashgan ildizlar sonini sanasin. U holda K ning qiymatiga mos quyidagi rekkurent ifodalarni hosil qilamiz:

K=1 da y_1:=sqrtl(5);

K=2 da y_2:=sqrtl(5+sqrtl(5))=sqrtl(5+y_1);

K=3 da y_3:=sqrtl(5+sqrtl(5+sqrtl(5)))=sqrtl(5+y_2);

...

K=N da y_N=sqrtl(5+y_{N-1});

Dasturini mustaqil tuzing!

while 126. Natural N soni berilgan. Agar y=sin(sin(sin(...sin((x))...))) ifodada N ta sin(x) funksiyasi qatnashgan bo'lsa, y ning qiymatini aniqlang.

while 127. Natural N soni berilgan. Agar y=cos(1+cos(2+cos(3...+cos(K...+ cos(N))...))) ifodada N ta cos(x) funksiyasi qatnashgan bo'lsa, y ning qiymatini aniqlang.

while 128. Natural N soni berilgan. Agar y=a/(b+a/(b+(...+a/b)...)) ifodada haqiqiy sonni bo'lish amali N marta qatnashgan bo'lsa, y ning qiymatini aniqlang.

while 129. Natural N soni berilgan. Agar $y=\sqrt{2+\sqrt{4+\sqrt{...+\sqrt{2\cdot N}}}}$ bo'lsa, y ning qiymatini aniqlang.

while 130. P soni polindromlik xususiyatiga egaligini tekshiring. Son **polindrom** deb ataladi, agar sonni chapdan ham, o'ngdan ham bir xil o'qilsa. Masalan, 5, 44 , 727, 19391-polindrom sonlar (yo'llanma: P sonining raqamlarini ajratib o'nlikdagi sonlarni standart ko'rinishi yordamida raqamlari teskari tartibda yozilgan TP son hosil qilinadi, so'ng P va TP sonlar tengligi tekshiriladi).

while 131. Natural N soni berilgan. N sonidan kichik polindrom sonlarni ekranga chiqaring.

while 132. Klaviaturadan butun sonlar ketma-ket kiritib boriladi. Ketma-ketlikka tegishli bo'lmagan 0 soni kiritilgach sonlarni kiritish yakunlanadi. Kiritilgan sonlardan ketma-ket kelgan juft sonlarning eng ko'p sonini aniqlang. Masalan, 1 2 3 6 4 5 0 kiritilganda ketma-ket kelgan juft sonlarning eng ko'p soni 2 ga teng, chunki 2 ta juft son 6 va 4 ketma-ket kelgan. Agar juft sonlar yo'q bo'lsa, u holda "–1" sonini chiqaring.

while 133. Klaviaturadan butun sonlar ketma-ket kiritib boriladi. Ketma-ketlikka tegishli bo'lmagan 0 soni kiritilgach sonlarni kiritish yakunlanadi. Kiritilgan sonlardan ketma-ket kelgan manfiy sonlarning eng ko'p sonini aniqlang. Masalan, 1 –2 2 –3 6 4 5 0 kiritilganda ketma-ket kelgan manfiy sonlarning eng ko'p soni 1 ga teng, chunki ketma-ket 1 ta donadan manfiy sonlar –2 yoki –3 bor. Agar manfiy sonlar yo'q bo'lsa, u holda "–1" sonini chiqaring.

while 134. Klaviaturadan butun sonlar ketma-ket kiritib boriladi. Ketma-ketlikka tegishli bo'lmagan 0 soni kiritilgach sonlarni kiritish yakunlanadi. Kiritilgan sonlardan ketma-ket kelgan bir xil sonlarning eng ko'p sonini aniqlang. Masalan, 1 2 2 6 –1 –1 0 kiritilganda ketma-ket kelgan bir xil sonlarning eng ko'p soni 2 ga teng, ya'ni 2 ta 2 soni yoki 2 ta –1 soni ketma-ket kelgan.

= D =

Takrorlanuvchi 135. Butun w, h, u1, d1, u2, d2 ($0 \le w, u1, u2 \le 100$, $1 \le h \le 100$, $0 \le d1, d2 \le h$) sonlari berilgan. Parashyutchi h metr balandlikdan w tezlik bilan sakradi. Yerning tortish kuchi ta'siri sababli har bir metrda parashyutchining tezligiga shu balandlik qiymati qo'shiladi. Lekin d1 va d2 balandliklarda yuqoriga ko'taruvchi shamol oqimi bo'lib, parashyutchi shu balandliklardan o'tganda tezligi, mos ravishda, u1 va u2 qiymatlarga kamayadi. Tezlik hech manfiy bo'lib qolmaydi, bu holda 0 ga aylanadi. Parashyutchining yerga tekkandagi, ya'ni 0 balandlikdagi tezligini aniqlang.

K:	4 3 1 1 1 2	4 3 9 2 0 1	41 2 1 1 67 2	87 2 10 2 76 1	94 3 71 3 12 2	30 2 88 1 2 2	8 2 29 1 23 2	85 3 47 1 92 3	34 5 82 2 52 5	19 7 14 7 28 3	94 30 83 11 85 27	19 50 36 15 90 16
CH:	8	1	0	4	17	0	0	0	1	5	391	1168

Takrorlanuvchi 136. Butun T, a va b ($1 \le T \le 1000$, $1 \le a < b \le 998244353$) sonlari berilgan. Berilgan T ta a va b juftligi uchun shunday x va y juftligini aniqlangki, $a \le x, y \le b$, $x \ne y$ va y soni x ga bo'linsin. Har bir testda yechim mavjud. Yechimlar ko'p bo'lsa ixtiyoriy bittasini chiqaring. Masalan:

K:	3 1 10 3 14 1 10	3 1 2 1 3 1 4	3 6969 696969 6969 696969 6969 696969	2 51949127 103898254 194778305 389556610	3 316383223 632766446 460123701 920247402 332067337 664134674
CH:	1 7 3 9 5 10	1 2 1 2 1 2	6969 13938 6969 13938 6969 13938	51949127 103898254 194778305 389556610	316383223 632766446 460123701 920247402 332067337 664134674

Takrorlanuvchi 137. Butun N, y, m, b (1≤ N, y ≤100, 2≤ m ≤100, 3≤ b ≤100) sonlari berilgan. Qopda y ta yong'oq, m ta mayiz va b ta bodom bor. Shoh salatini tayyorlash uchun bu masalliqlardan maxsus nisbatda ishlatiladi. Shoh salatida mayizlar soni yong'oqlar sonidan roppa-rosa 1 taga ko'p bo'lishi, bodomlar soni esa mayizlar sonidan roppa-rosa 1 taga ko'p bo'lishi kerak. N ta uchlikning har biri uchun shu nisbatda Shoh salati tayyorlansa, eng ko'pi bilan qancha yong'oq, mayiz va bodom birgalikda ishlatilishini aniqlang.

K:	2 8 13 9 13 3 6	3 3 8 20 1 2 3 100 100 100	1 9 5 5	3 88 89 7 50 80 70 80 81 82	2 100 98 99 65 69 67	1 55 56 76	3 78 3 79 4 49 50 40 50 50
CH:	24 9	12 6 297	12	18 153 243	294 198	168	9 15 123

Takrorlanuvchi 138. Butun T va N (1≤ T, N ≤100) soni berilgan. Berilgan T ta N sonining har biri uchun quyidagi shartlarni qanoatlanturuvchi x va y sonlarni aniqlang: 1) 1≤x, y≤N; 2) x soni y soniga bo'linadi; c) x·y>N; d) $\frac{x}{y} < N$. Agar yechimlar ko'p bo'lsa, ixtiyoriysini, aks holda, ya'ni bunday sonlar topilmasa –1 chiqaring. Masalan:

K:	2 10 1	3 5 8 20	1 55	3 89 100 1
CH:	6 3 –1	4 2 5 5 20 2	54 2	88 2 100 2 –1

Takrorlanuvchi 139. Natural Q, A va B (1≤ T ≤100, 1≤ A ≤ B ≤10^9) sonlari berilgan. Q ta testning har birida –1, 2, –3, 4, –5, … ishora almashtiruvchi ketma-ketlikning A-hadidan B-hadigacha bo'lgan (ular ham qo'shiladi) sonlar yig'indisini aniqlang.

K:	5 1 3 2 5 5 5 4 4 2 3	5 1 1 1000000000 1000000000 500000000 500000000 1 1000000000 3923 1000000000	4 617758920 825919887 775957146 950878973 404173573 553845184 25837072 795166931	4 449347852 741090797 541919355 938202256 442458747 465872502 463474612 789217678
CH:	-2 -2 -5 4 -1	-1 1000000000 500000000 500000000 499998039	-104080484 -87460914 74835806 -384664930	-145871473 198141451 11706878 626346145

Takrorlanuvchi 140. Natural N va k ($1 \le N \le 10^6$, $2 \le k \le 1000$) sonlari berilgan. Agar div – butun bo'lishni, mod qoldiq hisoblashni bildirsa, quyidagi tenglamaning eng kichik yechimi x ni aniqlang: (x div k)·(x mod k)=N.

K:	6 3	1 2	4 6	1000000 1000	999983 1000	101 10	524288 2	123456 789
CH:	11	3	10	1250800	999983001	1011	1048577	152131

Takrorlanuvchi 141. Natural A dan B gacha bo'lgan (A va B ham kiradi) sonlar berilgan bo'lib, doimo A<B, $(B-A+1) \le 3 \cdot 10^5$ va (B–A) toq. Bu sonlarni $\frac{B-A+1}{2}$ juftlikka shunday ajratingki, har bir juftlikning eng katta umumiy bo'luvchisi 1 bo'lsin hamda har bir son juftliklarda bir marta qatnashsin. Hosil qilingan barcha juftliklarni yangi satrlarda orachiq bilan ajratib, aks holda, ya'ni agar bunday sonlar yo'q bo'lsa –1 chiqaring. Agar yechimlar ko'p bo'lsa, ixtiyoriy bittasini chiqaring.

K:	1 8	2 3	4 9	3 12	3 6	2 7
CH:	YES 1 2 3 4 5 6 7 8	YES 2 3	YES 4 5 6 7 8 9	YES 3 4 5 6 7 8 9 10 11 12	YES 3 4 5 6	YES 2 3 4 5 6 7

Takrorlanuvchi 142. Natural W, H va k ($3 \le W, H \le 100$, $1 \le k \le \left[\frac{\min(W,H)+1}{4}\right]$) sonlari berilgan. O'lchami WxH bo'lgan jadval berilgan. Avval jadvalning tashqi halqa kataklari, ya'ni tashqi chegarasidagi barcha kataklar * belgisi bilan bilan to'ldiriladi. So'ngra bitta halqa kataklari tashlab o'tilib keyingi halqa kataklari * belgisi bilan to'ldiriladi, va shu kabi davom ettiriladi. Agar k ta halqaga * belgilari joylashtirilsa, u holda jadvaldagi * belgilari sonini aniqlang. Quyida 5x6 va k=2 ga mos rasm keltirilgan:

*	*	*	*	*	*
*					*
*		*	*		*
*					*
*	*	*	*	*	*

Bu misolda * belgilari soni 20 ta.

K:	5 6 2	3 3 1	7 9 1	7 9 2	18 26 3	63 34 8	100 100 25	11 12 2	100 8 2
CH:	20	8	28	40	204	1072	5100	68	408

Takrorlanuvchi 143. Natural N va k ($2 \le N \le 10^9$, $1 \le k \le 50$) sonlari berilgan. Ali N sonini quyidagi qonuniyat asosida o'zgartiradi:

- agar sonning oxirgi raqami 0 dan farqli bo'lsa, u holda sonni 1 taga kamaytiradi;
- agar sonning oxirgi raqami 0 ga teng bo'lsa, u holda sonni 10 ga bo'ladi.

Agar shu amallar k marta bajarilgan bo'lsa, u holda hosil bo'lgan sonni aniqlang.

K:	512 4	1000000000 9	131203 11	999999999 50	999999999 49	900000000 16
CH:	50	1	12	9999	99990	1

Takrorlanuvchi 144. Natural N ($2\leq N \leq 10^5$) soni berilgan. Ali rahbarlik qilayotgan kompaniyada N ta xizmatchi bor. Ali yaxshi ishlagan xizmatchilardan kichik rahbarlar tayinlamoqchi bo'ldi. Ali quyidagicha taqsimlashni xohlaydi:

- barcha kichik rahbarlarning qo'l ostidagi xizmatchilar soni bir xil bo'lishi shart;
- kichik rahbar bo'lmagan har bir xizmatchining faqat bitta kichik rahbari bo'ladi;
- birorta ham kichik rahbarning boshqa kichik rahbari yo'q.

Shu talablar asosida Ali necha xil usulda kichik rahbarlarni tayinlay olishini aniqlang.

K:	2	10	3	4	100000	1024	99999	10007	65536	30030
CH:	1	3	1	2	35	10	11	1	16	63

Takrorlanuvchi 145. Natural N ($2\leq N \leq 10^6$) soni berilgan. Ikkita turli A va B sonlar do'st sonlar deyiladi, agar A sonining o'zidan kichik bo'luvchilari yig'indisi B songa, B sonining o'zidan kichik bo'luvchilari yig'indisi A songa teng bo'lsa. N sonidan kichik barcha do'st sonlar juftligini tartib raqami bilan yangi satrda chiqaring. Agar do'st sonlar topilmasa – 1 chiqaring.

K:	200	201	3000	10000	5000	11000
CH:	–1	1. 200 284	1. 220 284 2. 1184 1210 3. 2620 2924	1. 220 284 2. 1184 1210 3. 2620 2924 4. 5020 5564 5. 6232 6368	1. 220 284 2. 1184 1210 3. 2620 2924	1. 220 284 2. 1184 1210 3. 2620 2924 4. 5020 5564 5. 6232 6368 6. 10744 10856

Takrorlanuvchi 146. Natural N ($3\leq N \leq 1000$) soni berilgan. Shunday a va b sonlarni aniqlangki:

- a+b=n bo'lsin;
- $\frac{a}{b}$ eng katta qiymatli qisqarmas kasr bo'lsin.

Shartni qanoatlantiruvchi a va b sonlarini orachiq bilan ajratib chiqaring.

Yo'llanma. Matematika, EKUB, realizatsiya

K:	3	4	12	34	13	10	69	1000	999	998
CH:	1 2	1 3	5 7	15 19	6 7	3 7	34 35	499 501	499 500	497 501

Takrorlanuvchi 147. Butun A va B ($1 \le A, B \le 10^9, \min(A, B) \le 12$) sonlari berilgan. EKUB(A!, B!) ni aniqlang (EKUB – eng katta umumiy bo'luvchi, k!=1·2·…·k, ya'ni k faktorial).

K:	4 3	10 399603090	6 973151934	11 562314608	1 1	9 7	4 999832660
CH:	6	3628800	720	39916800	1	5040	24

Takrorlanuvchi 148. Butun c, v0, v1, a va s ($1 \le c \le 1000$, $0 \le s < v0 \le v1 \le 1000$, $0 \le a \le 1000$) sonlari berilgan. Ali c sahifali kitobni o'qishni boshladi. Alining o'qish tezligi 1-kun v0 sahifa bo'lib, keyingi har bir kun avvalgi kunga qaraganda a sahifa ko'p o'qiydi. Lekin Ali qancha harakat qilmasin kuniga v1 sahifadan ortiq o'qiy olmaydi. Ali kitob mazmunini esdan chiqarmaslik uchun ikkinchi kundan boshlab oxirgi o'qigan sahifalardan s tasini qayta o'qib, keyin yangi sahifalarni o'qishni davom ettiradi. Ali birinchi marta kitobning oxirgi betini o'qib bo'lishi bilan o'qishni tugatadi. Ali kitobni to'liq o'qib bo'lishi uchun sarflaydigan kunlar sonini aniqlang.

K:	5 5 10 5 4	12 4 12 4 1	1000 999 1000 1000 998	1000 2 2 5 1	737 41 74 12 11
CH:	1	3	2	999	13

Takrorlanuvchi 149. Natural N ($2 \le N \le 100\,000$) soni berilgan. N sonini maksimal sondagi tub sonlar yig'indisi ko'rinishida tasvirlang. Javobning birinchi satrida qo'shiluvchilar soni, ikkinchi satrida qo'shiluvchi tub sonlarni orachiq bilan ajratib istalgan tartibda chiqaring.

K:	5	6	2	3	13	23	19
CH:	2 2 3	3 2 2 2	1 2	1 3	6 2 2 2 2 2 3	11 2 2 2 2 2 2 2 2 2 2 3	9 3 2 2 2 2 2 2 2 2

Takrorlanuvchi 150. Natural N va K ($1 \le N \le 10$, $1 \le K \le 240$) sonlari berilgan. Aliga soat 10:00 da boshlangan musobaqada dasturlash uchun N ta masala berildi. Masalalar soddadan murakkabga qarab tartiblangan bo'lgani uchun u 1-masalaga 5 minut, 2-masalaga 10 minut, m-masalaga 5·m minut vaqt sarflaydi. Lekin Ali shu kuni uyga soat 14:00 ga yetib borishi shart bo'lib, musobaqa bo'layotgan binodan uyga yetib olishga K minut vaqt ketadi. Ali uyiga vaqtida yetib borishi uchun eng ko'pi bilan nechta masala yecha olishini aniqlang.

K:	3 222	10 135	7 1	10 136	1 240	9 235	5 225	4 210	10 166
CH:	2	6	7	5	0	1	2	3	4

Takrorlanuvchi 151. Natural N ($2 \le N \le 10^9$) soni berilgan. 1, 2, 3, …, N ketma-ketlik elementlarini 2 ta A va B to'plamga shunday ajratingki, A to'plam elementlari yig'indisi sumA va B to'plam elementlari yig'indisi sumB orasidagi farq eng kichik bo'lsin. Avval yig'indilar orasidagi farqning absolyut qiymatini, keyin yangi satrdan "1:" dan keyin A

to‘plam elementlarini ixtiyoriy tartibda, so‘ng yangi satrdan “2:” dan keyin B to‘plam elementlarini ixtiyoriy tartibda chiqaring. Masalan:

K:	2	3	7	10	12	15
CH:	1 1: 2 2: 1	0 1: 3 2:	0 1: 7 4 3 2: 6 5 2 1	1 1: 10 7 6 3 2 2: 9 8 5 4 1	0 1: 12 9 8 5 4 1 2: 11 10 7 6 3 2	0 1: 15 12 11 8 7 4 3 2: 14 13 10 9 6 5 2 1

Takrorlanuvchi 152. Natural N ($1 \le N \le 10^6$) soni berilgan. Ali kompyuteri uchun monitor sotib olmoqchi. Uning talabi quyidagicha:

- monitor N ta pikseldan iborat bo‘lishi shart;
- satrlar soni a ustunlar soni b dan katta emas, ya’ni a≤b;
- ustun va satr orasidagi farq b–a qiymati minimal bo‘lishi kerak.

Aliga u xohlagandek bo‘lgan monitor o‘lchamini aniqlang.

K:	8	64	5	999999	716539	1	2	101	100001
CH:	2 4	8 8	1 5	999 1001	97 7387	1 1	1 2	1 101	11 9091

Takrorlanuvchi 153. Butun $k2, k3, k5, k6$ ($0 \le k2, k3, k5, k6 \le 5 \cdot 10^6$) sonlari berilgan. Alida 2 raqami yozilgan k2 ta, 3 raqami yozilgan k3 ta, 5 raqami yozilgan k5 ta va 6 raqami yozilgan k6 ta kubik bor edi. Ali mos kubiklarni yonma-yon joylashtirib, o‘zi yoqtirgan 32 va 256 sonlarini tuzmoqchi. Shu bilan birga, tuzilgan sonlar yig‘indisi maksimal bo‘lishi kerak. Har bir kubik faqat bitta sondagina qatnashishi mumkin va ba’zi kubiklar umuman ishlatilmay qolishi ham mumkin. Masalan, k2=2, k3=1, k5=1, k6=2 bo‘lsa, u holda 1 ta 32 va 1 ta 256 sonlarini hosil qilish mumkin, maksimal yig‘indisi 288 bo‘ladi, 1 ta 6 raqami yozilgan kubik ishlatilmay qoladi. Ali hosil qilishi mumkin bo‘lgan eng katta yig‘indini aniqlang.

K:	5 1 3 4	10 2 1 5	4 2 7 2	9557 5242 1190 7734	1480320 1969946 1158387 3940412
CH:	800	320	576	472384	306848928

Takrorlanuvchi 154. Natural K va r ($1 \le K \le 1000$, $1 \le r \le 9$) sonlari berilgan. Alining cho‘ntagida faqat yaxlit 10 so‘mlik tangalar juda ko‘p bo‘lib, faqat 1 dona r qiymatli tanga bor edi. Ali do‘kondan 1 dona daftar sotib olmoqchi bo‘ldi. Daftarning narxi K so‘m ekan. Lekin do‘konda hali savdo qilinmagani uchun unga qaytim berishning iloji yo‘q ekan. Ali cho‘ntagidagi tangalardan foydalanib qaytimsiz sotib oladigan minimal sondagi daftarlar sonini aniqlang.

K:	117 3	237 7	15 2	1 9	1000 3	999 8	105 6	403 9	23 4
CH:	9	1	2	9	1	2	2	3	8

Takrorlanuvchi 155. Natural N ($1 \le N \le 100$) soni berilgan. Klaviaturadan N ta manfiymas butun son kiritiladi. Bu sonlar N ta savatga har biriga solingan konfetlar sonidir. Savatlardan konfetlarni olmasdan ularni ichidagi konfetlar sonini tenglashtirish zarur. Buning uchun hammasi bo'lib kerak bo'ladigan konfetlar sonini aniqlang.

K:	5 0 1 2 3 4	5 1 1 0 1 1	3 1 3 1	1 12	2 32576 550340	3 910648 542843 537125	4 751720 572344 569387 893618	1 1000000	5 0 0 0 0 0
CH:	10	1	4	0	517764	741328	787403	0	0

Takrorlanuvchi 156. Natural N ($1 \le N \le 100000$) soni berilgan. Alining tug'ilgan kuniga N ta kubik sovg'a qilishdi. Kengligi k ta kubik to'plami deganda boshqalaridan ajratib yonma-yon joylashtirilgan k ta kubikni tushunamiz. Ali zerikkanidan bir o'yin o'ylab topdi. O'yin jarayoni shunday: avval u stolga 1 ta kubikni qo'yadi va qolganlarini quyidagicha joylashtiradi:

- keyingi kubiklar avvalgilaridan o'ngda joylashtiriladi;
- agar stolda bittadan ortiq kubik bo'lsa va o'ngdagi oxirgi ikkitasining kengligi k ta bo'lsa, ularni birlashtirib (ba'zi kubiklarni ustma-ust qo'yib) kengligi (k+1) ga teng kubik to'plami hosil qiladi.

Shu jarayon oxirida hosil bo'lgan har bir kubiklar to'plamining kengligini aniqlang. Masalan, N=4 bo'lsa, o'yin jarayoni quyidagicha kechadi:

1. 1
2. 1 1 → 2
3. 2 1
4. 2 1 1 → 2 2 → 3

K:	1	2	3	8	100000	12345	32	12705	8192
CH:	1	2	2 1	4	17 16 11 10 8 6	14 13 6 5 4 1	6	14 13 9 8 6 1	14

Takrorlanuvchi 157. Natural N ($1 \le N \le 1000000000$) soni berilgan. Quyidagi ketma-ketlikning N ta hadi yig'indisini aniqlang:

–1, –2, 3, –4, 5, 6, 7, –8, …

Ya'ni bu ketma-ketlikda 2 sonining darajalariga teng hadlar minus ishoralidir.

K:	4	10	15	17	1000000000	901	3	712	836912
CH:	–4	25	90	91	499999998352516354	404305	0	251782	350209169178

Takrorlanuvchi 158. Natural N ($1 \le N \le 10^9$) soni berilgan. Ali basseynda zuluklarni ko'paytirishni yaxshi ko'radi. Avvaliga basseyn bo'sh edi. Ali har kuni ertalab basseynga

xohlagancha zuluk tashlashi mumkin. Har kuni tunda har bir zuluk ikkiga bo'linadi. Ali qachondir basseynda roppa-rosa N dona zuluk bo'lishini xohlaydi. Agar zuluklar soni N ta bo'lgan bo'lsa, u holda Ali hammasi bo'lib basseynga tashlagan eng kam zuluklar sonini aniqlang.

K:	5	8	536870911	1	343000816	559980448	697681824	954746654
CH:	2	1	29	1	14	12	14	15

Takrorlanuvchi 159. Natural N ($1 \le N \le 20$) soni berilgan. O'lchami NxN jadval quyidagicha hosil qilinadi:

- birinchi ustunda va birinchi satrda barcha elementlar 1 ga teng;
- jadvalning qolgan har bir elementi o'zidan yuqoridagi va o'zidan chapdagi elementlar yig'indisiga teng.

Shu qoida bo'yicha hosil qilingan jadvaldagi maksimal elementni aniqlang.

Yo'llanma. Paskal uchburchagi yoki binomial koeffitsiyentlar.

K:	1	2	10	5	15	11	13	20
CH:	1	2	48620	70	40116600	184756	2704156	35345263800

Takrorlanuvchi 160. Natural N ($1 \le N \le 10000$) soni berilgan. Valiga N ta kubik sovg'a qilishdi. Valining do'sti Ali bu kubiklardan quyidagicha usulda piramida hosil qilmoqchi:

- piramidaning uchida 1 ta kubik;
- ikkinchi qavatda 1+2=3 ta kubik;
- uchinchi qavatda 1+2+3=6 ta kubik;
- va hokazo, k-qavatda 1+2+…+(k–1)+k ta kubik.

Shu usul asosida hosil qilingan piramidaning balandligini aniqlang.

K:	1	2	4	6	15	707	909	999	9879	9999	10000
CH:	1	1	2	3	3	15	16	17	37	38	38

Takrorlanuvchi 161. Natural va juft N ($2 \le N \le 100$) soni berilgan. Alining N ta do'sti bo'lib, u do'stlariga N^2 dona konfet sotib oldi. Konfetlar esa quyidagicha usulda xaltalarga solingan edi: 1-xaltada 1 ta konfet, 2-xaltada 2 ta konfet, …, k-xaltada k ta konfet …. Alining har bir do'stiga N tadan xaltani shunday taqsimlangki, barcha do'stlariga bir xil sondagi konfet tegsin. Sonlarni ixtiyoriy tartibda chiqarish mumkin.

K:	2	4	6	8
CH:	1 4 2 3	1 16 2 15 3 14 4 13 5 12 6 11 7 10 8 9	1 36 2 35 3 34 4 33 5 32 6 31 7 30 8 29 9 28 10 27 11 26 12 25 13 24 14 23 15 22 16 21 17 20 18 19	1 64 2 63 3 62 4 61 5 60 6 59 7 58 8 57 9 56 10 55 11 54 12 53 13 52 14 51 15 50 16 49 17 48 18 47 19 46 20 45 21 44 22 43 23 42 24 41 25 40 26 39 27 38 28 37 29 36 30 35 31 34 32 33

Takrorlanuvchi 162. Natural N va M ($1 \le N, M \le 100, M \le N$) sonlari berilgan. Ali M ta do'sti uchun bor puliga N ta konfet sotib oldi. U konfetlarni do'stlariga teng miqdorda taqsimlashga, agar bunday taqsimlashning iloji bo'lmasa, taqsimlangan eng ko'p sondagi konfet bilan eng kam sondagi konfet orasidagi farq kichik bo'lishini xohladi. Uning xohishiga mos taqsimlash dasturini tuzing. Sonlarni chiqarish tartibining ahamiyati muhim emas. Masalan:

K:	15 4	12 3	18 7	20 6	21 11
CH:	3 4 4 4	4 4 4	3 3 3 3 2 2 2	4 4 3 3 3 3	1 2 2 2 2 2 2 2 2 2 2

Takrorlanuvchi 163. Natural N ($1000 \le N \le 9000$) soni berilgan. Berilgan N-yildan keyingi barcha raqamlari turlicha bo'lgan eng kichik yilni aniqlang.

K:	1987	2013	2018	2019	2987	2988	3012	9000	8800
CH:	2013	2014	2019	2031	3012	3012	3014	9012	8901

Takrorlanuvchi 164. Natural N ($1 \le N \le 2000$) soni berilgan. Qulf 1 dan N gacha tartib raqami berilgan N ta tugmadan iborat bo'lib, to'g'ri kodni kiritish uchun tugmalar kodga mos tartibda bosilishi kerak. Kod to'g'ri terilayotganda bosilgan tugma bosilgancha qoladi, agar biror tugma xato bosilsa, xato bosilgan tugma bilan birga avval to'g'ri bosilgan barcha tugmalar o'z joyiga qaytadi. Aql bilan o'ylab tugmalar bosilganda eng ko'pi bilan nechta tugma bosilishini aniqlang.

Masalan, tugmalar 3 ta bo'lib {3-tugma; 2-tugma; 1-tugma } tartibda bosilishi kerak bo'lsin. Tugmalar ketma-ketligini bosib kodni aniqlash jarayonini tahlil qilamiz:

- avval 1-tugma yoki 2-tugma bosilsa qaytib chiqadi, 3-tugma bosilsa qaytib chiqmaydi, demak, eng ko'pi bilan 2 marta bosishda birinchi bo'lib 3-tugma bosilishi kerakligi aniqlandi;
- 3-tugmani bosgach (chunki avval 1-tugma yoki 2-tugmani bosgan bo'lishimiz mumkin), 1-tugma bosilsa, barcha tugmalar qaytib chiqadi yoki 2-tugma bosilsa qaytib chiqmaydi, lekin ikkinchi bo'lib 2-tugma bosilishi kerakligini eng ko'pi bilan 2 ta tugma bosilganda aniqlandi;
- 2 ta tugma aniqlangach oxirgi 3-tugma o'z-o'zidan ma'lum bo'ladi;
- endi 3 marta (3-tugmani bosgach 1-tugma bosilgandagi barcha tugmalar qaytib chiqqan holda) tugmalar bosilgach qulf ochiladi, ya'ni tugmalar hammasi bo'lib 2+2+3=7 marta bosiladi.

K:	2	3	4	10	2000	1747	889	1999
CH:	3	7	14	175	1333335000	888644743	117099969	1331335999

Takrorlanuvchi 165. Natural N ($1 \le N \le 100$) soni berilgan. N ta N dan katta bo'lmagan turli $p_1, p_2, \ldots, p_N$ sonlarning tartibli ketma-ketligiga **o'rinlashtirish** deb aytiladi. N soni

o‘rinlashtirishning o‘lchami deb atalib, o‘rinlashtirishning k-elementi p_k bilan belgilanadi. Ali ham o‘rinlashtirishni yaxshi ko‘radi, lekin uning uchun sevimlisi shunday o‘rinlashtirishki, uning ixtiyoriy k-elementi uchun $p_{p_k}=k$ va $p_k \neq k$ o‘rinli bo‘ladi. Berilgan N uchun Alining biror sevimli o‘rinlashtirishini chiqaring, agar bunday o‘rinlashtirish mavjud bo‘lmasa –1 chiqaring.

K:	1	2	4	7	20
CH:	-1	2 1	2 1 4 3	-1	2 1 4 3 6 5 8 7 10 9 12 11 14 13 16 15 18 17 20 19

Takrorlanuvchi 166. Natural n va m ($1 \leq n, m \leq 1000$) sonlari berilgan. Quyidagi tenglamalar sistemasini qanoatlantiradigan manfiymas butun (a; b) juftliklar sonini aniqlang:

$$\begin{cases} a^2 + b = n \\ a + b^2 = m \end{cases}.$$

K:	9 3	14 28	4 20	18 198	1 1	227 975	840 780	1000 1000	471 921
CH:	1	1	0	1	2	1	0	0	1

Takrorlanuvchi 167. Natural N ($2 \leq N \leq 10^9$) soni berilgan. Ali N soni ustida quyidagicha amallarni bajarmoqda:

1) N sonini qog‘ozga yozib qo‘yadi;
2) agar $N=a \cdot b$ va $a>1$ bo‘lsa qog‘ozga b ni yozib qo‘yadi, aks holda qog‘ozga 1 ni yozib qo‘yadi va jarayon tugaydi;
3) N=b deb olib 2) bandga qaytiladi.

Demak, a va b sonlarning tanlanishiga qarab qog‘ozga yozilgan sonlar farqlanar ekan. Qog‘ozga yozilgan sonlar yig‘indisi maksimal bo‘ladigan holdagi yig‘indi qiymatini aniqlang.

K:	10	8	4	36	32	49	200	1000000000	999002449	500000000
CH:	16	15	7	67	63	57	381	1998535156	999034057	998535156

Takrorlanuvchi 168. Natural k, l, m, n va d ($1 \leq k, l, m, n \leq 10$, $1 \leq d \leq 10^5$) sonlari berilgan. Alida 1 dan d gacha tartiblangan oq rangli qog‘ozga o‘ralgan d ta konfet bor edi. Ali har k-tartib raqamli konfet qog‘ozini sariq rangga, har l-tartib raqamli konfet qog‘ozini qizil rangga, har m-tartib raqamli konfet qog‘ozini yashil rangga va har n-tartib raqamli konfet qog‘ozini ko‘k rangga bo‘yadi. Bitta konfet qog‘ozi bir necha xil rangga bo‘yalgan bo‘lishi ham mumkin. Oq rangdan farqli rangdagi qog‘ozli konfetlar sonini aniqlang.

K:	1 2 3 4 12	2 3 4 5 24	1 1 1 1 100000	10 9 8 7 6	8 4 4 3 65437	5 5 5 10 55592
CH:	12	17	100000	0	32718	11118

Takrorlanuvchi 169. Natural N ($1 \leq N \leq 10000$) soni berilgan. Sonni go‘zal deymiz, agar son faqat 4 va 7 raqamlaridan iborat bo‘lsa. Masalan, 4, 7, 44, 47 va 74 sonlari go‘zal, 5 va 100 sonlari go‘zal emas. Sonni deyarli go‘zal deymiz, agar son biror go‘zal songa bo‘linsa. Agar berilgan N soni deyarli go‘zal bo‘lsa “YES”, aks holda “NO” chiqaring.

K:	47	100	99	10000	78	107	477	49	298
CH:	YES	YES	NO	YES	NO	NO	YES	YES	NO

Takrorlanuvchi 170. Natural N, a va b ($1 \le N, a, b \le 1000$) sonlari berilgan. Ali bilan Vali N ta konfet uyumi bilan bog'liq shunday o'yin o'ynashmoqda:

- Ali a sonni, Vali b sonni tanlaydi;
- har bir o'yinchi uyumdan o'z soni bilan uyumdagi konfetlar sonining eng katta umumiy bo'luvchisiga teng sondagi konfet oladi;
- birinchi bo'lib Ali yurishni boshlaydi;
- kim yura olmasa, ya'ni uyumda kerakli sondagi konfet qolmasa, yutqazadi.

O'yinda yutgan o'yinchi nomini chiqaring.

K:	3 5 9	1 1 100	23 12 16	95 26 29	73 32 99	100 100 10	42 81 17
CH:	0	1	1	1	1	0	0

Takrorlanuvchi 171. Natural N ($1 \le N \le 10^{18}$) soni berilgan. Raqam go'zal deymiz, agar 4 yoki 7 raqami bo'lsa. Sonni go'zal deymiz, agar u go'zal raqamlardan iborat bo'lsa. Masalan, 4, 7, 44, 47 va 74 sonlari go'zal, 5 va 100 sonlari go'zal emas. Sonni qisman go'zal deymiz, agar sondagi go'zal raqamlar soni go'zal bo'lsa. Agar berilgan N soni qisman go'zal bo'lsa "YES", aks holda "NO" chiqaring.

K:	7747774	40047	1000000000000000000	10000000004744744	7777
CH:	YES	NO	NO	YES	YES

Takrorlanuvchi 172. Natural N va M ($1 \le N \le 50$, $1 \le M \le 10^4$) sonlari berilgan. Ali N ta idishni doira shaklida joylashtirib, 1 dan N gacha tartib raqamlari yozib chiqildi. Vali M ta konfetni quyidagicha taqsimlaydi:

- 1-idishga 1 ta konfet soladi;
- 2-idishga 2 ta konfet soladi;
- ...
- k-idishga k ta konfet soladi;
- N-idishdan keyin navbat 1-idishga o'tadi.

Agar biror qadamda keragicha konfet qolmasa, qolgan konfetni Ali olib ketadi. Ali olib ketadigan konfetlar sonini aniqlang.

K:	4 11	17 107	3 8	46 7262	32 6864	36 6218	50 10000	1 10000
CH:	0	2	1	35	0	14	40	0

Takrorlanuvchi 173. Natural N va M ($1 \le N < M \le 100$) sonlari berilgan. Berilgan N soni tub bo'lib, agar M soni N dan keyingi birinchi tub son bo'lsa "YES", aks holda "NO" chiqaring.

K:	2 3	3 5	7 9	5 11	7 11	23 29	83 97	89 97	89 91
CH:	YES	YES	NO	NO	YES	YES	NO	YES	NO

Takrorlanuvchi 174. Natural N ($2 \le N \le 100$) soni berilgan. Doira shaklida joylashgan N ta bolaga 1 dan N gacha tartib raqami berilgan. Koptok 2-bolada turibdi. U koptokni 1 ta bolani o'tkazib keyingi, ya'ni 4-bolaga oshirdi, 4-bola 2 ta bolani o'tkazib keyingi, ya'ni 7-bolaga oshirdi, 7-bola 3 ta bolani o'tkazib keyingi, ya'ni 11-bolaga oshirdi va hokazo. Koptok oshirishlar N–2 marta bajarildi. Koptok qo'lida bo'lgan bolalar tartib raqamini aniqlang.

K:	3	10	4	8	13
CH:	2 1	2 4 7 1 6 2 9 7 6	2 4 3	2 4 7 3 8 6 5	2 4 7 11 3 9 3 11 7 4 2 1

Takrorlanuvchi 175. Natural N ($2 \le N \le 10^{10}$) soni berilgan. Unga quyidagi algoritm qo'llanadi:

1. agar N=0 bo'lsa, u holda algoritm tugatilsin;
2. N sonining eng kichik tub bo'luvchisi p topilsin;
3. N sonini N–p soniga tenglashtirib 1-qadamga o'tilsin.

Algoritm bajargan ayirishlar sonini aniqlang.

K:	5	4	2	9999999999	473	9999999967	9998200081	6969696
CH:	1	2	1	4999999999	232	1	4999050046	3484848

Takrorlanuvchi 176. Natural N ($1 \le N \le 10^{18}$) soni berilgan. Sonni sehrli deymiz, agar u faqat 1, 14 va 144 sonlarini turli ketma-ketlikda "ulash"dan hosil bo'lgan bo'lsa. Masalan, 14144, 141414 va 1411sonlari sehrli, 1444, 514 va 414 sonlari sehrli emas. Agar berilgan N soni sehrli bo'lsa "YES", aks holda "NO" chiqaring.

K:	114114	1111	441231	144	414	111444	141414141	114414441
CH:	YES	YES	NO	YES	NO	NO	YES	NO

6-BOB. C++ TILIDA MASSIVLAR VA FOYDALANUVCHI FUNKSIYALARI

Bu bobda dasturlash tillarida eng muhim ahamiyatga ega bo'lgan massivlar hamda C++ tilida foydalanuvchi funksiyalarini tashkil etish, o'z-o'ziga murojaat qiluvchi, ya'ni rekursiv funksiyalar hosil qilish haqida so'z boradi. Shu bilan birga, massivlarda "saralash" bilan bog'liq (umuman, boshqa obyektlarda ham qo'llash mumkin bo'lgan) ba'zi algoritmlar yoritiladi. Ma'lumki, dasturlash tili beradigan har bir yangi imkoniyat, birinchidan, yechish mumkin bo'lgan masalalar to'plamini kengaytirsa, ikkinchidan, bir usulda yechilgan masalalarni boshqa usulda soddaroq yechish imkonini beradi.

1-§. C++ DASTURLASH TILIDA MASSIVLAR

Kundalik hayotimizda ko'p turdagi jadvallardan foydalanamiz: dars jadvali, shaxmat yoki futbol o'yinlari bo'yicha musobaqa jadvali, Pifagor (karra) jadvali, kelishiklar jadvali va boshqalar. Jadvalni tashkil etuvchi ma'lumotlar uning **elementlari** deb ataladi.

Jadvallar bir o'lchovli (chiziqli), ikki o'lchovli (to'g'ri to'rtburchakli), uch o'lchovli (parallelopipedli) va hokazo bo'ladi.

Chiziqli jadvallar satr yoki ustun shaklida ifodalanadi. Masalan, sinfdagi o'quvchilar ro'yxati sinf jurnalida ustun shaklidagi jadval ko'rinishida yoziladi. O'quvchilarning familiyalari bu jadvalning elementlarini tashkil etadi. Ularning har biri o'z tartib raqamiga ega.

Ikki o'lchovli jadvallar satrlar va ustunlar kesishmasidan tashkil topadi (MS Word yoki MS Excel jadvallari kabi). Ularning elementlari ustun va satrlar kesishgan kataklarda joylashadi. Bunday jadvallarda biror elementni ko'rsatish uchun uning nechanchi satr va nechanchi ustunda joylashganligini, ya'ni satr va ustun bo'yicha tartib raqamlarini bilish kerak bo'ladi. Demak, ikki o'lchovli jadvalning har bir elementiga ikkita (satr va ustun bo'yicha) tartib raqami mos keladi.

C++ dasturlash tilida jadvallardagi ma'lumotlarni ifodalash, umuman, bir turdagi bir necha o'zgaruvchini tartiblab guruhlash hamda shu ma'lumotlar bilan ishlash qulay bo'lishi uchun massiv tushunchasi kiritilgan. **Massiv** – bu jadval ko'rinishidagi ma'lumotlarni ifodalash uchun dasturchiga dasturlash tili beradigan ajoyib guruhlash imkoniyati bo'lib, massiv aniq sondagi bir turdagi tartib raqamiga ega ma'lumotlar majmuyidir. Tartib raqamiga ega ma'lumotlar **massiv elementlari** (**qiymati**) deb ataladi. Massiv elementlarining tartib raqami **indeks** deb ataladi va indeks [ ] operatori ichida yoziladi. Massiv elementlarining **indeksi** manfiymas butun sonlardan iborat. Kompilyator massiv elementlarini xotirada ketma-ket joylashtiradi.

BIR O'LCHOVLI STANDART MASSIVLAR

Bir o'lchovli massivlar C++ tilida quyidagicha tavsiflanadi:

```
tur identifikator[o'lchami];
```

Bu yerda `tur` – massiv elementlarining turi (barcha ma'lumot bitta umumiy turda ifodalanadi), `identifikator` – massiv nomi, `o'lchami` – massiv elementlarining soni. Massiv tavsifidan ko'rinib turibdiki, massiv elementlarining soni massivni tavsiflash jarayonida ko'rsatiladi va bu qiymat keyinchalik o'zgarishsiz qoladi. Massiv o'lchami qiymati butun (**unsigned int**) turdagi son yoki ifoda bo'lishi shart. C++ tilida massiv elementlari 0-indeksdan boshlanadi. Masalan, **int** (barcha elementlar qiymati, ya'ni ma'lumotlar int) turidagi 10 ta elementdan iborat bo'lgan **a** nomli massiv, **double** turidagi 100 ta elementdan iborat bo'lgan **x** nomli massiv, **bool** turidagi 123 ta elementdan iborat bo'lgan **b** nomli massiv quyidagicha tavsiflanadi:

```
#include <iostream>
using namespace std;
int main(){
    int a[10];
    double x[10*10];
    bool b[123];
    return 0;
}
```

Misol-1. Bir o'lchovli jadval beshta elementga ega bo'lsin:

210	–6005	700	19	1

Jadvalni, tartib raqamlari berib, quyidagi **a** massiv kabi tasavvur qilamiz:

Tartib raqami	0	1	2	3	4
a[ ] massiv elementlari qiymati	210	–6005	700	19	1

Endi C++ tilida bu jadval massiv kabi elementlari quyidagicha tavsiflanadi: **int a[5];**

Massiv elementlarini esa quyidagicha ifodalash mumkin:

a[0] = 210; a[1] = –6005; a[2] = 700; a[3] = 19; a[4] = 1;

Massiv elementlari indeksini biror butun qiymatli o'zgaruvchi (masalan, k) orqali ifodalash ham mumkin, masalan, k = 0 bo'lsa, a[k] = 210, k = 3 bo'lgani uchun, a[k] = 19 bo'ladi. Boshqa tomondan, k = 0 bo'lsa, u holda k + 3 = 0 + 3 = 3 bo'lgani uchun a[k+3] = 19 bo'ladi.

MASSIVGA QIYMAT O‘ZLASHTIRISH

Massivga qiymat turli xil usullarda o‘zlashtirilishi mumkin.

1) **Massivni tavsiflashda boshlang‘ich qiymat berish:**

   a. **Massiv o‘lchamini ko‘rsatmasdan boshlang‘ich qiymat o‘zlashtirish.** Umumiy ko‘rinishi quyidagicha:

```
tur identifikator[] = {element_1, element_2, …,element_n};
```

bu yerda **element_1**, **element_2**, ..., **element_n** – massiv elementlari boshlang‘ich qiymati. Bu qiymatlar massivga 0-indeksdan boshlab ketma-ket joylashtirib chiqiladi. Massiv bunday tavsiflanganda massiv o‘lchami, ya’ni elementlari soni **n** ga teng bo‘ladi. Masalan:

```
int a[] = {2, 7, 2, 10};// massiv o‘lchami n=4
char b[] = {'b', 'a', 'a'};// massiv o‘lchami n=3
double c[] = {19.2, 21.07, 10.2};// massiv o‘lchami n=3
```

Bu massivlar elementlari 0 dan n-1 gacha indeksdagi o‘rinlarda joylashadi:

indeks	0	1	2	3
a[ ]	**2**	**7**	**2**	**10**

indeks	0	1	2
b[ ]	**b**	**a**	**a**

indeks	0	1	2
c[ ]	**19.2**	**21.07**	**10.2**

   b. **Massiv o‘lchamini ko‘rsatib boshlang‘ich qiymat o‘zlashtirish.** Umumiy ko‘rinishi:

```
tur identifikator[o‘lchami] = {element_1, element_2, …, element_n};
```

bunda agar elementlar soni ko‘rsatilgan massiv o‘lchamidan ko‘p bo‘lsa, xato xabari chiqadi. Aksincha, agar elementlar soni ko‘rsatilgan massiv o‘lchamidan kam bo‘lsa, elementlar 0-indeksdan boshlab ketma-ket joylashtiriladi va qolgan joylar bo‘sh (qolgan elementlar qiymati noma’lum bo‘lib, ya’ni initsializatsiya qilinmay) qoladi. Masalan:

```
int a[3] = {1, 2, 3};// massiv o‘lchami = elementlar soni
double b[4] = {2.3, 1.3};// massiv o‘lchami > elementlar soni
bool c[2] = {true, true};// massiv o‘lchami= elementlar soni
```

2) Massiv tavsifidan so‘ng qiymat o‘zlashtirish.

a. Massiv tavsifidan so‘ng massiv elementlariga boshlang‘ich qiymat berish yoki qiymatini o‘zgartirish to‘g‘ridan to‘g‘ri massivning kerakli elementiga murojaat qilish orqali amalga oshiriladi. Bunda **[ ]** operatoridan foydalanish maqsadga muvofiq (**[ ]** operatori ichida massivning kerakli elementi indeksi ko‘rsatiladi):

```
#include <iostream>
using namespace std;
int main(){
    int a[5];
    a[0] = 1; a[1] = 2; a[2] = 3;
    a[3] = 4; a[4] = 5;
    return 0;
}
```

b. C++ standart massivlari kompyuter xotirasida massiv elementlari turiga va massiv o‘lchamiga mos ravishda ketma-ket joylashgan xotira bloklarini egallaydi. Bunda massivni ifoda etuvchi o‘zgaruvchi aslida massiv elementlari turiga mos ko‘rsatkich bo‘lib, u massivning birinchi (0-indeksdagi) elementi adresini o‘zida saqlaydi. Misol uchun `int a[10];` massiv tavsifida **a** o‘zgaruvchisi aslida **int** turidagi ko‘rsatkichdir. Shuni hisobga olgan holda va ko‘rsatkichlar ustida amallar yordamida massiv elementlariga quyidagicha murojaat qilishimiz ham mumkin:

`*(a + 0) = 1;`

Bu ko‘rsatma `a[0] = 1;` ko‘rsatmasi bilan bir xil ma’noga ega.

```
#include <iostream>
using namespace std;
int main(){
    int a[5];
    *(a + 0) = 1; *(a + 1) = 2; *(a + 2) = 3;
    *(a + 3) = 4; *(a + 4) = 5;
    return 0;
}
```

Massiv bilan ishlashda takrorlanish operatorlaridan foydalanish juda qulaydir. Masalan, massiv elementlariga ketma-ket murojaat qilish, qiymatlarini o‘zgartirish, massivni chiqarish va boshqalarda takrorlanish operatorlari dastur yozishni soddalashtiradi. Masalan, qiymat o‘zlashtirishda:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  int a[10], t;`
`  for(t=0;t<10;t++)a[t]=t+1;`
`  for(t=0;t<10;t++)cout<<a[t]<<" ";`
`  return 0;`
`}`
	`1 2 3 4 5 6 7 8 9 10`

Masalan, massiv elementlari qiymatlarini klaviatura yordamida kiritishda:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  int a[10], t;`
`  cout<<"Kiruvchi ma'lumotlar:"<<endl;`
`  for(t=0;t<10;t++)cin>>a[t];`
`  cout<<"Chiquvchi ma'lumotlar:"<<endl;`
`  for(t=0;t<10;t++)cout<<a[t]<<" ";`
`  return 0;`
`}`
	`Kiruvchi ma'lumotlar:`
	`10 9 8 7 6 5 4 3 2 1`
	`Chiquvchi ma'lumotlar:`
	`10 9 8 7 6 5 4 3 2 1`

C++ dasturlash tilida bir massivni ikkinchi massivga o'zlashtirish mumkin emas!
Ya'ni quyidagi dastur kompilyatsiya jarayonida xatolik xabari chiqaradi:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  int a[10], b[10], t;`
`  for(t=0;t<10;t++)a[t]=t;`
`  b = a;`
`  for(t=0;t<10;t++)cout<<b[t]<<" ";`
`  return 0;`
`}`
	error: invalid array assignment

KO‘P O‘LCHOVLI MASSIVLAR

C++ dasturlash tilida ko‘p o‘lchovli massivlar bilan ham ishlash mumkin. Ko‘p o‘lchovli massivlar tavsifi quyidagicha amalga oshiriladi:

```
tur identifikator[o‘lcham_1][o‘lcham_2]…[o‘lcham_n];
```

Masalan, ikki o‘lchovli **int** turidagi 10 ta satrli va 20 ta ustunli massiv (10x20) quyidagicha tavsiflanadi: `int a[10][20];`

Quyida 2, 3 va 4 o‘lchovli massivlar tavsifi ifodalangan:

```
#include <iostream>
using namespace std;
int main(){
    int a[10][20];
    double b[100][3][2];
    bool d[1][1][1][1];
    return 0;
}
```

Dasturda tavsiflangan ikki o‘lchovli **a** massiv elementlari sonini hisoblaymiz. Matematik nuqtayi nazardan olib qaraganda 10 – bu satrlar soni va 20 – ustunlar soni. Ikkinchi tomondan, bu yerda 10 – ustunlar soni va 20 esa satrlar soni deb ham qaralishi mumkin. Albatta, qanday tushunish dasturchining o‘ziga bog‘liq. Ya’ni massiv elementlari soni 10·20=20·10=200 ta ekan. Lekin shuni ta’kidlash joizki, C++ tilida massivni xotirada saqlash tartibi hozir aytib o‘tilgan holatlarning ikkinchisiga to‘g‘ri keladi, ya’ni avval ustun, keyin esa satr tartibi sanab boriladi.

Misol-2. Ikki o‘lchovli butun qiymatli jadval berilgan bo‘lsin:

$$\begin{bmatrix} 2 & 10 & 6 \\ -1 & -23 & 5 \end{bmatrix}$$

Massiv elementlarini tartib bilan b[0][0], b[0][1], b[0][2], b[1][0], ... kabi yozib olamiz:

$$b = \begin{bmatrix} 2 & 10 & 6 \\ -1 & -23 & 5 \end{bmatrix} = \begin{bmatrix} b_{00} & b_{01} & b_{02} \\ b_{10} & b_{11} & b_{12} \end{bmatrix}$$

Bu jadval C++ tilida quyidagicha tavsiflanadi: **int b[1][2];**.

Aytib o‘tilganidek, indeks sifatida o‘zgaruvchi yoki ifoda qo‘llaniladi. Masalan, t=0, k=2 bo‘lsa, u holda b[t][k] = 6 va (t+1=0+1=1 va k–2=2–2=0 bo‘lgani uchun) b[t+1][k–2] = –1 bo‘ladi. Umumiy holda **b[t][k]** massiv uchun satr tartibini ifodalovchi **t** o‘zgaruvchisi 0 va 1, ustun tartibini ifodalovchi **k** o‘zgaruvchisi esa 0, 1 va 2 qiymatlarni qabul qiladi.

Ko‘p o‘lchovli massivlarga har bir elementga ustun, satr va boshqa o‘lchovlardagi tartibi ko‘rsatilgan holda boshlang‘ich qiymat berish mumkin. Masalan:

Dastur

```
#include <iostream>
using namespace std;
int main(){
    int t, k, m;
    int a[2][2]={{1, 2},{3, 4}};
    double b[1][2]={{1.2, 1.3}};
    char c[1][2][3]={{{'a', 'b', 'c'}, {'m', 'n', 'k'}}};
        cout<<"a massiv elementlari: "<<endl;
    for(k=0;k<2;k++){
        for(t=0;t<2;t++)cout<<a[k][t]<<" ";cout << endl;
                        }

    cout<<endl<<"b massiv elementlari: "<<endl;
    for(k=0;k<1;k++){
        for(t=0;t<2;t++)cout<<b[k][t]<<" ";cout<<endl;
                    }

    cout<<endl<<"c massiv elementlari: "<<endl;
    for(k=0;k<1;k++){
        for(m=0;m<2;m++){
            for(t=0;t<3;t++)cout<<c[k][m][t]<<" ";cout<<endl;
                            }
                    }
    return 0;
}
```

Natijasi

```
a massiv elementlari:
1 2
3 4

b massiv elementlari:
1.2 1.3

c massiv elementlari:
a b c
m n k
```

LOKAL VA GLOBAL TAVSIFLASHGA OID

C++ tilida global o‘zgaruvchi sifatida tavsiflangan o‘zgaruvchilarga, massiv elementlarining har biriga boshlang‘ich qiymat o‘zlashtiriladi.

Masalan, sonli massiv elementlarining barchasiga 0 qiymati o‘zlashtiriladi. Lokal o‘zgaruvchilarda yoki lokal o‘zgaruvchilar sifatida tavsiflangan sonli massivda esa bunday emas. Shu sababli lokal o‘zgaruvchi sifatida tavsiflangan massivga boshlang‘ich qiymat o‘zlashtirishdan avval murojaat qilinsa, xato natijalarga olib kelishi mumkin. Quyidagi dastur natijasida shu kabi holatlarni ko‘rish mumkin.

Dastur

```
#include <iostream>
using namespace std;
int a[5];bool b[5];char h[5];
int main(){
    int aa[5]; bool bb[5]; char hh[5],t;
    cout<<"Qiymatlar:"<<endl;
    for(t=0;t<5;t++)cout<<a[t]<<" ";cout<<endl;
    for(t=0;t<5;t++)cout<<aa[t]<<" ";cout<<endl;
    for(t=0;t<5;t++)cout<<b[t]<<" ";cout<<endl;
    for(t=0;t<5;t++)cout<<bb[t]<<" ";cout<<endl;
    for(t=0;t<5;t++)cout<<h[t]<<" ";cout<<endl;
    for(t=0;t<5;t++)cout<<hh[t]<<" ";cout<<endl;
    return 0;
}
```

Natijasi

```
Qiymatlar:
0 0 0 0 0
-2 1976766818 1977113532 4354608 2293652
0 0 0 0 0
117 200 212 127 66
a a a a a
" a ₣ M ₺
```

Lokal yoki global tavsiflanayotgan massivga boshlang‘ich qiymat quyidagicha o‘zlashtirilishi ham mumkin:

```
tur identifikator[o‘lcham_1][o‘lcham_2]…[o‘lcham_n]={qiymat};
```

Bunday tavsiflangan massivning birinchi elementi `qiymat` ni o‘zlashtiradı, qolgan elementlariga global tavsiflangan massivlar kabi boshlang‘ich qiymat (masalan, sonli massivlarda 0) o‘zlashtiriladi.

Massivlarni global tavsiflash bilan lokal tavsiflashning yana bir farqi **massiv o'lchami** bilan bog'liq bo'lib, **global tavsiflashda** massiv o'lchami lokal tavsiflangan massiv o'lchamiga qaraganda **bir necha o'n baravar** katta tanlanishi mumkin.

MASSIVLARGA OID FOYDALI FUNKSIYALAR

C++ tilining **algorithm** kutubxonasiga tegishli quyidagi funksiyalar massivlar bilan ishlashda qulayliklar beradi:

Funksiya	Vazifasi
count(a+start, a+end, qiymat)	a massivning qaralayotgan indeksi **start** ga teng elementidan indeksi **end** ga teng elementigacha bo'lgan barcha elementlari ichidan **qiymat**ga tenglari sonini qaytaradi
*min_element(a+start, a+end)	a massivning qaralayotgan indeksi **start** ga teng elementidan indeksi **end** ga teng elementigacha bo'lgan barcha elementlari ichidan **eng kichik elementiga ko'rsatkich** qaytaradi
*max_element(a+start, a+end)	a massivning qaralayotgan indeksi **start** ga teng elementidan indeksi **end** ga teng elementigacha bo'lgan barcha elementlari ichidan **eng katta elementiga ko'rsatkich** qaytaradi

Quyidagi dastur bu funksiyalarni qo'llashga doir:

Dastur	Natijasi

`#include <iostream>`
`#include <algorithm>`
`using namespace std;`
`int main(){`
`  int a[] = {1, 6, -3, 3, 0, 8, 3}, s;`
`  s= count(a, a + 7,3);`
`  cout<<"3 soni "<<s << " ta"<<endl;`
`  s=*min_element(a, a+7);`
`  cout<<"Min= "<<s<<endl;`
`  cout<<"Max= "<<*max_element(a, a+4);`
`  return 0;`
`}`
	`3 soni 2 ta`
	`Min= -3`
	`Max= 6`

2-§. MASSIVLARNI TARTIBLASH USULLARI

ALGORITMNING BAJARILISH VAQTI

Shuni aytib o'tish kerakki, dasturlashda katta hajmdagi axborotni qayta ishlashda qo'llanadigan usullarda bajariladigan amallar soni bilan taqqoslanadi. Misol uchun yuqoridagi N o'lchovli massivning maksimal elementini aniqlash masalasini olaylik. Maksimal elementni topish uchun qo'llangan algoritmda taqqoslashlar soni (N-1) ta bo'ldi, demak, amallar soni (N-1) ga teng. Bunda kiritish va chiqarish amallari e'tiborga olinmadi, chunki qo'llangan algoritm uchun bu ahamiyatsiz. Dasturchilar algoritmdagi amallar sonini "algoritmning bajarilish vaqti" tushunchasi bilan bog'lashadi. Umuman olganda algoritmning bajarilish vaqti faqat amallar soniga emas, balki yana protsessor tezligi (masalan, chastotasi 2 Ggs liga nisbatan 3 Ggs li protsessor tezroq ishlaydi), dasturlash tili imkoniyati (masalan, Basic tiliga nisbatan Paskal tili tezroq ishlaydi, Paskal yoki Java tiliga nisbatan C++ tili tezroq ishlaydi), ma'lumotlar turi va hokazolarga bog'liq. Shu sababli, odatda, **algoritmning tezligi** yoki **algoritmning bajarilish vaqti** deganda **bajariladigan amallarning maksimal soni** nazarda tutiladi.

Algoritmning bajarilish vaqtini hisoblashda qayta ishlanayotgan axborot hajmining eng yuqori darajasi bilan bog'liq amallar soni e'tiborga olinadi. Masalan, biror algoritmda qayta ishlanayotgan N ta axborotga nisbatan amallar soni $21 \cdot N^3+7 \cdot N^2+1963$ ta bo'lsa, u holda bu algoritmning bajarilish vaqti $\mathbf{O(N^3)}$ (o'qilishi: "O n kub"), ya'ni **deyarli** $\mathbf{N^3}$ ga teng deb olinadi. Bunda yuqori darajadan kichik darajalar yoki koeffitsiyentlar qaralmaydi. Qo'llanma doirasida **O(X)** bilan bir qatorda **deyarli X** mazmundagi atama qo'llanadi.

ODDIY TANLOV, ALMASHTIRISH VA JOYLASHTIRISH USULLARI

Dasturlashda juda ko'p masalalar massivlar bilan bog'liq bo'lib, bir o'lchamli massiv elementlari qiymatlarini o'sish yoki kamayish yo'nalishida tartiblash masalasi dolzarb masalalardan biridir.

Tartiblashning juda ko'p usullari bor bo'lib, ulardan ba'zilarini ko'rib chiqamiz.

Bir o'chovli N ta elementli A massiv berilgan bo'lsin.

Oddiy tanlov usulida tartiblashda (qo'yilgan masala shartidan kelib chiqib) eng kichik (eng katta) elementini aniqlab massivning 1-elementi bilan almashtiriladi. Keyin tartiblangan qismga tegmasdan massivning qolgan qismi uchun ham shu kabi ish bajarilaveradi. Bunda 1-elementni aniqlash uchun (N–1) ta taqqoslash, 2-elementni aniqlash uchun (N-2) ta taqqoslash, ..., (N-1)-elementni aniqlash uchun 1 ta taqqoslash kerak bo'ladi. Jami $1+2+\ldots+(N-1)=\frac{N \cdot (N+1)}{2}$ (deyarli N^2) marta taqqoslash bajariladi. Dastur qismi quyidagicha:

```
for(t=1;t<=n-1;t++) {                    //o‘tishlar soniga mos sikl
   k=t;          //k – massivning tekshirilayotgan qismidagi birinchi element tartib raqami
   m=k;          //m – eng kichik qiymatli massiv elementi tartib raqami uchun
   for(tt=k+1;tt<=n;tt++)   //massivning qolgan qismi bilan taqqoslash sikli
     if (a[tt]<a[m]) m=tt;  //tt ning qiymatini eslab qolish
       swap(a[k],a[m]);     //elementlar qiymatini almashtirish
}
```

Dasturda avval massivni 1-elementidan (k=1) oxirgi elementigacha qaraladi. Bunda 1-element (m=1) qiymati eng kichik deb olinadi. Qolgan elementlar bilan birma-bir taqqoslab undan kichik qiymatli biror element topilsa, u holda uning indeksi eslab qolinadi (m=tt). Endi 1-element (k=1 edi) bilan m-element qiymati almashtiriladi (swap(a[k],a[m])). Keyingi o‘tishda hosil bo‘lgan yangi massivni 2-elementidan (k=2) oxirgi elementigacha qaraladi. Bunda 2-elementi (m=2) qiymati eng kichik deb olinadi. Qolgan elementlar bilan birma-bir taqqoslab undan kichik qiymatli biror element topilsa, uni indeksi eslab qolinadi (m=tt). Ikkinchi (k=2 edi) element bilan m-element qiymati almashtiriladi. Keyin 3-elementi, …, (N–1)-elementi (m=3, …, N–1) qiymati eng kichik deb olib, 1-element va 2-elementlardagi kabi amallar ketma-ketligi bajariladi.

Oddiy almashtirish (pufakcha) usulida tartiblash ikkita yonma-yon joylashgan elementlar juftligini ketma-ket taqqoslashga asoslangan. Agar yonma-yon joylashgan elementlar juftligi kerakli tartibda joylashmagan (masalan, o‘sish yo‘nalishida tartiblashda 1-element 2-elementdan katta) bo‘lsa, u holda ularning qiymati almashtiriladi. Masalan, birinchi o‘tishda 1-element 2-elementdan katta bo‘lsa o‘rnini almashtiramiz, 2-element 3-elementdan katta bo‘lsa o‘rnini almashtiramiz, …, (N–1)-element N-elementdan katta bo‘lsa o‘rnini almashtiramiz va natijada eng oxirgi element o‘zidan oldingi barcha elementlarga nisbatan eng katta qiymatli bo‘ladi. Ikkinchi o‘tishda (N–1)-element o‘zidan oldingi barcha elementlarga nisbatan eng katta qiymatli bo‘ladi. Shu usulda davom ettirsak, oxirida 2-element o‘zidan oldingi 1-elementga nisbatan eng katta qiymatli bo‘ladi. Bu usulda N-elementni aniqlash uchun (N–1) ta taqqoslash, (N–1)-elementni aniqlash uchun (N–2) ta taqqoslash, …, 2-elementni aniqlash uchun 1 ta taqqoslash kerak bo‘ladi. Jami taqqoslashlar soni $1+2+\ldots+(N-1)=\frac{N\cdot(N+1)}{2}$ (deyarli N^2) ta bo‘ladi. Dastur qismi quyidagicha:

```
for(t=1;t<=n–1;t++)                    //o‘tishlar soniga mos sikl
   for(tt=1;tt<=n–t;tt++)              //massivni qolgan qismi bilan taqqoslash sikli
     if(a[tt]>a[tt+1]) swap(a[tt],a[tt+1]); //elementlar qiymatini almashtirish
```

Parametrli takrorlash operatori yordamida tuzilgan dasturda biror o‘tishda taqqoslash

sharti bir marta ham rost qiymat qabul qilmasa ham, ya'ni massiv tartiblanib qolgan bo'lsa ham, tekshirish davom etaveradi. Agar tartiblash dasturi shart bo'yicha takrorlash operatori yordamida tuzilsa, behuda o'tishlarni olib tashlash mumkin. Bu holda taqqoslashlar soni kamayadi.

```
t=0;
do { t++; bor=0;                  //bor – o'tish kerakligi belgisi
     for(tt=1;tt<=n–t;tt++)       //tekshirilayotgan qismda elementlarni taqqoslash sikli
       if(a[tt]>a[tt+1]){bor=1;   //belgi qiymatini o'zgartirish
          swap(a[tt],a[tt+1]);}}  //elementlar qiymatini almashtirish
while(bor==1);                    //sikldan chiqish sharti
```

Dasturda har bir qo'shni elementlar taqqoslanayotgani uchun biror o'tishda bor=0 bo'lsa, u holda massiv tartiblanib qolgan bo'ladi va bor==1 shart **false** qiymat qabul qilgani uchun shart bo'yicha takrorlash operatori ham o'z ishini tugatadi.

Oddiy joylashtirish usulida tartiblashda tartiblash shartini bajargan elementni massivni tartiblangan qismidan keyingi o'ringa joylashtirishga asoslanadi. Birinchi qadamda 2-element 1-element bilan taqqoslanadi, ikkinchi qadamda 3-element birinchi ikkita element bilan taqqoslanadi, uchinchi qadamda 4-element birinchi uchta element bilan taqqoslanadi, va hokazo. Tartiblangan element (k-1) ta elementli qismga k-element ularni tartibini buzmagan holda "joylashtiriladi", ya'ni k-elementni m-o'ringa (m<k) joylashtirish uchun m dan katta va k dan kichik tartib raqamli elementlar indekslari bittaga oshiriladi.

```
for(t=2;t<=n;t++){    //qadamlar soni
  b=a[t];             //massivni tartiblangan qismidan keyingi element qiymati
  m=1;
  while(b>a[m]) m++;        //joylashtirish uchun m tartib raqamni aniqlash
  for(tt=t;tt>=m+1;tt--)a[tt]=a[tt–1];  //elementlar indeksini oshirish
  a[m]= b;}                             //b qiymatni m-element o'rniga joylashtirish
```

Avvalgi tartiblash usullaridan farqli ravishda bu usulda elementlarni taqqoslash jarayoni element tartiblash shartini bajargan zahoti to'xtatiladi. Tartiblashning oddiy joylashtirish usulida ham ko'pi bilan $1+2+\ldots+(N-1)=\frac{N\cdot(N+1)}{2}$ (deyarli N^2) marta taqqoslash bajariladi.

C++ TILINING TARTIBLASH FUNKSIYALARI

C++ tilida massivlarni (boshqa obyektlarni ham) tartiblash uchun **stdlib.h** kutubxonasiga tegishli **qsort** va **algorithm** kutubxonasiga tegishli **sort** funksiyasi mavjud. Bu funksiyalardan **sort** funksiyasi **qsort** funksiyasiga qaraganda tezroq ishlaydi.

Mazkur **sort** funksiyaning **n** ta elementli (agar indekslari tartibi 0 dan n-1 gacha bo'lsa) biror **a** massivni o'sish yo'nalishida tartiblashga tatbiqi quyidagicha yoziladi:

```
sort(a, a+n);
```

kamayish yo'nalishida tartiblashga tatbiqi quyidagicha yoziladi:

```
sort(a, a+n, greater<int>());
```

Agar **a** massivning k-elementidan m-elementiga bo'lgan qismi tartiblanishi kerak bo'lsa, u holda **sort** funksiyasi quyidagicha yoziladi:

```
sort(a+k, a+m+1, greater<int>());
```

Quyida dasturda **6** ta elementli **a** massivni **sort** funksiyasi yordamida turli yo'nalishlarda tartiblash ko'rsatilgan:

Dastur	Natijasi

`#include <iostream>`
`#include <algorithm>`
`using namespace std;`
`int main(){`
`  int t, a[]={5, 2, 7, 1, 9, 6};`
`  for(t=0;t<=5;t++)cout<<a[t]<<" ";`
`  cout<<endl;`
`  sort(a+1, a+4);`
`  for(t=0;t<=5;t++)cout<<a[t]<<" ";`
`  cout<<endl;`
`  sort(a, a+6);`
`  for(t=0;t<=5;t++)cout<<a[t]<<" ";`
`  cout<<endl;`
`  sort(a, a+6,greater<int>());`
`  for(t=0;t<=5;t++)cout<<a[t]<<" ";`
`  return 0;`
`}`
	`5 2 7 1 9 6`
	`5 1 2 7 9 6`
	`1 2 5 6 7 9`
	`9 7 6 5 2 1`

Agar **a** massiv indekslari tartibi 1 dan n gacha bo'lsa, u holda (o'sish va kamayish yo'nalishida):

`sort(a+1, a+n+1);` va `sort(a+1, a+n+1, greater<int>());`

C++ tili **sort** funksiyasi ishi avvalgi usullardan tezroq bo'lib, amallari soni deyarli N*log(N) ga (bunday belgilashlarda asos 2 deb tushuniladi) teng. Ya'ni masalan, N=1024 bo'lsa, u holda yuqoridagi usullarda deyarli $1024 \cdot 1024 = 1048576$ marta amal bajarilsa, **sort** funksiyasi ishida deyarli 100 marta kam, ya'ni deyarli $1024 \cdot \log(1024) = 1024 \cdot 10 = 10240$ marta amal bajariladi.

3-§. C++ TILIDA FOYDALANUVCHI FUNKSIYALARINI HOSIL QILISH

Dasturlash tillarida foydalanuvchi funksiyalari kabi qism dasturlar yozish imkoniyati kiritilgan bo'lib, bunday qism dasturlarga asosiy dastur tanasidan "murojaat" qilinadi. "Murojaat qilish" tushunchasini ba'zan "chaqirish" deb ham atashadi.

Foydalanuvchi funksiyalari – bu dasturchi tomonidan tuziladigan funksiyalar bo'lib, ular dasturni soddalashtirish, takroran uchraydigan dastur qismlarini qayta-qayta yozishning oldini olish va boshqa maqsadlarda dasturchiga qulayliklar berish uchun tuziladi.

Masalan, agar 1 dan N gacha bo'lgan sonlardan 2 ta xossa, ya'ni birinchi va oxirgi raqami teng (1-xossa) hamda biror K songa karralilarini (2-xossa) aniqlash kerak bo'lsa, u holda har bir xossani tekshirishni alohida funksiya sifatida tashkil etib, asosiy dasturda shu funksiyalarga murojaat bilan kifoyalanish mumkin. Yana, masalan, biror son tub ekanligini alohida funksiya sifatida tashkil etish ham foydali, chunki shu funksiyani ixtiyoriy dasturga nusxalab foydalanish mumkin.

Foydalanuvchi funksiyalarining umumiy ko'rinishi quyidagicha:

```
natija_turi identifikator(tur_1 o'zgaruvchi_1,…, tur_n o'zgaruvchi_n)
    {
        …
        return ifoda;
    }
```

Bu yerda **natija_turi** – funksiya qaytaradigan qiymat turi, **identifikator** – dasturchi tomonidan funksiyaga berilgan nom (nom sifatida **main** tanlash mumkin emas), **tur_1, …, tur_n** – funksiyaga beriladigan argumentlarning turlari, **o'zgaruvchi_1, …, o'zgaruvchi_n** – funksiya argumentlari nomlari bo'lib, asosiy dasturdagi o'zgaruvchilar qiymati shu argumentlar tomonidan o'zlashtiriladi. Tushunish qiyin emaski, bu holda **ifoda** qiymati **natija_turi** ga mos bo'lishi shart. Bu funksiyada **return** orqali funksiya qiymati qaytariladi. Sodda misol sifatida quyida tashkil etilgan funksiyani keltirish mumkin:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int nol()`
`{`
`  return 0;`
`}`
`int main(){`
`  cout << nol();`
`  return 0;`
`}`
	`0`

Ko‘rinib turibdiki, yuqoridagi funksiya faqatgina 0 qiymatini qaytaradi.

Dasturchi xohishiga ko‘ra funksiya tavsifi **main** funksiyasidan avval, uning tanasini esa **main** funksiyasidan keyin tashkil etsa ham bo‘ladi:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int nol();`
 	
`int main(){`
`  cout << nol();`
`  return 0;`
`}`
`int nol()`
`{`
`  return 0;`
`}`
	`0`

Keyingi misol sifatida 3 ta sondan kattasini topish funksiyasi tuzilmasini ko‘ramiz:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int max3(int a,int b,int c){`
`  if(a>=b&&a>=c)return a;`
`  if(b>c)return b;else return c;`
`}`
`int main(){`
`  int a = 3, b = 2, c = 1;`
`  int m = 12, n = 21, k = 4;`
`  cout<<"max("<<a<<","<<b<<","<<c<<")= ";`
`  cout<<max3(a, b, c)<<endl;`
`  cout<<"max("<<m<<","<<n<<","<<k<<")= ";`
`  cout<<max3(m, n, k);`
`  return 0;`
`}`
	`max(3,2,1)= 3`
	`max(12,21,4)= 21`

Bu misol orqali o‘zgaruvchilar orasidagi bog‘lanishni tushunish oson.

Keyingi misolda sonni tub yoki tub emasligini tekshirish dasturi foydalanuvchi funksiyasi sifatida dasturga kiritilishi beradigan imkoniyatni ko‘rish mumkin:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`bool tub(int ts){`
`  if(ts==1) return false;`
`  for(int k=2;k*k<=ts;k++)`
`  if(ts%k==0)return false;`
`  return true;`
`}`
`int main(){`
`  int n,t;`
`  cout<<"N= "; cin>>n;`
`  for(t=1;t<=n;t++)`
`  if(tub(t))cout<<t<<" tub son"<<endl;`
`  else cout<<t<<" tub son emas"<<endl;`
`  return 0;`
`}`
	`N=15`
	`1 tub son emas`
	`2 tub son`
	`3 tub son`
	`4 tub son emas`
	`5 tub son`
	`6 tub son emas`
	`7 tub son`
	`8 tub son emas`
	`9 tub son emas`
	`10 tub son emas`
	`11 tub son`
	`12 tub son emas`
	`13 tub son`
	`14 tub son emas`
	`15 tub son emas`

Bu dasturda mantiqiy qiymat qaytaruvchi foydalanuvchi funksiyasi nomi **tub()** bo'lib, agar son tub bo'lsa **true**, aks holda **false** qiymatni qaytaradi. Aytish joizki, **tub()** funksiyasining qanday tashkil etilishi qo'yilgan masalaning mohiyati va dasturchining xohish-imkoniyatiga bog'liq.

Dastur ishida bir foydalanuvchi funksiyasidan boshqa bir foydalanuvchi funksiyasiga murojaat etish ham mumkin. Bunda murojaat etilayotgan funksiya murojaat etuvchi funksiyadan avval tavsiflangan bo'lishi kerak. Ya'ni quyidagi holat xatolikka olib keladi:

```
int birinchi(){
    int a = ikkinchi();
    return a;
}
int ikkinchi(){
    return 2;
}
```

Bunday xato holatlar yuzaga kelmasligi uchun foydalanuvchining barcha funksiyalarini **main** funksiyasidan avval tavsiflash maqsadga muvofiq:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
 	
`int kvadrat(int a);`
`int summa(int a, int b);`
 	
`int main(){`
	`25`

```
    int a=2,b=3;
    cout<<kvadrat(summa(a, b));
    return 0;
}
int summa(int a,int b){
    return a+b;}

int kvadrat(int a){
      return a*a;}
```

Ba'zi funksiyalar biror natija qaytarishi shart emas. Bunday funksiyalar ba'zi dasturlash tillarida **protsedura** sifatida keltirilgan (masalan, Pascal). C++ tilida bunday funksiyalarni (protsedura kabi) tashkil etish uchun **void** turidan foydalaniladi, ya'ni funksiyaning qiymati **void** turida bo'ladi. Bu holda hech qanday qiymat qaytarilmayotgani uchun **return** xizmatchi so'zini yozish shart emas yoki biror qiymat qaytarmaydigan **return;** kabi yozish ham yetarli. Masalan:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
 	
`void print_summa(int a,int b){`
`    cout<<a+b<<endl;}`
`int main(){`
`    print_summa(2, 3);`
`    return 0;`
`  }`
	5

FUNKSIYALARDA KO'RSATKICHLAR

Shunday holatlar bo'ladiki, unda funksiyaga berilgan argument qiymati funksiya bajarilish vaqtida o'zgarishi hamda bu o'zgarish **main** asosiy funksiya ishiga ta'sir qilishi kerak bo'ladi. Avval ko'rib chiqilqan barcha funksiyalarda funksiyaga argument sifatida berilgan o'zgaruvchi qiymati asosiy funksiya tanasidan tashqarida funksiya bajarilishidan oldingi qiymatini saqlab turar edi. Bunga sabab C++ dasturlash tilida funksiyalarga murojaat qilishda **call-by-value** (ing. qiymat orqali chaqirish) uslubiga asoslanganligidir. Bu usulda funksiyaga berilgan har bir argumentga mos lokal yangi o'zgaruvchilar hosil qilinadi va ularga argumentdagi o'zgaruvchilar qiymati nusxalanadi. Funksiya hosil qilingan shu lokal o'zgaruvchilar bilan ishlaydi. Shu sababli dastur ishlashi jarayonida lokal o'zgaruvchilar qiymati o'zgarsada, bu o'zgarishlar funksiya argumentlariga ta'sir ko'rsatmaydi. O'zgarishlar funksiya argumentlariga ta'sir ko'rsatishi uchun ko'rsatkichlardan foydalanish mumkin. Ma'lumki, ko'rsatkichlar o'zida o'sha ko'rsatkich ko'rsatayotgan o'zgaruvchi adresini saq-

laydi. O‘zgaruvchi adresi lokal o‘zgaruvchiga nusxalansa ham, o‘sha adresdagi o‘zgaruvchiga murojaat qilish va uning qiymatini o‘zgartirish imkoniyati saqlanib qoladi. Hozirgina aytib o‘tilganlarni quyidagi A va B dasturlar ishidagi farqlanish orqali tushunib olish mumkin:

Dastur A	Natijasi

`#include <iostream>`
`using namespace std;`
`void summa(int a, int b, int c){`
`  c=a+b; cout<<"sum= "<< c <<endl;`
`  return;}`
`int main(){`
`  int a=10, b=5, c=2;`
`  cout<<"c= "<< c <<endl;`
`  summa(a, b, c);`
`  cout<<"c= "<< c <<endl;`
`  return 0;}`
	`c= 2`
	`sum= 15`
	`c= 2`

Dastur B	**Natijasi**

`#include <iostream>`
`using namespace std;`
`void summa(int a, int b, int *c){`
`  *c = a + b; cout<<"sum= "<< *c <<endl;`
`  return;}`
`int main(){`
`  int a=10, b=5, c=2;`
`  cout<<"c= "<< c <<endl;`
`  summa(a, b, &c);`
`  cout << "c = " << c << endl;`
`  return 0;}`
	`c= 2`
	`sum= 15`
	`c= 15`

Foydalanuvchi funksiyalari parametri sifatida sodda turlardan tashqari standart massivlar va (keyinchalik ko‘rib chiqiladigan) boshqa turlar (konteynerlar) ham ko‘rsatilishi mumkin.

Standart massivlarni quyidagi dasturlardagi ko‘rinishlarda funksiya parametri sifatida ko‘rsatish mumkin. Ta’kidlab o‘tish joizki, bunday ko‘rinishda funksiyaga berilgan massivlarning o‘lchamini funksiya tanasida aniqlashning imkoni yo‘q. Shu bois standart massivlar bilan birga ularning o‘lchami ham yana bir parametr sifatida funksiya e’lonida ko‘rsatilishi maqsadga muvofiqdir.

Dastur 1	Natijasi

`#include <iostream>`
`using namespace std;`
`void orttirish(int *a, int n){`
`  for(int i=0;i<n;i++)a[i]++;`
`  return;`
`}`
	`1 2`
	`2 3`

```
int main(){
    int a[2] = {1, 2};
    cout<<a[0]<<" "<<a[1]<<endl;
    orttirish(a, 2);
    cout<<a[0]<<" "<<a[1]<<endl;
    return 0;
}
```

Dastur 2	Natijasi

```
#include <iostream>
using namespace std;
void orttirish(int a[], int n){
    for(int i=0;i<n;i++)a[i]++;
    return;
}
int main(){
    int a[2] = {1, 2};
    cout<<a[0]<<" "<<a[1]<<endl;
    orttirish(a, 2);
    cout<<a[0]<<" "<<a[1]<<endl;
    return 0;
}
```

```
1 2
2 3
```

Yuqoridagi ikkala dasturda ham **orttirish** funksiyasi e'loni va ishi C++ kompilyatori uchun bir xil ma'noga ega.

Bundan tashqari C++ da **reference** turidagi o'zgaruvchi va o'zgarmaslar ham mavjud. Bunday **reference** turdagi o'zgaruvchilar (o'zgarmaslar) boshqa bir o'zgaruvchiga (o'zgarmasga) birikib olib, shu o'zgaruvchining (o'zgarmasning) xususiyatlarini o'zlashtirib olishi mumkin. Bunday xususiyatlar asosiy o'zgaruvchining qiymati va adresidir. Shu sababli agar asosiy o'zgaruvchining qiymati o'zgarsa, u holda unga birikib olgan **reference** turidagi o'zgaruvchi qiymati ham o'zgaradi va aksincha.

Odatda, **reference** turidagi o'zgaruvchilar doimo ularning e'lonida (boshqa bir o'zgaruvchi bilan) initsializatsiya qilinishi zarur. Ular ko'rsatkichlar kabi e'lon qilinadi, ammo bunda * belgisi o'rniga & belgisi qo'llaniladi:

```
int a = 10;
int& b = a;// b o'zgaruvchi a o'zgaruvchiga birikib oldi
```

Quyidagi dastur misolida bu o'zgaruvchilar qanday birikishini tushunib olish mumkin:

Dastur	Natijasi

```
#include <iostream>
using namespace std;
int main(){
    int a{10};
```

```
b = 10
b = 11 a = 11
b = 12 a = 12
```

```
    int& b = a;
    cout<< "b = " << b << endl;
    a = 11;
    cout<<"b = "<<b<<" a = "<<a<<endl;
    b = 12;
    cout<<"b = "<<b<<" a = "<<a<<endl;
    return 0;
}
```

C++ dasturlash tilida **reference** turdagi o‘zgaruvchilarning borligi **call-by-reference** uslubini ishlatish imkoniyatini beradi. Nomidan ham ma’lumki, **call-by-reference** uslubida **reference** turidagi o‘zgaruvchilar qo‘llaniladi. Bu uslubda funksiyaga berilgan argumentlar qiymatlari lokal o‘zgaruvchilarga nusxalanmaydi, balki ularga biriktiriladi va shu orqali ularning qiymatiga ta’sir o‘tkazish imkoniyati saqlab qolinadi. Masalan, yuqorida ko‘rsatkichlar yordamida yozilgan dasturni **call-by-reference** uslubi yordamida quyidagicha yozish ham mumkn:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`void summa(int a, int b, int& c){`
`  c = a + b; cout<<"sum= "<< c <<endl;`
`  return;}`
`int main(){`
`  int a=10, b=5, c=2;`
`  cout<<"c= "<< c <<endl;`
`  summa(a, b, c);`
`  cout << "c = " << c << endl;`
`  return 0;`
`}`
	`c= 2`
	`sum= 15`
	`c= 15`

4-§. REKURSIV FUNKSIYALAR TUZISHGA OID NAMUNALAR

Agar qism dastur tanasida yana o‘ziga, qism dasturga murojaat joylashtirilgan bo‘lsa, ya’ni qism dastur yana o‘zini chaqirsa, u holda bunday murojaat **rekursiv murojaat** deb ataladi. Tanasida o‘ziga murojaat joylashtirilgan funksiyalar esa **rekursiv** funksiyalar deb ataladi.

Avvalgi bobdagi rekkurent formulalar rekursiv murojaatni tushunish uchun sodda misollar bo‘lib xizmat qiladi.

Namuna-1. N butun manfiymas son berilgan. N faktorialni rekursiv funksiya yordamida hisoblash dasturini tuzing.

Yechish. Masalani yechish uchun natija N! qiymatining tahlilidan boshlaymiz. Ma'lumki, N!=N·(N–1)!, ya'ni N! ni hisoblash uchun avval (N–1)! ning qiymatini, (N–1)!=(N–1)·(N–2)! bo'lganligidan (N–1)! qiymatini bilish uchun esa (N–2)! qiymatini, va h.k. bilish kerak. Lekin bilamizki, boshlang'ich qiymatlar 1!=1 va 0!=1 aniqdir. Shularga asoslanib N! ni hisoblash uchun rekkurent formula keltirib chiqaramiz: N!=(N–1)!*N. Endi rekursiv funksiya tuzib olishga tayyormiz. Rekursiv funksiya tuzish jarayonida dasturlash tillariga oid ba'zi tushuncha va qoidalarni eslatib o'tamiz.

Rekursiv funksiya tuzish qat'iy ketma-ketliklarga asoslanadi:

1. Funksiyaga nom berish. Funksiya nomi dasturlash tilining identifikatori talablariga mos istalgancha bo'lishi mumkin. Ba'zi katta dasturlarda funksiyalar soni ko'p bo'lishi mumkinligini hisobga olib funksiyaga vazifasiga mos **factorial** nomini berish maqsadga muvofiqdir.

2. Funksiya natijasi. Rekursiv funksiya tuzishdan avval funksiyaning natijasini aniqlab olishimiz zarur. Hisoblanayotgan N! soni yagona hamda butun son, demak, funksiyamiz yagona butun qiymat qaytarishi kerak. Natijaning juda tez o'sishini va katta son bo'lishini bilgan holda C++ tilida **long long** yoki **unsigned long long** turda tavsiflashimiz mumkin. Ya'ni masalan:

long long factorial(...){...};

3. Funksiya argumentlari. Keyingi qadamda funksiya argumentlarini aniqlab olamiz. Yuqorida aytib o'tilganidek, N! ni hisoblash uchun (N–1)! ni bilishimiz zarur. Shu sababli argument sifatida faqat butun turdagi N o'zgaruvchini qarash yetarli bo'ladi. U qaysi sonning faktoriali hisoblanayotganini bildiradi:

long long factorial(long long n){...};

4. Funksiya tanasi. Funksiya tanasini yuqoridagi rekkurrent formulaga asoslanib yozamiz:

```
long long factorial(long long n)
{
      return factorial(n-1) * n;
}
```

Funksiyalarning yozilishi birinchi qarashda to'g'ridek tuyulishi mumkin. Aslida esa unday emas. Masalan, N=2 bo'lsin. Funksiya **factorial(2)** da o'z ishini boshlaydi, keyin funksiya tanasida: **factorial(N–1)** qismida **factorial(1)** ga murojaat qiladi, ya'ni **factorial(0)** ga, **factorial(–1)** ga, va hokazo, murojaat qiladi. Ya'ni funksiya to'xtamasdan ishlayveradi (dasturchilar tilida: siklga tushib qoladi). Bu esa funksiya tanasidagi xatolikdan dalolat bermoqda. Xatolik qayerda? Xatolik shundaki, funksiyadan chiqish, ya'ni funksiya ishini to'xtatishi uchun kerakli shart kiritilmaganidadir. **Bu rekursiv funksiyalarda eng muhim qadamlardan biridir**.

5. Funksiyadan chiqish sharti. Rekursiv funksiya qandaydir holatda hisoblash bajarmasdan faqat qiymat qaytarishi kerak va bu **funksiyadan chiqish** deyiladi. Chunki faqat shu holatda rekursiv funksiya o'z ishini to'xtatadi va "orqaga qarab yurishni" boshlaydi. Ya'ni joriy holatdan kelib chiqishi mumkin bo'lgan boshqa holatlarga o'tmaydi va javob qaytaradi. Bu holatning kerakli tomoni shundaki, bu holatdan keyin hosil bo'ladigan qiymatlar xato yo'lga olib boradi. Masalaning maqsadi N!=1*2*...*N sonini hisoblash bo'lgani uchun, bizga –1!, –2!, ... faktorialning qiymatlari kerak emas (bu xato yo'l). Demak, rekursiv funksiya tuzishda ahamiyatli qadamlardan biri funksiyadan chiqish shartini to'g'ri aniqlashdir. Shu sababli rekursiv funksiya tanasini yozish jarayonida, birinchi navbatda, funksiyadan chiqish shartini yozib olish tavsiya etiladi.

Demak, N! sonini hisoblash uchun rekursiv funksiya quyidagi ko'rinishga ega bo'ladi:

```
long long factorial(long long n)
{
      if(n==0)return 1;
      return factorial(n-1)*n;
}
```

Rekursiv **`factorial()`** funksiya bajarayotgan ishini tahlil etamiz. Agar asosiy dastur tanasida funksiyaga **`factorial(n)`** kabi murojaat etilsa, funksiya N! sonini hisoblab beradi: **factorial(n)** ichida **factorial(n–1)**, uning uchida **factorial(n–2)** va h.k. joylashgan bo'ladi. Masalan, N=5 bo'lsin:

Factorial(5) 5≠0 (shuning uchun factorial(4) ni hisoblashga o'tadi)
  Factorial(4) 4≠0
    Factorial(3) 3≠0
      Factorial(2) 2≠0
        Factorial(1) 1≠0
          Factorial(0), 0=0 va factorial(0) **1** qiymatni qaytaradi
        Factorial(1) //factorial(0)*1 qiymatni qaytaradi: 1*1=**1**
      Factorial(2) //factorial(1)*2 qiymatni qaytaradi: 1*2=**2**
    Factorial(3) //factorial(2)*3, ya'ni 2*3=**6**
  Factorial(4) // factorial(3)*4, ya'ni 6*4=**24**
Factorial(5) // factorial(4)*5 = **120**

Funksiya o'z ishini tugatadi va **120** qiymatni qaytaradi. Javob: **5!=120.**

Dasturning to'liq ko'rinishi quyidagicha bo'lishi mumkin:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
 	
`long long factorial(long long n)`
`{ if(n==0)return 1;`
`return factorial(n-1)*n;}`
 	
`int main(){`
`int n;`
`n=10;`
`cout<<n<<"!="<<factorial(n)<<endl;`
`n=5;`
`cout<<n<<"!="<<factorial(n)<<endl;`
`return 0;`
`}`
	`10!=3628800`
	`5!=120`

Namuna-2. Natural N soni uchun N-Fibonachchi soni F_N ni hosil qiluvchi dastur tuzing.

Yechish. Masalani rekursiv algoritm yordamida yechish uchun rekkurent formuladan foydalanamiz: $F_N=F_{N-2}+F_{N-1}$. Ma'lumki, $F_1=0$ va $F_2=1$. Rekursiv funksiyani tuzishga o'tamiz:

1. Funksiyaga nom berish. Fibonachchi sonlarini hosil qilayotganimiz uchun funksiya nomini **fib** deb ataymiz.

2. Funksiya natijasi. Fibonachchi sonlari butun sonlar bo'lgani uchun bizning funksiyamiz qaytaradigan qiymat ham butun son bo'ladi, demak long long.

3. Funksiya argumentlari. Dastur davomida faqatgina nechanchi Fibonachchi soni hosil qilinayotganini eslab turish yetarli, shuning uchun funksiya argumenti sifatida faqat **N** butun sonini olish kifoya.

4. Funksiyadan chiqish sharti. Avval aytib o'tilganidek, funksiya tanasini yozishdan avval funksiyadan chiqish shartini yozib olish shart. Masala shartidan Fibonachchi sonlarining birinchi va ikkinchi elementlari ma'lum. Shuning uchun funksiyadan Fibonachchi elementlarining birinchi va ikkinchi elementlarini hisoblash vaqtida chiqiladi.

5. Funksiya tanasi. Yuqoridagi rekkurent formulaga ko'ra: fib(n)=fib(n–2)+fib(n–1);

Demak, rekursiv funksiya quyidagi ko'rinishga ega bo'ladi:

```
long long fib(long long n)
{
      If (n==1) return 0;
      If (n==2) return 1;
      return fib(n-2)+fib(n-1);
}
```

Namuna-3. Rekursiya asosida N sonining raqamlari yig'indisini hisoblash dasturini tuzing.

Yechish.

1. Funksiya nomini **digsum** deb ataymiz (dig – digit – raqam, sum – summa – yig'indi).
2. Funksiya natijasi butun son (raqamlar butun turda).
3. Funksiya argumenti bitta: N soni (butun turda).
4. Funksiyadan chiqish sharti: N=0 bo'lganda (N sonining raqamlari qolmasa).
5. N soni biror k ta xonali desak, uning raqamlari yig'indisini hisoblash uchun birinchi k-1 ta xonadagi raqamlari yig'indisiga oxirgi k-chi xonadagi raqamni qo'shish kerak, ya'ni:

digsum(n)=digsum(n / 10)+(n %10).

Rekursiv funksiyaning umumiy ko'rinishi:

```
long long digsum(long long n)
{
      If (n==0) return 0;
      return digsum(n-1)+(n%10);
}
```

Namuna-4. Rekursiya asosida butun A va B sonlarining eng katta umumiy bo'luvchisini (EKUB(A,B) ni) hisoblash dasturini tuzing.

Yechish. Masalani yechishda umumlashgan Evklid algoritmidan foydalanamiz. Evklid algoritmiga asosan EKUB(A,B)=EKUB(min(A,B), max(A,B)%min(A,B)), ya'ni A va B sonlarining eng katta umumiy bo'luvchisi shu sonlarning kichigining va shu sonlardan kattasini kichigiga bo'lgandagi qoldiqning eng katta umumiy bo'luvchisiga teng. Endi bu algoritmga asoslanib rekursiv algoritm tuza olamiz.

1. Funksiya EKUB ni hisoblayotgani uchun **ekub** deb nom berish maqsadga muvofiq.
2. Funksiyamiz butun qiymat qaytaradi.
3. Funksiya argumentlari sifatida A va B butun sonlar olinadi.
4. Funksiyadan A va B sonlardan birontasi 0 ga teng bo'lib qolganda chiqamiz. Bunda 0 ga teng bo'lmagan son A va B sonlarining EKUB ini beradi.
5. Funksiya tanasida yuqorida ifodalangan Evklid algoritmidan foydalanamiz.

```
long long ekub(long long a, long long b)
{
    if (b==0) return a;
    return ekub(b,a%b);
}
```

Funksiya tanasidagi ifodani shartli o'zlashtirish operatori yordamida qisqaroq ko'rinishda yozish mumkin. Bu holga mos dasturning to'liq ko'rinishi quyidagicha bo'lishi mumkin:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
 	
`int ekub(int a, int b){`
`  return (b==0)?a:ekub(b,a%b);}`
 	
`int main(){`
`  int x, y;`
`  cin>>x>>y;`
`  cout<<"EKUB("<<x<<","<<y<<")="<<ekub(x,y);`
`  return 0;`
`}`
	`21 14`
	`EKUB(21,14)=7`

Namuna-5. Rekursiya asosida berilgan butun A va N sonlari uchun $A^N\%(10^9+7)$ qoldiqni hisoblash dasturini tuzing ($0< A \le 10^9$, $1 \le N \le 10^{18}$).

Yechish. N juda katta son bo'lishi mumkinligini e'tiborga olsak, u holda masala yechimini oddiy **for** operatoridan foydalanib hisoblash xatolikka olib kelishi mumkin. Shu sababli **darajaga binar ko'tarish** usulidan foydalanamiz. Bunda quyidagi ikki xossani e'tiborga olamiz:

$$A^N = \begin{cases} A^{N-1} * A, & \text{agar N toq bo'lsa,} \\ A^{\frac{N}{2}} * A^{\frac{N}{2}}, & \text{agar N juft bo'lsa.} \end{cases}$$

1. Funksiyaga **daraja** deb nom beramiz.
2. Funksiya natijasi butun son.
3. Funksiya argumentlari sifatida A va N ni olamiz.
4. Funksiyadan chiqish sharti: agar N=1 bo'lsa, funksiya **A** qiymat qaytaradi. Ya'ni A sonining 1-darajasi A ga teng. Undan oldingi darajalar (0, –1, –2, ...) masala uchun kerak emas.
5. Yuqoridagi formulaga asosan funksiya tanasini quyidagicha yozish mumkin:

```
const int b=1e9+7;
long long daraja(long long a, long long n){
    If (n==1) return a%b;
    long long y;
    if (n%2==0){ y= daraja(a,n/2); return (y*y)%b;}
       else return (a* daraja(a,n-1))%b;
}
```

Namuna-6. N ta sonni kiritib, ularni teskari tartibda chiqaruvchi dastur tuzing.

Yechish. Bu masala **massiv** yordamida osongina bajariladi, lekin bizning maqsadimiz masalani massivlarsiz hal etishdir. Masaladagi muammo: kiritilayotgan sonlarni xotirada

saqlab qolishdir! Muammoni qanday hal etish mumkin? Albatta rekursiya asosida. Rekursiya tanasida tavsiflangan o'zgaruvchi yordamida sonlarni kiritib olish orqali xotirada saqlash imkoniyatiga ega bo'lamiz. Chunki bu holda har safar rekursiv funksiyaga murojaat etilganda "eski son" xotirada saqlanib qoladi va shu bilan birga, "yangi son" ni kiritib borish mumkin bo'ladi. Ya'ni sonlar teskari tartibda kiritib borilsa, u holda dastlab N-son, keyingi qadamda (N–1)-son, ..., va nihoyat N=0 bo'lganda, ya'ni kiritiladigan son qolmaganida, kiritish to'xtatiladi va funksiyadan chiqiladi. Har safar funksiyadan chiqilayotganda xuddi shu qadamda kiritilgan son ekranga chiqarib boriladi, ya'ni 1-son, 2- son, ..., N-son. 0-son mavjud emas, shuning uchun N=0 bo'lganda rekursiv funksiyadan chiqiladi (orqaga qarab yurish boshlanadi). Endi navbat rekursiv funksiyani tuzishga:

1. Funksiyaga **revord** deb nom beramiz (reverse – "teskari", ord – order – "tartib").

2. Funksiya bajaradigan ish faqat sonlarni kiritish va chiqarish, ya'ni funksiya hech qanday qiymat qaytarmaydi. Demak, C++ tilida **void** turida bo'ladi.

3. Funksiya argumenti sifatida yagona N butun sonini berish mumkin, chunki faqatgina nechanchi sonni kiritilayotganini (chiqarilayotganini) eslab turish kifoya.

4. Yuqorida aytib o'tilganidek, funkisyadan chiqish sharti N=0.

5. Funksiya tanasini yozishga kirishamiz. Bunda, har doimgidek, dastlab funksiyadan chiqish sharti tekshirib olinadi. So'ngra k-son kiritiladi. Undan keyin esa rekursiv funksiyaga murojaat qilinadi ((k–1)-hadga) va h.k. Keyingi qatorda k-son chiqariladi.

```
void revord(int k){
    if (k==0) return;
    int a; cin>>a;
    revord(k-1); cout<<a<<' ';}
```

Dasturning to'liq ko'rinishi quyidagicha bo'lishi mumkin:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
 	
`void revord(int k){`
`  if (k==0) return;`
`  int a; cin>>a;`
`  revord(k-1); cout<<a<<' ';}`
 	
`int main(){`
`  int n;`
`  cout<<"N="; cin>>n;`
`  revord(n);`
`  return 0;`
`}`
	`N=5`
	`5 4 3 2 1`
	`1 2 3 4 5`

5-§. MASSIVLARGA OID MASALALAR YECHISH NAMUNALARI

Dasturlashga oid masalalarda massivlar katta ahamiyat kasb etadi. Masalalarning ba'zilari massivlarsiz ham hal etilishi mumkin, lekin har bir masalada ba'zi muammolarni hal etishga to'g'ri keladi. Massiv qo'llanganda esa masala dasturi matni qisqa va mazmuni sodda bo'lib qoladi.

Massivlarga oid berilgan masalalarda massiv o'lchami uchun yuqori chegara qiymati ko'rsatilmasligi ham mumkin. Bunday hollarda massiv o'lchami dasturlash tili beradigan imkoniyatdan ortib ketmasligi nazarda tutilgan bo'ladi.

Massiv indeksining 0 dan boshlanishi ba'zan dasturchining sanog'ini adashtirib yuborishi, indeksga mos ifodalarni murakkablashtirishi yoki boshqa muammolar hosil qilishi mumkin. Bunday holatlar bo'lmasligi uchun massivning elementlarini 1-indeksdan boshlab to'ldirish mumkin. Xotirada ajratilgan joy ba'zan yetmay qolishi yoki boshqa biror sababli dastur ishiga ta'sir ko'rsatishini hisobga olib massiv tavsiflanganda o'lchamini keragidan biroz kattaroq berish kifoya.

Agar masalalarda butun turdagi massiv elementlari uchun qiymat chegarasi ko'rsatilmagan bo'lsa, u holda, odatda, yechim uchun **int** turida tavsiflash yetarli bo'ladi.

Namuna 1. N ta elementli butun qiymatli A massiv berilgan. Massivning eng katta va eng kichik elementlarini aniqlang.

Yechim. Albatta, bu masala yechimini **min_element** va **max_element** funksiyalari asosida ham yozish mumkin. Biroq, biz qo'llamoqchi bo'lgan algoritm bir tomondan, bu funksiyalar bajaradigan ishni ifodalab bersa, ikkinchi tomondan, masaladagi boshqa shartlarga ko'ra mazkur funksiyalarni qo'llab bo'lmaganda foydalidir, masalan, Namuna – 2 masalasidagi kabi shartlarda.

Massivning eng katta (eng kichik element uchun ham shu tarzda amallar bajariladi) elementini aniqlash uchun quyidagicha tartibda ish bajaramiz:

1) massivning 1-elementini maksimal son deb olamiz;

2) maksimal son (ya'ni 1-element) bilan massivning 2-elementini taqqoslaymiz va agar 2-element maksimal sondan katta bo'lsa, u holda maksimal son deb 2-elementni olamiz;

3) maksimal son bilan massivning 3-elementini taqqoslaymiz va agar 3-element maksimal sondan katta bo'lsa, u holda maksimal son deb 3-elementni olamiz;

...

N) maksimal son bilan massivning N-elementini taqqoslaymiz va agar N-element maksimal sondan katta bo'lsa, u holda maksimal son deb N-elementni olamiz.

Algoritmdan ko‘rinib turibdiki, massivning har bir elementi qarab chiqilmoqda va N ta qadamdan so‘ng kerakli natija olinadi.

Masalada minimal va maksimal elementlarni aniqlash talab qilingani uchun butun turdagi **mina** va **maxa** o‘zgaruvchilarni kiritamiz. Massiv elementlarini kiritish va qayta ishlash (ya’ni taqqoslashda har bir elementni qarab chiqish) uchun takrorlash operatorlaridan foydalanamiz. Takrorlash operatori esa, o‘z navbatida, yana bir o‘zgaruvchi kiritishini talab qiladi. Dasturi quyidagicha:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  int n,t,mina,maxa;`
`  cout<<"N= "; cin>>n;`
`  int a[n+1];`
`  for(t=1;t<=n;t++)cin>>a[t];`
`  mina=a[1]; maxa=a[1];`
`  for(t=2;t<=n;t++){`
`  if(mina>a[t])mina=a[t];`
`  if(maxa<a[t])maxa=a[t]; }`
`  cout<<"Javob: "<<maxa<<" "<<mina;`
`  return 0;`
`}`
	`N= 5`
	`7 21 -1 -23 0`
	
	`Javob: 21 -23`

Dasturda A massiv o‘lchamning qiymati kiritilgandan so‘ng tavsiflangani va elementlari 1 dan boshlab tartiblangani uchun xotirada faqatgina A[0] element uchun ajratilgan joygina bo‘sh qoldi. Umuman, katta xotiraga ega kompyuterlar uchun yuzlab bu kabi bo‘sh qolgan joylar muammo keltirib chiqarmaydi.

Dasturda **mina** va **maxa** o‘zgaruvchilarning boshlang‘ich qiymati (etaloni) sifatida olinadigan sonlarning ahamiyati katta. Masalan, dasturda etalon sifatida massiv element-larining birortasiga ham teng bo‘lmagan sonlar olinsa, ya’ni **mina = 100** va **maxa = –100** bo‘lsa, u holda quyidagicha xato natija olinardi:

Dastur (xato)	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  int n,t,mina,maxa,minind,maxind;`
`  cout<<"N= "; cin>>n;`
	`N= 3`
	`-1963 -230 -101`
	
	`Javob: -100 -1963`

```
    int a[n+1];
    for(t=1;t<=n;t++)cin>>a[t];
    mina=100;   maxa=100;
    for(t=1;t<=n;t++){
    mina=min(mina,a[t]);
    maxa=max(maxa,a[t]); }
    cout<<"Javob: "<<maxa<<" "<<mina;
    return 0;
}
```

Boshlang'ich qiymatlar sifatida boshqa qiymatlar olinsa ham xato natija kelib chiqishi ehtimoli bo'ladi, chunki massiv elementlari ichida boshlang'ich qiymatdan katta va kichik sonlar bo'lishi mumkin.

Bunday xatolik kelib chiqmasligi uchun etalon sifatida massiv elementlaridan birini (yuqoridagi dasturdagi **mina=a[1]; maxa=a[1];** kabi) yoki mina va maxa o'zgaruvchilar tavsiflangan turning eng katta va eng kichik qiymatlarni olish mumkin, ya'ni: **mina=INT_MAX; maxa=INT_MIN;**

Namuna 2. N ta turli elementli butun qiymatli A massiv berilgan. Massivning eng katta va eng kichik elementlari o'rnini almashtirib massivni chiqaring.

Yechim. Masalada talab qilingan almashtirishni bajarish uchun minimal va maksimal elementlarning tartib raqamini aniqlash yetarli. Dastur ishi davomida bu tartib raqamlarni saqlab turish uchun butun turdagi **minind** va **maxind** nomli o'zgaruvchilarni kiritamiz. Lekin dastur ishida minimal va maksimal elementlarni aniqlash uchun taqqoslab borish zarur bo'ladi. Minimal va maksimal elementlar uchun esa, avvalgi misol kabi, butun turdagi **mina** va **maxa** o'zgaruvchilarni kiritamiz. Massiv elementlarini kiritish va qayta ishlash (ya'ni taqqoslashda har bir elementni qarab chiqish) uchun takrorlash operatorlaridan foydalanamiz. Takrorlash operatori uchun butun turdagi t o'zgaruvchini tavsiflaymiz. Yechim dasturini quyidagicha tuzish mumkin:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  int n,t,mina,maxa,minind,maxind;`
`  cout<<"N= "; cin>>n;`
`  int a[n+1];`
`  for(t=1;t<=n;t++)cin>>a[t];`
`  mina=a[1]; minind=1;`
`  maxa=a[1]; maxind=1;`
	`N= 5`
	`7 21 -1 -23 0`
	`Javob:`
	`7 -23 -1 21 0`

```
    for(t=2;t<=n;t++){
    if(mina>a[t]){mina=a[t]; minind=t;}
    if(maxa<a[t]){maxa=a[t]; maxind=t;}
                        }
    swap(a[minind], a[maxind]);
    cout<<"Javob: "<<endl;
    for(t=1;t<=n;t++)cout<<a[t]<<" ";
    return 0;
}
```

Namuna 3. N ta (N>1) elementli butun qiymatli A massiv berilgan. Massivning juft indeksli elementlari yig'indisidan toq indeksli elementlari yig'indisini ayirib chiqaring.

Yechim. Avval ko'rilgan ketma-ketliklarning yig'indisini hisoblagan usulda massiv elementlari yig'indisini hisoblaymiz. Masalada juft va toq indekslar haqida so'z borgani uchun har safar indekslarni juft yoki toqligini tekshirib, mos yig'indilarni hosil qilamiz. Dasturi quyidagicha:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  int n,t,sumjuft=0, sumtoq=0;`
`  cout<<"N= "; cin>>n;`
`  int a[n+1];`
`  for(t=1;t<=n;t++)cin>>a[t];`
`  for(t=1;t<=n;t++)`
`  if(t%2==0)sumjuft+=a[t];`
`  else sumtoq+=a[t];`
`  cout<<"Javob: "<<sumjuft-sumtoq;`
`  return 0;`
`}`
	`N= 5`
	`5 -1 -3 4 7`
	`Javob: -6`

Namuna 4. Ox to'g'ri chizig'ida N ta (N>1) nuqta o'zining koordinatalari orqali berilgan. Shu nuqtalardan bir-biriga eng yaqin bo'lgan ikkitasini aniqlang.

Yechim. Barcha nuqta koordinatalarini kiritish va ular orasidagi masofalarni hisoblash uchun takrorlash operatoridan foydalanamiz. Agar berilgan nuqtalardan ikkitasi ustma-ust tushsa, ular orasidagi masofa d=0 bo'ladi. Bunday nuqtalarni bitta nuqta sifatida qaraymiz. Demak, bizni qiziqtiradigan nuqtalar orasidagi masofa d>0 va boshqa nuqtalar orasidagi noldan farqli masofalar ichida eng kichigidir.

Avval boshlang'ich qiymat (ya'ni etalon: Namuna 1 ga qarang) sifatida olish mumkin bo'lgan noldan farqli **mind** masofani aniqlaymiz. Buning uchun 1-nuqta bilan 2-, 3-, …, N-

nuqtalar orasidagi masofalarni qarab chiqamiz. Agar 1-nuqta uchun noldan farqli masofani aniqlay olmasak, unda 2-nuqta bilan 3-, 4-, ..., N-nuqtalar orasidagi masofalarni qaraymiz. Shu kabi usulda davom ettirib birinchi noldan farqli masofani **mind** va mos nuqtalar indekslarini esa **indx1** hamda **indx2** deb olib jarayonni to'xtatamiz. Agar barcha nuqtalar ustma-ust tushsa, bu haqida xabar chiqaramiz.

Keyingi bosqichda yuqoridagi kabi tartibda tekshirib noldan farqli masofaga ega nuqtalar ichidan orasidagi masofa **mind** sonidan kichiklarini qidiramiz.

Masalada sonli qiymatga ega koordinatalarning turi aytilmagan, demak, dasturda aniqlikka ta'sir etmaydigan ayirish amali ishlatilgani uchun ularni **double** turda tavsiflaymiz.

Dastur	Natijasi

`#include <iostream>`
`#include <cmath>`
`using namespace std;`
`int main(){`
`  int n,t,k,indx1,indx2;`
`  cout<<"N= "; cin>>n;`
`  double x[n+1],mind; bool bor=0;`
`  for(t=1;t<=n;t++)cin>>x[t];`
`  for(t=1;t<=n-1&&bor==0;t++)`
`  for(k=t+1;k<=n&&bor==0;k++)`
`  if(x[t]-x[k]!=0)`
`  {mind=abs(x[t]-x[k]);indx1=t;indx2=k;bor=1;}`
`  if(bor==0)cout<<"Nuqtalar ustma-ust tushgan!";`
`  else {`
`  for(t=indx1;t<=n-1;t++)`
`  for(k=t+1;k<=n;k++)`
`  if(x[t]-x[k]!=0&&abs(x[t]-x[k])<mind)`
`  {mind=abs(x[t]-x[k]);indx1=t;indx2=k;}`
`  cout<<"Nuqtalar: "<<indx1<<" "<<indx2;`
`  }`
`  return 0;`
`}`
	`N= 5`
	`5.6 -1.23 -3 4 7`
	`Nuqtalar: 1 5`

Dasturda birinchi takrorlash operatorlari ishini noldan farqli **mind** topilgan zahoti to'xtatish uchun **bor** mantiqiy o'zgaruvchidan foydalandik. Bu o'zgaruvchi barcha nuqtalar ustma-ust tushgan holda xabar chiqarish uchun ham foydali bo'ldi.

Namuna 5. Sanayotgan kishi atrofida doira bo'ylab N ta kishi turibdi. Har bir kishiga 1 dan N gacha tartib raqami berilgan. Sanovchi doira bo'ylab chapdan o'ngga qarab 1 dan M gacha sanab boradi va M-sanalgan kishi doiradan chiqadi, sanovchi esa keyingi kishida

sanoqni yana 1 dan boshlaydi. Sanoq bitta kishi qolguncha davom etadi. Oxirida qolgan kishining tartib raqamini aniqlang.

Yechim. Sanalayotgan (butun turdagi o'zgaruvchi) N ta kishining har biri tartib raqamiga ega bo'lgani uchun a[1], a[2], ..., a[n] butun sonlar massiviga quyidagicha mos qo'yish mumkin: massiv elementi qiymati kishi doirada qolgan bo'lsa 1 ga, doiradan chiqqan bo'lsa 0 ga teng bo'lsin. Massiv elementlarini bittalab qarab chiqish uchun **t** nomli butun turdagi o'zgaruvchi kiritib, ixtiyoriy takrorlash operatorini qo'llash mumkin. Doiradan chiqish sanoqda ishtirok etmaslik bilan bir xil, ya'ni agar a[t]=0 bo'lsa, u holda bu elementga mos t-kishi doiradan chiqqan va sanoqda ishtirok etmaydi. Doirada qolgan kishilarning sonini saqlab turish uchun **son** nomli, sanash uchun esa **s** nomli butun turidagi o'zgaruvchilar kiritish maqsadga muvofiq.

Masala yechimi shartdagidek ish bajaradigan ko'rinishda quyidagicha aks etadi:

Sanoq boshida barcha kishi doirada bo'lgani uchun a[1], a[2], ..., a[N] massiv elementlariga 1 qiymat beriladi. Sanoq A[1] dan boshlanadi va qiymati 1 ga teng har bir element qaralganda S bittaga oshiriladi hamda S=M shart tekshiriladi. Agar S=M shart natijasi True qiymat hosil qilsa, u holda oxirgi qaralgan elementga 0 qiymat beriladi va S=0 deb olinadi, ya'ni sanoq boshidan boshlanadi. Agar K=N bo'lsa, u holda K=1 deb olinadi, bu esa sanoq doiraviy davom etishini bildiradi. Sanoq son=1 bo'lganda to'xtaydi va 0 dan farqli element tartib raqami chiqariladi.

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  int n,t,m,s=0,son;`
`  cout<<"N= "; cin>>n;`
`  int a[n+1];son=n;`
`  cout<<"M= "; cin>>m;`
`  for(t=1;t<=n;t++)a[t]=1;`
`  t=1;`
`  while(son>1){`
`  if(a[t]==1)s++;`
`  if(s==m){s=0;son--;a[t]=0;}`
`  if(t==n)t=0;`
`  t++; }`
`for(t=1;t<=n;t++)`
`  if(a[t]==1)cout<<"Javob: "<<t;`
`  return 0;`
`}`
	`N= 210`
	`M= 70`
	`Javob: 161`

Namuna 6. N ta (N>0) elementli manfiymas butun qiymatli A massiv ($0 \le A[1..N] \le 10^5$) berilgan. Massivdagi turli sonlar sonini aniqlang.

Yechim. Berilgan A massiv elementlari [0; 10^5] sonlar oralig'ida yotgani uchun elementlari soni 10^5+1 tadan oshmaydigan B massiv kiritamiz. B massiv elementlarining boshlang'ich qiymatini va sanoq o'zgaruvchisi S ni 0 ga tenglashtirib, ulardan A massiv elementlarini sanash uchun foydalanamiz. Sanashda 0 dan (N-1) gacha bo'lgan har bir k uchun quyidagi amallarni bajaramiz: B[A[k]]++ va har safar (B[A[k]]==1) bo'lganda, ya'ni yangi son uchraganda S sanoqni bittaga oshiramiz. Masalan, A massiv quyidagicha bo'lsin:

Tartibi	0	1	2	3	4	5	6	7
A[ ]	5	3	1	7	0	5	1	5

U holda

B[A[0]]++ natijasi B[A[0]]=B[5]=1; B[A[1]]++ natijasi B[A[1]]=B[3]=1;
B[A[2]]++ natijasi B[A[2]]=B[1]=1; B[A[3]]++ natijasi B[A[3]]=B[7]=1;
B[A[4]]++ natijasi B[A[4]]=B[0]=1; B[A[5]]++ natijasi B[A[5]]=B[5]=2;
B[A[6]]++ natijasi B[A[6]]=B[1]=2; B[A[7]]++ natijasi B[A[7]]=B[5]=3.

Ko'rinib turibdiki, B massivning 5 ta elementi 1 ga teng ekan, demak, A massivning turli elementlari soni 5 ta bo'lib, ular quyidagilar: 5; 3; 1; 7; 0.

Shu algoritmga mos dastur matnini quyidagicha yozish mumkin:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  int n,t,s=0;`
`  cout<<"N= "; cin>>n;`
`  int a[n],b[100001]={0};`
`  for(t=0;t<n;t++)cin>>a[t];`
`  for(t=0;t<n;t++){`
`  b[a[t]]++; s=s+(b[a[t]]==1);}`
`  cout<<"Javob: "<<s;`
`  return 0;`
`}`
	`N= 8`
	`5 3 1 7 0 5 1 5`
	`Javob: 5`

Mazkur masalaning bundan ham samaraliroq yechimini keyinroq ko'rib chiqamiz.

Namuna 7. N ta elementli sonli A massiv va B soni berilgan. Massivning B soniga teng biror elementi bo'lsa, shu element tartib raqamini aniqlang. Agar bunday element yo'q bo'lsa, u holda bu haqida xabar chiqaring.

Yechim. Massiv elementlarini bittalab B soni bilan taqqoslaymiz. Taqqoslashda B soniga teng element borligini bildiruvchi mantiqiy Bor belgisini kiritamiz. Avval Bor=0 (ya'ni bunday element yo'q) bo'ladi. Agar massiv elementlaridan birortasi B soniga teng bo'lsa Bor=1 deb olamiz hamda element tartibini chop etib takrorlanishni to'xtatamiz. Agar massiv elementlarining birortasi B soniga teng bo'lmasa, ya'ni bu holda Bor=0 bo'ladi, bu haqida xabar chiqaramiz.

Masala shartida B soni va sonli A massiv elementlarining turi aytilmagani uchun **double** turda tavsiflaymiz. Yuqoridagi algoritmga mos dastur matnini quyidagicha yozish mumkin:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  int n,t;`
`  bool bor=0;`
`  cout<<"N= "; cin>>n;`
`  double a[n],b;`
`  for(t=0;t<n;t++)cin>>a[t];`
`  cout<<"B= "; cin>>b;`
`  for(t=0;t<n;t++)`
`  if(a[t]==b){bor=1;`
`  cout<<"Javob: "<<t;break;}`
`  if(bor==0)`
`  cout<<"Bunday element yo'q";`
`  return 0;`
`}`
	`N= 8`
	`5 3 1 7 0 5 1 5`
	`B= 1`
	`Javob: 2`

Qo'llangan algoritm **chiziqli qidiruv** deb atalib, undagi amallar soni **deyarli N** ga teng.

Namuna 8. O'sish yo'nalishida tartiblangan N ta elementli sonli A massiv va B soni berilgan. Massivning B soniga teng biror elementi bo'lsa, shu element tartib raqamini aniqlang. Agar bunday element yo'q bo'lsa, u holda bu haqida xabar chiqaring.

Yechim. Namuna 7 masalasidan bu masala massiv tartiblanganligi bilan farq qiladi. Shuning uchun Namuna 7 da qo'llangan algoritmning amallar soni massiv tartiblangan bo'lgan holda ham **deyarli N** ga teng. Massiv tartiblangan bo'lganda qo'llanadigan quyidagi usul amallar sonini keskin kamaytiradi.

TARTIBLANGAN MASSIVLAR UCHUN BINAR (IKKILIK) QIDIRUV

N ta elementli tartiblangan (o'sish yoki kamayish yo'nalishida) sonli A massiv va B soni berilgan bo'lsin. B soni massivning qaysi elementiga teng bo'lishini aniqlash kerak.

Binar qidiruv mazmuni quyidagidan iborat:

1) Massivning eng chapdagi elementi tartib raqamini Chap deb belgilaymiz, demak, tartib raqami berilishiga bog'liq ravishda Chap=0 yoki Chap=1. Massivning eng o'ngdagi elementi tartib raqamini O'ng deb belgilaymiz, demak, tartib raqami berilishiga bog'liq ravishda O'ng=N-1 yoki O'ng=N. O'rta=(Chap+O'ng)/2 belgilash kiritamiz (eslatma: C++ da butun bo'lish natijasi (Chap+O'ng)/2 butun son bo'ladi). Massivning B soniga teng elementi borligini bildirish uchun Bor o'zgaruvchisini kiritamiz, demak, boshlang'ich holatda Bor=0.
2) Massiv elementlarini 2 guruhga, ya'ni chap va o'ng guruhga ajratamiz. Bunda o'rtadagi bitta A[O'rta] element massivni ikkiga ajratuvchi element bo'lib xizmat qiladi. Massiv tartiblangan bo'lgani uchun A[O'rta]=B yoki B<A[O'rta] yoki A[O'rta]>B shartlardan faqat bittasi bajariladi. Agar A[O'rta]=B bo'lsa, u holda Bor=1 va natija sifatida O'rta qiymati chiqarilib dastur ishi yakunlanadi.
3) Aks holda, ya'ni A[O'rta]≠B bo'lsa, u holda B soni qaysi guruhda ekanligiga bog'liq ravishda yangi Chap yoki O'ng qiymatlarini aniqlaymiz. Agar B soni chap guruhda bo'lsa, ya'ni B<A[O'rta] bo'lsa, u holda O'ng=O'rta-1 kabi, agar B soni o'ng guruhda bo'lsa, ya'ni A[O'rta]>B bo'lsa, u holda Chap=O'rta+1 kabi, so'ngra O'rta=(Chap+O'ng)/2 kabi olamiz;
4) Agar Chap≤O'ng shart bajarilsa (ya'ni hozircha massivning qaralmagan elementlari mavjud), u holda 2-bandga qaytamiz;
5) Aks holda Bor=0 bo'lgani uchun, ya'ni massivning birorta ham elementi B ga teng bo'lmagani uchun bu haqida xabar chiqaramiz.

Bu algoritmda amallar soni deyarli logN ga teng. Masalan, N=1024 bo'lganda chiziqli qidiruv algoritmi deyarli 1024 marta amal bajarsa, binar qidiruv algoritmi deyarli log1024=10 marta amal bajaradi.

Algoritmni quyidagi misolda 8 ta elementli A massiv va B=10 soni uchun tatbiqini ko'ramiz:

Indeks	1	2	3	4	5	6	7	8
A[ ]	2	5	6	7	10	11	19	21

1-qadam. Chap=1, O'ng=8, O'rta=(1+8)/2=9/2=4, Bor=0:

A[O'rta]=A[4]=7 va A[4]≠B;

2-qadam. A[4]<B, demak, Chap=O'rta+1=4+1=5, O'ng=8, O'rta=(5+8)/2=13/2=6:

A[O'rta]=A[6]=11 va A[6]≠B;

3-qadam. A[6]>B, demak, Chap=5, O‘ng=O‘rta-1=6-1=5, O‘rta=(5+5)/2=10/2=5:
A[O‘rta]=A[5]=10 va A[5]=B, demak, Bor=1 va Javob: 5

Algoritmning binar (ikkilik) qidiruv deb atalishi qo‘llanayotgan usuldan kelib chiqqan: har safar qidiruv oralig‘i 2 baravar kichraytirilmoqda.

Endi Namuna 8 masalaning yechimini beradigan binar qidiruv dasturini quyidagicha yozish mumkin (o‘zgaruvchilar nomini inglizcha ifodalaymiz, chunki o‘zbek tilidagi O‘ harfini identifikator nomiga yozib bo‘lmaydi):

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  int n,t,left,right,middle;`
`  bool bor=0;`
`  cout<<"N= "; cin>>n;`
`  double a[n+1],b;`
`  for(t=1;t<=n;t++)cin>>a[t];`
`  cout<<"B= "; cin>>b;`
`  //-------------------------//`
`  left=1; right=n;`
`  while(left<=right&&bor==0){`
`  middle=(left+right)/2;`
`  if(a[middle]==b)bor=1;else {`
`  if(a[middle]<b)left=middle+1;`
`  else right=middle-1;} }`
`  //-------------------------//`
`  if(bor==1)cout<<"Javob: "<<middle;`
`  else cout<<"Bunday element yo‘q";`
`  return 0;`
`}`
	`N= 8`
	`2 5 6 7 10 11 19 21`
	`B= 10`
	`Javob: 5`

Dasturda **binar qidiruv** algoritmining o‘zagi ajratib ko‘rsatilgan.

Namuna 9. Elementlari [1963; 2107] oralig‘ida yotadigan N ta elementli A massivni tasodifiy sonlar generatori yordamida hosil qiling va chiqaring.

Yechim. Barcha dasturlash tillarida bo‘lgani kabi C++ tilida ham massivning **tasodifiy sonlar generatori** mavjud. Masalani yechish uchun avval shu imkoniyatni ko‘rib chiqamiz.

(PSEUDO) RANDOM GENERATORI

C++ dasturlash tilida tasodifiy sonlarni hosil qilish uchun **cstdlib** kutubxonasiga standart **rand()** funksiyasi kiritilgan. Bu funksiya [0, RAND_MAX) diapazonidagi butun sonlarni tasodifiy hosil qiluvchi **random** (ing. random – tasodifiy) generatori bo'lib, RAND_MAX – konstanta. Bu yerda RAND_MAX soni C++ tili konstantasi bo'lib, qiymati dastur ishlayotgan platformaga bog'liq.

Ammo kompyuterda haqiqiy tasodifiy sonlarni hosil qilish mumkin emas va faqat qo'shimcha qurilma (ba'zi zamonaviy kompyuterlar bu qurilmaga ega) yordamida bunga erishish mumkin.

Tasodifiy sonlar generatori **rand()** aslida psevdo (yolg'on ma'nosida) tasodifiy sonlarni hosil qiladi, ya'ni sonlar tasodifiyga o'xshaydi, lekin ular parametrga bog'liq aniq algoritm asosida hosil qilinadi. Tasodifiy sonlar generatori qo'llangan quyidagi dasturni qaraymiz:

Dastur	Natijasi

`#include <iostream>`
`#include <cstdlib>`
`using namespace std;`
`int main(){`
`  int n,t;`
`  cout<<"N= "; cin>>n;`
`  int a[n+1];`
`  for(t=1;t<=n;t++)`
`  a[t]=rand();`
`  cout<<"RAND_MAX= "<<RAND_MAX<<endl;`
`  for(t=1;t<=n;t++)cout<<a[t]<<" ";`
`  return 0;`
`}`
	`N= 4`
	`RAND_MAX= 32767`
	`41 18467 6334 26500`

Yuqoridagi dasturni bir necha marta ishlatish natijasida shuni anglash mumkinki, A massiv elementlari **har safar bir xil qiymat qabul qiladi**! Bunga esa yuqoridagi aytib o'tilgan algoritm sababchidir. Bu muammoni bartaraf etish uchun dasturga **ctime** kutubxonasini va quyidagi ko'rsatmani kiritish zarur:

```
srand(time(0));
```

U holda dastur va uning natijasi quyidagicha ko'rinish oladi:

Dastur	Natijasi

`#include <iostream>`
`#include <cstdlib>`
`#include <ctime>`
`using namespace std;`
`int main(){`
	`N= 4`
	`RAND_MAX= 32767`
	`1970 101 20698 10757`

```
    srand(time(0));
    int n,t;
    cout<<"N= "; cin>>n;
    int a[n+1];
    for(t=1;t<=n;t++)a[t]=rand();
    cout<<"RAND_MAX= "<<RAND_MAX<<endl;
    for(t=1;t<=n;t++)cout<<a[t]<<" ";
    return 0;
}
```

Izoh 10: RAND_MAX qiymati platformaga bog'liqdir. Agar RAND_MAX dan katta sonlar kerak bo'lsa, u holda formula tuzish orqali hosil qilish mumkin. Masalan, 0 dan M gacha (32767<M<65534) bo'lgan a sonni hosil qilish kerak bo'lsa, u holda a=rand()*rand()/65535 kabi yozish mumkin.

Ta'kidlash joizki, yuqoridagi dastur ham haqiqiy (son ma'nosida emas) tasodifiy sonlar hosil qilmaydi, faqat vaqt parametriga bog'liq ravishda aniq algoritm asosida sonlar hosil qiladi. Aytib o'tilganidek, buning uchun maxsus yordamchi qurilma zarur.

Ba'zi holatlarda biror oraliqdagi tasodifiy sonlarni hosil qilish kerak bo'ladi. Bunda quyidagi formulalardan foydalanish mumkin:

Oraliq	Tasodifiy son
[0, N] , bu yerda N natural son	int a = rand() % N;
[N, M], bu yerda N va M butun sonlar: N<M	int a = N + rand() % (M – N)
[0, 1] oralig'ida haqiqiy turda	float r = static_cast <float> (rand()) / static_cast <float> (RAND_MAX);
[0, X], bu yerda X – haqiqiy son	float r2 = static_cast <float> (rand()) / (static_cast <float> (RAND_MAX/X));
[LO, HI], bu yerda LO va HI haqiqiy sonlar: LO<HI	float r3 = LO + static_cast <float> (rand()) /(static_cast <float> (RAND_MAX/(HI-LO)));

Izoh 11: Bu jadvaldagi ba'zi formulalar **stackoverflow.com** saytidan olindi. Shu ma'lumotlar asosida boshqa formulalarni hosil qilish ham mumkin.

Yuqoridagi ma'lumotlar asosida massiv elementlari [1963; 2107] oralig'idagi haqiqiy sonlar ekanligini unutmagan holda masala yechimini quyidagicha yozamiz:

Dastur	Natijasi

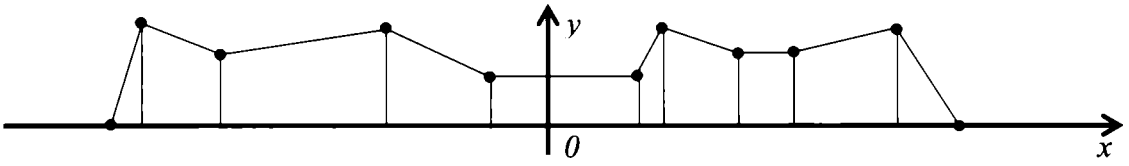
`#include <iostream>`
`#include <cstdlib>`
`#include <ctime>`
`using namespace std;`
`int main(){`
`  srand(time(0));`
	`N= 2`
	`2011.52 1965`

```
    int n,t;
    cout<<"N= "; cin>>n;
    long double a[n+1];
    for(t=1;t<=n;t++)
     a[t]=1963+rand()*1.0/(RAND_MAX/(2107-1963));
    for(t=1;t<=n;t++)cout<<a[t]<<" ";
    return 0;
}
```

Namuna 10. Oxy tekisligida N ta nuqta butun (x; y) koordinatalari orqali berilgan (y>0). Berilgan nuqtalardan faqat eng chapdagi va eng o'ngdagi nuqtalarning ikkinchi koordinatasi 0 ga teng, qolgan barcha nuqtalarning birinchi koordinatasi bir-biridan farqlanadi. Shu nuqtalar birinchi koordinatasiga nisbatan tartiblanganda Ox o'qi bilan birgalikda hosil qilgan shakl yuzini aniqlang.

Yechim. Masala shartidan eng chapdagi va eng o'ngdagi nuqtalar aniq Ox o'qida yotishini, qolgan (N-2) ta nuqtalar esa Ox o'qidan yuqorida yotishini tushunish mumkin.

Tasavvur qilish uchun shakl va ba'zi yordamchi kesmalarni chizib olamiz.



Agar asoslaridan birining uzunligi 0 ga teng trapetsiyaning uchburchakka aylanishini e'tiborga olsak, u holda chizmadan ko'rinadiki, shakl yuzasi (N-1) ta trapetsiyaning yuzalari yig'indisiga teng ekan. Bunda har bir yonma-yon nuqta uchun birinchi koordinatalar farqi trapetsiyaning balandligi, ikkinchi koordinatalar trapetsiyaning asoslari bo'lib xizmat qilar ekan. Trapetsiya yuzi formulasiga va aytilganlarga ko'ra quyidagi rekkurent amal ifodasini yozishimiz mumkin:

$$S=S+(y[k] + y[k+1])*(x[k+1] - x[k])/2,$$

bu yerda (x[k]; y[k]) va (x[k+1]; y[k+1]) nuqtalar yonma-yon joylashgan nuqtalar.

Masalada nuqtalar qanday tartibda kiritilishi aytilmagani uchun ularni birinchi koordinatasi bo'yicha tartiblab olish zarur bo'ladi. Lekin C++ tilining **sort** funksiyasidan foydalanish xatolikka olib keladi, chunki har bir nuqta 2 ta koordinatalar juftligi orqali ifodalanadi va bu koordinatalar bir-biri bilan qat'iy bog'liqdir. **Ya'ni agar ikkita birinchi koordinata o'rnini almashtirsak, u holda ularning ikkinchi koordinatasi o'rnini ham almashtirishimiz shart**. Demak, tartiblash uchun yuqorida keltirilgan oddiy tanlov, oddiy almashtirish yoki oddiy joylashtirish algoritmlaridan birortasini ozgina o'zgartirish bilan qo'llashimiz kerak ekan.

Biz bu masalada C tilida **qsort** (funksiyasi) yoki dasturlash tillarida **QuickSort** (tezkor tartiblash) deb ataladigan va **deyarli N^2** da, lekin juda ko'p amaliy masalalarda **deyarli N·logN** da ishlaydigan quyidagi tartiblash algoritmidan foydalanamiz.

TEZKOR (QUICKSORT) TARTIBLASH ALGORITMI

Bizga indekslari 1 dan N gacha bo'lgan bir o'lchovli A massiv berilgan bo'lsin.

Bu tartiblash usulida ham **binar qidiruv algoritmi** kabi massiv elementlarini ikki guruhga bo'lishdan foydalaniladi:

1) massiv o'rtasidagi element **Markaz** sifatida tanlab olinadi, ya'ni **Markaz**=A[(1+N)/2] (eslatma: C++ da butun bo'lish natijasi (1+N)/2 butun son bo'ladi);

2) Massivning **Markaz** dan kichik elementlarini **Markaz** dan chap qismga, **Markaz** dan katta elementlarini **Markaz** dan o'ng qismga o'tkaziladi;

3) Massivning **Markaz** dan chapdagi va o'ngdagi qismlari uchun 1-2 bandlar qo'llanadi.

Algoritmni izohlash uchun quyidagi misolni ko'ramiz:

1	2	3	4	5	Indekslar
7	2	2	-1	6	saralanmagan

7	2	2	-1	6	(1+5)/2=3
1 ↑		Markaz		N ↑	Markaz=2

7	2	2	-1	6	7>2>-1
c ↑		Markaz	u ↑		swap(7,-1)

-1	2	2	7	6	c>u
	u ↑	Markaz c ↑			to'xtadi

Massivda **Markaz** dan chap qismi tartiblanib qolgani uchun endi faqat **Markaz** dan o'ng tomon tartiblanadi.

1	2	3	4	5	Indekslar
-1	2	2	7	6	(4+5)/2=4
			Markaz	↑	Markaz=7

-1	2	2	7	6	7≥7>6
			Markaz c ↑	u ↑	swap(7,6)

-1	2	2	6	7	c>u
			Markaz u ↑	c ↑	to'xtadi

Algoritmning izohlaridan ko‘rinadiki, dastur rekursiv funksiya qo‘llashga mos ekan. Algoritmga mos dasturni quyidagicha yozish mumkin:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
 	
`void tezkor(int arr[], int left, int right) {`
`  int markaz,c,u;`
`  c=left; u=right;`
`  markaz=arr[(left+right)/2];`
`  do {while(arr[c] < markaz) c++;`
`  while(arr[u] > markaz) u--;`
`  if (c<=u) {swap(arr[c],arr[u]); c++; u--;}`
`  } while (c<=u);`
 	
`  if (left<u) tezkor(arr,left, u);`
`  if (c<right) tezkor(arr,c, right);`
`  return;`
`}`
`int main(){`
`  int t,n;`
`  cout<<"N= ";cin>>n;`
`  int a[n];`
`  for(t=1;t<=n;t++)cin>>a[t];`
`  tezkor(a,1,n);`
`  for(t=1;t<=n;t++)cout<<a[t]<<" ";`
`  return 0;`
`}`
	`N= 5`
	`7 2 2 -1 6`
	`-1 2 2 6 7`

Endi **Namuna 10** yechimini quyidagicha yozish mumkin:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`void tezkor(int arr[], int arr2[],int left, int right) {`
`  int markaz,chap,ung;`
`  chap=left; ung=right;`
`  markaz=arr[(left+right)/2];`
`  do {while(arr[chap] < markaz) chap++;`
`  while(arr[ung] > markaz) ung--;`
	`N= 6`
	`3 2`
	`-1 1`
	`-5 0`
	`0 3`
	`4 0`
	`-4 1`
	`S= 14`

```
        if (chap <= ung) {
            swap(arr[chap],arr[ung]);
            swap(arr2[chap],arr2[ung]);
            chap++; ung--;}
    } while (chap<=ung);

  if (left<ung) tezkor(arr,arr2,left, ung);
  if (chap<right) tezkor(arr,arr2,chap, right);
  return;
}
int main(){
    int t,n; long double s=0;
    cout<<"N= ";cin>>n;
    int x[n+1],y[n+1];
    for(t=1;t<=n;t++)cin>>x[t]>>y[t];
    tezkor(x,y,1,n);
      for(t=1;t<=n-1;t++)
            s=s+(y[t]+y[t+1])*(x[t+1]-x[t])*1.0/2;
        cout<<"S= "<<s;
      return 0;
}
```

Namuna 11. Natural N va M sonlari berilgan. Butun qiymatli N ta satrli va M ta ustunli A massivning satrlari ichidan satr elementlari modullari yig'indisining eng kattasini aniqlang.

Yechim. Massivimiz ikki o'lchovli bo'lgani uchun satr tartib raqamini sanab borish uchun **s**, ustun tartib raqamini sanab borish uchun **u**, masala yechimi bo'ladigan satr tartib raqami uchun **sm**, satr elementlari modullari yig'indisi uchun **y**, maksimal yig'indi uchun **my** o'zgaruvchilarini kiritamiz.

Avval **my**= –**1** kabi olamiz. Har safar yangi satrdagi elementlar modullari yig'indisi **y** ni hisoblagach, **y** va **my** ni taqqoslaymiz. Agar biror **k**-satr uchun **y>my** bo'lsa (hech bo'lmasa, 1-satr uchun bu shart o'rinli bo'ladi), u holda **my=y** va **sm=k** deb xotirada saqlab boraveramiz.

Masala yechimiga mos dasturni ichma-ich joylashgan sikllar asosida quyidagicha yozish mumkin:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  int n,m,s,u,sm,y,my=-1;`
`  cout<<"N= ";cin>>n;`
`  cout<<"M= ";cin>>m;`
	`N= 3`
	`M= 2`
	`-1 1`
	`-5 0`
	`0 3`
	`Satr= 2`

```
    int a[n+1][m+1];
    for(s=1;s<=n;s++)
        for(u=1;u<=m;u++)cin>>a[s][u];
    for(s=1;s<=n;s++){
        y=0;
        for(u=1;u<=m;u++)y=y+abs(a[s][u]);
        if(y>my){my=y;sm=s;}
    }
        cout<<"Satr= "<<sm;
      return 0;
}
```

Namuna 12. "Kiritish.txt" faylining birinchi satrida ketma-ketlik hadlari soni bo'lgan natural N soni, keyingi N ta satrda ketma-ketlikning butun qiymatli A_1, A_2, ..., A_N hadlari berilgan. Ketma-ketlik hadlaridan faqat ikkitasi 0 ga tengligi ma'lum. Shu 0 ga teng ikkita had orasidagi hadlarni o'sish yo'nalishida tartiblang. Hosil bo'lgan ketma-ketlik hadlarini "Chiqarish.txt" fayliga orachiq bilan ajratib yozing.

Yechim. Demak, masala shartidagi ma'lumotlarni o'qib olish va natijani yozish uchun matnli fayl bilan ishlashimiz kerak ekan. Avval shu vazifani amalga oshirish imkoniyatini ko'rib chiqamiz.

C++ TILIDA MATNLI FAYLLAR BILAN ISHLASH

Ma'lumki, massivlar yuzlab, hatto minglab elementdan iborat bo'lishi mumkin. Buncha ma'lumotni klaviatura orqali kiritish uchun qancha vaqt behuda sarf bo'lishini tushunish qiyin emas. Shuning uchun dasturlashda, odatda, katta hajmdagi ma'lumotlar matnli fayldan o'qib olinadi. Bunday ma'lumotlar matnli fayllar sifatida turli usullar bilan hosil qilinadi. Masalan, ba'zi qurilmalarni nazorat testidan o'tkazish vaqtida olingan natijalar maxsus qurilmalar yordamida matnli faylga yozib boriladi.

C++ tilida matnli fayllar bilan ishlashning bir necha xil usullari bor. Qulay imkoniyatlardan biri **freopen** funksiyasidan foydalanish bo'lib, bu funksiyaning umumiy ko'rinishi quyidagicha:

```
FILE* freopen(const char*name, const char* mode, FILE *v);
```

bu yerda **name** – fayl nomi, **mode** – faylni ochish turi, **v** – ochish kerak bo'lgan oqimga ko'rsatkich. O'qiladigan **name** nomli fayl tayyorlanayotgan loyiha papkasida bo'lishi yoki shu faylga to'liq yo'l ko'rsatilishi shart. O'z navbatida yozish fayli loyiha papkasida hosil bo'ladi. Odatda, **freopen** funksiyasi qulay bo'lganligi uchun standart kiritish-chiqarish oqimlari bilan birga ishlatiladi. Misol sifatida quyidagi dasturni keltiramiz:

```
#include <iostream>
#include <cstdio>
using namespace std;
int main(){
    int n;
    freopen("in.txt", "r", stdin);
    freopen("out.txt", "w", stdout);
    cin>>n;
    cout<<5*n;
    return 0;
}
```

Bu dasturda standart kiritish oqimi sifatida **in.txt** fayli, standart chiqarish oqimi sifatida **out.txt** fayllari ochilmoqda. Dastur ishga tushgach **cin** operatori **in.txt** fayli ichidagi ma'lumotni butun turdagi **n** o'zgaruvchiga o'qiydi, so'ngra **cout** operatori **5*n** ifoda qiymatini **out.txt** fayliga yozadi. Agar bundan so'ng boshqa faylni ochish kerak bo'lsa, uni ham xuddi shu usulda ochish kerak bo'ladi. Yangi fayl ochayotgan **freopen** funksiyasi o'zidan oldingi faylni yopadi va yangisini ochadi.

Izoh 12: Agar **freopen** funksiyasi biror platformada ishlamasa, u holda dasturga **cstdio** yoki **fstream** kutubxonalaridan birortasini qo'shish kerak bo'ladi.

Fayl ochish turlari (**mode** o'rniga yozish mumkin bo'lgan qiymatlar) quyidagilar:

mode	Ma'nosi
"r"	Faylni o'qish uchun ochadi (ingl. read – o'qish)
"w"	Faylni yozish uchun hosil qiladi (ingl. write – yozish)
"a"	Fayl davomiga qo'shish uchun ochadi (ingl. append – oxiriga qo'shish)
"rb"	Ikkilik faylini o'qish uchun ochadi
"wb"	Ikkilik faylini yozish uchun hosil qiladi
"ab"	Ikkilik faylini oxiriga qo'shish uchun ochadi
"r+"	Faylni o'qish va yozish uchun ochadi
"w+"	O'qish va yozish uchun fayl hosil qiladi
"a+"	Faylni o'qish va oxiriga qo'shish uchun ochadi
"r+b"	Ikkilik faylini o'qish va yozish uchun ochadi
"w+b"	Ikkilik faylini o'qish va yozish uchun hosil qiladi
"a+b"	Ikkilik faylini o'qish va oxiriga yozish uchun ochadi

Ba'zan qulay bo'lishi uchun ochilgan faylni o'zgaruvchiga biriktirib qo'yish ham mumkin:

```
FILE *f = freopen("in.txt", "r", stdin);
```

Faylda ma'lumotlar tugaganligini bilish uchun fayl oxirini tekshiradigan **feof** funksiyasidan foydalaniladi. Masalan:

```
if(feof(f))cout<<"Ma'lumotlar tugadi";
   else cout <<" Ma'lumotlar hali tugamadi ";
```

Endi **Namuna 12** da berilgan vazifani quyidagi dastur ko'rinishida bajaramiz.

Dastur	Natijasi

`#include <iostream>`
`#include <algorithm>`
`using namespace std;`
`int main(){`
`  int n,t,nol1=-1, nol2;`
`  freopen("Kiritish.txt", "r", stdin);`
`  freopen("Chiqarish.txt", "w", stdout);`
`  cin>>n;`
`  int a[n+1];`
`  for(t=1;t<=n;t++)cin>>a[t];`
`  for(t=1;t<=n;t++)if(a[t]==0)`
`  {if(nol1==-1)nol1=t; else nol2=t;}`
`  sort(a+nol1+1,a+nol2);`
`  for(t=1;t<=n;t++)cout<<a[t]<<" ";`
`  return 0;`
`}`
	`Kiritish.txt da`
	`6`
	`100`
	`0`
	`99`
	`-1`
	`0`
	`1`
	`Chiqarish.txt da`
	`100 0 -1 99 0 1`

Ketma-ketlikda uchraydigan birinchi 0 ga teng had tartib raqami uchun **nol1** va ikkinchi 0 ga teng had tartib raqami uchun **nol2** o'zgaruvchilari kiritildi. Ketma-ketlikda aniq ikkita 0 ga teng had uchrashi **nol1** va **nol2** o'zgaruvchilar qiymatini aniqlashni osonlashtirdi. Tartiblashda **sort** funksiyasi uchun birinchi 0 ga teng haddan keyingi va ikkinchi 0 ga teng haddan oldingi hadlar tartib raqamlari kiritildi.

6-§. MASSIVLARGA DOIR VAZIFALAR

Quyida massivlarga oid turli mazmundagi vazifalar tavsiya etilmoqda. Ba'zi bir masalalarda massiv elementlarini hosil qilish kerak bo'lsa, boshqa bir masalalarda massiv elementlari ustida amallar bajarish, yana bir masalalarda massivlarni tatbiq etish kerak bo'ladi. Agar boshqa talab qo'yilmagan bo'lsa, u holda hosil qilingan yoki qayta ishlangan massivlar chiqarilishi kerak.

Masala shartida massivlar turli usullarda ifodalanishi, masalan, N ta elementli sonlar massivi quyidagicha yozilishi mumkin: "N ta elementli sonlar massivi" yoki "N ta elementli A sonlar massivi" yoki "A[1..N] sonlar massivi" yoki "A[0..N–1] sonlar massivi" kabi.

= A =

Massiv 1. Natural N son berilgan. Birinchi N ta natural toq sonni o'zida saqlovchi massiv hosil qiling va chiqaring.

Massiv 2. Natural N va K sonlari berilgan. N ta elementli sonlar massivini hosil qiling, bunda massivning har bir elementi o'zidan avvalgi elementidan K songa katta bo'lsin.

Massiv 3. Natural N soni, A va D butun sonlar berilgan. Quyidagicha N ta elementli massiv hosil qiling va chiqaring: A, A+D, A+2·D, A+3·D, …

Massiv 4. Natural N son berilgan. N ta elementli manfiy butun qiymatli A sonlar massivini tasodifiy sonlar generatori yordamida hosil qiling va chiqaring.

Massiv 5. Natural N soni, B va Q butun sonlar berilgan. Quyidagicha N ta elementli A massiv hosil qiling va chiqaring: B, B·Q, $B \cdot D^2$, $B \cdot D^3$,…

Massiv 6. Natural N son berilgan. N ta elementli A butun sonlar massivini hosil qiling, bunda massiv elementlari 2 ning darajalarini o'sish tartibida saqlasin.

Massiv 7. Natural N son berilgan. N ta elementli A butun sonlar massivini hosil qiling, bunda massiv elementlari faqat 0 va 1 dan iborat bo'lsin.

Massiv 8. Natural N son, butun X va Y berilgan. N ta elementli A butun sonlar massivini quyidagicha hosil qiling: 1-elementi X ga, 2-elementi Y ga, keyingi har bir elementi oldingi 2 element yig'indisiga teng bo'lsin.

Massiv 9. Juft natural N son berilgan. N ta elementli A butun sonlar massivini hosil qiling, bunda massiv elementlari yarmigacha o'suvchi, yarmidan keyin kamayuvchi bo'lsin.

Massiv 10. Juft natural N son berilgan. N ta elementli A butun sonlar massivining birinchi N/2 ta elementi berilgan. Massivning keyingi N/2 elementi avvalgi qism nusxasidan iborat. Massiv elementlarini chiqaring.

Massiv 11. Natural N son va N ta elementli A butun sonlar massivi berilgan. Massiv elementlarini teskari tartibda chiqaruvchi dastur tuzing.

Massiv 12. Natural N son va N ta elementli A butun sonlar massivi berilgan. Massiv elementlarini yangi B massivga teskari tartibda ko'chiring va chiqaring, ya'ni $B_1=A_N$, $B_2=A_{N-1}$, …, $B_N=A_1$.

Massiv 13. Natural N son va N ta elementli A butun sonlar massivi berilgan. Avval massivning juft elementlari indekslarini o'sish tartibida, keyin toq elementlari indekslarini kamayish tartibida chiqaring.

Massiv 14. Natural N va K (1<K<N) sonlari va A[1..N] butun sonlar massivi berilgan. Massivning har K-o'rinda turgan, ya'ni A[K], A[2·K], A[3·K], ..., elementlarini chiqaring.

Massiv 15. 20 ta elementli A massivi elementlarining tartibini quyidagicha o'zgartirib, har bir vazifaga mos izoh bilan chiqaring:

A) $A_{20}, A_1, A_2, A_3, \ldots, A_{19}$;

B) $A_2, A_3, A_4, \ldots, A_{20}, A_1$;

D) $A_{11}, A_{12}, A_{13}, \ldots, A_{20}, A_1, A_2, A_3, \ldots, A_{10}$;

E) $A_{11}, A_{12}, A_{13}, \ldots, A_{20}, A_{10}, A_9, A_8, \ldots, A_1$;

F) $A_{20}, A_{19}, A_{18}, \ldots, A_{11}, A_1, A_2, A_3, \ldots, A_{10}$;

G) $A_{20}, A_{19}, A_{18}, \ldots, A_{11}, A_{10}, A_9, A_8, \ldots, A_1$;

Massiv 16. Juft natural N son va N ta elementli A butun sonlar massivi berilgan. Massiv elementlarini yarmigacha o'sish, yarmidan keyin kamayish yo'nalishida tartiblab chiqaring.

Massiv 17. Natural N son va N ta elementli A butun sonlar massivi berilgan. Massiv elementlarining manfiylari ichida eng kattasi va musbatlari ichida eng kichigini aniqlang.

Massiv 18. Natural N son va N ta elementli A butun sonlar massivi berilgan. Massivning nolga teng elementlari indeksini chiqaring.

Massiv 19. Natural N, L, R sonlari (1≤L<R≤N) va N ta elementli A butun sonlar massivi berilgan. Massivning indekslari [L; K] oraliqda yotgan elementlari yig'indisini aniqlang.

Massiv 20. Natural N, L, R sonlari (1≤L<R≤N) va N ta elementli A butun sonlar massivi berilgan. Massivning indekslari [L; K] oraliqda yotmagan elementlari yig'indisini aniqlang.

Massiv 21. Natural N, L, R sonlari (1≤L<R≤N) va N ta elementli A butun sonlar massivi berilgan. Massivning indekslari L va K ga teng bo'lmagan elementlari ko'paytmasini aniqlang.

Massiv 22. Natural N son va N ta elementli A butun sonlar massivi berilgan. Massivning toq indeksli elementlari yig'indisidan juft indeksli elementlari yig'indisini ayirib chiqaring.

Massiv 23. Natural N son va N ta elementli A butun sonlar massivi berilgan. Massivning toq elementlari ko'paytmasidan juft elementlari yig'indisini ayirib chiqaring.

Massiv 24. Natural N son va N ta elementli A butun sonlar massivi berilgan. Massivning 5 ga karrali elementlari yig'indisini chiqaring.

Massiv 25. Natural N soni berilgan. N ta elementli A sonlar massivini 0, 1 va 2 raqamlari bilan tasodifiy sonlar generatori yordamida to'ldiring. A massivini B massivga quyidagicha usulda ko'chiring: agar A[k]=0 bo'lsa, u holda B[k]=1, agar A[k]=1 bo'lsa, u holda B[k]=2 va agar A[k]=2 bo'lsa, u holda B[k]=0 bo'lsin. A va B massivlarni yonma-yon chiqaring.

Massiv 26. Natural N son va N ta elementli A sonlar massivi berilgan. Massiv elementlari ichida noldan kichik elementlar mavjud bo'lsa, bu elementlarni nol bilan almashtiring, noldan katta elementlarni esa bir bilan almashtiring.

Massiv 27. Natural N, K sonlari va N ta elementli A sonlar massivi berilgan. Avval massiv elementlari ichida K sonidan katta, so'ng K sonidan kichik bo'lgan elementlari indeksini chiqaring.

Massiv 28. Natural N, butun K sonlari va N ta elementli butun A sonlar massivi berilgan. Massiv elementlari ichida K soniga teng bo'lgan elementlar sonini va ularning indeksini chiqaring.

Massiv 29. Natural N, L, R sonlari ($1 \le L < R \le N$) va N ta elementli A sonlar massivi berilgan. Massivning indekslari [L; K] oraliqda yotgan elementlari o'rta arifmetigini toping.

Massiv 30. Natural N son va N ta elementli A sonlar massivi berilgan. Massiv elementlari ichida manfiy elementlarni B massivga ko'chiring.

Yo'llanma. Avval manfiy elementlar sanagichi S ni 0 deb olamiz. Biror qadamda manfiy element aniqlansa, S ni bittaga oshiramiz va aniqlangan elementni B[S] ga o'zlashtiramiz. Javob chiqarishda S=0 bo'lsa "Massivning manfiy elementlari yo'q" matnini, aks holda B massivni chiqaramiz. Dasturni mustaqil tuzing.

Massiv 31. Natural N son va N ta elementli A sonlar massivi berilgan. Massiv elementlari 0 yoki 1 dan iborat. Massiv elementlarida 0 qiymatni 1 bilan, 1 qiymatni 0 lar bilan almashtiring (yo'llanma: 1-usulda agar A[k]=0 bo'lsa, u holda A[k]=1, aks holda A[k]=0; 2-usulda taqqoslash o'rniga bitta formula yozish kifoya, ya'ni A[k]=1–A[k]).

Massiv 32. Natural N son berilgan. N ta elementli A massivni 4 va 20 sonlar bilan tasodifiy sonlar generatori yordamida to'ldiring. Massiv elementlarida 4 qiymatni 20 bilan, 20 qiymatni 4 bilan almashtiring dastur tuzing. Hosil qilingan va qayta ishlangan massivlarni chiqaring. Tarmoqlash va tanlash operatorlarini qo'llash mumkin emas.

Massiv 33. Natural N son berilgan. Bir o'lchovli A[1..N] sonlar massivi elementlarining ba'zilari 0 bo'lishi mumkin. Qo'shimcha massivdan foydalanmasdan, shu massivda avval 0 dan farqli elementlar, keyin 0 lar joylashtirilsin.

Massiv 34. Natural N son va A[1..N] sonlar massivi berilgan. Qo'shimcha massivdan foydalanmasdan shu massivda avval manfiy sonlarni, keyin 0 larni, oxirida musbat sonlar joylashtirilsin.

Massiv 35. Natural N son va A[1..N] sonlar massivi berilgan. Qo'shimcha massivdan foydalanmasdan shu massivda avval musbat sonlarni, keyin 0 larni, oxirida manfiy sonlar joylashtirilsin.

Massiv 36. Natural N son va A[1..N] sonlar massivi berilgan. Qo'shimcha massivdan foydalanmasdan shu massivda avval musbat sonlarni o'sish tartibida, keyin manfiy sonlar kamayish tartibida, oxirida 0 lar joylashtirilsin.

Massiv 37. A[1..100] sonlar massivini shunday to'ldiringki, elementlari ham 7 ga, ham 11 ga bo'linadigan dastlabki 100 ta sondan iborat bo'lsin.

Massiv 38. Natural juft N soni va A[1..N] sonlar massivi berilgan. Massivning "juft o'rindagi element – toq o'rindagi element" juftligidagi elementlarning o'zaro o'rinlarini almashtiring.

Massiv 39. Natural N son va A[1..N] sonlar massivi berilgan. Massivning juft o'rinlarda joylashgan elementlarini massivdan o'chirib, yangi A[1..(N+1)/2] massiv hosil qiling (bu yerda (N+1)/2 butun bo'lish). Qo'shimcha massiv ochish mumkin emas.

Massiv 40. Natural N, haqiqiy M va B sonlar, A[1..N] sonlar massivi berilgan. Massivning (M, B) intervalga tegishli bo'lgan elementlari ichidan eng kattasining va eng kichigining indeksini aniqlang.

Massiv 41. Natural N son va A[1..N] sonlar massivi berilgan. Indekslari o'sib borganda A[K]<A[N] shartni birinchi bo'lib qanoatlantiradigan massivning K tartib raqamli elementini chiqaring. Agar bunday elementlar mavjud bo'lmasa, u holda bu haqida xabar chiqaring.

Massiv 42. Natural N son va A[1..N] sonlar massivi berilgan. Indekslari o'sib borganda A[1]<A[K]<A[N] shartni birinchi bo'lib qanoatlantiradigan massivning K tartib raqamli elementini chiqaring. Agar bunday elementlar mavjud bo'lmasa, u holda bu haqida xabar chiqaring.

Massiv 43. Natural N son, A[1..N] va B[1..N] sonlar massivi berilgan. Quyidagi ko'paytmani hisoblash dasturini tuzing:

$$(A_1+B_N)\cdot(A_2+B_{N-1})\cdot(A_3+B_{N-2})\cdot\ldots\cdot(A_N+B_1)$$

Massiv 44. N natural son, A[1..N] va B[1..N] sonlar massivi berilgan. Quyidagi yig'indini hisoblash dasturini tuzing: $A_1\cdot B_N+A_2\cdot B_{N-1}+A_3\cdot B_{N-2}+\ldots+A_N\cdot B_1$

Massiv 45. Natural N son va A[1..N] sonlar massivi berilgan. Quyidagilarni aniqlang:

a) $MAX\{A_1+A_N, A_2+A_{N-1},\ldots, A_N+A_1\}$;

b) $MIN\{A_1\cdot A_N, A_2\cdot A_{N-1}, A_3\cdot A_{N-2},\ldots, A_N\cdot A_1\}$;

Massiv 46. Natural N son va A[1..N] sonlar massivi berilgan. Yangi B[1..2·N], C[1..2·N], D[1..2·N] massivlarini quyidagicha hosil qiling:

B[1..2·N]: $A_1, A_2, A_3,\ldots, A_N, A_1, A_2, A_3,\ldots, A_N$;

C[1..2·N]: $A_1, A_2, A_3,\ldots, A_N, A_N, \ldots, A_3, A_2, A_1$;

D[1..2·N]: $A_N, \ldots, A_3, A_2, A_1, A_1, A_2, A_3,\ldots, A_N$.

Massiv 47. Natural N son va A[1..N] sonlar massivi berilgan. Shunday $B_1, B_2,\ldots, B_{10}$ ketma-ketlikni hosil qilingki, ular quyidagicha bo'lsin:

$B_1 = A_1 + A_2 + \cdots + A_N$, $B_2 = A_1^2 + A_2^2 + \cdots + A_N^2$, ..., $B_{10} = A_1^{10} + A_2^{10} + \cdots + A_N^{10}$.

Massiv 48. Natural N son va 0 va 1 lardan iborat B[1..N] sonlar massivi berilgan. Massiv elementlari ichida ketma-ket keluvchi 0 va 1 sonlar juftligi mavjud bo'lsa, bunday juftliklar sonini aniqlang (masalan: 0, 1, 0, 1 juftlikda faqat 2 ta ketma-ketlik bor).

Massiv 49. Natural N son va N ta elementli A butun sonlar massivi berilgan. Massivdagi eng kichik elementni massivning birinchi elementi o'rniga, eng katta elementni esa massivning oxirgi elementi o'rniga joylashtiring.

Massiv 50. Natural N son va a_1, a_2, a_3,..., a_N sonlar ketma-ketligi berilgan. Shu sonlar ketma-ketligidagi:

a) biror toq sonni ikkilanganiga teng bo'lganlarini aniqlang;

b) 7 ga bo'lganda 1 yoki 2 yoki 5 qoldiq qoladigan sonlarni aniqlang.

Massiv 51. Natural N son va a_1, a_2, a_3,..., a_N sonlar ketma-ketligi berilgan. Shu sonlar ketma-ketligidagi:

a) 5 ga karrali bo'lgan sonlar yig'indisini hisoblang;

b) 7 ga bo'lganda 3 yoki 4 qoldiq qoladigan sonlar ko'paytmasini hisoblang;

d) toq va manfiy bo'lgan sonlar yig'indisini hisoblang;

e) K (natural son) songa karrali bo'lgan sonlar ko'paytmasini hisoblang;

f) 5 ga karrali bo'lgan va 7 ga karrali bo'lmagan sonlar yig'indisini hisoblang.

Massiv 52. Natural N son va a_1, a_2, a_3,..., a_N sonlar ketma-ketligi berilgan. Shu sonlar ketma-ketligidagi barcha ikkidan kichik bo'lgan hadlarni nollar bilan almashtirib chiqaring hamda [4, 7] kesmaga tegishli elementlar sonini va yig'indisini aniqlang.

Massiv 53. Natural N son va a_1, a_2, a_3,..., a_N sonlar ketma-ketligi berilgan. Quyidagilarni aniqlang:

a) $MAX(a_1, a_2, a_3,..., a_N)$; b) $MIN(a_1, a_2, a_3,..., a_N)$; d) $MAX(a_2, a_4,...)$;

e) $MIN(a_1, a_3,...)$; f) $MAX(a_2, a_4,...)+MIN(a_1, a_3,...)$; g) $MAX(|a_1|, |a_2|, |a_3|,...,|a_N|)$.

Massiv 54. Natural N son va haqiqiy a_1, a_2, a_3,..., a_N sonlar berilgan. Shu sonlar ketma-ketligida quyidagi amallarni bajaring:

a) ikkita ketma-ket keluvchi musbat elementlar bo'lsa, bunday ketma-ketliklar sonini aniqlang;

b) ikkita ketma-ket keluvchi turli ishorali elementlar mavjud bo'lsa, bunday ketma-ketliklar sonini aniqlang.

Massiv 55. Natural N son va N ta elementli A butun sonlar massivi berilgan. Massivda musbat va manfiy qiymatli elementlar yonma-yon kelishini tekshiring. Agar shu xossa o'rinli bo'lsa 0 chiqaring, aks holda bu qonuniyatni buzgan birinchi element indeksini chiqaring.

Massiv 56. Natural N son va N ta elementli A butun sonlar massivi berilgan. Massivda juft va toq qiymatli elementlar yonma-yon kelishini tekshiring. Agar shu xossa o'rinli bo'lsa 0 chiqaring, aks holda bu qonuniyatni buzgan birinchi element indeksini chiqaring.

Massiv 57. Natural N son va N ta elementli A butun sonlar massivi berilgan. Massivning o'zidan keyingi elementdan katta bo'lgan elementlari indeksini va sonini chiqaring. Agar bunday elementlar yo'q bo'lsa 0 chiqaring.

Massiv 58. Natural N son va N ta elementli A butun sonlar massivi berilgan. Massivning o'zidan oldingi elementdan katta bo'lgan elementlari indeksini kamayish tartibida chiqaring. Agar bunday elementlar yo'q bo'lsa 0 chiqaring.

Massiv 59. Natural N son va N ta elementli A butun sonlar massivi berilgan. Massivning faqat ikkita bir xil elementi bor bo'lsa, shu elementlar indekslarini chiqaring.

Massiv 60. Natural N son va N ta elementli A butun sonlar massivi berilgan. Massivning bir-biriga eng yaqin ikkita elementi indekslarini chiqaring.

Massiv 61. Natural N son va N ta elementli A butun sonlar massivi berilgan. Massivning faqat juft elementlaridan iborat bo'lgan B massivni hosil qiling va chiqaring. Agar A massivning juft elementlari bo'lmasa, u holda bu haqida xabar chiqaring.

Massiv 62. Natural N son va N ta elementli A butun sonlar massivi berilgan. Quyidagicha hosil qilingan B massivni chiqaring:

$$B[k] = \begin{cases} 2 \cdot A[k], & \text{agar } A[k] > 7 \\ \dfrac{A[k]}{2}, & \text{aks holda} \end{cases}$$

Massiv 63. Natural N, L, R sonlari ($1 \le L < R \le N$) va N ta elementli A sonlar massivi berilgan. Massivning A_L va A_K elementlari orasida yotgan elementlarni teskari tartibda joylashtirib chiqaring.

= B =

Massiv 64. Natural N son va N ta elementli A butun sonlar massivi berilgan. Agar massiv 1 dan N gacha bo'lgan sonlarning **o'rinlashtirishi** bo'lsa (ya'ni massivda 1 dan N gacha bo'lgan barcha sonlar ishtirok etsa) 0 sonini, aks holda o'rinlashtirish shartini buzuvchi birinchi uchragan mumkin bo'lmagan sonni chiqaring.

Massiv 65. Natural N son va N ta elementli A butun sonlar massivi berilgan. Massiv 1 dan N gacha bo'lgan sonlarning o'rinlashtirishidan iborat. Shu o'rinlashtirishdagi **inversiyalar** sonini, ya'ni k<m bo'lganda A[k]>A[m] tengsizlik o'rinli bo'ladigan juftliklar sonini chiqaring.

Massiv 66. Natural N son va N ta elementli A butun sonlar massivi berilgan. Massivning elementlari faqat o'suvchi bo'lgan oraliqlar sonini chiqaring.

Massiv 67. Natural N son va N ta elementli A butun sonlar massivi berilgan. Massivning elementlari faqat kamayuvchi bo'lgan oraliqlar sonini chiqaring.

Massiv 68. Natural N son va N ta elementli A butun sonlar massivi berilgan. Massivning elementlari kamayuvchi yoki o'suvchi bo'lgan oraliqlar sonini chiqaring.

Massiv 69. Natural N son va N ta elementli A butun sonlar massivi berilgan. Massivning yonma-yon elementlari ichida orasidagi farq maksimal bo'lgan elementlarini chiqaring.

Massiv 70. Natural N soni, R va N ta elementli A sonlar massivi berilgan. Massivning elementlari ichida R soniga eng yaqin bo'lganini chiqaring.

Massiv 71. Natural N soni, R va N ta elementli A sonlar massivi berilgan. Massivning yig'indisi R soniga eng yaqin bo'lgan ikkita turli elementlari indekslarini chiqaring.

Massiv 72. Natural N son va N ta elementli A butun sonlar massivi berilgan. Massivning barcha lokal minimumlarini chiqaring. Agar bunday element yo'q bo'lsa 0 chiqaring. Massiv elementi **massivning lokal minimumi** deyiladi, agar o'ziga qo'shni ikkita elementdan kichik bo'lsa.

Massiv 73. Natural N son va N ta elementli A butun sonlar massivi berilgan. Massivning barcha lokal maksimumlarini chiqaring. Agar bunday element yo'q bo'lsa 0 chiqaring. Massiv elementi **massivning lokal maksimumi** deyiladi, agar o'ziga qo'shni ikkita elementdan katta bo'lsa.

Massiv 74. Natural K, N va haqiqiy $a_1, a_2, ..., a_{K \cdot N}$ sonlar berilgan. $b_1=\max(a_1, \ldots, a_K)$, $b_2=\max(a_{K+1}, \ldots, a_{2 \cdot K})$, $b_N=\max(a_{K \cdot (N-1)}, \ldots, a_{K \cdot N})$ ketma-ketlikni hosil qiling.

Massiv 75. Natural K, N va haqiqiy $a_1, a_2, ..., a_{K \cdot N}$ sonlar berilgan. $\mathrm{MIN}(\max(a_1, \ldots, a_K), \max(a_{K+1}, \ldots, a_{2 \cdot K}), \max(a_{K \cdot (N-1)}, \ldots, a_{K \cdot N}))$ sonni aniqlang.

Massiv 76. Natural N son va N ta elementli A butun sonlar massivi berilgan. Massivning lokal minimumlari ichida eng kattasini chiqaring. Agar bunday element yo'q bo'lsa 0 chiqaring.

Massiv 77. Natural N son va N ta elementli A butun sonlar massivi berilgan. Massivning lokal maksimumlari ichida eng kichigini chiqaring. Agar bunday element yo'q bo'lsa 0 chiqaring.

Massiv 78. Natural N son va N ta elementli A butun sonlar massivi berilgan. Massivning lokal minimum ham, lokal maksimum ham bo'lmagan elementlari ichida eng kattasini chiqaring. Agar bunday element yo'q bo'lsa 0 chiqaring.

Massiv 79. Natural N son va N ta elementli A sonlar massivi berilgan. K-elementi A massivning birinchi K ta elementining o'rta arifmetigiga teng bo'lgan B massivni hosil qiluvchi dastur tuzing (yo'llanma: B[K] = (A[1] + A[2] + ... + A[K]) / K).

Massiv 80. Natural N son va N ta elementli A sonlar massivi berilgan. Quyidagi ifodani hisoblang: $A[1]-A[2]+...+(-1)^{N+1} \cdot A[N]$ (yo'llanma: ishora o'zgaruvchisini kiriting).

Massiv 81. Natural N son va butun sonlar A[1..N] massivi berilgan. Massiv elementlarini ketma-ket qo'shib borib, yig'indi berilgan N sonidan ortishi bilan yig'indi qiymatini va oxirgi qo'shilgan element indeksini chiqaruvchi dastur tuzing. Agar massiv elementlari yig'indisi N dan oshmasa, bu haqida xabar chiqaring.

Massiv 82. Natural N son va A[1..N] sonlar massivi berilgan. Massiv elementlari ichida ketma-ket keluvchi bir xil elementlar ketma-ketliklari sonini aniqlang (yo'llanma: A[k]=A[k+m], m=1, 2,..., shart tekshirib boriladi).

Massiv 83. Natural N son va A, B, C sonlar berilgan. X[1..N] massiv elementlari faqat A, B, C sonlardan iborat. Qo'shimcha massivdan foydalanmasdan shu massivda avval A ga teng sonlar, keyin B ga teng sonlar, so'ngra C ga teng sonlarni joylashtiring.

Massiv 84. Natural N son va butun qiymatli A[1..N] massiv berilgan. Massiv elementlari ichida eng ko'p takrorlanganini aniqlang. Bunday elementlar ko'p bo'lsa, birortasini chiqaring.

Massiv 85. Natural N son va butun qiymatli A[1..N] massiv berilgan. Agar massiv elementlari arifmetik progressiya tashkil etsa birinchi had va ayirmani, aks holda 0 sonini chiqaring.

Massiv 86. Natural N son va butun qiymatli A[1..N] massiv berilgan. Agar massiv elementlari geometrik progressiya tashkil etsa birinchi had va bo'linmani, aks holda 0 sonini chiqaring.

Massiv 87. Natural N son, Oxy tekisligida B(X0; Y0) nuqta va N ta (Xk; Yk) nuqtalar berilgan. Shu nuqtalar ichidan B nuqtaga eng yaqin joylashganini aniqlang.

Massiv 88. Natural N son va Oxy tekisligida N ta (Xk; Yk) nuqtalar berilgan. Shu nuqtalarning II kvadrantda yotganlari ichidan koordinata boshiga eng uzoq joylashganining indeksini chiqaring. Agar bunday nuqtalar yo'q bo'lsa, u holda -1 chiqaring.

Massiv 89. Natural N son va Oxy tekisligida N ta (Xk; Yk) nuqtalar berilgan. Shu nuqtalarning I kvadrant yoki III kvadrantda yotganlari ichidan koordinata boshiga eng yaqin joylashganining indeksini chiqaring. Agar bunday nuqtalar yo'q bo'lsa, u holda -1 chiqaring.

Massiv 90. Natural N son va Oxy tekisligida N ta (Xk; Yk) nuqtalar berilgan. Shu nuqtalar ichidan bir-biridan eng uzoq joylashgan ikki nuqta indekslarini va orasidagi masofani chiqaring.

Massiv 91. Butun sonli A[1..6][1..6] massiv elementlarini tasodifiy son kabi hosil qiling va jadval ko'rinishida ekranga chiqaring.

Massiv 92. Natural N soni (N<21) berilgan. A[1..N][1..N] massivni shunday hosil qilingki, k-satri elementlarining har biri 21·k ga teng bo'lsin. Massivni jadval ko'rinishida ekranga chiqaring.

Massiv 93. Natural N soni (N<21) berilgan. A[1..N][1..N] massivni shunday hosil qilingki, k-ustuni elementlarining har biri 10·k ga teng bo'lsin. Massivni jadval ko'rinishida ekranga chiqaring.

Massiv 94. Natural N soni (N<21) berilgan. A[1..N][1..N] massivni shunday hosil qilingki, k-satrda o'sish yo'nalishida (k-1)·N+1 dan k·N gacha bo'lgan sonlar joylashsin. Massivni jadval ko'rinishida ekranga chiqaring.

Massiv 95. Natural N soni (N<21) berilgan. A[1..N][1..N] massivni shunday hosil qilingki, k-ustunda kamayish yo'nalishida k·N dan (k-1)·N+1 gacha bo'lgan sonlar joylashsin. Massivni jadval ko'rinishida ekranga chiqaring.

Massiv 96. Natural N soni va butun qiymatli A[1..N][1..N] massivi berilgan. Massivning juft tartib raqamli satrlarini jadval ko'rinishida ekranga chiqaring.

Massiv 97. Natural N soni va butun qiymatli A[1..N][1..N] massivi berilgan. Massivning toq tartib raqamli ustunlarini satr kabi jadval ko'rinishida ekranga chiqaring.

Massiv 98. Natural N soni va butun qiymatli A[1..N][1..N] massivi berilgan. Massivning toq tartib raqamli satri elementlarini teskari tartibda, juft tartib raqamli satri elementlarini to'g'ri tartibda jadval ko'rinishida ekranga chiqaring.

Massiv 99. Natural N soni va butun qiymatli A[1..N][1..N] massivi berilgan. Massivning toq tartib raqamli ustuni elementlarini pastdan yuqoriga, juft tartib raqamli ustuni elementlarini yuqoridan pastga jadval ko'rinishida ekranga chiqaring.

Massiv 100. Natural N soni (N<31) berilgan. A[1..N][1..N] massivni shunday hosil qilingki, k-ustunda 2 sonining 0 dan N^2-1 gacha bo'lgan darajalari satrning ortib borish yo'nalishida ham, ustunning ortib borish yo'nalishida ham o'sish tartibida joylashsin.

Massiv 101. Natural N soni va butun qiymatli A[1..N][1..N] massivi berilgan. Massivning minimal va maksimal elementlari indeksini chiqaring.

Massiv 102. Natural N soni va butun qiymatli A[1..N][1..N] massivi berilgan. Massivning har bir satri elementlari yig'indisini alohida satrlarda chiqaring.

Massiv 103. Natural N soni va butun qiymatli A[1..N][1..N] massivi berilgan. Massivning ustunlari elementlari ko'paytmasini alohida satrlarda chiqaring.

Massiv 104. Natural N, M sonlari va butun qiymatli A[1..N][1..M] massivi berilgan. Massivning har bir satrining elementlari o'rta arifmetigini alohida satrlarda chiqaring.

Massiv 105. Natural N, M sonlari va butun qiymatli A[1..N][1..M] massivi berilgan. Massivning har bir ustunining elementlari kvadratlarining o'rta geometrigini alohida satrlarda chiqaring.

Massiv 106. Natural N, M sonlari va butun qiymatli A[1..N][1..M] massivi berilgan. Massiv elementlaridan juftlarining yig'indisini chiqaring.

Massiv 107. Natural N, M sonlari va butun qiymatli A[1..N][1..M] massivi berilgan. Massiv elementlaridan satr va ustun tartib raqamlari yig'indisi toq bo'lganlarining yig'indisini chiqaring.

Massiv 108. Natural N, M sonlari va butun qiymatli A[1..N][1..M] massivi berilgan. Massivning har bir satri elementlaridan eng katta va eng kichigini alohida satrlarda chiqaring.

Massiv 109. Natural N, M sonlari va butun qiymatli A[1..N][1..M] massivi berilgan. Massivning har bir toq ustuni elementlaridan eng katta va eng kichigini alohida satrlarda chiqaring.

Massiv 110. Natural N, M sonlari va butun qiymatli A[1..N][1..M] massivi berilgan. Massivning har bir satri elementlaridan eng kichiklari ichida eng kattasini chiqaring.

Massiv 111. Natural N, M sonlari va butun qiymatli A[1..N][1..M] massivi berilgan. Massivning har bir ustuni elementlaridan eng kattalari ichida eng kichigini chiqaring.

Massiv 112. Natural N, M sonlari va butun qiymatli A[1..N][1..M] massivi berilgan. Massivning har bir satri elementlaridan shu satr elementlari o'rta arifmetigidan kichiklari sonini alohida satrlarda chiqaring.

Massiv 113. Natural N, M sonlari va butun qiymatli A[1..N][1..M] massivi berilgan. Massivning har bir ustuni elementlaridan shu ustun elementlari o'rta arifmetigidan kattalari sonini alohida satrlarda chiqaring.

Massiv 114. Natural N, M sonlari va butun qiymatli A[1..N][1..M] massivi berilgan. Massivning satrlaridan musbat va manfiy elementlari soni tenglarining tartib raqamlarini alohida satrlarda chiqaring. Bunday satrlar yo'q bo'lsa 0 chiqaring.

Massiv 115. Natural N, M sonlari va butun qiymatli A[1..N][1..M] massivi berilgan. Massivning faqat juft elementlardan iborat bo'lgan ustunlarining tartib raqamlarini alohida satrlarda chiqaring. Bunday ustunlar yo'q bo'lsa 0 chiqaring.

Massiv 116. Natural N, M sonlari va butun qiymatli A[1..N][1..M] massivi berilgan. Massivning maksimal sondagi bir xil elementlardan tashkil topgan satri tartib raqamini chiqaring. Bunday satrlar ko'p bo'lsa, u holda ulardan tartib raqami eng kichigini chiqaring.

Massiv 117. Natural N, M sonlari va butun qiymatli A[1..N][1..M] massivi berilgan. Massivning maksimal sondagi bir xil elementlardan tashkil topgan ustuni tartib raqamini chiqaring. Bunday ustunlar ko'p bo'lsa, u holda ulardan tartib raqami eng kattasini chiqaring.

Massiv 118. Natural N, M sonlari va butun qiymatli A[1..N][1..M] massivi berilgan. Massivning o'zi joylashgan satrdagi elementlarga nisbatan eng kichik va o'zi joylashgan ustundagi elementlarga nisbatan eng katta elementlar tartib raqamlarini chiqaring. Bunday elementlar yo'q bo'lsa 0 chiqaring.

Massiv 119. Natural N, M, L, R ($1 \le L < R \le N$) sonlari va butun qiymatli A[1..N][1..M] massiv berilgan. Massivning L-satri bilan R-satri o'rnini almashtirib chiqaring.

Massiv 120. Natural N, M, L, R ($1 < L \le N$, $1 \le R < N$) sonlari va butun qiymatli A[1..N][1..M] massiv berilgan. Massivning avval L-ustuni bilan 1-ustuni, so'ngra R-satri bilan N-satri o'rnini almashtirib chiqaring.

Massiv 121. Natural N, M sonlari va butun qiymatli A[1..N][1..M] massivi berilgan. Massivning maksimal elementi joylashgan biror satri bilan minimal elementi joylashgan biror satri o'rnini almashtirib chiqaring.

Massiv 122. Natural N, M sonlari va butun qiymatli A[1..N][1..M] massivi berilgan. Massivning faqat manfiy elementlardan iborat biror ustuni bilan faqat musbat elementlardan iborat biror ustuni o'rnini almashtirib chiqaring. Bunday ustunlar yo'q bo'lsa 0 chiqaring.

Massiv 123. Natural N, M sonlari va butun qiymatli A[1..N][1..M] massivi berilgan. Massivning yuqori yarmi bilan quyi yarmi o'rnini almashtirib chiqaring.

Massiv 124. Natural N, M sonlari va butun qiymatli A[1..N][1..M] massivi berilgan. Massivning satrlarini o'rtasiga nisbatan simmetrik almashtirib chiqaring.

Massiv 125. Natural N (<21) son va butun qiymatli A[1..N][1..N] massiv berilgan. Massivning satrlarini ustunlari bilan almashtiring (ya'ni k-satr elementlari k-ustun elementlariga, k-ustun elementlari k-satr elementlariga aylanadi).

Yo'llanma. Shartga ko'ra A[k,m] elementi A[m,k] elementi bilan o'rin almashtrilishi talab qilingan. Parametrli ikkita takrorlash operatori yordamida almashtirishni amalga oshirish mumkin. Faqatgina e'tiborni parametrning oxirgi qiymatiga qaratish kerak bo'ladi. Chunki agar ikkala takrorlash operatorida parametr qiymatini 1 da N gacha olinsa, u holda massiv yana o'ziga qaytib qoladi. Shu sababli ichki parametrning oxirgi qiymatini [N/2] gacha olish lozim.

Massiv 126. Natural N, M sonlari va butun qiymatli A[1..N][1..M] massivi berilgan. Massivning barcha lokal minimumlarini chiqaring. Agar bunday element yo'q bo'lsa 0 chiqaring. Massiv elementi **massivning lokal minimumi** deyiladi, agar o'zi atrofidagi barcha qo'shni elementlardan kichik bo'lsa.

Massiv 127. Natural N, M sonlari va butun qiymatli A[1..N][1..M] massivi berilgan. Massivning barcha lokal maksimumlarini chiqaring. Agar bunday element yo'q bo'lsa 0 chiqaring. Massiv elementi **massivning lokal maksimumi** deyiladi, agar o'zi atrofidagi barcha qo'shni elementlardan katta bo'lsa.

Massiv 128. Natural N, M sonlari va butun qiymatli A[1..N][1..M] massivi berilgan. Massivning eng katta elementini, shu element massivda qatnashishlari sonini aniqlang.

Massiv 129. Natural N, M sonlari va butun qiymatli A[1..N][1..M] massivi berilgan. Massivning eng kichik elementini, shu elementga teng elementlar indekslarini aniqlang.

= C =

Massiv 130. Natural N son va Oxy tekisligida N ta (Xk; Yk) nuqtalar berilgan. Berilgan nuqtalar ichidan shunday uchtasini topingki, ular eng katta yuzali uchburchak tashkil qilsin. Yuza doimo musbat hisoblanadi.

Massiv 131. Natural N son va Oxy tekisligida N ta (Xk; Yk) nuqtalar berilgan. Berilgan nuqtalar ichidan shunday uchtasini topingki, ular eng kichik yuzali uchburchak tashkil qilsin. Yuza doimo musbat hisoblanadi.

Massiv 132. Natural N son va Oxy tekisligida N ta (Xk; Yk) nuqtalar berilgan. Berilgan nuqtalar ichidan shunday uchtasini topingki, ulardan hosil qilingan uchburchak perimetri eng kichik bo'lsin. Yuza doimo musbat hisoblanadi.

Massiv 133. Natural N son va Oxy tekisligida N ta (Xk; Yk) nuqtalar berilgan. Bu nuqtalar ichidan shundayini topingki, qolgan nuqtalargacha bo'lgan masofalar yig'indisi eng kichik bo'lsin.

Massiv 134. Natural N son va Oxy tekisligida N ta (Xk; Yk) nuqtalar to'plami va M ta (Xt; Yt) nuqtalar to'plami berilgan. Berilgan ikkala to'plam orasidagi eng yaqin masofani aniqlang. Ikki to'plam orasidagi masofa to'plamlarning har biridan bittadan olingan ikki nuqta orasidagi masofa orqali aniqlanadi.

Massiv 135. Natural N son va Oxy tekisligida N ta (Xk; Yk) nuqtalar berilgan. Tekislikda tartiblashni quyidagicha aniqlaymiz: $(x_k; y_k) < (x_m; y_m)$, agar $(x_k < x_m)$ yoki $(x_k = x_m$ va $y_k < y_m)$ bo'lsa. Berilgan nuqtalarni tekislikda o'sish yo'nalishida tartiblang.

Massiv 136. Natural N son va Oxy tekisligida N ta (Xk; Yk) nuqtalar berilgan. Tekislikda tartiblashni quyidagicha aniqlaymiz: $(x_k; y_k) < (x_m; y_m)$, agar $(x_k+y_k < x_m+ y_m)$ yoki $(x_k+y_k = x_m+ y_m$ va $x_k < x_m)$ bo'lsa. Berilgan nuqtalarni tekislikda o'sish yo'nalishida tartiblang.

Massiv 137. Natural N (N<135) son va butun qiymatli A[1..N] massiv berilgan. Quyidagi yig'indi qiymatini aniqlang:

|A[1]-A[2]|+...+ |A[1]-A[N]|+|A[2]-A[3]|+...+ |A[2]-A[N]|+|A[3]-A[4]|+...+|A[N-1]-A[N]|

Massiv 138. Natural N son va butun qiymatli A[1..N] massiv berilgan. Massivning eng ko'p ketma-ket kelgan bir xil elementi sonini aniqlang. Masalan, 1, 2, 2, 2, 3, 1, 1 bo'lsa, 2 soni 3 ta.

Massiv 139. 1 soni M to'plamga tegishli. Agar k soni M ga tegishli bo'lsa, u holda 2·k+1 va 3·k+1 ham M ga tegishli bo'ladi. M to'plamning 1000 dan kichik barcha elementlarini kamayish yo'nalishida chiqaring.

Massiv 140. 1 soni M to'plamga tegishli. Agar k soni M ga tegishli bo'lsa, u holda 2·k+1 va 3·k+1 ham M ga tegishli bo'ladi. M to'plamning 1000 dan kichik barcha elementlarini o'sish yo'nalishida chiqaring.

Massiv 141. 1 soni M to'plamga tegishli. Agar k soni M ga tegishli bo'lsa, u holda 2·k-1 va 3·k-1 ham M ga tegishli bo'ladi. M to'plamning 1000 dan kichik barcha elementlarini kamayish yo'nalishida chiqaring.

Massiv 142. 1 soni M to'plamga tegishli. Agar k soni M ga tegishli bo'lsa, u holda 2·k-1 va 3·k-1 ham M ga tegishli bo'ladi. M to'plamning 1000 dan kichik barcha elementlarini o'sish yo'nalishida chiqaring.

Massiv 143. Natural N son va butun qiymatli hamda elementlari o'sish yo'nalishida tartiblangan A[1..N] massiv berilgan. Massivning turli elementlari sonini aniqlang.

Massiv 144. Natural N, M sonlari va butun qiymatli A[1..N][1..M] massivi berilgan. Massivning elementlari turli sonlardan iborat bo'lgan satrlari tartib raqamlarini alohida satrlarda chiqaring. Bunday satrlar yo'q bo'lsa 0 chiqaring.

Massiv 145. Natural N, M sonlari va butun qiymatli A[1..N][1..M] massivi berilgan. Massivning elementlari turli sonlardan iborat bo'lgan ustunlari sonini chiqaring.

Massiv 146. Natural N, M sonlari va butun qiymatli A[1..N][1..M] massivi berilgan. Massivning toq tartib raqamli satrlarini o'sish yo'nalishida, juft tartib raqamli satrlarini kamayish yo'nalishida tartiblab chiqaring.

Massiv 147. Natural N, M sonlari va butun qiymatli A[1..N][1..M] massivi berilgan. Massivning juft tartib raqamli ustunlarini o'sish yo'nalishida, toq tartib raqamli ustunlarini kamayish yo'nalishida tartiblab chiqaring.

Massiv 148. Natural N son va butun qiymatli A[1..N][1..N] kvadrat matritsa berilgan. Matritsaning asosiy diagonali elementlari yig'indisini aniqlang. Kvadrat matritsaning asosiy diagonali elementlari deb quyidagilarga aytiladi: A[1][1], A[2][2], A[3][3], ..., A[N][N].

Massiv 149. Natural N son va butun qiymatli A[1..N][1..N] kvadrat matritsa berilgan. Matritsaning qo'shma diagonali elementlarining maksimalini aniqlang. Kvadrat matritsaning qo'shma diagonali elementlari deb quyidagilarga aytiladi: A[1][N], A[2][N-1], A[3][N-2], ..., A[N][1].

Massiv 150. Natural N son va butun qiymatli A[1..N][1..N] kvadrat matritsa berilgan. Matritsaning asosiy diagonaliga parallel diagonallar elementlari yig'indilarini alohida satrlarda chiqaring. Matritsaning asosiy diagonaliga parallel diagonallar: 1) A[1][N]; 2) A[1][N-1], A[2][N]; 3) A[1][N-2], A[2][N-1], A[3][N]; ...; 2·N-1) A[N][1].

Massiv 151. Natural N son va butun qiymatli A[1..N][1..N] kvadrat matritsa berilgan. Matritsaning qo'shma diagonaliga parallel diagonallar elementlari ko'paytmalarini alohida satrlarda chiqaring. Matritsaning qo'shma diagonaliga parallel diagonallar: 1) A[1][1]; 2) A[1][2], A[2][1]; 3) A[1][3], A[2][2], A[3][1]; ...; 2·N-1) A[N][N].

Massiv 152. Natural N son va butun qiymatli A[1..N][1..N] kvadrat matritsa berilgan. Matritsaning asosiy diagonaliga parallel diagonallar elementlari eng kattalarini alohida satrlarda chiqaring.

Massiv 153. Natural N son va butun qiymatli A[1..N][1..N] kvadrat matritsa berilgan. Matritsaning qo'shma diagonaliga parallel diagonallar elementlari eng kichiklarini alohida satrlarda chiqaring.

Massiv 154. Natural N son va butun qiymatli A[1..N][1..N] kvadrat matritsa berilgan. Matritsaning asosiy diagonalidan yuqorida joylashgan elementlarning maksimalini chiqaring.

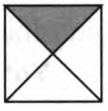
Massiv 155. Natural N son va butun qiymatli A[1..N][1..N] kvadrat matritsa berilgan. Matritsaning qo'shma diagonalidan pastda joylashgan elementlarning minimalini chiqaring.

Massiv 156. Natural N va M sonlar, butun qiymatli A[1..N] va B[1..M] sonlar massivlari berilgan. Bu massivlarni o'xshash yoki o'xshash emasligini tekshiring. Massivlar **o'xshash** deyiladi, agar ularda bir xil sonlar ishtirok etgan bo'lsa.

Yo'llanma. A massivning har bir elementi B massivda borligi va B massivning har bir elementi A massivda borligi tekshiriladi.

Massiv 157. Natural N, M sonlari va butun qiymatli A[1..N][1..M] massivi berilgan. Massivning birinchi satriga o'xshash satrlarining tartib raqamlarini alohida satrlarda chiqaring. Bunday satrlar yo'q bo'lsa 0 chiqaring. Massiv satrlari **o'xshash** deyiladi, agar ularda bir xil sonlar ishtirok etgan bo'lsa.

Massiv 158. Natural N, M sonlari va butun qiymatli A[1..N][1..M] massivi berilgan. Massivning oxirgi ustuniga o'xshash ustunlarning tartib raqamlarini alohida satrlarda chiqaring. Bunday ustunlar yo'q bo'lsa 0 chiqaring. Massiv ustunlari **o'xshash** deyiladi, agar ularda bir xil sonlar ishtirok etgan bo'lsa.

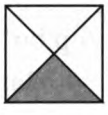


Massiv 159. Natural N son va butun qiymatli ikki o'lchovli A[1..N][1..N] massiv berilgan. Massivning ma'lum qismi bo'yalgan bo'lib, shu bo'yalgan qismdagi elementlar ichidan eng katta va eng kichigini aniqlang.

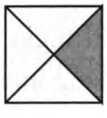
Yo'llanma. Shuni aniqlash mumkinki, agar N juft bo'lsa, bo'yalgan qismdagi satrlar soni M=N/2 ga, agar N toq bo'lsa, bo'yalgan qismdagi satrlar soni M=(N/2)+1 ga teng bo'ladi, ya'ni bo'yalgan qismdagi satrlar soni C++ da M=(N+1)/2 ga teng bo'ladi.

Bo'yalgan qismning har bir satrining chap va o'ng chegarasini aniqlab olamiz. Satr tartib raqamini **s**, ustun tartib raqamini **u** bilan belgilaymiz. Bo'yalgan qismda satr va ustunlar orasidagi quyidagi bog'liqlikni ko'rish qiyin emas:

s	u
1	$1 \le u \le N$
2	$2 \le u \le N-1$
3	$3 \le u \le N-2$
...	...
M	$M \le u \le N-(M-1)$

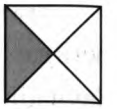


Massiv 160. Natural N son va butun qiymatli ikki o'lchovli A[1..N][1..N] massiv berilgan. Massivning ma'lum qismi bo'yalgan bo'lib, shu bo'yalgan qismdagi elementlar yig'indisini aniqlang.



Massiv 161. Natural N son va butun qiymatli ikki o'lchovli A[1..N][1..N] massiv berilgan. Massivning ma'lum qismi bo'yalgan bo'lib, shu bo'yalgan qismdagi elementlar ichidan eng katta va eng kichigining ko'paytmasini aniqlang.

Massiv 162. Natural N son va butun qiymatli ikki o'lchovli A[1..N][1..N] massiv berilgan. Massivning ma'lum qismi bo'yalgan bo'lib, shu bo'yalgan qismdagi elementlar o'rta arifmetigini aniqlang.



Massiv 163. Natural N son va butun qiymatli ikki o'lchovli A[1..N][1..N] massiv berilgan. Agar massivning k-satri elementlari o'suvchi ketma-ketlik tashkil etsa, u holda B[k]=1, aks holda B[k]=0 bo'lsin. Shu B[1..N] massivini chiqaring.

Massiv 164. Natural N son va butun qiymatli ikki o'lchovli A[1..N][1..N] massiv berilgan. Agar massivning k-satri elementlari o'smaydigan ketma-ketlik tashkil etsa, u holda B[k]=1, aks holda B[k]=0 bo'lsin. Shu B[1..N] massivini chiqaring.

Massiv 165. N ta kulrang va M ta oq sichqon aylana bo'ylab o'tiribdi. Mushuk aylana bo'ylab soat mili yo'nalishida yurib har S-sichqonni yeydi. Hisob kulrang sichqondan boshlanadi. Ma'lum vaqtdan so'ng K ta kulrang va L ta oq sichqon qolgan bo'lsa, boshida sichqonlar qanday tartibda o'tirganini aniqlang. Agar joylashtirish usullari ko'p bo'lsa, bittasini chiqarish yetarli.

Yo'llanma. Masala yechimlaridan biri sifatida quyidagi qadamlarni keltirish mumkin:

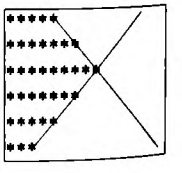
1-qadam. Sichqonlarning t-siga rangidan qat'iy nazar, son(t) massiv elementini mos qo'yamiz. Agar son(t)=1 bo'lsa t-sichqon tirik, son(t)=0 bo'lsa t-sichqon yeyilgan. Masala shartiga ko'ra boshlang'ich vaqtda barcha sichqonlar uchun son(t)=1.

2-qadam. Har t-sichqon yeyilsa, ya'ni son(t)=0 bo'lsa, ma'lum qadamdan so'ng yeyilgan sichqonlar soni N+M-K-Lga teng bo'lishi kerak, chunki masala shartiga ko'ra sichqonlarning umumiy soni N+M ta, tirik qolishi kerak bo'lgan kulrang sichqonlar soni K ta, tirik qolishi kerak bo'lgan oq sichqonlar soni L ta.

3-qadam. Masala shartiga ko'ra hisob kulrang sichqondan boshlanadi. Shuning uchun natijani chop etishda hisobni (t=1dan boshlab) kulrang sichqonlardan boshlaymiz. H1 tirik (son(t)=1), H2 (son(t)=0) yeyilgan kulrang sichqonlar soni bo'lsa, ularning soni mos ravishda (H1 va H2 noldan boshlangani uchun) K-1 va N-K-1dan oshmaydi. Kulrang sichqonlardan ortgan tirik sichqonlarni oq sichqon deb qaraymiz.

Massiv 166. Oq va kulrang B ta sichqon aylana bo'ylab joylashgan. Mushuk har M-sichqonni yeydi. Agar A ta sichqon tirik qolsa, ulardan kamida 2 tasi kulrang bo'lsa, sichqonlarning boshlang'ich joylashishini aniqlovchi dastur tuzing. Agar joylashtirish usullari ko'p bo'lsa, bittasini chiqarish yetarli.

Massiv 167. Natural N son va ikki o'lchovli butun qiymatli A[1..N][1..N] massiv berilgan. Massivdan ikkinchi shunday B[1..N][1..N] massiv hosil qilingki, bunda B[k][m] element A massivni A[k][m] elementi orqali o'tuvchi diagonallari bilan o'ng tomondan chegaralangan qismidagi elementlarning eng kattasi bo'lsin.



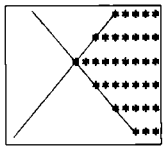
Yo'llanma. Masalani to'g'ridan to'g'ri hal etish uchun har bir A[k][m] elementga mos B[k][m] element sifatida chizmada ko'rsatilganidek, chegaralangan qismdagi elementlar ichidan eng kattasini olishimiz kerak. Buning uchun har bir A[k][m] element uchun chegaralangan qismni aniqlovchi qonuniyat topish (bu oson ish emas) lozim bo'ladi. Va natijada chegaralangan qism elementlarining eng kattasi topiladi. Lekin masala shartiga e'tibor bersak, undagi "eng katta" iborasi ishimizni osonlashtiradi. Quyidagicha mulohaza yuritamiz:

1) B[k][1] = A[k][1], k=1, 2, ..., N, chunki A[k][1] dan chapda elementlar yo'q;

2) B[k][m] elementni topish uchun to'rtta elementni taqqoslash yetarli, ya'ni :

B[k][m] =MAX(B[k-1][m-1], B[k][m-1], B[k+1][m-1], A[k][m]), k=2, ..., N-1, m=2, ..., N, chunki B[k-1][m-1], B[k][m-1], B[k+1][m-1] elementlar o'zidan chapdagi chegaralangan qismdagi elementlarning eng kattasi sifatida tanlab olingan edi;

3) B[1][m] va B[N][m] elementlarni uchta elementni taqqoslash orqali aniqlash mumkin, m=2,...,N. Lekin 2)-mulohazadan foydalanish dasturni soddalashtiradi. Buning uchun quyidagi yordamchi elementlarni kiritamiz: B[0][m]=A[1][m], B[N+1][m]=A[N][m], m=1,...,N.



Massiv 168. Natural N son va ikki o'lchovli butun qiymatli A[1..N][1..N] massiv berilgan. Massivdan ikkinchi shunday B[1..N][1..N] massiv hosil qilingki, bunda B[k][m] element A massivni A[k][m] elementi orqali o'tuvchi diagonallari bilan chap tomondan chegaralangan qismidagi elementlarning eng kattasi bo'lsin.

Massiv 169. Natural N, M sonlari va kamaymaydigan yo'nalishda tartiblangan A[1..N], B[1..M] massivlar berilgan. Ikkala massivni birlashtirib kamaymaydigan yo'nalishda tartiblangan C[1..N+M] massivni hosil qiling.

Massiv 170. Natural N son va elementlari 0 yoki 1 bo'lgan A[1..N][1..N] massiv berilgan. Massivning faqat 1 dan iborat bo'lgan eng katta o'lchamli kvadrat qism massivini chiqaring.

Massiv 171. Natural N son va elementlari 0 yoki 1 bo'lgan A[1..N][1..N] massiv berilgan. Massivning faqat 1 dan iborat bo'lgan eng katta o'lchamli to'g'ri to'rtburchak shakldagi qism massivini chiqaring.

Massiv 172. Natural N son va butun qiymatli A[1..N] massiv berilgan. Har bir k-kishini shu N ta odamdan aniq A[k] sondagi kishilar bilan tanishtirish mumkin yoki mumkin emasligini aniqlang.

Massiv 173. Natural N son, A_1, A_2, ..., A_N va B_1, B_2, ..., B_N sonlar ketma-ketligi berilgan. Ulardan shunday N ta (A_K, B_M) juftlik tuzish kerakki, barcha juftliklar ko'paytmasining yig'indisi eng katta bo'lsin. Har bir ketma-ketlik elementi yig'indi ifodasida faqat bir marta ishtirok etadi!

Yo'llanma. Juftliklarning ko'paytmasining yig'indisi eng katta qiymatli bo'lishi uchun ketma-ketliklarni bir xil (masalan, o'sish) yo'nalishda tartiblash zarur, chunki A<B va C<D bo'lsa, u holda:

$$A \cdot C + B \cdot D >= A \cdot D + B \cdot C$$

o'rinli bo'ladi (mustaqil isbotlang).

Massiv 174. Natural N son, A_1, A_2, ..., A_N va B_1, B_2, ..., B_N sonlar ketma-ketligi berilgan. Ulardan shunday N ta (A_K, B_M) juftlik tuzish kerakki, barcha juftliklar ko'paytmasining yig'indisi eng kichik bo'lsin. Har bir ketma-ketlik elementi yig'indi ifodasida faqat bir marta ishtirok etadi.

Massiv 175. Natural N son, Oxy tekisligida markazi B(x0, y0) nuqtadagi M radiusli doira va N ta nuqta koordinatalari orqali berilgan. Berilgan nuqtalar ichida doiraga tegishli bo'lganlariga mos Ox o'qiga nisbatan simmetrik va Oy o'qiga nisbatan simmetrik nuqtalarning koordinatalarini izoh bilan chiqaring.

Massiv 176. Natural N son, Oxy tekisligida markazi B(x0, y0) nuqtadagi M radiusli doira va N ta nuqta koordinatalari orqali berilgan. Berilgan nuqtalar ichida doiraga tegishli bo'lganlariga mos koordinata boshiga nisbatan simmetrik nuqtalarning koordinatalarini chiqaring.

Massiv 177. Natural N son, Oxy tekisligida tomonlarining uzunliklari B va M ga teng to'g'ri to'rtburchak va N ta nuqta koordinatalari orqali berilgan. To'g'ri to'rtburchakning tomonlari koordinata o'qlariga parallel va diagonallari D(x0, y0) nuqtada kesishadi. Berilgan nuqtalar ichida to'g'ri to'rtburchakka tegishli bo'lganlariga mos Ox o'qiga nisbatan simmetrik, Oy o'qiga nisbatan simmetrik nuqtalarning koordinatalarini izoh bilan chiqaring.

Massiv 178. Natural N soni berilgan (N<21). A[1..N][1..N] massivga 1, 2, 3,..., N^2 sonlarni o'sish tartibida quyidagi qonuniyatga asosan joylashtirng va chiqaring (o'ng spiral):

1	2	→	→	→	↓
→	→	→	→	↓	↓
↑	→	→	↓	↓	↓
↑	↑	N^2	←	↓	↓
↑	↑	←	←	←	↓
↑	←	←	←	←	←

Massiv 179. Natural N soni berilgan (N<21). A[1..N][1..N] massivga 1, 2, 3,..., N^2 sonlarni o'sish tartibida quyidagi qonuniyatga asosan joylashtirng va chiqaring (chap spiral):

1	↓	←	←	←	←
2	↓	↓	←	←	↑
↓	↓	↓	N^2	↑	↑
↓	↓	→	↑	↑	↑
↓	→	→	→	↑	↑
→	→	→	→	→	↑

= D =

Massiv 180. Natural N son berilgan. N ta birdan iborat sonning kvadratini aniqlang.

K:	1	2	3	10	17
CH:	1	121	12321	1234567900987654321	12345679012345678765432098765432 1

Massiv 181. Natural N son va A[1..N] butun sonlar massivi berilgan. Massivning har bir elementini o'zidan keyin kelgan eng yaqin katta qiymatli elementga almashtirilsin. Agar bunday element bo'lmasa 0 ga almashtirilsin. Masalan, A={1, 3, 2, 5, 3, 4} bo'lsa, u holda javob A={3, 5, 5, 0, 4, 0}.

K:	5 5 1 2 3 4	7 1 5 8 1 0 12 3	9 1 2 3 4 5 6 7 8 9	11 12 0 5 7 3 9 2 15 55 10 11
CH:	0 2 3 4 0	5 8 12 12 12 0 0	2 3 4 5 6 7 8 9 0	15 5 7 9 9 15 15 55 0 11 0

Massiv 182. Natural N son va kamaymaydigan yo'nalishda tartiblangan A[1:N] natural sonlar massivi berilgan, ya'ni A[1] ≤ A[2] ≤ ... ≤ A[N]. Berilgan massiv elementlarining yig'indisi orqali ifodalab bo'lmaydigan eng kichik natural sonni aniqlang. Massivning har bir elementi yig'indida faqat bir marta qatnashishi mumkin. Yig'indi sifatida massivning bitta elementi qatnashishi ham mumkin.

K:	4 2 3 4 5	7 1 2 4 6 8 12 34	9 1 2 7 9 13 17 19 20 21	12 1 2 3 4 5 6 7 8 9 55 56 57
CH:	1	68	4	46

Massiv 183. Birinchi satrda natural N soni berilgan. Ikkinchi satrda 1 ta, 3-satrda 2 ta, ... k-satrda (k–1) ta, ..., (N+1)-satrda N ta butun son berilgan. Ikkinchi satrdan boshlab oxirigi satrgacha joylashgan sonlar hosil qilgan uchburchakning uchidan asosiga qarab harakatlanib, yo'lda uchragan sonlar qo'shib boriladi. Harakatning har bir qadamida bitta pastki qatorga quyidagicha o'tiladi: vertikal bo'yicha pastga yoki diagonal bo'yicha o'ngga. Uchburchak asosiga yetib kelganda hosil bo'ladigan yig'indilar ichidan eng kattasini aniqlang.

K:	2	2	3	3	3	3	5	8

	5 3 9	5 9 3	5 9 3 4 5 4	5 3 9 4 5 4	5 9 3 5 4 4	5 3 9 5 4 4	7 3 8 8 1 0 2 7 4 4 4 5 2 6 5	5 14 67 21 34 96 19 25 77 54 23 89 41 24 19 11 62 29 1 87 24 3 66 0 2 7 13 21 1 8 3 90 31 80 17 2
CH:	14	14	19	19	19	18	30	449

Massiv 184. Natural N son va A_1, A_2, ..., A_N butun sonlar ketma-ketligi berilgan. Ketma-ket kelgan hadlar yig'indilari ichida maksimali hosil bo'ladigan eng uzun oraliqni va maksimal yig'indini aniqlang. Bunday oraliqlar bir nechta bo'lsa ixtiyoriysini chiqaring.

K:	6 -1 1 3 -2 4 -1	5 -1 -1 -1 -1 -1	6 -1 0 1 0 -1 0	8 -1 0 2 -1 -1 2 0 -1
CH:	2 5 6	1 1 -1	2 4 1	2 7 2

Massiv 185. Natural N son va A_1, A_2, ..., A_N sonlar ketma-ketligi berilgan. Ketma-ket kelgan hadlar yig'indilari ichida maksimali hosil bo'ladigan eng qisqa oraliqni va maksimal yig'indini aniqlang. Bunday oraliqlar bir nechta bo'lsa ixtiyoriysini chiqaring.

K:	6 -1 1 3 -2 4 -1	5 -1 -1 -1 -1 -1	6 -1 0 1 0 -1 0	8 -1 0 2 -1 -1 2 0 -1
CH:	2 5 6	1 1 -1	3 3 1	3 3 2

Massiv 186. Natural N, M, K ($1 \le N \le 10^5$, $N \le M \le 10^9$, $1 \le K \le N$) sonlar va o'sish bo'yicha tartiblangan A[1..N] butun sonlar massivi ($1 \le A[\ ] \le M$) berilgan. Ali asfalt yotqizuvchilar guruhi rahbari bo'lib, guruh K ta bo'limdan iborat. Aliga uzunligi M kilometr bo'lgan Alfa va Betta shaharlari orasidagi yo'lning A[1]-kilometrida, A[2]-kilometrida, ..., A[N]-kilometrida asfalt yorilgani haqida xabar berishdi. Ali K ta bo'limni shu joylarga asfalt yotqizishga yuborishi kerak. Har bir asfalt yotqizuvchi bo'lim quyidagicha ishlaydi:

1) Yorilgan joyni o'z ichiga olgan kilometrga to'liq asfalt yotqizishadi;
2) Bir asfalt yotqizilishi kerak bo'lgan kilometrdan ikkinchi asfalt yotqizilishi kerak bo'lgan kilometrga borishlari kerak bo'lsa, shu ikkala kilometr orasiga ham asfalt yotqizishadi.

Ali bo'limlarni samarali shunday taqsimladiki, eng kam yo'l va barcha yoriqlar asfalt qilindi. Eng kamida necha kilometr yo'l asfalt qilinganini aniqlang.

K:	4 100 2 20 30 75 80	5 100 3 1 2 4 60 87	1 1000000000 1 228	2 1000000000 1 1 1000000000	4 10 4 1 3 6 10
CH:	17	6	1	1000000000	4

Massiv 187. Natural B va K ($2 \le B \le 1000$, $1 \le K \le 10^5$) sonlar va A[1..K] butun sonlar massivi ($0 \le A[\ ] \le B-1$) berilgan. Ma'lumki, B asosli sanoq sistemasida N soni quyidagicha ko'rinishida tasvirlanadi:

$$N=A[1]\cdot B^{K-1}+A[2]\cdot B^{K-2}+\ldots+A[K-1]\cdot B+A[K]$$

Agar N soni juft bo'lsa 0 sonini, aks holda 1 sonini chiqaring.

K:	13 3 3 2 7	10 9 1 2 3 4 5 6 7 8 9	99 5 32 92 85 74 4	8 2 6 5	99 10 97 2 2 2 2 2 2 2 2 2 2	4 2 1 2
CH:	0	1	1	1	1	0

Massiv 188. Natural N, D ($1 \le N \le 1000$, $1 \le D \le 10^9$) sonlar va A[1..N] butun sonlar massivi ($1 \le A[\ ] \le 10^9$) berilgan. Massivning elementlari ichida ikkita elementi farqining absolyut qiymati D sonidan ortmaydigan juftliklar sonini aniqlang. Bunda (A[t], A[k]) va (A[k], A[t]) juftliklarni har xil juftlik deb hisoblang.

K:	5 10 10 20 50 60 65	5 1 55 30 29 31 55	6 10 4 6 4 1 9 3	7 100 19 1694 261 162 1 234 513
CH:	6	6	30	8

Massiv 189. Natural N va M ($2 \le N \le M \le 100$) sonlar va A[1..M] butun sonlar massivi ($4 \le A[\ ] \le 1000$) berilgan. Ali N ta do'stiga sovg'a qilish uchun do'kondan pazzl o'yinini sotib olmoqchi bo'ldi. Do'konda M ta pazzl bo'lib, 1-pazzl A[1] bo'lakdan, 2-pazzl A[2] bo'lakdan,..., M-pazzl A[M] bo'lakdan iborat ekan. Do'stlari xafa bo'lmasligi uchun Ali eng ko'p bo'lakli pazzl bilan eng kam bo'lakli pazzl orasidagi farq eng kam bo'lishini xohladi. Ali erishishi mumkin bo'lgan eng kam farqni aniqlang.

K:	2 2 4 4	4 6 10 12 10 7 5 22	2 10 4 5 6 7 8 9 10 11 12 12	2 2 1000 4	4 5 818 136 713 59 946
CH:	0	5	0	996	759

Massiv 190. Natural N ($3 \le N \le 1000$) son va A[1..N] butun sonlar massivi ($1 \le A[\ ] \le 10^9$) berilgan. Massiv elementlari asosida yasash mumkin bo'lgan barcha uchburchaklar sonini aniqlang. Har bir yangi uchburchakda massivning hech bo'lmasa bitta yangi elementi qatnashishi shart.

K:	3 1 1 1	3 1 1000 1000000	4 1 2 4 3	5 3 2 4 1 5	9 11 5 20 10 7 6 19 21 100
CH:	1	0	1	3	27

Massiv 191. Natural N ($1 \le N \le 1000$) son berilgan. N ta kartochka yonma-yon joylashtirilgan. Kartochkalarda 1 dan N gacha bo'lgan turli sonlar yozilgan. Ali va Vali shunday o'yin o'ynashmoqda:

- Birinchi yurishni Ali amalga oshiradi;
- O'yinchi eng chapdagi yoki eng o'ngdagi kartochkani olishi mumkin;
- O'yinchi har doim katta son yozilgan kartochkani oladi.

Agar kartochkalardagi sonlar ketma-ketligi berilgan bo'lsa, shu qoida bo'yicha o'ynalganda Ali va Vali yig'adigan kartochkalardagi sonlar yig'indisini aniqlang.

K:	4 4 1 2 10	7 1 2 3 4 5 6 7	1 3	6 580 376 191 496 73 44	5 10 15 18 1 28 16
CH:	12 5	16 12	3 0	844 916	44 28

Massiv 192. Lampochkalardan iborat 3x3 jadvalda avval barcha lampochka yoniq holatda. Agar biror lampochka bosilsa, shu va undan yuqoridagi va pastdagi, chapdagi va o'ngdagi lampochkalar holatini teskarisiga o'zgartiradi, ya'ni yoniq (1 ga teng) bo'lsa o'chadi (0 ga teng bo'ladi), va aksincha, o'chiq (0 ga teng) bo'lsa yonadi (1 ga teng bo'ladi). Sizga qaysi lampochka necha marta bosilgani haqidagi ma'lumot A[1..3][1..3] jadval ($0 \le A[\,][\,] \le 100$) orqali berilgan. Har bir lampochkaning holatini aniqlang.

K:	1 0 0 0 0 0 0 0 1	1 0 1 8 8 8 2 0 3	13 85 77 25 50 45 65 79 9	96 95 5 8 84 74 67 31 61	24 54 37 60 63 6 1 84 26	36 48 42 45 41 66 26 64 1	0 0 0 0 0 0 0 0 0
CH:	0 0 1 0 1 0 1 0 0	0 1 0 0 1 1 1 0 0	0 0 0 0 1 0 0 0 0	0 1 1 0 1 1 1 0 1	1 1 0 1 0 1 0 1 1	0 0 1 1 1 1 0 1 0	1 1 1 1 1 1 1 1 1

Massiv 193. Natural N ($1 \le N \le 10^5$) son va H[1..N] butun sonlar massivi ($1 \le H[\,] \le 10^4$) berilgan. Gorizontal yo'l yoqasida chapdan o'ngga qarab tartib raqami berilgan N ta daraxt o'smoqda. K-daraxtning balandligi H[K] metrga teng (K=1, 2, ..., N). Olmaxon har bir daraxtning uchidagi yong'oqlarning barchasini yeb tugatishi shart. Olmaxon hozir 1-daraxtning asosida turibdi. U 1 sekundda quyidagilardan bittasini bajarishi mumkin:

- Daraxt bo'ylab bir birlik yuqoriga ko'tariladi yoki bir birlik pastga tushadi;
- Daraxt uchidagi yong'oqni yeydi;
- Agar keyingi daraxtning balandligi olmaxon turgan balandlikdan kichik bo'lmasa, u holda u daraxtning o'zi turgan H-metr balandlikdan keyingi daraxtning H-metr balandligiga sakray oladi.

Barcha yong'oqlarni yeb tugatish uchun olmaxon sarflaydigan eng kam vaqtni aniqlang.

K:	2 1 2	5 2 1 2 1 1	4 3152 2250 2 9577	3 10000 10000 10000	1 1	3 10000 1 10000
CH:	5	14	15884	10005	2	30003

Massiv 194. Natural N, K ($1 \le N$, $K \le 100$) sonlar va A[1..N] butun sonlar massivi ($1 \le A[\,] \le 10^9$) berilgan. Oxy tekisligida N ta kvadrat berilgan bo'lib, t-kvadratning chap quyi

uchi (0; 0) nuqtada, o'ng yuqori uchu esa (A[t]; A[t]) nuqtada joylashgan. Aniq K ta kvadratga tegishli bo'lgan biror butun koordinatali (x; y) nuqtaning koordinatalarini chiqaring. Agar bunday nuqta mavjud bo'lmasa -1 chiqaring. Masalan:

K:	4 3 5 1 3 4	3 1 2 4 1	4 50 5 1 10 2	5 2 10 9 19 12 14	5 2 1 2 5 4 3	4 1 1 2 999999991 999999999
CH:	2 1	4 0	-1	14 14	4 4	999999999 999999999

Massiv 195. Natural N, K ($1 \le N$, $K \le 100$) sonlar va A[1..N] butun sonlar massivi ($1 \le A[\,] \le 10^9$) berilgan. Sonni go'zal deymiz, agar son faqat 4 va 7 raqamlaridan iborat bo'lsa. Masalan, 4, 7, 44, 47 va 74 sonlari go'zal, 5 va 100 sonlari go'zal emas. Go'zal raqamlari soni K tadan oshmaydigan massiv elementlari sonini aniqlang.

K:	3 4 1 2 4	3 2 447 44 77	2 2 507978501 180480073	2 3 247776868 480572137	2 2 447 77777
CH:	3	2	2	1	0

Massiv 196. Natural N ($1 \le N \le 1000$) son va A[1..N] manfiymas butun sonlar massivi ($0 \le A[\,] \le 10000$) berilgan. Ali do'sti Valining dasturlash sporti bo'yicha muntazam o'tkazib boriladigan N ta musobaqada yiqqan ballarini A massivga yozib borar edi. Ali faqat ikki holda **ajablanish** holatiga tushardi: agar Vali musobaqada rekord qo'ysa, ya'ni avval olgan ballaridan ko'p ball to'plasa va agar Vali musobaqada anti-rekord qo'ysa, ya'ni avval olgan ballaridan kam ball to'plasa. Birinchi musobaqa natijasiga ko'ra Ali ajablanish holatiga tushmaganligini hisobga olib Alining ajablanish holatiga tushgan musobaqalar sonini aniqlang.

K:	5 100 50 200 150 200	1 6	2 2 1	5 7 36 53 81 100	10 1 3 3 4 6 7 7 8 9 10	2 5 5	1 0
CH:	2	0	1	7	7	0	0

Massiv 197. Natural N ($1 \le N \le 100$) son va S[1..N] butun sonlar massivi ($1 \le S[\,] \le 100$) berilgan. Ali N ta do'stiga N xil sovg'a sotib oldi. U do'stlariga 1 dan N gacha tartib raqami berdi va har biriga bittadan sovg'a topshirdi. Do'stlari sovg'ani olgach biri boshqasiga sovg'a qila boshladi. Ali kuzatib turib t-tartib raqamli do'sti sovg'asini S[t]-tartib raqamli do'stiga sovg'a qilganini ko'rdi. Ali sovg'asini hech kimga sovg'a qilmagan do'stlarini sovg'asini o'z-o'ziga sovg'a qilgan deb hisobladi. Ali qarasa, hamma do'stida yana bir donadan sovg'a bo'lib qolibdi. Har bir t-tartib raqami uchun Alining t-do'stiga sovg'a bergan do'stini tartib raqamini aniqlang.

K:	4 2 3 4 1	3 1 3 2	5 5 4 3 2 1	2 1 2	10 2 9 4 6 10 1 7 5 3 8	8 1 3 5 2 4 8 6 7
CH:	4 1 2 3	1 3 2	5 4 3 2 1	1 2	6 1 9 3 8 4 7 10 2 5	1 4 2 5 3 7 8 6

Massiv 198. Natural N ($1 \le N \le 100$) son va A[1..N] butun sonlar massivi ($1 \le S[\,] \le 100$) berilgan. Massiv elementlaridan faqat bittasini shunday olib tashlash kerakki, qolgan

elementlar yig'indisi juft son bo'lsin. Olib tashlanadigan elementni necha xil usul bilan tanlash mumkinligini aniqlang.

K:	1 1	2 1 1	7 7 7 7 7 7 7 7	5 1 1 3 2 2	10 1 2 2 3 4 4 4 2 2 2	8 1 2 3 4 5 6 7 8
CH:	1	0	7	3	8	4

7-§. FOYDALANUVCHI FUNKSIYALARINI TUZISHGA DOIR VAZIFALAR

Quyida berilgan barcha vazifalarda foydalanuvchi funksiyalarini tashkil etish talab qilinadi. Funksiya nomi dasturchi tomonidan o'zgartirilishi mumkin, lekin funksiya argumentlari masala talabiga mos bo'lishi shart. Ko'pgina masalalar shartida tasodifiy sonlar hosil qilish, ya'ni generatsiya qilish talab etilgan.

= A =

Funksiya 1. Butun turdagi S sonining 3-darajasini qaytaradigan Daraja3(S) nomli foydalanuvchi funksiyasini tuzing. Asosiy dasturda 5 ta butun son kiriting hamda funksiyadan foydalanib ularning 3-darajasini hisoblang.

Funksiya 2. Ikkita sonning o'rta arifmetigi qiymatini qaytaradigan Mid(X, Y) nomli foydalanuvchi funksiyasini tuzing. Asosiy dasturda A, B, C, D sonlar kiriting hamda funksiyadan foydalanib (A, B), (A, C), (A, D), (B, C), (B, D) va (C, D) juftliklarning o'rta arifmetigi qiymatini hisoblang.

Funksiya 3. Uchta natural X, Y va Z sonlardan uchburchak hosil qilish mumkin yoki yo'qligiga mos mantiqiy qiymat qaytaradigan UchbHa(X, Y, Z) nomli foydalanuvchi funksiyasini tuzing. Asosiy dasturda A, B, C, D sonlar kiriting hamda funksiyadan foydalanib (A, B, C), (A, B, D), (A, C, D) va (B, C, D) uchliklar uchun uchburchak hosil bo'lishini tekshiring.

Funksiya 4. Teng tomonli uchburchakning A tomoniga ko'ra perimetri va yuzasini qaytaradigan Peryuz(A) nomli foydalanuvchi funksiyasini tuzing. Asosiy dasturda 5 ta son kiriting hamda funksiyadan foydalanib ularga mos teng tomonli uchburchakning perimetri va yuzasini hisoblang.

Funksiya 5. Butun turdagi S soni raqamlarining ko'paytmasini qaytaradigan RaqP(S) nomli foydalanuvchi funksiyasini tuzing. Asosiy dasturda 5 ta butun son kiriting hamda funksiyadan foydalanib ularning raqamlari ko'paytmasini hisoblang.

Funksiya 6. Butun turdagi S sonining raqamlarini teskari tartibda joylashtirib yangi son hosil qilib qaytaradigan AksSon(S) nomli foydalanuvchi funksiyasini tuzing. Asosiy dasturda

5 ta butun son kiriting hamda funksiyadan foydalanib ularning raqamlarini teskari tartibda joylashtirishdan hosil bo'lgan sonlarni chiqaring.

Funksiya 7. X va Y sonlarining qiymatlarini almashtirib qaytaradigan Almash(X, Y) nomli foydalanuvchi funksiyasini tuzing. Asosiy dasturda 5 ta sonlar juftligini kiriting hamda funksiyadan foydalanib shu juftliklardagi sonlar qiymatlarini almashtirib izoh bilan chiqaring.

Funksiya 8. X, Y va Z sonlarini o'sish yo'nalishida tartiblab qaytaradigan Tartib(X, Y, Z) nomli foydalanuvchi funksiyasini tuzing. Asosiy dasturda 5 ta sonlar uchligini kiriting hamda funksiyadan foydalanib shu uchliklarni tartiblab izoh bilan chiqaring.

Funksiya 9. X soni musbat bo'lsa 1, manfiy bo'lsa –1, 0 ga teng bo'lsa 0 qiymatlarni qaytaradigan Ishora(X) nomli foydalanuvchi funksiyasini tuzing. Asosiy dasturda 5 ta son kiriting hamda funksiyadan foydalanib shu sonlarning ishorasini aniqlang.

Funksiya 10. A, B va C sonlarga asosan $Ax^2+Bx+C=0$ kvadrat tenglamaning yechimlari sonini qaytaradigan IldizSoni(A, B, C) nomli foydalanuvchi funksiyasini tuzing. Asosiy dasturda 5 ta sonlar uchligini kiriting hamda funksiyadan foydalanib shu uchliklarga mos kvadrat tenglamalarning yechimlari sonini izoh bilan chiqaring.

Funksiya 11. Musbat X soni uchun X radiusli aylana uzunligi va doira yuzini qaytaradigan AylanaLS(X) nomli foydalanuvchi funksiyasini tuzing. Asosiy dasturda 5 ta son kiriting hamda funksiyadan foydalanib shu sonlarga mos aylana uzunligi va doira yuzini aniqlang.

Funksiya 12. Butun X sonidan boshlab butun Y sonigacha bo'lgan barcha sonlar yig'indisini qaytaradigan Sum(X, Y) nomli foydalanuvchi funksiyasini tuzing. Asosiy dasturda 5 ta sonlar juftligini kiriting hamda funksiyadan foydalanib shu juftliklarga mos sonlar yig'indisini izoh bilan chiqaring.

Funksiya 13. Oxy koordinata tekisligida (X, Y) koordinatali nuqta yotgan kvadrantning tartib raqamini qaytaradigan Chorak(X, Y) nomli foydalanuvchi funksiyasini tuzing. Asosiy dasturda 5 ta nuqta kiriting hamda funksiyadan foydalanib shu nuqtalar yotgan kvadrantlarning tartib raqamini izoh bilan chiqaring.

Funksiya 14. Musbat X soni biror butun sonning kvadrati bo'lishi yoki bo'lmasligini tekshiruvchi Kvadrat(X) nomli mantiqiy foydalanuvchi funksiyasini tuzing. Asosiy dasturda 5 ta son kiriting hamda funksiyadan foydalanib shu sonlar biror butun sonning kvadrati bo'lishi yoki bo'lmasligi haqida xabar chiqaring.

Funksiya 15. Musbat X soni butun Y sonining kvadrati bo'lishi yoki bo'lmasligini tekshiruvchi Kvadrat(X, Y) nomli foydalanuvchi funksiyasini tuzing. Asosiy dasturda 5 ta sonlar juftligi kiriting hamda funksiyadan foydalanib shu juftliklarni tekshiring.

= B =

Funksiya 17 Butun turdagi S sonini natural N-darajasini qaytaradigan Daraja(S, N) nomli foydalanuvchi funksiyasini tuzing. Asosiy dasturda A butun son kiriting hamda funksiyadan foydalanib uning 5 xil darajasini hisoblang.

Funksiya 16. Butun turdagi S sonining 5-darajasini qaytaradigan Daraja5(S, A) nomli foydalanuvchi funksiyasini tuzing. Bunda funksiya natijasi A parametr orqali qaytarilsin. Asosiy dasturda 5 ta butun son kiriting hamda funksiyadan foydalanib ularning 5-darajasini hisoblang.

Funksiya 18. Ikkita sonning o'rta arifmetigi va o'rta geometrigi qiymatini qaytaradigan Qiymat(X, Y, SumXY, ProdXY) nomli foydalanuvchi funksiyasini tuzing. Bunda funksiya natijasi SumXY va ProdXY parametrlar orqali qaytarilsin. Asosiy dasturda A, B, C, D sonlar kiriting hamda funksiyadan foydalanib (A, B), (A, C), (A, D), (B, C), (B, D) va (C, D) juftliklar uchun o'rta arifmetigi va o'rta geometrigi qiymatlarini hisoblang.

Funksiya 19. Teng tomonli uchburchakning A tomoniga ko'ra perimetri va yuzasini qaytaradigan PeryuzT(A, P, S) nomli foydalanuvchi funksiyasini tuzing. Bunda funksiya natijasi P va S parametrlar orqali qaytarilsin. Asosiy dasturda 5 ta son kiriting va funksiyadan foydalanib ularga mos teng tomonli uchburchakning perimetri va yuzasini hisoblang.

Funksiya 20. Natural X soni tub son bo'lsa TRUE, aks holda, ya'ni murakkab son bo'lsa FALSE qiymat qaytaradigan Tub(X, P) nomli mantiqiy foydalanuvchi funksiyasini tuzing. Bunda funksiya natijasi P parametr orqali qaytarilsin. Asosiy dasturda 5 ta son kiriting va funksiyadan foydalanib ularning tub yoki murakkabligini aniqlang.

Funksiya 21. Natural X va Y sonlari o'zaro tub bo'lsa TRUE, aks holda, ya'ni EKUB(X, Y)>1 bo'lsa FALSE qiymat qaytaradigan Otub(X, Y, P) nomli mantiqiy foydalanuvchi funksiyasini tuzing. Bunda funksiya natijasi P parametr orqali qaytarilsin. Asosiy dasturda A, B, C, D natural sonlar kiriting hamda funksiyadan foydalanib (A, B), (A, C), (A, D), (B, C), (B, D) va (C, D) juftliklar uchun o'zaro tublik xossasini tekshiring.

Funksiya 22. Natural S soni palindrom xossaga ega bo'lsa, ya'ni raqamlarini teskari tartibda joylashtirib hosil qilingan yangi Sy son S ga teng bo'lsa TRUE, aks holda FALSE qiymat qaytaradigan Palindrom(S, B) nomli mantiqiy foydalanuvchi funksiyasini tuzing. Bunda funksiya natijasi B parametr orqali qaytarilsin. Asosiy dasturda 5 ta natural son kiriting hamda funksiyadan foydalanib ularning palindromlik xossasini tekshiring.

Funksiya 23. Manfiymas butun F sonining ikkilangan faktorialini qaytaradigan Fact2(F, B) nomli foydalanuvchi funksiyasini tuzing. Bunda funksiya natijasi B parametr orqali qaytarilsin. Asosiy dasturda 5 ta natural son kiriting hamda funksiyadan foydalanib ularning ikkilangan faktorialini hisoblang. F sonining ikkilangan faktoriali quyidagicha ta'riflanadi:

- F!!=1, agar F=0;
- F!!=1·3·…·F, agar F toq bo‘lsa;
- F!!=2·4·…·F, agar F juft bo‘lsa.

Funksiya 24. Natural X va Y sonlarining eng kichik umumiy karralisini qaytaradigan EKUK(X, Y, K) nomli foydalanuvchi funksiyasini tuzing. Bunda funksiya natijasi K parametr orqali qaytarilsin. Asosiy dasturda A, B, C, D natural sonlar kiriting hamda funksiyadan foydalanib (A, B), (A, C), (A, D), (B, C), (B, D) va (C, D) juftliklar uchun EKUK hisoblang.

Funksiya 25. Manfiymas butun T sekund ichida necha to‘liq minut va to‘liq soat borligini qaytaradigan Vaqt1(T, M, H) nomli foydalanuvchi funksiyasini tuzing. Bunda funksiya natijasi M va H parametrlar orqali qaytarilsin. Asosiy dasturda 5 ta manfiymas butun son kiriting hamda funksiyadan foydalanib ularning necha to‘liq minut va soatligini aniqlang.

Funksiya 26. Manfiymas butun T sekunddan keyin necha sutka, soat va minut o‘tganligini qaytaradigan Vaqt2(T, S, H, M) nomli foydalanuvchi funksiyasini tuzing. Bunda funksiya natijasi S, H va M parametrlar orqali qaytarilsin. Asosiy dasturda 5 ta manfiymas butun son kiriting hamda funksiyadan foydalanib ularga mos sekundda necha sutka, soat va minut o‘tganligini aniqlang.

Funksiya 27. Manfiymas butun Y songa mos yil oddiy bo‘lsa FALSE va kabisa yil bo‘lsa TRUE javob qaytaradigan Yil(Y, N) nomli mantiqiy foydalanuvchi funksiyasini tuzing. Bunda funksiya natijasi N parametr orqali qaytarilsin. Asosiy dasturda 5 ta manfiymas butun son kiriting hamda funksiyadan foydalanib ularning oddiy yoki kabisa yil ekanini aniqlang. 100 ga bo‘linib 400 ga bo‘linmaydigan yillardan tashqari barcha 4 ga bo‘linadigan yillar kabisa yili hisoblanadi.

Funksiya 28. Ikki nuqtaning (A; B) va (C; D) koordinatalariga asosan ular orasidagi masofani qaytaradigan MasNuqta(A, B, C, D, M) nomli foydalanuvchi funksiyasini tuzing. Bunda funksiya natijasi M parametr orqali qaytarilsin. Asosiy dasturda funksiyadan foydalanib Oxy tekisligida uchlari (X1; Y1), (X2; Y2) va (X3; Y3) nuqtalarda bo‘lgan uchburchakning perimetri va yuzini aniqlang.

Funksiya 29. MasNuqta funksiyasiga murojaat qilib Oxy tekisligida A(x1; y1), B(x2; y2) va C(x3; y3) uchlari orqali berilgan ABC uchburchakning perimetrini qaytaradigan PerUchb(x1, y1, x2, y2, x3, y3) nomli foydalanuvchi funksiyasini tuzing. Asosiy dasturda funksiyadan foydalanib berilgan A, B, C va D nuqtalarga mos ABC, ABD va ACD uchburchaklarning perimetrini aniqlang.

Funksiya 30. MasNuqta va PerUchb funksiyalariga murojaat qilib Oxy tekisligida A(x1; y1), B(x2; y2) va C(x3; y3) uchlari orqali berilgan ABC uchburchakning yuzasini qaytaradigan YuzaUchb(x1, y1, x2, y2, x3, y3) nomli foydalanuvchi funksiyasini tuzing. Asosiy dasturda funksiyadan foydalanib berilgan A, B, C va D nuqtalarga mos ABC, ABD va ACD uchburchaklarning yuzasini aniqlang.

= C =

Funksiya 31. Oxy tekisligida A(x1; y1), B(x2; y2) va K(x0; y0) koordinatalariga asosan K nuqtadan A va B nuqtalardan o'tuvchi to'g'ri chiziqqacha bo'lgan masofani qaytaradigan MasTCH(x1, y1, x2, y2, x0, y0, D) nomli foydalanuvchi funksiyasini tuzing. Bunda funksiya natijasi D parametr orqali qaytarilsin. Foydalanuvchi funksiyasida masofa quyidagi formula asosida hisoblansin:

$$\text{Masofa(K, AB)} = \frac{2 \cdot S_{KAB}}{\text{Masofa(AB)}}$$

bu yerda S_{KAB} – K, A va B nuqtalar hosil qilgan uchburchakning yuzi, Masofa(AB) – A va B nuqtalar orasidagi masofa.

Funksiya 32. MasN ta elementli butun turdagi Mas massivining eng kichik elementini qaytaradigan MinElem(Mas, MasN) nomli foydalanuvchi funksiyasini tuzing. Asosiy dasturda [2; 11], [19; 21] va [50; 63] kesmalarda, mos ravishda, yotadigan N1, N2 va N3 sonlarini generatsiya qiling. A[1..N1], B[1..N2] va C[1..N3] massiv elementlarini [–50; 50] oraliqda generatsiya qilib ekranga chiqaring. Har bir massiv chiqarilganidan so'ng keyingi satrda uning minimal elementini ekranga chiqaring.

Funksiya 33. MasN ta elementli butun turdagi Mas massivining eng katta elementi indeksini qaytaradigan MakInd(Mas, MasN) nomli foydalanuvchi funksiyasini tuzing. Asosiy dasturda [5; 10], [6; 11] va [91; 99] kesmalarda, mos ravishda, yotadigan N1, N2 va N3 sonlarini generatsiya qiling. A[1..N1], B[1..N2] va C[1..N3] massiv elementlarini [–55; 55] oraliqda generatsiya qilib ekranga chiqaring. Har bir massiv chiqarilganidan so'ng keyingi satrda uning maksimal elementi indeksini ekranga chiqaring.

Funksiya 34. MasN ta elementli Mas massivi elementlarini teskari tartibda joylashtirib qaytaradigan AksMas(Mas, MasN) nomli foydalanuvchi funksiyasini tuzing. Asosiy dasturda [5; 21] kesmada yotadigan N sonini generatsiya qiling. A[1..N], B[1..N] va C[1..N] massiv elementlarini [-69; 69] oraliqda generatsiya qilib ekranga chiqaring. Har bir massiv chiqarilganidan so'ng AksMas funksiyasi qaytargan massivni ekranga chiqaring.

Funksiya 35. MasN ta elementli butun turdagi Mas massivi elementlarini kelish tartibini o'zgartirmagan holda juftlarini Juft massiviga va toqlarini Toq massiviga ajratib qaytaradigan AjratMas(Mas, MasN, Juft, NJ, Toq, NT) nomli foydalanuvchi funksiyasini tuzing (NJ – Juft massivi elementlari soni, NT – Toq massivi elementlari soni). Bunda funksiya natijasi Juft, NJ, Toq va NT parametrlar orqali qaytarilsin. Asosiy dasturda [6; 19] kesmada yotadigan N sonini generatsiya qiling. A[1..N], B[1..N] va C[1..N] massiv elementlarini [–63; 63] oraliqda generatsiya qilib ekranga chiqaring. Har bir massiv chiqarilganidan so'ng AjratMas funksiyasi qaytargan massivlar va elementlari sonini ekranga chiqaring.

Funksiya 36. N ta satrli va M ta ustunli butun turdagi Mat jadvalining K-satri elementlari yig'indisini qaytaradigan MatSatr(Mat, N, M, K, S) nomli foydalanuvchi funksiyasini tuzing. Bunda funksiya natijasi S parametr orqali qaytarilsin. Asosiy dasturda [10; 19] kesmada yotadigan N sonini, [11; 21] kesmada yotadigan M sonini, [1; N] kesmada yotadigan K sonini generatsiya qiling. A[1..N][1..M] jadval elementlarini [–50; 63] oraliqda generatsiya qilib jadval ko'rinishida ekranga chiqaring. Keyingi satrdan K-satr elementlarini, so'ng ularning yig'indisini yangi satrdan izoh bilan ekranga chiqaring.

Funksiya 37. N ta satrli va M ta ustunli butun turdagi Mat jadvalining K-ustuni elementlarini kamayish yo'nalishida tartiblab qaytaradigan MatUst(Mat, N, M, K) nomli foydalanuvchi funksiyasini tuzing. Asosiy dasturda [10; 21] kesmada yotadigan N sonini, [11; 19] kesmada yotadigan M sonini, [1; N] kesmada yotadigan K sonini generatsiya qiling. A[1..N][1..M] jadval elementlarini [–50; 63] oraliqda generatsiya qilib jadval ko'rinishida ekranga chiqaring. So'ng MatUst natijasida hosil bo'lgan yangi jadvalni izoh bilan ekranga chiqaring.

Funksiya 38. N ta satrli va M ta ustunli butun turdagi Mat massivining toq tartib raqamli satri elementlarini kamayish yo'nalishida, juft tartib raqamli satri elementlarini o'sish yo'nalishida tartiblab qaytaradigan MSatr(Mat, N, M) nomli foydalanuvchi funksiyasini tuzing. Asosiy dasturda [6; 10] kesmada yotadigan N sonini, [2; 7] kesmada yotadigan M sonini generatsiya qiling. A[1..N][1..M] massivi elementlarini [91; 97] oraliqda generatsiya qilib jadval ko'rinishida ekranga chiqaring. So'ng MSatr natijasida hosil bo'lgan yangi massivni izoh bilan ekranga chiqaring.

Funksiya 39. NxN butun turdagi Mat massivini asosiy diagonaliga nisbatan transponirlab qaytaradigan MatTransA(Mat, N, TrA) nomli foydalanuvchi funksiyasini tuzing (asosiy diagonalga nisbatan transponirlash – bu elementlar o'rnini asosiy diagonalga nisbatan simmetrik almashtirish). Bunda funksiya natijasi TrA parametr orqali qaytarilsin. Asosiy dasturda [5; 19] kesmada yotadigan N sonini generatsiya qiling. A[1..N][1..N] jadval elementlarini [–6; 10] oraliqda generatsiya qilib jadval ko'rinishida ekranga chiqaring. So'ng MatTransA natijasida hosil bo'lgan yangi jadvalni izoh bilan ekranga chiqaring.

Funksiya 40. NxN butun turdagi Mat massivini qo'shma diagonaliga nisbatan transponirlab qaytaradigan MatTransQ(Mat, N, TrA) nomli foydalanuvchi funksiyasini tuzing (qo'shma diagonalga nisbatan transponirlash – bu elementlar o'rnini qo'shma diagonalga nisbatan simmetrik almashtirish). Bunda funksiya natijasi TrA parametr orqali qaytarilsin. Asosiy dasturda [11; 21] kesmada yotadigan N sonini generatsiya qiling. A[1..N][1..N] jadval elementlarini [–2; 7] oraliqda generatsiya qilib jadval ko'rinishida ekranga chiqaring. So'ng MatTransQ natijasida hosil bo'lgan yangi jadvalni izoh bilan ekranga chiqaring.

= D =

Funksiya 41. Manfiymas butun F sonining ikkilangan faktorialini qaytaradigan FactR2(F) nomli foydalanuvchi rekursiv funksiyasini tuzing. Asosiy dasturda 5 ta natural sonni [10; 19] oraliqda generatsiya qiling va funksiyadan foydalanib ularning ikkilangan faktorialini hisoblang.

Funksiya 42. Natural N va manfiymas butun K sonlari uchun quyidagicha aniqlangan C(N, K) sonni qaytaradigan Comb(N, K) nomli foydalanuvchi rekursiv funksiyasini tuzing:

$$C(N, 0)=C(N, N)=1;$$

$$C(N, K)=C(N-1, K)+C(N-1, K-1), \text{ agar } 0<K<N.$$

N=10 bo'lganda asosiy dasturda [0; 10] oraliqda K sifatida 5 ta butun sonni generatsiya qiling va funksiyadan foydalanib C(N, K) qiymatlarini hisoblang.

Funksiya 43. MasN ta elementli butun qiymatli Mas massivining minimal elementini qaytaradigan MinRF(Mas, MasN) nomli foydalanuvchi rekursiv funksiyasini tuzing. Funksiya tanasida takrorlash operatori qo'llanmasin! Asosiy dasturda [6; 11] oraliqda N sifatida butun sonni, A massivining N ta elementini [5; 21] oraliqda generatsiya qiling hamda funksiyadan foydalanib massivning minimal elementini aniqlang.

Funksiya 44. Natural S sonini palindrom xossaga ega bo'lsa, ya'ni raqamlarini teskari tartibda joylashtirib hosil qilingan yangi Sy son S ga teng bo'lsa TRUE, aks holda FALSE qiymat qaytaradigan PalindromR(S, B) nomli mantiqiy foydalanuvchi rekursiv funksiyasini tuzing. Funksiya tanasida takrorlash operatori qo'llanmasin! Asosiy dasturda 5 ta natural son kiriting hamda funksiyadan foydalanib ularning palindromlik xossasini tekshiring.

Funksiya 45. X sonining (natural) K-darajali (natural) N ta qadamdagi taqribiy ildizini quyidagi formula bo'yicha hisoblab qaytaradigan Ildiz(X, K, N) nomli foydalanuvchi rekursiv funksiyasini tuzing:

$$Y[0]=1;\ Y[N+1]=Y[N]-(Y[N]-X/Y[N]^{K-1}/K).$$

X=81 va K=2 va K=4 bo'lganda [10; 19] oraliqda 5 ta butun N sonini generatsiya qiling va funksiyadan foydalanib ildizlarni hisoblang.

7-BOB. C++ TILINING AJOYIB IMKONIYATLARI

Bu bobda dasturlash tillarida muhim ahamiyatga ega bo‘lgan satrlar, ya’ni C++ tilida **string** turi deb ataluvchi maxsus tuzilma, **andazalar** (ing. **templates**, rus. **шаблоны**) hosil qilish, C++ tili tarkibiga kiritilgan va hozirgi kunda tilning ajralmas qismi sifatida qaralayotgan **Standart Andazalar Kutubxonasi** (ing. **Standard Template Library**, rus. **Библиотека Стандартных Шаблонов**) ko‘rib chiqiladi.

1-§. FUNKSIYALAR ANDAZALARI – FUNCTION TEMPLATES

Avvalgi bobda foydalanuvchi funksiyalarini tuzish va ulardan dasturda foydalanish usullari ko‘rib chiqilgan edi. Foydalanuvchi funksiyalari dasturchiga murakkab dasturni soddaroq bo‘laklarga bo‘lish hamda har bir bo‘lakni bir-biridan ajralgan holda tuzish imkoniyatini beradi. Shu yo‘l bilan dasturchi o‘z ishini birmuncha osonlashtiradi.

Ba’zi bir holatlarda ma’no jihatidan bir xil, ammo funksiyaga beriladigan qiymatlar turlari bir-biridan farqli bo‘lgan bir nechta funksiya tuzishga to‘g‘ri kelishi ham mumkin. Masalan:

Misol. Butun turdagi va haqiqiy turdagi sonlar uchun qo‘llash mumkin bo‘lgan 3 ta sondan kattasini topish funksiyasini tuzing.

Bu masalani hal etish uchun dastlab bir tur (masalan, butun sonlar) uchun funksiya tuzish, keyin esa funksiyaning “nusxasini” boshqa tur (masalan, haqiqiy sonlar) uchun yozish kerak bo‘ladi, ya’ni 2 xil o‘zgaruvchilar turi uchun funksiyalar tuzish kerak bo‘ladi. Bu vazifani quyidagicha hal etish mumkin:

Dastur A	Natijasi

`#include <iostream>`
`using namespace std;`
 	
`int Imax3(int a, int b, int c) {`
`  int max = (a > b) ? a : b;`
`  max = (max > c) ? max : c;`
`  return max;}`
`double Dmax3(double a, double b, double c) {`
`  double max = (a > b) ? a : b;`
`  max = (max > c) ? max : c;`
`  return max;}`
	`6`
	`6.3`

```
int main(){
    cout << Imax3(1, 6, 3) << endl;
    cout << Dmax3(2.4, 5.1, 6.3) << endl;
    return 0;
}
```

Yuqoridagi Dastur A matnidan **Imax3** va **Dmax3** funksiyalarining tanasi bir xil ko'rsatmalardan iborat ekanligini ko'rish mumkin, ya'ni faqat funksiyalarda qatnashayotgan o'zgaruvchilar turigina bir-biridan farq qiladi, xolos (dasturda o'zgaruvchilar turi tagchiziq bilan ajratib ko'rsatilgan).

Bu kabi holatlarda dasturchilar ishini yengillashtirish maqsadida C++ dasturlash tilida andazalar qo'llash usuli kiritilgan.

Yuqoridagi masala shartidan ham murakkabroq holatga javob bera oladigan, ya'ni o'zgaruvchilar turi 4 xil (butun, haqiqiy, mantiqiy va belgili) bo'lgan, quyidagi dasturda funksiya uchun **andaza** imkoniyatidan foydalanish ko'rsatilgan:

Dastur B	Natijasi

`#include <iostream>`
`using namespace std;`
 	
`template<typename T>`
`T max3(T a, T b, T c){`
`  T max = (a > b) ? a : b;`
`  max = (max > c) ? max : c;`
`  return max;}`
`int main(){`
`  cout << max3(1, 6, 3) << endl;`
`  cout << max3(2.5, 6.2, 7.3) << endl;`
`  cout << boolalpha << max3(true, false, true)<<endl;`
`  cout << max3('a', 'b', 'c') << endl;`
`  return 0;`
`}`
	`6`
	`7.3`
	`true`
	`c`

Ko'rib turganingizdek, **Dastur B** da **Dastur A** da tagchiziq bilan ajratib ko'rsatilgan tur nomlari bitta **T andaza turi** kabi ifodalangan.

Ushbu dastur **andaza**lar yordamida nafaqat **int**, **double**, balki **char**, **bool** va boshqa turdagi (keyinroq ko'rib chiqilishi rejalashtirilgan) qiymatlar qabul qiluvchi, balki bir xil vazifani bajaruvchi funksiyalarni ham birlashtirish imkoniyati borligini ko'rsatib bermoqda. **Dastur B** kompilyatsiya qilinishi vaqtida kompilyator dasturda ishlatilgan turlarga (**max3** funksiyasiga berilgan qiymat turlariga) mos **max3** funksiyasi "nusxalarini" hosil qiladi. Shuning uchun yuqoridagi dastur muammosiz bir necha turdagi qiymatlar uchun ham xizmat qila oladi.

Endi **andaza** funksiyalar hosil qilishni **max3** funksiyasi misolida izohlaymiz:

```
template<typename T>
T max3(T a, T b, T c) {…}
```

Bu yerda: **template** – andaza hosil qiluvchi kalit so'z; **typename** – tur nomi (ing., o'zb.: tur nomi); **T** – funksiya e'lonida "tur" ning o'rniga ko'rsatiladigan belgi bo'lib, **T** belgisining o'rnida ixtiyoriy identifikator ishlatilishi mumkin. Bu belgini funksiya tanasidagi o'zgaruvchilar va o'zgarmaslar, funksiyaga beriladigan parametrlar va funksiya qaytaradigan qiymat turini ifodalashda "aniq" turlar o'rniga ishlatish mumkin.

Shuni ta'kidlab o'tish joizki, bunday usulda tuzilgan funksiyalarga beriladigan qiymatlar bir xil turda bo'lishi shart. Ya'ni oddiy foydalanuvchi funksiyalarida funksiyaga berilayotgan qiymat funksiya parametri turiga mos qiymatga "keltirilishi" mumkin bo'lsa, andaza funksiyalarda buning imkoni yo'q, chunki kompilyator ko'rsatilgan turlardan qaysi birini tanlashni bilmay qoladi. Misol uchun quyidagi dasturlarni qaraylik:

Dastur C	Dastur D

`#include <iostream>`
`using namespace std;`
 	
`template<typename T> T sum(T a, T b){`
`  return a + b;}`
`int main(){`
`  cout << sum(2, 3.1);`
`  return 0;`
`}`
	`#include <iostream>`
	`using namespace std;`
	
	`double sum(double a, double b){`
	`  return a + b;}`
	`int main(){`
	`  cout << sum(2, 3.1);`
	`  return 0;`
	`}`

Kompilyatsiya jarayonida **Dastur C** xatolik xabarini beradi, bunga funksiyaga **int** va **double** turidagi qiymatlar berilayotgani sabab bo'ladi. **Dastur D** esa hech qanday muammosiz ishlaydi.

Bu kabi muammoni hal qilishning turli yo'llari mavjud. Masalan, funksiyaga faqatgina bir xil turdagi qiymatlar berilishini dasturchi nazorat qilishi mumkin yoki funksiyaga berilayotgan qiymatlarni dasturchining o'zi bir xil turga keltirishi mumkin. Bundan tashqari, andaza funksiyalarga murojaat qilinayotganda turni oshkora ko'rsatish ham mumkin. Bu ikki usulni quyidagi misollarda ko'rishimiz mumkin:

Dastur 1	Dastur 2

`#include <iostream>`
`using namespace std;`
 	
`template<typename T> T sum(T a, T b){`
`  return a + b;}`
`int main(){`
	`#include <iostream>`
	`using namespace std;`
	
	`template<typename T> T sum(T a, T b){`
	`  return a + b;}`
	`int main(){`

```
    int a = 2;
    double b = 3.1;
cout<<sum(static_cast<double>(a),b);
    return 0;
}
```

```
    int a{2};
    double b{3.1};
    cout << sum<double>(a, b);
    return 0;
}
```

Dastur 2 dagi **sum<double>(a,b)** ko'rsatmasi, kompilyator tomonidan **sum** andaza funksiyasining aynan **double** turiga mos ko'rinishini hosil qilishini ta'minlaydi. Bu ko'rinish **Dastur D** dagi `sum` funksiyasi ko'rinishi bilan aynan mos tushadi va demak, bu dastur ham **Dastur D** kabi muammosiz ishlaydi.

Andaza funksiyalarni **call-by-reference** uslubi bilan ham birlashtirish mumkin. Bunda, tushunib turganingizdek, hech qanday o'zgartirish kiritishga hojat yo'q. Masalan, ikkita o'zgaruvchi qiymatini almashtirib chiqaradigan funksiya andazasini quyidagicha tuzamiz:

```
template <typename T> void myswap(T& a, T& b){
            T c = b; b = a; a = c;          }
```

2-§. STANDART ANDAZALAR KUTUBXONASI

Standard Template Library (qisqartmasi: STL) – standart andazalar kutubxonasi bo'lib, u **konteynerlar**, **iteratorlar** va **algoritmlarni** o'z ichiga oladi. Bu kutubxona dasturchilar orasida **STL** qisqartmasi bilan mashhur bo'lgani bois biz ham ushbu qo'llanmada **STL** qisqartmasidan foydalanamiz. STL ayni bir qat'iy ajralib turgan, alohida kutubxona emas, balki kutubxonalar jamlanmasidir. Biz mazkur kutubxonalar taqdim etadigan ba'zi imkoniyatlarni ko'rib chiqamiz.

Konteynerlar – bir xil turdagi qiymatlarni o'zida saqlaydigan maxsus tuzilma (tur deyish ham mumkin). Konteyner o'zida saqlanayotgan qiymatlar ustida o'ziga xos amallar bajarish, tayyor algoritmlardan foydalanish imkoniyatini beradi. Biz qo'llanma doirasida **string**, **vector**, **list**, **queue**, **stack**, **set**, **multiset**, **map** va **multimap** konteynerlarini ko'rib chiqamiz.

Har bir konteyner o'zida **a'zo funksiya**larini (ing. member function) saqlaydi. Konteynerning a'zo funksiyalariga murojaat qilish uchun shu konteynerni ifodalaydigan identifikator, a'zolikni ifodalovchi "." va a'zo funksiya yoziladi. Misol sifatida **a.size();** yozish mumkin, bu yerda **a** – identifikator, "." belgisi va **size**() – a'zo funksiya.

Iteratorlar – konteyner elementlariga murojaat qilish uchun foydalaniladigan tuzilmalar. Avvalgi boblarda ko'rib chiqilgan standart massivlar va ba'zi konteynerlar elementlari xotiraning ketma-ket joylashgan "xotira uyachalari" blokida saqlanadi. Lekin ushbu bobda ko'rib chiqiladigan murakkab tuzilmali ba'zi konteynerlar haqida bunday fikr bildirib bo'lmaydi. Bu konteynerlarda "mantiqan" ketma-ket kelgan elementlar xotiraning turli qismlarida joylashgan bo'lishi ham mumkin. Iteratorlar esa ushbu holatlarda ham bizga konteyner

elementlarini ketma-ket ko'rib chiqish imkoniyatini beradi. Iteratorlar ma'lum bir ma'noda ko'rsatkichlarga o'xshaydi. Ko'rsatkichlar istalgan turdagi o'zgaruvchi yoki o'zgarmas, massiv elementlarini ko'rsatishi mumkin bo'lsa, iteratorlar, asosan, konteyner elementlari uchun qo'llaniladi.

Iteratorlar standart **<iterator>** kutubxonasida e'lon qilingan bo'lib, ular ustida quyidagi amallarni bajarish mumkin:

- **Iteratorlarni taqqoslash:** == va != operatorlari yordamida;
- **Keyingi iteratorga o'tish:** ++ operatori;
- **Iterator ko'rsatgan elementga murojaat:** * operatori yordamida.

STL tarkibiga kiruvchi har bir konteyner o'zida **begin()** va **end()** a'zo funksiyalarini saqlaydi. Bu funksiyalar ushbu konteyner saqlayotgan elementlarning boshini va oxirini ko'rsatuvchi iteratorlarni qaytaradi. Bunda **begin()** funksiyasi konteynerning birinchi elementini ko'rsatuvchi iterator qaytarsa, **end()** funksiyasi konteyner tugaganligini bildiruvchi iterator qaytaradi.

Ba'zi konteynerlar uchun iteratorlar imkoniyati cheklab qo'yilgan. Lekin barcha konteynerlar iteratorida **inkrement**, **dekrement** operatorlari yoki quyidagi funksiyadan foydalanish mumkin:

```
advance(nom, qadam);
```

Bu yerda **nom** – iterator nomi, **qadam** – iterator surilishi kerak bo'lgan qadam (butun son).

3-§. C++ DASTURLASH TILIDA SATRLAR

Dastlabki 6 ta bobda biz satr literallaridan (dastur matnidagi "" orasida yozilgan belgilar ketma-ketligidan) keng foydalanib keldik. Biroq satr turidagi o'zgaruvchilar va o'zgarmaslar haqida deyarli so'z yuritganimiz yo'q. Bunga sabab, C++ dasturlash tili satrlarni ifodalash uchun "sodda" tur joriy etmagan. C++ dasturlash tilida satrlarni **char** turidagi massivlar orqali hosil qilishimiz mumkin. Ammo bu usul satrlar bilan ishlashda ba'zi qiyinchiliklarga sabab bo'ladi. Bu muammoni hal etish maqsadida **string** (ing., o'zb.: satr) konteyneri qo'shilgan.

Ma'lumki, **string** konteyneri **<string>** kutubxonasi tarkibiga kiradi. Matnlar bilan ishlash qulay bo'lishi uchun **<string>** kutubxonasi **<iostream>** kutubxonasi tarkibiga qo'shilgan bo'ladi, shuning uchun ba'zi hollarda **<string>** kutubxonasini dasturda yozmaslik mumkin. Ba'zi dasturchilar bu konteyner STL tarkibiga kiradi desa, boshqalari esa (balki **<string>** kutubxonasi **<iostream>** kutubxonasi tarkibiga qo'shilgan bo'lganligi uchundir) buni inkor etadi. Asosiy maqsadimiz satrlar bilan ishlashda qulaylik beradigan **string** konteyneridan foydalanishni o'rganish bo'lgani uchun bu fikrlar qo'llanma doirasida ahamiyatga ega emas.

SATRNING E'LONI, QIYMAT BERISH VA CHIQARISH

Dasturda **string** turidagi o'zgaruvchi va o'zgarmaslar quyidagicha e'lon qilinadi:

```
string a;
const string b;
```

Satrga boshlang'ich qiymat berish oddiy turlardagi kabidir:

```
string a{"Toshkent"};
string b = "Samarqand";
string c("Buxoro");
```

Boshlang'ich qiymat berilmagan **string** turidagi o'zgaruvchilar "**bo'sh**" satr qiymatini oladi.

Kiritish va chiqarishda **string** turidagi satrlar uchun **cin** va **cout** operatorlari qo'llanadi:

```
string s;
cin >> s;
cout << s;
```

Kiritilayotgan matnda qatnashgan orachiq (rus.: пробел) belgisi matnni "bo'lib" yuborishga sabab bo'lishi mumkin. Masalan, quyidagi dastur natijasini qaraylik:

Dastur	Izoh

`#include <iostream>`
`#include <string>`
`using namespace std;`
`int main(){`
`  string s;`
`  cin >> s;`
`  cout << s << endl;`
`  return 0;`
`}`
	Kiruvchi ma'lumotlar:
	`Satr elementi`
	
	Natijasi:
	`Satr`

Bu dastur ishlaganida matnda 2 ta so'z, ya'ni "Satr elementi" kiritilganiga qaramay **s** satrga faqatgina "Satr" so'zi o'zlashtirildi. Bunga, aytib o'tganimizdek, **cin** operatori orachiq belgisi uchrashi bilanoq o'qishni to'xtatishi sababdir. Agar biz kiritilayotgan matndagi barcha belgilarni (orachiq belgilarini ham qo'shib) o'qishimiz kerak bo'lsa, u holda standart kutubxonada e'lon qilingan **getline** funksiyasidan foydalanishimiz zarur bo'ladi. Yuqoridagi dasturning tuzatilgan ko'rinishi quyidagicha:

Dastur	Izoh

`#include <iostream>`
`#include <string>`
`using namespace std;`
`int main(){`
`  string s;`
	Kiruvchi ma'lumotlar:
	`Satr elementi`
	
	Natijasi:
	`Satr elementi`

```
    getline(cin, s);
    cout << s << endl;
    return 0;
}
```

SATR KONSTRUKTORLARI

Yuqoridagilardan tashqari, **string** elementini e'lon qilishda quyidagi konstruktorlardan (**string** elementini "quruvchi" funksiyalardan) foydalanish mumkin.

1. Biror belgini **n** marta takrorlash yordamida:

```
string a(n, c) // n - takrorlash soni, c - belgi
```

Masalan:

```
string a(10, 'a'); //a o'zgaruvchisi qiymati: "aaaaaaaaaa"
```

2. Ma'lum **string** turidagi o'zgaruvchidan yangi **string** elementini e'lon qilishda foydalanish mumkin. Masalan:

```
string s{"Toshkent shahri"};//qiymati ma'lum o'zgaruvchi
string d(s);
cout << d; // "Toshkent shahri" qiymatini chiqaradi
```

3. Boshqa satrning **k**-o'rnidan boshlab oxirigacha bo'lgan belgilaridan yangi satrni hosil qilish mumkin. Agar k soni satr uzunligidan katta bo'lsa, yangi satr "bo'sh" qiymatini oladi.

```
string a{"Mening yurtim"};
string b(a, 7);
cout << b; // "yurtim" qiymatini chiqaradi
```

Bu misoldan shuni anglash mumkinki, C++ ning massivlari kabi, string turi ham 0-indeksli, ya'ni sanoq 0 dan boshlanadi. "Mening yurtim" satrining 7-indeksdagi belgisi – bu 'y'. Shu bois yuqoridagi konstruktor 'y' belgisidan boshlab satr oxirigacha bo'lgan belgilarni yangi satrga "ko'chiradi".

4. Boshqa satrning **k**-indeksdagi belgisidan boshlab keyingi **n** ta ketma-ket belgilardan yangi satrni hosil qilish mumkin. Ya'ni bunda satrning **[k, k + n - 1]** oraliqdagi belgilaridan yangi satr tuziladi. Bu yerda n soniga manfiy qiymat berish xatolikka olib keladi. Agar **(k + n - 1)** soni satr uzunligidan katta bo'lsa, satrning k-o'rnidan boshlab oxirigacha bo'lgan belgilardan yangi satr tuziladi. Masalan:

```
string a{"Toshkent shahri"};
string c(a, 0, 8);
cout << c; // "Toshkent" qiymatini chiqaradi
```

5. Belgilar ro'yxatidan satr hosil qilish mumkin:

```
string e({'1', '2', '3'}); // e = "123"
```

6. Belgili, ya'ni **char** turdagi massivlardan satr hosil qilish mumkin:

```
char a[] = "Salom";
string s(a); // s = "Salom";
```

SATR ELEMENTIGA MUROJAAT VA QIYMATINI O'ZGARTIRISH

Satrning elementiga murojaat etish va qiymatini o'zgartirish juda ham oson. Buning asosiy 3 ta usulini izohlaymiz:

1-usul. [ ] operatori yordamida:

```
string a{"davlat"};
cout << a[0] << endl; //ekranga 'd' belgisi chiqadi
a[0] = 's';
cout << a; // ekranga "savlat" so'zi chiqadi
```

2-usul. at() (o'zb.: da) a'zo funksiyasidan foydalanib:

```
string a{"satr"};
cout << a.at(0) << a.at(2) << endl; //"st" satri ekranga chiqadi
a.at(0) = 'd'; //0-o'rindagi belgini 'd' ga o'zgartiradi
a.at(2) = 'v'; //2-o'rindagi belgini 'v' ga o'zgartiradi
cout << a; //ekranga "davr" so'zi chiqadi
```

3-usul. Biroz murakkabroq bo'lsa ham iterator tuzilmasidan foydalanish ham mumkin:

```
string a="satr";
string::iterator it = a.begin();//'s' belgisiga mos iterator
cout <<*(it+2) <<endl;// ekranga 't' belgisi chiqadi
*(it+2) = 'b';   // iteratorga mos 't' ni 'b' ga o'zgartiradi
cout << a << endl; //ekranga "sabr" so'zi chiqadi
```

SATR O'LCHAMLARI VA SIG'IMI

Satr o'lchamlari haqida ma'lumot olish uchun bir qator a'zo funksiyalar tuzilgan.

A. Satr bo'sh yoki bo'sh emasligini tekshirish uchun **empty()** (o'zb.: bo'sh) a'zo funksiyasi mavjud. U **bool** turidagi qiymat qaytaradi. Qo'llanishiga misol:

```
string s;
string d{"satr"};
cout << boolalpha << s.empty() << endl; // ekranga true chiqadi
cout << d.empty() << endl;              // ekranga false chiqadi
```

B. Satr uzunligini aniqlash uchun **length()** (o‘zb.: uzunligi) yoki **size()** (o‘zb.: o‘lchami) a’zo funksiyalaridan foydalanish mumkin. Bu ikki funksiya **size_t** (nomanfiy butun) turdagi qiymatni qaytaradi va bir-biridan umuman farq qilmaydi.

```
string s{"Toshkent"};
cout << s.size() << endl;// 8 qiymatini chiqaradi
cout << s.length();      // 8 qiymatini chiqaradi
```

C. Satrning maksimal sig‘imini aniqlash uchun **max_size()** a’zo funksiyalaridan foydalanish mumkin.

```
string s{"Toshkent"};
cout << s.max_size() << endl;// 2147483647 qiymatini chiqaradi
```

SATRGA OID FOYDALI A’ZO FUNKSIYALAR

Satr o‘zgaruvchisini “bo‘shatish” uchun **clear()** (o‘zb.: tozalash) a’zo funksiyasidan foydalanish yoki unga “” (bo‘sh) qiymatini berish mumkin.

```
string s{"London"}, d{"Moskva"};
cout << s.size() << endl;  // qiymati 6
cout << d.size();          // qiymati 6
s.clear();
d= "";
cout << s.size() << endl;  // qiymati 0
cout << d.size();          // qiymati 0
```

Satr oxiriga yangi belgi qo‘shish yoki satr oxiridagi belgini “o‘chirish” uchun mos ravishda **push_back()** va **pop_back()** a’zo funksiyalaridan foydalanish mumkin.

```
string s{"12346"};
s.push_back('7'); // s = "123467"
s.pop_back();       // s = "12346"
```

Bundan tashqari, satrlarni “qo‘shish” (ulash) ham mumkin. Buning natijasida, odatda, yangi satr hosil bo‘ladi.

```
string a{"31"}, b{"92"};
string s;
s = a + b;  // s = "3192"
a = a + b;  // s = "3192"
a += b;     // s = "319292"
```

Satr oxiriga belgilarni **append()** (o‘zb.: oxiriga qo‘shish) a’zo funksiyasi yordamida ham qo‘shish mumkin. Mazkur funksiyaning turli xil ko‘rinishlari mavjud:

1. **append(n, b)** - **n** ta **b** belgisini qo‘shadi;

2. **append(s)** - **s** satrni qo‘shadi;
3. **append(s, k, n)** - **s** satrning **k**-o‘rnidan boshlab **n** ta belgisini qo‘shadi;
4. **append({a, b, c})** - **a**, **b**, **c** belgilarini ketma-ket, ko‘rsatilgan tartibda qo‘shadi.

Misol sifatida quyidagi dasturni keltirish mumkin.

Dastur	Natijasi

`#include <iostream>`
`#include <string>`
`using namespace std;`
`int main(){`
`  string s;`
`  string b{"1"};`
`  string a{"aa3333de"};`
`  s.append(b);`
`  cout << s << endl;`
`  s.append(2, '2');`
`  cout << s << endl;`
`  s.append(a, 2, 3);`
`  cout << s << endl;`
`  s.append({'4', '4', '4', '4'});`
`  cout << s << endl;`
`  return 0;`
`}`
	`1`
	`122`
	`122333`
	`1223334444`

Ikkita satrning qiymatlarini o‘zaro almashtirish uchun **swap()** (o‘zb.: ayirboshlash) a’zo funksiyasidan foydalanish mumkin. Masalan,

```
string a{"kitob"};
string b{"daftar"};
a.swap(b);
cout << a << endl;   // "daftar"
cout << b << endl;   // "kitob"
```

Swap funksiyasi satrlarning qiymatlarini almashtirishni juda tez amalga oshiradi. Shuning uchun ham quyidagi kabi dastur o‘rniga **swap()** funksiyasini ishlatish tavsiya etiladi.

```
string a{"kitob"};
string b{"daftar"};
string d = a;  // a satrining nusxasi d ga o‘zlashtiriladi
a = b;         // bunda satrning belgilari bittalab
a = d;         // nusxalanayotgani uchun KO‘P VAQT sarflanadi!
```

SATRLARNI TAQQOSLASH

Satrlarni taqqoslash satrlarning belgisini taqqoslash orqali amalga oshiriladi. O‘z navbatida, belgilarni taqqoslash uchun ularning **ASCII** (yoki **Unicode**) kodi taqqoslanadi. Ma’lumki, har bir belgi kompyuter uchun kodlangan bo‘lib, bu kodlar jadvali ASCII (American Standart Code for Information Interchange – Amerika axborot almashinuvining standart kodi) deb yuritiladi. Quyida lotin harflarining ASCII jadvalidagi kodi o‘nlik sanoq sistemasiga mos holda keltirilgan:

A	B	C	D	E	F	G	H	I	J	K	L	M
65	66	67	68	69	70	71	72	73	74	75	76	77
N	O	P	Q	R	S	T	U	V	W	X	Y	Z
78	79	80	81	82	83	84	85	86	87	88	89	90
a	b	c	d	e	f	g	h	i	j	k	l	m
97	98	99	100	101	102	103	104	105	106	107	108	109
n	o	p	q	r	s	t	u	v	w	x	y	z
110	111	112	113	114	115	116	117	118	119	120	121	122

Umuman, ASCII jadvalida boshqa belgilarning ham kodi keltirilgan bo‘lib, masalan, 0 dan 9 gacha bo‘lgan raqamlar kodi, mos ravishda, 48 dan 57 gachadir.

Belgilarni taqqoslashga oid quyidagi dasturni keltirish mumkin.

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  cout<<boolalpha;`
`  cout<<"A<a: "<<('A'<'a')<<endl;`
`  cout<<"A=a: "<<('A'=='a')<<endl;`
`  cout<<"A>a: "<<('A'>'a')<<endl;`
`  return 0;`
`}`
	`A<a: true`
	`A=a: false`
	`A>a: false`

Satrlarni taqqoslash ASCII (yoki Unicode) alifbosidagi leksikografik tartiblangan “so‘zlar” to‘plamida qaraladi. Biror A va B satrlarni leksikografik tartiblashda A satr B satrdan kichik (A satr B satrdan oldin) bo‘ladi, agar:

- yoki A satrning birinchi m ta belgisi B satrning birinchi m ta belgisi bilan bir xil, lekin A satrning (m+1)-belgisi B satrning (m+1)-belgisidan kichik (ASCII (yoki Unicode) kodi bo‘yicha) bo‘lsa;
- yoki A satr B satrning bosh qismi (prefiksi) bo‘lsa.

Demak, bu holda B satr A satrdan katta bo‘ladi.

Satrlarni taqqoslashda ham oddiy turlardagi kabi amallar qo‘llanadi:

Taqqoslash operatori	Izoh
==	**Teng.** Agar satrlar bir xil bo‘lsa, **true** qiymatini, aks holda **false** qiymatini qaytaradi.
!=	**Teng emas**. Agar satrlar teng bo‘lmasa **true**, aks holda **false** qiymatini qaytaradi.
<	**Kichik.** Agar birinchi satr ikkinchi satrdan kichik bo‘lsa **true**, aks holda **false** qiymatini qaytaradi.
>	**Katta.** Agar birinchi satr ikkinchi satrdan katta bo‘lsa **true**, aks holda **false** qiymatini qaytaradi.
<=	**Kichik yoki teng.** Satrlar teng yoki birinchi satr ikkinchi satrdan kichik bo‘lsa **true**, aks holda **false** qiymatini qaytaradi.
>=	**Katta yoki teng.** Satrlar teng yoki birinchi satr ikkinchi satrdan katta bo‘lsa **true**, aks holda **false** qiymatini qaytaradi.

Misol sifatida quyidagi dasturni keltirish mumkin.

Dastur

```
#include <iostream>
#include <string>
using namespace std;
int main(){
    string s1,s2;
    s1="21";s2=s1;
    cout<<boolalpha;
    cout<<"s1="<<s1<<" s2="<<s2<<endl;
    cout<<"s1==s2: "<<(s1==s2)<<endl;
    cout<<"s1!=s2: "<<(s1!=s2)<<endl;
    cout<<"s1<s2: "<<(s1<s2)<<endl;
    cout<<"s1>s2: "<<(s1>s2)<<endl;
    cout<<"s1<=s2: "<<(s1<=s2)<<endl;
    cout<<"s1>=s2: "<<(s1>=s2)<<endl;
    s2='a'+s1;
    cout<<"s1="<<s1<<" s2="<<s2<<endl;
    cout<<"s1==s2: "<<(s1==s2)<<endl;
    cout<<"s1!=s2: "<<(s1!=s2)<<endl;
    cout<<"s1<s2: "<<(s1<s2)<<endl;
    cout<<"s2<s1: "<<(s2<s1)<<endl;
    cout<<"s1<=s2: "<<(s1<=s2)<<endl;
    cout<<"s1>=s2: "<<(s1>=s2)<<endl;
    cout<<"s1>s2: "<<(s1>s2)<<endl;
    cout<<"s2>s1: "<<(s2>s1)<<endl;
    return 0;
}
```

Natijasi

```
s1=21 s221
s1==s2: true
s1!=s2: false
s1<s2: false
s1>s2: false
s1<=s2: true
s1>=s2: true
s1=21 s2=a21
s1==s2: false
s1!=s2: true
s1<s2: true
s1>s2: false
s1<=s2: true
s1>=s2: false
s1>s2: false
s2>s1: true
```

Murakkab taqqoslash amallarini **compare()** (o'zb.: taqqoslash) a'zo funksiyasi yordamida bajarish mumkin. Funksiyaning umumiy ko'rinishi quyidagicha:

```
compare(int bosh1, int soni1, string satr2, int bosh2, int soni2)
```

Funksiya o'ziga murojaat etgan satrning **bosh1**-o'rnidan boshlab **soni1** ta belgisidan iborat qism satri bilan **satr2** satrining **bosh2**-o'rnidan boshlab **soni2** ta belgisidan iborat qism satrini taqqoslaydi. Agar qism satrlar teng bo'lsa **0**, birinchi qism satr katta bo'lsa **1**, birinchi qism satr kichik bo'lsa **-1** qiymatlarini qaytaradi. Masalan:

```
string s{"Toshkent shahri"};
string d{"Samarqand shahri"};
cout << s.compare(9, 6, d, 10, 6);
```

Bu namuna ishlashi natijasida ekranga 0 qiymati chiqadi. Bunga sabab, birinchi satrning 9-o'rnidan boshlab 6 belgisidan iborat qism satri "shahri" bo'ladi, ikkinchi satrning 10-o'rnidan boshlab 6 ta belgisidan iborat qism satri ham "shahri" bo'ladi. Bu ikki satr teng bo'lganligi sababli **compare** funksiyasi 0 qiymatini qaytardi.

Ba'zi funksiya parametrlarini "tushirib" qoldirish ham mumkin. Masalan, agar birinchi satr va ikkinchi satrlarni to'liqligicha taqqoslash kerak bo'lsa, quyidagicha kabi yozish mumkin:

```
s1.compare(s2);
```

Bu holda shunday xulosa qilish mumkin: **compare()** a'zo funksiyasida **satr2** dan tashqari barcha parametrlarni tushirib qoldirish mumkin.

SATR QISMI USTIDA AMALLAR

Ko'p hollarda satrning biror-bir qismini ajratib olish (biror-bir uzluksiz qismi nusxasini olish) kerak bo'lishi mumkin. Buning uchun quyidagi **substr()** (ing. substring, o'zb.: satr qismi) a'zo funksiyasidan foydalanish qulay:

```
substr(k, n)
```

Bu a'zo funksiya **k**-o'rindagi belgidan boshlab ketma-ket kelgan **n** ta belgini satr ko'rinishida qaytaradi. Agar **n** ko'rsatilmagan bo'lsa yoki **(k + n – 1)** soni satr uzunligidan katta bo'lsa, **k**-o'rindagi belgidan boshlab satr oxirigacha bo'lgan elementlarini, agar hech narsa ko'rsatilmagan bo'lsa, to'liq satrni qaytaradi. Quyidagi dastur hamda natijasi shu hollarni ifodalaydi.

Dastur	Natijasi

`#include <iostream>`
`#include <string>`
`using namespace std;`
	`Sirdaryo`
	`Sir`
	`daryo`

```
int main(){
    string s{"Sirdaryo"};
    string d = s.substr();
    cout << d << endl;
    d = s.substr(0, 3);
    cout << d << endl;
    d = s.substr(3);
    cout << d << endl;
    return 0;
}
```

Avvalroq satr oxiriga boshqa satrni qo‘shish (ulash) ko‘rilgan edi. “Satr orasiga boshqa satrni joylashtirish mumkinmi?” degan savol yuzaga kelishi tabiiy. Bu savolning javobi albatta “ha”. Bunday imkoniyat uchun **insert()** (o‘zb.: joylashtirish) a’zo funksiyasi bir necha ko‘rinishlarda qo‘llanadi:

1. **`insert(size_t k, size_t n, char b)`** – satrning **k**-o‘rnidan boshlab oraga **b** belgisining **n** ta nusxasini joylashtiradi;
2. **`insert(size_t k, string s2)`** – satrning **k**-o‘rnidan boshlab oraga **s2** satrini joylashtiradi;
3. **`insert(size_t k, string s2, size_t p, size_t n)`** – satrning **k**-o‘rnidan boshlab **s2** satrining **p**-belgisidan boshlab **n** ta belgisini joylashtiradi.

Quyida **insert**() funksiyasi ishini izohlashga doir dastur namunasi keltirilgan:

Dastur	Natijasi

`#include <iostream>`
`#include <string>`
`using namespace std;`
`int main(){`
`  string s{"abc"};`
`  string c{"abccdef"};`
`  s.insert(0, 2, 'a');`
`  cout << s << endl;`
`  s.insert(4, "bb");`
`  cout << s << endl;`
`  s.insert(6, c, 2, 2);`
`  cout << s << endl;`
`  return 0;`
`}`
	`aaabc`
	`aaabbbc`
	`aaabbbccc`

Ba’zan satrning biror qismini o‘chirish zarur bo‘ladi. Buning uchun **erase()** (o‘zb.: o‘chirish) a’zo funksiyasi xizmat qiladi. Uning umumiy ko‘rinishi quyidagicha:

`erase(size_t k, size_t n)`

Bu funksiya **k**-o'rindagi belgidan boshlab keyingi **n** ta belgini o'chiradi. Agar **n** ko'rsatilmagan yoki **(k + n - 1)** soni satr uzunligidan katta bo'lsa, u holda satrning **k**-belgisidan boshlab oxirigacha bo'lgan belgilarini o'chiradi. Agar hech nima ko'rsatilmagan bo'lsa, u holda satrning barcha belgilarini o'chiradi.

```
string s{"Toshkent shahri"};
string d(s);
s.erase(8);      // s = "Toshkent"
d.erase(0, 9);   // d = "shahri"
```

Satr qismini boshqa satr qismiga almashtirish **replace()** (o'zb.: almashtirish) a'zo funksiyasi yordamida amalga oshiriladi. Bu funksiyaning ham bir nechta ko'rinishlari mavjud.

1. **replace(size_t k, size_t n, string s2)** – satrning **k**-o'rindagi belgisidan boshlab **n** ta belgisini **s2** satriga almashtiradi.
2. **replace(size_t k, size_t n, string s2, size_t k2, size_t n2)** – satrning **k**-o'rindagi belgisini **s2** satrining **k2**-o'rindagi belgisidan boshlab **n2** ta belgisiga almashtiradi. Agar **(n + k – 1)** soni satr uzunligidan katta bo'lsa, satr oxirigacha bo'lgan belgilarini almashtiradi. Agar **(k2 + n2 – 1)** soni **s2** satri uzunligidan katta bo'lsa, **s2** satrining oxirigacha bo'lgan belgilarini oladi.
3. **replace(size_t k, size_t n, size_t n2, char b)** – satrning **k**-o'rnidan boshlab **n** ta belgisini **b** belgisining **n2** ta nusxasiga almashtiradi.

Satr qismini almashtirishga doir misollar quyida keltirilgan.

Dastur	Natijasi

`#include <iostream>`
`#include <string>`
`using namespace std;`
`int main(){`
`  string s{"abcde"};`
`  string d{"ert"};`
`  s.replace(0, 1, d);`
`  cout << s << endl;`
`  s.replace(2, 3, d, 1, 2);`
`  cout << s << endl;`
`  s.replace(1, 2, 2, 'a');`
`  cout << s << endl;`
`  return 0;`
`}`
	`ertbcde`
	`errtde`
	`eaatde`

SATRDA QIDIRISH

Satrlar ustida ko‘p bajariladigan amallardan yana biri – bu satrda qism satr sifatida boshqa bir satrning bor yoki yo‘qligini aniqlashdir. Ya’ni masalan, a = “kutubxona” va b = “xona” bo‘lsa, “b satr a satrida bor” deyiladi, chunki a satrida “xona" qism satri mavjud. Bu ishda **find()** (o‘zb.: topish) a’zo funksiyasi yordam beradi. Bu funksiyaning umumiy ko‘rinishi quyidagicha:

```
find(string s2, size_t k)
```

Funksiya qaralayotgan satrning **k**-o‘rnidan boshlab keyingi qismlarida **s2** satrini qidiradi va agar **s2** satr bor bo‘lsa, u holda qidirilayotgan **s2** satrining qaralayotgan satrdagi boshlang‘ich o‘rnini, aks holda **string::npos** konstantasining qiymatini qaytaradi, bu qiymat esa **size_t** ning maksimal qiymatiga teng. Umuman olganda, **satrdan belgini ham qidirish mumkin.**

Dastur	Natijasi
`#include <iostream>` `#include <string>` `using namespace std;` `int main(){` `  string s{"Toshkent, Samarqand, Buxoro"};` `  string d{"Toshkent"};` `  if(s.find(d) != string::npos) cout << "Bor";` `  else cout << "Yo`q";` ` cout << endl<< d.find(s) << endl;` ` if(s.find(d, 1) != string::npos) cout << "Bor";` ` else cout << "Yo`q";` `  return 0;` `}`	`Bor` `4294967295` `Yo`q`

Mazkur **find()** funksiyasi qidirilayotgan satrni qaralayotgan satrda birinchi marta uchragan o‘rniga (pozitsiyasiga) mos indeks qiymatini qaytaradi. Agar qidirilayotgan satrni qaralayotgan satrdagi eng oxirgi uchrash o‘rnini aniqlash zarur bo‘lsa (demak, satrni oxiridan boshlab qarash kerak ekan), u holda **rfind()** funksiyasidan foydalanish mumkin. Bu holda, masalan, **s** = "**abdsaacdccaa**" va **d** = "**aa**" bo‘lsa, **find()** funksiyasi **4** qiymatni qaytaradi, lekin **d** satri esa **s** satrda ikki marta uchraydi va uning oxirgi uchrash o‘rni **10** ga teng. Ishlashi **find()** funksiyasi bilan bir xil bo‘lgani uchun **rfind()** funksiyasiga izoh bermasdan quyidagi misol bilan kifoyalanamiz.

Dastur	Natijasi
#include <iostream> #include <string> using namespace std; int main(){ string s{"abdsaacdccaa"}; string d{"aa"}; if(s.find(d) != string::npos) cout <<s.find(d); else cout << "Yo`q"; cout << endl; if(s.rfind(d) != string::npos) cout << s.rfind(d); else cout << "Yo`q"; return 0; }	4 10

C++11 standartidan boshlab qo‘shilgan rbegin() va rend() funksiyalari ishini mustaqil o‘rganishni tavsiya etamiz.

SATRLARDA MURAKKAB QIDIRISH FUNKSIYALARI

Tarkibi murakkabroq ko‘rinishdagi qidiruv funksiyalari ba’zi hollarda qulay bo‘lgani uchun ularni o‘rganish ayni muddaodir.

1. **`find_first_of(string s2, size_t k)`** – qaralayotgan satrning **k**-o‘rnidan boshlab qidirilayotgan **s2** satrining qaralayotgan satrda birinchi uchragan belgisining satrdagi o‘rnini qaytaradi. Agar **k** ko‘rsatilmagan bo‘lsa, u holda satrning boshidan boshlab qaraladi. Masalan, **a** = **"1234"**, **b** = **"8732"** bo‘lsa, **a.find_first_of(b)** funksiyasi **1** qiymatni qaytaradi. Chunki qaralayotgan **a** satrda b satrining faqatgina **'3'** va **'2'** belgilari bor bo‘lib, ulardan birinchi bo‘lib uchragan **'2'** belgisining **a** satrdagi o‘rni **1** ga teng. Agar **s2** satrining hech bir belgisi qaralayotgan satrda uchramasa, u holda funksiya **string::npos** qiymatini qaytaradi.
2. **`find_first_not_of(string s2, size_t k)`** – qaralayotgan satrning **k**-o‘rnidan boshlab qidirilayotgan **s2** satrida uchramaydigan qaralayotgan satrning birinchi belgisining o‘rnini qaytaradi. Agar bunday belgisi yo‘q bo‘lsa, **string::npos** qiymatini qaytaradi. Masalan, **a** = **"1234"**, **b** = **"87321"** bo‘lsa, **a.find_first_not_of (b)** funksiyasi **3** qiymatni, ya’ni **'4'** belgisining o‘rnini qaytaradi.
3. **`find_last_of(string s2, size_t k)`** – qaralayotgan satrning **k**-o‘rnidan boshlab qidirilayotgan **s2** satrining qaralayotgan satrda eng oxirida uchragan belgisining satrdagi o‘rnini qaytaradi. Agar **s2** satrining hech bir belgisi satrda uchramasa, **string::npos** qiymatini qaytaradi. Masalan, **a** = **"1234"**, **b** = **"8732"** bo‘lsa, **a.find_last_of(b)** funksiyasi **2** qiymatni, ya’ni **'3'** belgisining o‘rnini qaytaradi.

4. **find_last_not_of(string s2, size_t k)** – qaralayotgan satrning **k**-o'rnidan boshlab qidirilayotgan **s2** satrida uchramaydigan qaralayotgan satrning oxirgi belgisining o'rnini qaytaradi. Agar bunday belgi bo'lmasa, u holda **string::npos** qiymatini qaytaradi. Masalan, **a = "1234"**, **b = "8722"** bo'lsa, **a.find_last_not_of (b)** funksiyasi **3** qiymatni, ya'ni **'4'** belgisining o'rnini qaytaradi.

Yuqoridagi funksiyalarning barchasida **k** ko'rsatilmagan bo'lsa, u holda qidirish jarayoni satr boshidan boshlanadi. Qidirilayotgan **s2** satri o'rniga belgini qarash ham mumkin. Yuqoridagi namuna misollarga qo'shimcha sifatida quyidagi dastur mazkur qiziqarli funksiyalar ishini izohlaydi.

Dastur	Natijasi

`#include <iostream>`
`#include <string>`
`using namespace std;`
`int main(){`
`  string s{"Maktab"};`
`  string d{"bulut"};`
`  cout << s.find_first_of(d) << endl;`
`  cout << s.find_first_not_of(d) << endl;`
`  cout << s.find_last_of(d) << endl;`
`  cout << s.find_last_not_of(d) << endl;`
`  return 0;`
`}`
	3
	0
	5
	4

STRINGSTREAM OBYEKTI

Aytaylik, bizga matn berilgan va biz shu matndagi so'zlar ustida amallar bajarishimiz kerak. Biroq bunda bir muammo paydo bo'ladi. Matndan qanday qilib so'zlarni ajratib olish mumkin? Matndagi so'zlar ' ' (orachiq), **'\n'** (keyingi satrga o'tish), **'\r'** (kursorni satrning boshiga qaytarish), **'\t'** (gorizontal tabulyatsiya berish) yoki **'\v'** (vertikal tabulyatsiya berish) kabi **whitespice** (o'zb.: orachiq ma'nosida) belgilari deb ataluvchi belgilar yordamida ajratilishi ma'lum. Demak, har safar keyingi **whitespice** belgisini topib ungacha bo'lgan belgilarni satrdan ajratib olishimiz kerak bo'ladi. C++ dasturi buni osonroq yechish imkonini beradi. Buning uchun **<sstream>** kutubxonasining **stringstream** obyektidan foydalanish kifoya.

Masalani soddalashtirish uchun, deylik, matndagi so'zlar faqat ' ' belgisi bilan ajratilgan.

Dastur	Natijasi
#include <iostream> #include <string> #include <sstream> using namespace std; int main(){ string s{"Bu gapning so`zlarini ajratib olishimiz kerak"}; string d; stringstream a(s); //yangi stringstream obyekti e'loni while(a >> d){//yana so`zlar bor bo`lsa, takrorlash davom etadi cout << d << endl; } return 0; }	Bu gapning so`zlarini ajratib olishimiz kerak

Stringstream obyektidan raqamlardan iborat satrni sonli turga yoki, aksincha, sonni satrga o'tkazishda ham foydalanish mumkin:

Dastur	Natijasi

#include <iostream>
#include <string>
#include <sstream>
using namespace std;
int main(){
string s{"123 432 252"};
string d;
stringstream str(s);
int a, b, c;
str >> a >> b >> c;
cout << "a = " << a << endl;
cout << "b = " << b << endl;
cout << "c = " << c << endl;
stringstream str2;
str2 << (a + b) << " " << (b + c);
str2 >> s >> d;
cout << "s = " << s << endl;
cout << "d = " << d << endl;
return 0;
}
	a = 123
	b = 432
	c = 252
	s = 555
	d = 684

Boshqa turdagi o‘zgaruvchilar qiymatini string turiga o‘tkazish uchun **to_string** funksiyasidan foydalaniladi. O‘zgaruvchilar qiymatini **string** turidan boshqa turga o‘tkazish uchun C++14 standartidan boshlab quyidagi funksiyalar mavjud:

Funksiya	Vazifasi
`int stoi(string s)`	s satrini **int** turiga o‘tkazib qaytaradi
`float stof(string s)`	s satrini **float** turiga o‘tkazib qaytaradi
`double stod(string s)`	s satrini **double** turiga o‘tkazib qaytaradi
`long double stold(string s)`	s satrini **long double** turiga o‘tkazib qaytaradi
`long long stoll(string s)`	s satrini **long long** turiga o‘tkazib qaytaradi

Quyidagi dastur yuqoridagi funksiyalardan foydalangan holda **int** turidagi 2107 sonini **string** turiga va yana qaytarib **int** turiga o‘tkazilish jarayonini ifodalaydi:

Dastur	Natijasi

`#include <iostream>`
`#include <string>`
`using namespace std;`
`int main(){`
`  int a = 2107;`
`  string s = to_string(a);`
`  cout << s << endl;`
`  int b = stoi(s);`
`  cout << b << endl;`
`  return 0;`
`}`
	`2107`
	`2107`

4-§. PAIR VA TUPLE

Hozirgacha ko‘rilgan o‘zgaruvchi va o‘zgarmaslar turlari o‘zida yagona qiymat saqlar edi. Massivlar ham ko‘p sondagi qiymatlarni o‘zida saqlash imkoniyatiga ega bo‘lsa-da, bu qiymatlarning barchasi yagona turda bo‘lishi shart edi, ya’ni bir massiv bittadan ortiq turdagi qiymatlarni saqlay olmaydi. Ba’zan shunday masalalar uchraydiki, ularga mos dasturda bir-biriga bog‘langan turli turdagi o‘zgaruvchilar yoki massivlar bilan ish ko‘rishga to‘g‘ri keladi. Masalan, tekislikdagi nuqtaning koordinatasi ikkita (turdagi) qiymatning juftligi sifatida qaralishi, massiv qiymatini indeksiga bog‘langan holda qaralishi kabi holatlar. STL beradigan imkoniyatlardan biri shu kabi bog‘liqliklarni ifodalash, ya’ni bir necha turdagi qiymatlarni guruhlash hamda amallar bajarishdir.

PAIR

Pair inglizcha so‘z bo‘lib, o‘zbek tiliga juftlik kabi tarjima qilinadi. Uning yordamida ikki xil turdagi o‘zgaruvchi yoki o‘zgarmas bir turga birlashtiriladi va yangi murakkab tur – konteyner hosil qilinadi. Pair turi **<utility>** kutubxonasida e’lon qilingan.

Mazkur **pair** turidagi **a** o‘zgaruvchi qiymatlariga murojaat etish quyidagicha amalga oshirilishi mumkin:

`a.first` – birinchi elementi, **`a.second`** – ikkinchi elementi.

Quyida **pair** turidagi o‘zgaruvchilar e’loni va qiymat berishga oid misollar keltirilgan:

Dastur	Natijasi

`#include <iostream>`
`#include <utility>`
`using namespace std;`
`int main(){`
`  pair<int, int> a{11, 2};`
`  pair<int, string> b,c;`
`  pair<bool, string> f;`
`  b.first=21; b.second="07";`
`  c=make_pair(20,"asr");`
`  pair<float, int> d{1.1, 2};`
`  pair<string, int> e({"rad", 3});`
`  f=make_pair(true,"true");`
`  cout << a.first << " " << a.second << endl;`
`  cout << b.first << " " << b.second << endl;`
`  cout << c.first << " " << c.second << endl;`
`  cout << d.first << " " << d.second << endl;`
`  cout << e.first << " " << e.second << endl;`
`  cout << f.first << " " << f.second << endl;`
`  return 0;`
`}`
	`11 2`
	`21 07`
	`20 asr`
	`1.1 2`
	`rad 3`
	`1 true`

Ko‘rib turganingizdek, **pair** turidagi o‘zgaruvchi (o‘zgarmas) nafaqat oddiy turdagi o‘zgaruvchilar (**int**, **float**, **char**), balki murakkab turdagi – **string** turidagi qiymatlarni (satrlarni) ham saqlay oladi. Bundan tashqari, ushbu bobda so‘z yuritilgan va yuritiladigan yangi turlar, tuzilmalar ham **pair** orqali birlashtirilishi mumkin.

Pair turi uchun **<utility>** kutubxonasida e’lon qilingan **get** funksiyasidan ham foydalanish mumkin. Bu funksiyaning umumiy ko‘rinishi quyidagicha:

`get<N>(a)` – **N**-qiymat tartib raqami: **0** yoki **1**, a – **pair** turidagi element

`get<T>(a)` – **T** – qiymat turi (**int**, **bool**, **char**, ...), a – **pair** turidagi element

Albatta, **T** turi **a** elementi saqlaydigan qiymatlardan birining turi bilan mos bo‘lishi kerak. Lekin agar **a** elementi bir xil turdagi ikkita qiymatni saqlaydigan bo‘lsa, bu ko‘rinishdagi **get** funksiyasini qo‘llab bo‘lmaydi. Quyida **pair** va **get** uchun dastur misoli keltirilgan.

Dastur	Natijasi

`#include <iostream>`
`#include <utility>`
`using namespace std;`
`int main(){`
`  pair<int, string> a{11, "satr"};`
`  pair<float, bool> b(1.2, false);`
`  pair<int, int> d{1, 2};`
`  cout << a.first << " " << a.second << endl;`
`  cout << get<0>(b) << " " << get<1>(b) << endl;`
`  cout << get<bool>(b) <<" "<< get<float>(b) << endl;`
`  //cout << get<int>(d) << " " << get<int>(d) << endl;`
`  //d o‘zgaruvchisi bir xil turdagi qiymatga ega!`
`  return 0;`
`}`
	`11 satr`
	`1.2 0`
	`0 1.2`

Yuqorida **pair** turidagi o‘zgaruvchilar bir nechta qiymatni saqlay oladi deb ta’kidlangan edi. Bu tur ikkitadan ortiq turdagi qiymatlarni saqlashi uchun ichma-ich joylashgan “pair” lardan foydalanish mumkin:

```
pair<int, pair<float, char> > a(1, {2.3, 'a'});
```

Bu holda ham o‘zgaruvchi qiymatlariga **first** va **second** kalit so‘zlari yordamida quyidagicha murojaat qilinishi mumkin:

```
cout<<t.first<<", ("<<t.second.first<<", "<<t.second.second<<")";
```

Oxirgi ikki buyruq birlashtirilgan dastur parchasining natijasi shunday bo‘ladi: **1, (2.3, a)**

Pair turidagi o‘zgaruvchilarni ekranga chiqarish uchun, yuqorida ko‘rilganidek, ancha uzun ko‘rsatmalarni takroran yozishga to‘g‘ri keladi. Agar bir dasturda bir necha marotaba pair turidagi o‘zgaruvchilarni chop etishga to‘g‘ri kelsa, u holda bunday o‘zgaruvchilarni chop etish uchun quyidagi dasturdagi kabi alohida funksiya tuzib olish maqsadga muvofiq bo‘ladi.

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`void print(pair<int, char> & t){`
`  cout << t.first << " "`
`  << t.second << endl;`
	2 b
	4 c
	11 z

```
}//pair turidagi elementni
 // ekranga chiqarish uchun
int main(){
    pair<int, char> a{1, 'a'};
    a = {2, 'b'}; print(a);
    a = make_pair(4, 'c'); print(a);
    a.first = 11; a.second = 'z';
    print(a);
    return 0;
}
```

AUTO KALIT SO‘ZI

Kalit so‘zi bo‘lgan **auto** C++ dasturlash tilining imkoniyatlaridan biri bo‘lib, C++11 standartida birinchi bor taqdim etilgan. Hozirgi kunda bu kalit so‘z beradigan imkoniyatlar ancha kengaytirilgan. Kalit so‘zi o‘zgaruvchilarni e’lon qilishda birmucha qulaylik beradi. Bunda o‘zgaruvchi turi o‘rniga **auto** kalit so‘zi qo‘llaniladi. Kompilyator **auto** so‘zini o‘qishi bilan shu o‘zgaruvchiga berilgan qiymatdan kelib chiqib o‘zgaruvchi turini aniqlaydi. Masalan:

Dastur	Natijasi

`#include <iostream>`
`using namespace std;`
`int main(){`
`  auto a{1}; auto f = 2.4;`
`  auto b = string("satr");`
`  auto d = make_pair(1.3, 'a');`
`  cout << a << ' ' << b << endl;`
`  cout << f << ' ' << '(' << d.first;`
`  cout << ", " << d.second << ')' << endl;`
`  return 0;`
`}`
	`1 satr`
	`2.4 (1.3, a)`

Avval misol tariqasida berilgan dasturda bir necha marotaba pair turidagi o‘zgaruvchilarni chop etishga to‘g‘ri kelganda funksiya tuzib olingan edi. Biroq u yerda taqdim etilgan funksiya faqat ikkita **int** va **char** turidagi qiymatni saqlovchi **pair** uchun edi. Umumiy ko‘rinishda, deyarli barcha turdagi **pair** turlarni chop etuvchi funksiya tuzish uchun esa yuqorida ko‘rilgan **andaza** funksiyalardan foydalanish maqsadga muvofiq. Quyidagi misol shu usulni yoritadi.

Dastur	Natijasi

#include <iostream>
using namespace std;
template<typename T1, typename T2>
void print(pair<T1, T2>& p){
cout << '(' << p.first << ", "
<< p.second << ')' << endl;}
int main(){
pair<int, string> a(11, "Toshkent");
auto b = make_pair(3.14, true);
print(a); print(b);
return 0;
}
	(11, Toshkent)
	(3.14, 1)

STRUCTURED BINDINGS – TUZILMALI BOG'LASHLAR

Pair turi saqlayotgan biron-bir qiymatga murojaat etishi uchun, avval aytib o'tilganidek, **get** funksiyasidan yoki **first** va **second** kalit so'zlaridan foydalaniladi. C++17 standartida bu vazifani yanada osonroq hal etish uchun **auto** kalit so'zi bilan bog'lab qo'shimcha imkoniyat taqdim etilgan:

```
pair<int, string> a{1, "ikki"};
auto [x, y] = a;  //x = 1, y = "ikki"
```

Umuman olganda, bu usuldan boshqa maqsadlarda ham foydalanish mumkin. Quyidagi dastur ba'zi imkoniyatlarni o'zida aks ettiradi:

Dastur	Natijasi

#include <iostream>
using namespace std;
int main(){
int a[3] = {1, 2, 3};
auto& [x, y, z] = a;
x++; y--; z = 0;
for(int i = 0;i < 3;i++) cout << a[i] << ' ';
return 0;
}
	2 1 0

TUPLE

Tuple – chekli sondagi turli xil turdagi qiymatlarni o‘zida saqlay oladigan tuzilma – turdir. **Pair** turidan farqli **tuple** turi faqat ikkita emas, **N** ta qiymatni bir vaqtda saqlay oladi. **Tuple** va **pair** turlari bir-biriga juda ham o‘xshaganligi sababli **tuple** turining ishlatilishiga ortiqcha izoh berib o‘tirmaymiz. Quyidagi dasturdagi misollar **tuple** turining aksariyat imkoniyatlarini ochib beradi:

Dastur	Natijasi

`#include <iostream>`
`#include <tuple>`
`using namespace std;`
`int main(){`
`  tuple<int, string, char> a{1, "ikki", '3'};`
`  auto b = make_tuple(1, 2, 3, 4);`
`  cout << get<0>(a) << ' ' <<get<1>(a) << ' ';`
`  cout << get<2>(a) << endl;`
`  auto [m, n, k, t] = b;`
`  cout << m <<' '<< n <<' '<< k <<' '<< t;`
`  return 0;`
`}`
	`1 ikki 3`
	`1 2 3 4`

Har ikki **pair** yoki **tuple** turdagi o‘zgaruvchilarning qiymatini boshqa bir o‘ziga mos bo‘lgan o‘zgaruvchiga o‘zlashtirish, oddiy turdagi o‘zgaruvchilarda bo‘lgani kabi, o‘zlashtirish (=) operatori yordamida amalga oshiriladi. Bunda ikkala o‘zgaruvchining mos o‘rindagi qiymatlari bir xil turda bo‘lishi shart. Masalan:

Dastur	Natijasi

`#include <iostream>`
`#include <tuple>`
`using namespace std;`
`int main(){`
`  auto a = make_pair(1, 2);`
`  pair<int, int> c = a;`
`  auto b = make_tuple(-1, 2, 3);`
`  tuple<int, int, int> d = b;`
`  auto [c0, c1] = c;`
`  auto [d0, d1, d2] = d;`
`  cout << c0 << ' ' << c1 << endl;`
`  cout << d0 << ' ' << d1;`
`  cout << ' ' << d2 << endl;`
`  return 0;`
`}`
	`1 2`
	`-1 2 3`

5-§. VECTOR KONTEYNERI

Vector konteyneri standart massivlar kabi bir turdagi bir necha qiymatni o'zida saqlay oladi. **Vector** konteynerining standart massivlardan asosiy farqi – **vector** konteyneri e'lon qilingandan so'ng ham konteyner o'lchami dastur ishi davomida o'zgarib turishi mumkin, standart massivlarda esa massiv e'lon qilingandan so'ng ularning o'lchamini o'zgartirib bo'lmas edi. Boshqa barcha jihatlardan bu ikki tur bir-biridan deyarli farq qilmaydi. Shunday ekan, **vector** konteyneri ustida ham oddiy massivlardagi kabi amallarni bajarish mumkin. Dasturda **vector konteyneri**ni qo'llash uchun **<vector>** kutubxonasini qo'shish zarur.

Vector turidagi o'zgaruvchilar quyidagi ko'rinishda e'lon qilinadi:

```
vector<T> nom;
```

Bu yerda **T** – **vector** elementlari turi, **nom** – identifikator. Masalan:

```
vector<int> a;
vector<string> b;
vector<pair<int, long double>> c;
```

Demak, **vector** konteyneri elementi boshqa bir konteyner (bu holda **string**) turida bo'lishi ham mumkin. Konteynerni **initsializatsiya** qilishda quyidagi usullardan birini tanlash mumkin:

`vector<T> a;` - uzunligi **0** bo'lgan bo'sh **vector** hosil qilindi.

`vector<T> b(N);` - uzunligi (aniq son) **N** bo'lgan **vector** hosil qilindi.

`vector<int> c{1, 2, 3, 4};` - uzunligi **4** ga teng bo'lgan va elementlari **1**, **2**, **3**, **4** sonlaridan iborat bo'lgan **int** turidagi **vector** hosil qiladi.

Umumiy holdagi quyidagi e'lon:

```
vector<T> d{E1, E2, E3, …, EN};
```

uzunligi **N** va **E1**, **E2**, **E3**, …, **EN** elementlardan iborat bo'lgan **vector** hosil qiladi. Bu holda **T** tur va **E1**, **E2**, **E3**, …, **EN** elementlarning turi mos kelishi shart.

Agar **b vector** turda bo'lsa, u holda quyidagi

```
vector<T> f(b);
```

e'loni orqali **f vector**i **b vector**i elementlaridan hosil qilinadi, bunda **f vector**i elementlarining **T** turi **b vector**i elementlari turiga mos kelishi shart. Hosil bo'lgan **f vector**i uzunligi **b vector**i uzunligiga teng bo'ladi.

Quyidagi e'lon uzunligi **N** ga teng bo'lgan **T** turidagi va har bir elementi **E** ga teng bo'lgan **vector** hosil qiladi:

```
vector<T> p(N, E);
```

Quyidagi e'lon **It1** va **It2** iteratorlari orasidagi elementlardan tashkil topgan **vector** hosil qiladi:

```
vector<T> v(It1, It2);
```

Vector elementlariga murojaat qilish ham standart massivlar kabi **[ ]** orqali amalga oshiriladi. **Vector** elementlariga murojaat qilish uchun **at(k)** a'zo funksiyasidan ham foydalanish mumkin. Bu funksiya **[k]** operatori kabi **vector**ning k-o'rindagi elementiga murojaat qiladi. Uning **[ ]** operatoridan farqi shundaki, agar **vector**ning **k**-elementi bo'lmasa, ya'ni agar **vector N** ta elementdan iborat bo'lib va **k >= N** bo'lsa, u holda bu haqida xabar beriladi, lekin **[ ]** operatori bu holni inobatga olmaydi. Shu sababli **[ ]** operatorini ishlatishdan avval murojaat etilayotgan indeksda **vector** elementi borligiga ishonch hosil qilish kerak, aks holda dastur noaniq natija berishi mumkin.

Dastur	Natijasi

`#include <iostream>`
`#include <vector>`
`#include <string>`
`using namespace std;`
`int main(){`
`  vector<int> mukammal{6, 28, 496, 8128};`
`  for(int i = 0;i < 4;i ++)  {`
`  cout << mukammal[i] << ' ';}`
`  cout << endl;`
`  vector<string> kitob(5, "kitob");`
`  for(int i = 0;i < 5;i ++)  {`
`  cout << kitob.at(i) << ' '; }`
`  return 0;`
`}`
	`6 28 496 8128`
	`kitob kitob kitob kitob`
	`kitob`

Aytib o'tilganidek, STL konteynerlari elementlariga iteratorlar yordamida birma-bir murojaat qilish mumkin. Bu imkoniyat quyidagi misolda aks ettirilgan:

Dastur

`#include <iostream>`
`#include <vector>`
`#include <iterator>`
`using namespace std;`
`int main(){`
`  vector<int> a{1, 2, 3, 4};`
`  for(vector<int>::iterator it = a.begin(); it != a.end(); it ++)`
`  { cout << *it << ' '; (*it) ++;  }  cout << endl;`
`  for(auto it = a.begin(); it != a.end();  it ++){`
`  cout << *it << ' ';  }`
`  return 0;`
`}`

Natijasi
`1 2 3 4` `2 3 4 5`

Dasturda ko‘rsatilganidek, **vector<int>** turidagi o‘zgaruvchiga mos iterator hosil qilish uchun **vector<int>::iterator it** ko‘rinishidagi ko‘rsatma yozish hamda **vector** elementlari tugaganligini tekshirish uchun har safar **it** iteratorini **a.end()** iteratori bilan taqqoslash kerak. Eslatib o‘tamiz, **a.end()** iteratori konteyner oxirini ko‘rsatuvchi iterator edi, **it++** esa iteratorni bir qadam oldinga suradi, ya’ni **vector**ning keyingi elementi iteratori hosil bo‘ladi. Dasturda yozilgan **vector<int>::iterator** ko‘rsatma ancha uzunligini hisobga olib, uning o‘rniga dasturning keyingi bo‘lagida **auto** kalit so‘zi qo‘llangan. Dasturdan ko‘rinadiki, iteratorlar ham ko‘rsatkichlar kabi o‘zi ko‘rsatayotgan element qiymatiga * operatori yordamida murojaat qiladi.

VECTOR KONTEYNERIGA A’ZO FUNKSIYALAR

ASSIGN

Vector konteyneriga **assign** a’zo funksiyasi yordamida qiymatlar o‘zlashtirish mumkin:

1. **assign(N, E)** – **N** ta **E** elementidan iborat massiv hosil qiladi;

assign(It1, It2) – **It1** va **It2** iteratorlari orasidagi elementlardan iborat **vector** hosil qiladi;

assign({E1, E2, …, EN}) – **E1**, **E2**, …, **EN** elementlaridan iborat **vector**ni hosil qiladi.

Quyidagi dastur misolida shu imkoniyatlar ko‘rsatib berilgan.

Dastur	Natijasi

`#include <iostream>`
`#include <vector>`
`using namespace std;`
`int main(){`
`  vector<int> a;`
`  a.assign({1, 2, 3, 4});`
`  for(int i = 0;i < 4;i ++)`
`  {cout << a[i] << ' '; } cout << endl;`
`  a.assign(5, 4);`
`  for(int i = 0;i < 5;i ++)`
`  { cout << a[i] << ' '; } cout << endl;`
`  vector<int> fib{1, 2, 3, 5, 8};`
`  a.assign(fib.begin(), fib.end());`
`  for(int i = 0;i < 5;i ++){`
`  cout << a[i] << ' '; }`
`  return 0;`
`}`
	`1 2 3 4`
	`4 4 4 4 4`
	`1 2 3 5 8`

INSERT

Vectorning ixtiyoriy o‘rniga yangi elementni qo‘shish uchun **insert()** funksiyasi qo‘llanadi. Bu funksiyaning bir necha ko‘rinishlari mavjud:

- **`insert(iterator it, T v)`** – **it** ko‘rsatgan elementdan oldingi o‘ringa **v** qiymatini joylashtiradi va shu joylashtirilgan o‘rindagi elementni ko‘rsatuvchi iteratorni qaytaradi;
- **`insert(size_t pos, T v)`** – **pos** o‘rnidagi elementdan oldingi o‘ringa **v** qiymatini joylashtiradi va shu joylashtirilgan o‘rindagi elementni ko‘rsatuvchi iteratorni qaytaradi;
- **`insert(iterator pos, size_t n, T v)`** – **pos** iteratori ko‘rsatgan elementdan oldingi o‘ringa **v** qiymatini **n** ta nusxasini joylashtiradi;
- **`insert(iterator pos, iterator it1, iterator it2)`** – [**it1; it2**) oraliqdagi qiymatlarni **pos** iteratori ko‘rsatgan elementdan oldingi o‘ringa joylashtiradi.

Quyida shu funksiya ishlashiga oid dastur keltirilgan.

Dastur	Natijasi

`#include <iostream>`
`#include <vector>`
`#include <iterator>`
`using namespace std;`
`void print(vector<int>& a){`
`  for(auto it = a.begin(); it!=a.end();it ++)`
`  {cout << *it << ' '; }  cout << endl;`
`}`
`int main(){`
`  vector<int> a{1, 2, 5, 6};`
`  a.insert(a.begin(), 0);  print(a);`
`  vector<int> b{3, 4};`
`  auto pos = a.begin();`
`  advance(pos, 3);`
`  a.insert(pos, b.begin(), b.end());  print(a);`
`  a.insert(a.begin(), 2, -1);  print(a);`
`  return 0;`
`}`
	`0 1 2 5 6`
	`0 1 2 3 4 5 6`
	`-1 -1 0 1 2 3 4 5 6`

Yuqoridagi dasturda avval aytib o‘tilgan **advance()** funksiyasi qo‘llangan bo‘lib, bu funksiya o‘ziga berilgan iteratorni ko‘rsatilgan qadam sonicha oldinga suradi. Masalan, **advance(it, 2)** funksiyasi **it** iteratorini **2** qadam oldinga suradi, ya’ni bu ma’no jihatidan ikki marotaba ++ operatorini yozish bilan teng kuchlidir. Ba’zi iteratorlarning turiga qarab, **advance()** funksiyasi ishlash tezligi bir necha ++ operatorlari yordamida surishdan ko‘ra tezroq ishlashi mumkin. Bunday holat **vector** elementlari iteratorlari bilan sodir bo‘ladi. Shu

sababli **vector** konteyneri uchun iteratorni bir necha qadam ilgari surish lozim bo'lsa, u holda **advance()** funksiyasidan foydalanish maqsadga muvofiq.

Bundan tashqari, dasturda **vector** konteyneri elementlarini ekranga chiqarish uchun **print** funksiyasi tuzilgan. Uning e'loniga e'tibor berilsa, **vector<int> a** emas balki **vector<int>& a** ifodasini ko'rish mumkin. Bunga sabab **vector** konteyneri o'zida bir emas, bir nechta elementni saqlashi va agar e'lon **vector<int> a** ko'rinishida, ya'ni **call-by-value** metodi bo'yicha funksiyaga argument beriladigan bo'lsa, keraksiz nusxalash amallari barajariladi va dastur ishi sekinlashadi. Chunki bu e'lon holida **a vector**ining har bir elementi **print** funksiyasi ichida hosil qilingan **vector**ga nusxalanadi. Shu bois **print** funksiyasida **call-by-reference** metodi qo'llangan, ya'ni **&** yozilgan, bu holda hech narsa nusxalanmaydi.

VECTOR KONTEYNERI UCHUN MUHIM A'ZO FUNKSIYALAR

Quyidagi jadvalda **vector** konteyneriga oid muhim a'zo funksiyalar keltirilgan:

A'zo funksiyalar	Vazifasi
front()	**vector**ning birinchi elementiga murojaat qiladi
back()	**vector**ning oxirgi elementiga murojaat qiladi
empty()	**vector** uzunligi 0 ga teng bo'lsa **true**, aks holda **false** qiymatini qaytaradi
size()	**vector** uzunligini qaytaradi, bu funksiya qaytaradigan qiymat **size_type** (yoki **size_t**) turidadir (**size_type** turi – **unsigned int** turidir)
max_size()	**vector**ning maksimal sig'imini qaytaradi
clear()	**vector**ni "bo'sh" holatga olib keladi, ya'ni barcha elementlarini o'chirib tashlaydi, demak, **vector** uzunligi **0** ga teng bo'ladi
push_back(v)	**vector** oxiriga **v** qiymatini qo'shadi, ya'ni **vector** uzunligi bittaga ortadi
pop_back()	**vector** oxiridagi elementni **vector**dan o'chiradi, ya'ni **vector** uzunligi bittaga kamayadi

Bu a'zo funksiyalar ishini quyidagi dasturlar aks ettirgan.

Dastur	Natijasi

`#include <iostream>`
`#include <vector>`
`#include <ios>`
`using namespace std;`
`int main(){`
`  vector<int> a{1, 2, 3, 4};`
`  for(size_t i{};i < a.size();i ++)`
`  { cout << a[i] << ' '; } cout << endl;`
`  cout << "front - " << a.front() << endl;`
`  cout << "back - " << a.back() << endl;`
	`1 2 3 4`
	`front - 1`
	`back - 4`
	`7 -4 100 0`
	`true`

```
    a.front() = 7; a.at(1) = -4;
    a[2] = 100;    a.back() = 0;
    for(size_t i{};i < a.size();i ++)
    { cout << a[i] << ' ';  } cout << endl;
    a.clear();
    cout << boolalpha << a.empty() << endl;
    return 0;
}
```

Dastur	Natijasi

`#include <iostream>`
`#include <vector>`
`#include <ios>`
`using namespace std;`
`int main(){`
`  vector<int> a;`
`  a.push_back(1);  a.push_back(2);`
`  cout << a.size() << endl;`
`  cout << a.max_size() << endl;`
`  for(size_t i{};i < a.size();i ++)`
`  { cout << a[i] << ' '; } cout << endl;`
`  a.pop_back();  a.pop_back();`
`  cout << boolalpha << a.empty() << endl;`
`  return 0;`
`}`
	`2`
	`1073741823`
	`1 2`
	`true`

VECTOR KONTEYNERI UCHUN FOYDALI FUNKSIYALAR

Vector konteynerini qo'llab masalalar yechishda algoritmni, qolaversa, dastur matnini soddalashtirish uchun turli foydali funksiyalar kiritilgan. Ularning ba'zilari haqidagi ma'lumot va misollarni quyida bayon etamiz. Bu funksiyalardan foydalanishda dasturga **<algorithm>** kutubxonasini qo'shish zarur.

1. all_of, any_of, none_of – konteynerning berilgan oraliqdagi **barcha/biror-bir/hech qaysi** elementi ko'rsatilgan **predikat**ga (argumenti mulohaza bo'lgan funksiyaga, soddaroq aytganda, shartga yoki shartlar to'plamiga mos natijani qaytaruvchi mantiqiy funksiyaga) mos kelish yoki kelmasligini tekshiradi. Bunda har bir funksiya quyidagicha e'lon qilingan:

```
..._of(iterator it1, iterator it2, Predicate p)
```

Predikat sifatida turli xil vositalardan foydalanish mumkin. Bular jumlasiga **bool** qiymat qaytaradigan **funksiyalar** yoki **lambda-ifoda**lar kiradi. Ushbu qo'llanma doirasida lambda-ifodalarni qamray olmasligimiz sababli, predikat sifatida faqat funksiyalardan foydalanamiz.

Yodda tuting, funksiya qiymati **bool** turiga mos bo'lishi va funksiya parametri turi konteyner elementlari turiga mos bo'lishi shart. Ushbu ko'rsatilgan funksiyalar, berilgan oraliqdagi barcha elementlarga predikat sifatida ko'rsatilgan funksiyani qo'llab chiqadi, ya'ni har bir elementni predikat sifatida berilgan funksiyaga argument sifatida beradi va funksiya qiymatini tekshiradi. Masalan, quyidagi dasturda **vector** elementlarini 10 dan katta yoki katta emasligini tekshiramiz:

Dastur

```
#include <iostream>
#include <algorithm>
#include <vector>
#include <ios>
using namespace std;
bool p(int& v){
    if(v > 10)return true;else return false;
}//bu funksiya - predikat
int main(){
    vector<int> a{11, 11, 12, 9};
    cout << boolalpha;
    cout << "Barcha elementlari 10 dan katta: ";
    cout << all_of(a.begin(), a.end(), p) << endl;
    cout << "10 dan katta elementi bor: ";
    cout << any_of(a.begin(), a.end(), p) << endl;
    cout <<"Hech bir elementi 10 dan katta emas: ";
    cout << none_of(a.begin(), a.end(), p);
    return 0;
}
```

Natijasi

```
Barcha elementlari 10 dan katta: false
10 dan katta elementi bor: true
Hech bir elementi 10 dan katta emas: false
```

Ko'rib turganingizdek, **predikat** sifatida berilgan funksiya bitta **int** turidagi qiymat qabul qiladi va ushbu qabul qilingan qiymatning 10 dan katta yoki katta emasligini tekshiradi.

2. **`for_each(iterator it1, iterator it2, function f)`** – **it1** va **it2** iteratorlari orasidagi barcha elementlarga **f** funksiyasini qo'llaydi, bunda **f** funksiya faqat bitta argument qabul qilishi shart.

Misol sifatidagi quyidagi dasturda **double** turidagi vector elementlarining kvadrat ildizini hisoblash va chop etishda shu imkoniyatdan foydalanamiz:

Dastur	Natijasi

#include <iostream>
#include <vector>
#include <algorithm>
#include <cmath>
using namespace std;
void f(double& t){
  if(t >= 0.0)t = sqrt(t);
}
void print(double& t){
  cout << t << ' ';
}
int main(){
  vector<double> a{4.0, 16.0, 11.0};
  for_each(a.begin(), a.end(), f);
  for_each(a.begin(), a.end(), print);
  return 0;
}
	2 4 3.31662

3. **count(iterator it1, iterator it2, T v)** – **it1** va **it2** iteratorlari oralig'idagi elementlardan **v** ga tenglari sonini qaytaradi;

count_if(iterator it1, iterator it2, Predicate p) – **it1** va **it2** iteratorlari orasidagi elementlardan **p** predikatiga mos keladiganlari sonini qaytaradi.

Bu kabi funksiyalar kerakli elementlar sonini aniqlashda (massivlarda ham shu kabi imkoniyat ko'rilgan edi) dasturni qanchalik soddalashtirishini quyidagi dastur misolida ko'rish mumkin.

Dastur	Natijasi

#include <iostream>
#include <vector>
#include <algorithm>
using namespace std;
int main(){
  vector<int> a{4, 14, 4, 24, 44};
  int el=4;
  cout<< count (a.begin(), a.end(), el);
  return 0;
}
	2

4. **find(iterator it1, iterator it2, T v)** – **it1** va **it2** iteratorlari orasidagi elementlardan **v** ga teng bo'lgan birinchi elementni ko'rsatuvchi iteratorni qaytaradi. Agar bunday element mavjud bo'lmasa, konteyner oxirini ko'rsatuvchi iteratorni qaytaradi;

find_if(iterator it1, iterator it2, Predicat p) – **p** predikatiga mos keladigan birinchi elementni ko'rsatuvchi iteratorni qaytaradi;

find_not_if(iterator it1, iterator it2, Predicate p) – **p** predikatiga mos kelmaydigan birinchi elementni ko'rsatuvchi iteratorni qaytaradi.

Funksiyalar ishini quyidagi dasturdan ko'rish mumkin.

Dastur	Natijasi
`#include <iostream>` `#include <algorithm>` `#include <vector>` `using namespace std;` `bool t_(string& s){` `  if(!s.empty() && s[0] == 'T') return true;` `  else return false;` `}` `int main(){` `  vector<string> a{"Toshkent", "Termiz", "Samarqand"};` `  auto it = find(a.begin(), a.end(), "Toshkent");` `  if(it != a.end())cout << "Bor";else cout << "Yo`q";` ` cout << endl;` ` it = find_if(a.begin(), a.end(), t_);` ` cout << *it << endl;` ` it = find_if_not(a.begin(), a.end(), t_);` ` cout << *it << endl;` ` return 0;` `}`	`Bor` `Toshkent` `Samarqand`

5. accumulate(iterator it1, iterator it2, T init) – **it1** va **it2** iteratorlari orasidagi elementlarning yig'indisini qaytaradi, **init** – boshlang'ich qiymat. Mazkur funksiya **<numeric>** kutubxonasida e'lon qilingan. Bu funksiyaning ishlashiga misol quyidagicha:

Dastur	Natijasi
`#include <iostream>` `#include <vector>` `#include <algorithm>` `#include <numeric>` `using namespace std;` `int main(){` `  int s; vector<int> a{1, -3, 5, 2};` `  s=accumulate(a.begin(), a.end(), 0);` `  cout <<"Yig`indisi: "<< s;` ` return 0;` `}`	`Yig`indisi: 5`

6. `reverse(iterator it1, iterator it2)` – [**it1**; **it2**) oraliqdagi elementlarni teskari tartibda joylashtiradi. Masalan:

Dastur	Natijasi

`#include <iostream>`
`#include <vector>`
`#include <algorithm>`
`using namespace std;`
`int main(){`
`  vector<int> a{1, -3, 5, 2};`
`  reverse(a.begin(), a.end());`
`  for(int i : a)cout << i << ' ';`
`  return 0;`
`}`
	`2 5 -3 1`

Yuqoridagi dasturda **for** operatorini "boshqa ko'rinishda" ishlatilganining guvohi bo'ldingiz. Bunday turdagi **for** operatorida ":" dan so'ng ko'rsatilgan konteyner elementlarini birma-bir konteyner nomidan avval ko'rsatilgan o'zgaruvchiga o'zlashtirib boriladi. Bu xuddi **for_each** funksiyasini to'liq konteynerga ishlatish bilan teng kuchlidir. Aksariyat hollarda bunday **for** operatorlarida o'zgaruvchi turi yaqqol ko'rsatilmay, buning o'rniga **auto** kalit so'zi ishlatiladi.

Quyidagi **binary_search()**, **lower_bound()** va **upper_bound()** funksiyalari qo'llanganda berilgan oraliqdagi elementlar **tartiblangan** bo'lishi shart, chunki bu funksiyalar avvalgi bobda bayon etilgan **binar qidiruv** algoritmidan foydalanadi. Shu bois bu funksiyalar qo'llaydigan algoritmning bajarilish vaqti **O(log(n))** ga teng bo'ladi.

7. `binary_search(iterator it1, iterator it2, T v)` – [**it1**; **it2**) orasidagi elementlar ichida **v** ga teng element bor yoki yo'qligini binar qidiruvi yordamida aniqlaydi. Bu funksiyaning vazifasi shunday element faqat bor yoki yo'qligini aniqlash ekan.

8. `lower_bound(iterator it1, iterator it2, T v)` – [**it1**; **it2**) orasidagi elementlar ichida **v** dan kichik bo'lmagan birinchi elementni ko'rsatuvchi iteratorni qaytaradi. Agar bunday element mavjud bo'lmasa **it2** ni qaytaradi. Bu funksiya natijasiga e'tibor beradigan bo'lsak, agar **v** ga teng element mavjud bo'lsa, shu elementni ko'rsatuvchi iteratorni qaytarar ekan.

9. `upper_bound(iterator it1, iterator it2, T v)` – [**it1**; **it2**) orasidagi elementlar ichida **v** dan katta bo'lgan birinchi elementni ko'rsatuvchi iteratorni qaytaradi. Agar bunday element mavjud bo'lmasa **it2** ni qaytaradi.

Yuqoridagi uchala funksiya ishini quyidagi dastur orqali ko'rish mumkin.

Dastur	Natijasi

#include <iostream>
#include <vector>
#include <algorithm>
#include <ios>
using namespace std;
int main(){
vector<char> a{'a', 'b', 'd', 'e'};
cout << boolalpha;
cout << binary_search(a.begin(), a.end(), 'a');
cout << endl;
auto it = lower_bound(a.begin(), a.end(), 'b');
if(it != a.end())cout << *it;
else cout << "Bunday element mavjud emas";
cout << endl;
it = upper_bound(a.begin(), a.end(), 'b');
if(it != a.end())cout << *it;
else cout << "Bunday element mavjud emas";
return 0;
}
	true
	b
	d

Umuman olganda, bu funksiyalarni massivlar uchun ham qo'llash mumkin. Quyidagi dasturda shu imkoniyat taqqoslab berilgan. Dasturda (tartiblangan **vector**da va massivda) qiymati 5 ga teng elementning indeksi aniqlanmoqda.

Dastur	Natijasi

#include <iostream>
#include <vector>
#include <algorithm>
using namespace std;
int main(){
vector<int> a = {1, 2, 3, 4, 5, 6};
auto it = lower_bound(a.begin(), a.end(), 5);
cout << it - a.begin() << endl;
int b[] = {1, 2, 3, 4, 5, 6};
int* pos = lower_bound(b, b + 6, 5);
cout << (pos - b) << endl;
return 0;
}
	4
	4

Avvalgi bobda **sort()** funksiyasi standart massivlar bilan ishlatilishini ko‘rgan edik. Sort funksiyasini **vector** konteyneri bilan ham ishlatish mumkin. Bu holda mos ravishda ko‘rsatkichlar o‘rnini iteratorlar egallaydi, xolos. Agar elementlarni o‘sish yoki kamayish tartibida emas, balki boshqa bir tartibda tartiblash kerak bo‘lsa, elementlarni taqqoslash uchun alohida funksiya yozish va uni **sort()** funksiyasida ishlatish ham mumkin. Masalan, elementlari **pair<int, int>** turida bo‘lgan, mos ravishda, koordinata o‘qlarini ifodalaydigan **vector**ning koordinatalarini koordinata boshiga yaqinligi bo‘yicha tartiblash vazifasi berilgan bo‘lsin. Bu quyidagicha mazmunda bajarilishi mumkin:

Dastur	Natijasi

`#include <iostream>`
`#include <vector>`
`#include <algorithm>`
`#include <cmath>`
`using namespace std;`
 	
`double distance(pair<double, double>& p){`
`  return sqrt(p.first*p.first+p.second*p.second);`
`}`
`bool cmp(pair<double, double>& a, pair<double, double>& b){`
`  if(distance(a) < distance(b)) return true;`
`  else return false;`
`}`
`int main(){`
`vector<pair<double,double>>a{{1, 2}, {2.3, -1.1}, {1.2, 1.8}};`
`  sort(a.begin(), a.end(), cmp);`
`  for(auto x : a){`
`  cout << x.first << ' ' << x.second << endl;`
`  }`
`  return 0;`
`}`
	`1.2 1.8`
	`1 2`
	`2.3 -1.1`

Izoh 13: Umuman olganda, keltirilgan bu foydali funksiyalarning barchasini faqatgina **vector** konteynerigagina emas, balki boshqa shunga o‘xshash **konteyner**larga, shu jumladan, **string** konteyneriga ham qo‘llash mumkin. Masalan, satrni teskarilash uchun:

```
reverse(satr.begin(), satr.end());
```

ko‘rsatmasidan foydalansh mumkin.

VECTORLARNI TAQQOSLASH

Ikkita **vector**ni ==, !=, <, >, <=, >= operatorlari yordamida taqqoslash imkoniyati bor. Quyidagi dastur orqali taqqoslash qanday talab asosida bajarilishini tushunish mumkin:

Dastur	Natijasi

`#include <iostream>`
`#include <vector>`
`using namespace std;`
`int main(){`
`  vector<int> a={1, 2}, b={1, 2},c={1, 1};`
`  vector<int> d={1, 2, 3}, e={1, 3, 2}, f={3};`
`  cout << "a = b: " <<`
`  ((a == b) ? "True" : "False") << endl;`
`  cout << "c < b: ";`
`  cout << ((c < b) ? "True" : "False") << endl;`
`  cout << "a < d: ";`
`  cout << ((a < d) ? "True" : "False") << endl;`
`  cout << "e < d: ";`
`  cout << ((e < d) ? "True" : "False") << endl;`
`  cout << "a < f: ";`
`  cout << ((a < f) ? "True" : "False") << endl;`
`  return 0;`
`}`
	`a = b: True`
	`c < b: True`
	`a < d: True`
	`e < d: False`

6-§. LIST, QUEUE VA STACK KONTEYNERLARI

LIST – RO‘YXAT

Avvalroq ko‘rib chiqilqan **vector** konteyneri va standart massivlar o‘zining elementlarini kompyuter xotirasining ketma-ket joylashgan uyachalarida saqlashi aytib o‘tilgan edi. Bunday xususiyat vector yoki massivning ixtiyoriy indeksdagi elementiga to‘g‘ridan to‘g‘ri murojaat qilish imkoniyatini beradi. Biroq, unga yangi element qo‘shish anchayin murakkabroq vazifadir. **Vector** konteyneri bunday imkoniyatni **insert()** va **push_back()** a’zo funksiyalari yordamida taqdim etsa-da, bu funksiyalar ancha sekin ishlaydi, ya’ni algoritmning bajarilish vaqti **O(n)** ga teng. Ko‘rilayotgan **list** konteyneri esa bunday amalni bajarilish vaqti **O(1)** bo‘lgan algoritm yordamida amalga oshiradi. Bunga sabab, avvalo **list** konteyneri o‘z elementlarini xotiraning ixtiyoriy qismida saqlashi mumkin, ya’ni ular xotirada ketma-ket joylashgan bo‘lishi shart emas. Shu bilan birga, har bir element bir-biri bilan iteratorlar yor-

damida bog‘langan bo‘ladi. Mazkur **list** konteynerining birinchi elementi **head** (ing., o‘zb.: bosh) va konteynerning oxiri **tail** (ing., o‘zb.: dum) so‘zlari yordamida belgilanadi. Tuzilishi jihatidan ikki xil turdagi **list** mavjud:

- **singly-linked-list (bir tomonlama bog‘langan)**
- **doubly-linked-list (ikki tomonlama bog‘langan)**

C++ da **singly-linked-list forward_list** konteynerida o‘z ifodasini topsa, **doubly-linked-list list** konteyneriga mos keladi. Ularning farqi shundaki, **list** konteynerining har bir elementi o‘zidan keyin va oldin kelgan elementlar adreslari haqida ma’lumotga ega bo‘ladi, **forward_list** konteynerida esa har bir element faqat o‘zidan keyin kelgan element adresini eslab qoladi.

List konteynerida **vector** konteynerida mavjud barcha funksiyalar bor. Ba’zi bir funksiyalarning ishlash tezligi yuqorida aytib o‘tilgan sababga ko‘ra o‘zgarishi mumkin, ammo funksiya bajaradigan vazifalar bir xil. Misol tariqasida quyidagi dasturni keltiramiz:

Dastur	Natijasi

`#include <iostream>`
`#include <list>`
`#include <algorithm>`
`#include <ios>`
`using namespace std;`
`void print(int& v){ cout << v << ' '; }`
`int main(){`
`  list<int> a{1, 2, 3, 5};`
`  for(auto& x : a){ cout << x << ' ';`
`  } cout << endl;`
`  auto it = a.begin();`
`  advance(it, 3);`
`  a.insert(it, 4);`
`  for_each(a.begin(), a.end(), print);`
`  cout << endl;`
`  a.erase(a.begin());`
`  a.push_back(6);`
`  cout << "size = " << a.size() << endl;`
`  for_each(a.begin(), a.end(), print);`
`  cout << endl; a.clear();`
`  cout << boolalpha << a.empty() << endl;`
`  return 0;`
`}`
	`1 2 3 5`
	`1 2 3 4 5`
	`size = 5`
	`2 3 4 5 6`
	`true`

Bularga qo‘shimcha **list** konteynerida **push_front()** va **pop_front()** a’zo funksiyalari mavjud bo‘lib ular, mos ravishda, konteyner boshiga yangi element qo‘shish va konteyner boshidagi elementni o‘chirishga xizmat qiladi. Elementlarini tartiblash uchun **list** konteyneri o‘zida **sort()** a’zo funksiyasini saqlaydi. Bundan tashqari **list** konteyneri ketma-ket kelgan bir xil qiymatli elementlarni bittasi qolguncha o‘chirish uchun **unique()** funksiyasiga ham ega. Bu funksiyalar ishini quyidagi dasturdan ko‘rish mumkin.

Dastur	Natijasi

`#include <iostream>`
`#include <list>`
`using namespace std;`
`int main(){`
`  list<int> a{1, 1, 4, 1, 2, 5, 5, 3};`
`  for(auto& x : a){cout << x << ' ';`
`  } cout << endl;`
`  a.push_front(1);`
`  a.sort();`
`  for(auto& x : a){cout << x  << ' ';`
`  } cout << endl;`
`  a.unique();`
`  for(auto& x : a){cout << x << ' ';`
`  }cout << endl;`
`  return 0;`
`}`
	`1 1 4 1 2 5 5 3`
	`1 1 1 1 2 3 4 5 5`
	`1 2 3 4 5`

QUEUE – NAVBAT KONTEYNERI

Queue konteyneri **<queue>** kutubxonasida e’lon qilingan juda sodda konteyner bo‘lib, oddiy ketma-ket keladigan **navbat** hosil qilish uchun ishlatiladi. Xuddi hayotdagi kabi navbat ustida bajarish mumkin bo‘lgan amallarga o‘xshash bo‘lgani uchun navbat konteyneri ham deyishadi: bu konteynerga yangi element faqat uning oxiridan qo‘shilishi hamda elementni o‘chirish faqat konteynerning boshidan bajarilishi mumkin. Bu amallarni bajarish uchun, mos ravishda, **push()** va **pop()** a’zo funksiyalari mavjud. Queue konteynerining boshidagi elementiga **front()**, oxiridagi elementiga **back()** funksiyalari yordamida murojaat qilish mumkin. Boshqa konteynerlarda bo‘lgani kabi **queue** konteyneri ham **size()**, **empty()** a’zo funksiyalariga ega.

Dastur	Natijasi

#include <iostream>
#include <queue>
using namespace std;
int main(){
queue<int> a;
a.push(1); a.push(2);
cout << a.front() << ' ' << a.back() << endl;
a.pop();
cout << "size = " << a.size() << endl;
a.pop();
cout << "empty: " << boolalpha << a.empty() << endl;
return 0;
}
	1 2
	size = 1
	empty:
	true

STACK – TAXLAM

Queue konteyneridan farqli o'laroq, bu konteyner elementlarni (taxlam ko'rinishida) ustma-ust joylashtiradi deyish mumkin. Ya'ni agar bu konteynerga yangi element qo'shiladigan bo'lsa, u holda bu element konteynerning eng ustiga joylashtiriladi, va, agar bu konteynerning biror-bir elementi kerak bo'lsa, avval uning ustidagi barcha elementlarni olib, keyin kerakli elementni olish kerak. Boshqa so'z bilan aytganda, **stack** turidagi konteynerda faqat eng yangi qo'shilgan elementga murojaat qilish mumkin, boshqa elementlarga murojaat qilish uchun esa avval undan so'ng qo'shilgan elementlarni konteynerdan o'chirish zarur. Bu kabi amallarni bajarish uchun **stack** konteyneri quyidagi a'zo funksiyalarga ega:

- **push**() – yangi element qo'shish uchun
- **pop**() – yangi qo'shilgan elementni o'chirish uchun
- **top**() – yangi qo'shilgan elementga murojaat etish uchun

Albatta, a'zo funksiyalar qatoridan **empty**() va **size**() funksiyalari ham joy olgan. Tushunarliki, **stack** konteyneri **<stack>** kutubxonasida e'lon qilingan. Bu konteyner quyidagi dasturda berilgan.

Dastur	Natijasi

#include <iostream>
#include <stack>
using namespace std;
int main(){
stack<int> a;
a.push(10); a.push(11);
cout << a.top() << endl;
a.pop();
	11
	10
	1
	true

```
    cout << a.top() << endl;
    cout << a.size() << endl;
    a.pop();
    cout << boolalpha << a.empty() << endl;
    return 0;
}
```

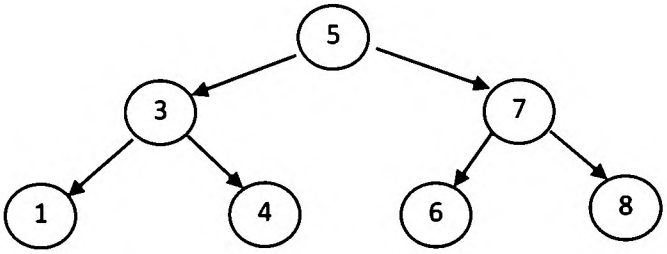
7-§. SET KONTEYNERI

Set – inglizchadan tarjimasi **to‘plam** bo‘lgan bu konteyner murakkab konteynerlardan biri hisoblanadi. Bunga konteynerni qurishda murakkab algoritmlar va tuzilmalar qo‘llanganligi sababdir. Qo‘llanma doirasida **set** uchun ishlatilgan tuzilma yoki algoritmlar haqida to‘liq ma’lumot emas, aksincha, umumiy tushuncha berish bilan kifoyalaniladi. **Set** konteyneri barcha boshqa konteynerlar kabi ma’lumotlarni saqlash uchun ishlatiladi. **Set konteyneri tarkibida qiymati bir xil bo‘lgan bir nechta element bo‘lishiga yo‘l qo‘ymaydi**. Yana bir xususiyat: quyida bayon etilgan **balanced binary search tree** asosidagi set tuzilmasi sharofati bilan **set konteyneri elementlarini chiqarishda elementlar o‘sish tartibida chiqadi.**

Set konteynerining boshqa konteynerlardan ustun tarafi shundaki, undagi deyarli barcha amallar **O(logN)** vaqtda bajariladi, bu yerda **N set** konteynerida saqlanayotgan elementlar sonini bildiradi. Ya’ni **set** konteyneriga yangi element qo‘shish, undan ma’lum qiymatli elementni qidirish, elementni o‘chirish kabi amallarga **O(logN)** vaqt sarflanadi. Ba’zi (masalan, **vector**) konteynerlarda ma’lum qiymatli elementning bor yoki yo‘qligini aniqlash uchun konteynerdagi barcha elementlarni birma-bir “qarab chiqish” lozim edi. Shu kabi holat konteynerga yangi element qo‘shish yoki biror elementni **vector**dan o‘chirishda sodir bo‘lishini kuzatish mumkin. Shuning uchun ham bunday amallarga kamroq vaqt sarflash uchun esa **set** konteyneridan foydalanish maqsadga muvofiq.

Aytib o‘tilganidek, **set** konteyneri murakkab tuzilmaga asoslangan konteyner hisoblanadi. Shu bois uning mazmunini birmuncha osonroq tuzilma qiyosida ochib berishga harakat qilamiz.

Quyidagi chizmada **balanced binary search tree** – ingliz tilidan tarjimasi: **muvozanatdagi ikkilik qidiruv daraxti** tasvirlangan. Doiralar – **node** (ing., o‘zb.: tugun) nuqtalar, ya’ni **set** elementlarini o‘zida saqlovchi ichki tuzilma. Har bir **node** nuqta o‘zida element qiymatini hamda boshqa **node** nuqtalar adreslarini saqlaydi.



Chizmada tasvirlangan tuzilma bunday tuzilmalarning xususiy hol bo‘lib, quyidagi xususiyatlarga ega:

- har bir **node** nuqta element qiymati bilan birga, ikkita boshqa **node** nuqtaning adresini o‘zida saqlaydi, bu **node** nuqtalar shu **node** nuqtaning chap va o‘ng “bolasi” deb ataladi;
- eng yuqoridagi **node** nuqta **root** (ing., o‘zb.: ildiz) **node** nuqta hisoblanadi;
- har bir **node** nuqtaning qiymati uning chap “bolasi” qiymatidan katta va uning o‘ng “bolasi” qiymatidan kichik bo‘lishi lozim;
- faqatgina eng quyidagi **node** nuqtalarning ikkitadan kam “bolasi” bo‘lishi mumkin, ya’ni o‘rtadagi **node** nuqtalarda doim chap va o‘ng “bola” bo‘lishi shart.

Endi ushbu tuzilmada 4 qiymatli element bor yoki yo‘qligi quyidagicha aniqlanadi:

1. Ildiz **node** nuqta qiymati bilan 4 qiymati taqqoslanadi. “4 < 5” shart bajarilgani uchun kerakli element ildiz **node** nuqtaning chap “bolasi”da bo‘lishi mumkin, degan xulosaga kelinadi.
2. Ildiz **node** nuqtaning chap “bolasi” taqqoslanadi. Uning qiymati 3 va u “3 < 4” shartni qanoatlantiradi. Demak, qidirilayotgan element joriy **node** nuqtaning o‘ng “bolasi”da bo‘lishi mumkin.
3. Joriy **node** nuqtaning o‘ng “bolasi” taqqoslanadi. Uning qiymati 4 va u “4 = 4” shartni qanoatlantiradi. Demak, qidirilayotgan qiymatli element tuzilmada mavjud.

Yangi element qo‘shish yoki biror elementni o‘chirish ham shu kabi usulda amalga oshiriladi. Yuqoridagi xususiyatlardan kelib chiqilsa, ushbu tuzilmaning eng yuqorisidagi **node** nuqta va eng quyisidagi **node** nuqtalar (bu ikki **node** nuqtani ham hisobga olganda) orasida eng ko‘pi bilan [**logN + 1**] ta **node** nuqta bo‘lishi mumkin. Demak, qidirish, yangi element qo‘shish, o‘chirish kabi amallar **O(logN)** vaqt talab qiladi.

Set konteyneri qanday tuzilmaga asoslangan bo‘lishi qat’iy belgilab qo‘yilmagan. Biroq ko‘p holatlarda u **Red-Black Tree** deb nomlanuvchi tuzilmaga asoslangan bo‘ladi. Bu tuzilma biz yuqorida tasvirlagan tuzilmaga juda o‘xshash, lekin biroz murakkabroqdir.

SET KONTEYNERINING A'ZO FUNKSIYALARI

Endi bevosita **set** konteyneri taqdim etadigan a'zo funksiyalarni qaraymiz. Tushunarliki, **set** konteyneri **<set>** kutubxonasida e'lon qilingan.

Set turidagi o'zgaruvchilar quyidagi ko'rinishda e'lon qilinadi:

- **`set<T> A;`** – elementlari **T** turda bo'lgan bo'sh **A** set konteynerini hosil qiladi;
- **`set<T> A (iterator it1, iterator it2)`** – [**it1**; **it2**) oraliqdagi elementlardan **A** set konteynerini hosil qiladi.

Set turidagi konteynerga yangi elementlar qo'shish uchun qo'llanadigan **insert()** funksiyasi quyidagi ko'rinishlarga ega:

- **`insert(T v)`** – agar qiymati **v** ga teng bo'lgan element konteynerda yo'q bo'lsagina konteynerga **v** qiymatidagi yangi elementni qo'shadi;
- **`insert(iterator it1, iterator it2)`** – [**it1**; **it2**) iteratorlari orasidagi elementlardan konteynerda yo'qlarini konteynerga qo'shadi.

Set turidagi konteynerdan elementlarni o'chirish uchun qo'llanadigan **`erase()`** funksiyasi quyidagi ko'rinishlarga ega:

- **`erase(T& v)`** – qiymati **v** ga teng bo'lgan element mavjud bo'lsa, uni o'chiradi;
- **`erase(iterator it)`** – **it** iteratori ko'rsatgan elementni o'chiradi;
- **`erase(iterator it1, iterator it2)`** – [**it1**; **it2**) oraliqdagi elementlarni o'chiradi.

Set turidagi konteynerda biror-bir elementni bor yoki yo'qligini tekshirish uchun **find()** funksiyasi xizmat qiladi:

- **`find(T& v)`** – qiymati **v** bo'lgan element mavjud bo'lsa, uni ko'rsatuvchi iteratorni, aks holda **set** konteyneri oxirini ko'rsatuvchi iteratorni qaytaradi.

Quyidagi dastur ushbu funksiyalar ishini izohlash uchun xizmat qiladi.

Dastur	Natijasi
`#include <iostream>` `#include <set>` `using namespace std;` `int main(){` `  set<int> a;` `  a.insert(1);  a.insert(2);` `  for(auto it = a.begin(); it != a.end();  it ++)` `  { cout << *it << ' ';} cout << endl;` `  cout << ((a.find(2) !=a.end()) ?"bor" :"yo`q") << endl;` ` a.erase(2);` ` cout << ((a.find(2) !=a.end()) ?"bor" :"yo`q") << endl;` `  return 0;` `}`	`1 2` `bor` `yo`q`

Ko‘rsatilgan qiymatdagi elementlar sonini **count()** funksiyasi qaytaradi. Aytib o‘tilganidek, **set** konteynerida bir xil qiymatdagi elementlar bo‘lishi mumkin emas. Shu sababli bu funksiyaning qiymati doimo **0** yoki **1** dan biri bo‘ladi:

- `count(T& v)` – qiymati **v** ga teng bo‘lgan elementlar sonini qaytaradi.

Quyidagi kabi funksiyalar bilan avval ham tanishtirgan edik:

Funksiya	Vazifasi
lower_bound(T& v)	**set** konteynerida qiymati **v** dan katta yoki **v** ga teng bo‘lgan birinchi elementni ko‘rsatuvchi iteratorni qaytaradi
upper_bound(T& v)	**set** konteynerida qiymati **v** dan katta bo‘lgan birinchi elementni ko‘rsatuvchi iteratorni qaytaradi
clear()	**set** konteynerini bo‘shatadi
size()	**set** konteyneri elementlari sonini qaytaradi
empty()	**set** konteyneri bo‘sh bo‘lsa **true**, aks holda **false** qiymatini qaytaradi

Ushbu funksiyalar ishiga doir dastur misoli quyidagicha:

Dastur	Natijasi

`#include <iostream>`
`#include <set>`
`#include <vector>`
`using namespace std;`
`int main(){`
`  set<int> st;`
`  vector<int> a{1, 5, 3, 6, 2};`
`  st.insert(a.begin(), a.end());`
`  cout << "size: " << st.size() << endl;`
`  cout << *st.lower_bound(2) << endl;`
`  cout << *st.upper_bound(2) << endl;`
 	
`  st.clear();`
`  cout << "empty: " << st.empty() << endl;`
`  return 0;`
`}`
	`size: 5`
	`2`
	`3`
	`empty: 1`

Set konteyneri bilan deyarli bir xil vazifani bajaruvchi va **<set>** kutubxonasida e’lon qilingan **multiset** konteyneri ham mavjud bo‘lib, uning **set** konteyneridan yagona farqi, **multiset konteyneri bir xil qiymatli elementlarni saqlashga ruxsat beradi**. Ya’ni **multiset** konteynerida bir xil qiymatdagi bir necha element bo‘lishi ham mumkin. Albatta, **multiset** konteyneri uchun ham **set** konteynerida ko‘rsatib o‘tilgan funksiyalarning barchasi o‘rinlidir.

8-§. MAP KONTEYNERI

Map konteynerini ba'zi jihatlarini **massiv**ga yoki **vector** konteyneriga o'xshatish mumkin, lekin tashkil etilishi ma'nosida esa **set** konteyneriga o'xshash tuzilma bo'lib, yagona farqi **map** konteyneri faqatgina qiymatlarni emas, balki [**kalit**, **qiymat**] juftligini (**pair**) saqlaydi. **Barcha kalitlar bir-biridan farq qilishi shart**. Ushbu konteynerning boshqa bir ko'rinishi bo'lgan **multimap** konteyneri esa bir xil qiymatli kalitlar bo'lishiga yo'l qo'yadi. Bu ikkala konteyner **<map>** kutubxonasida e'lon qilingan. **Map** konteyneri ham **set** konteyneri kabi ko'p holatlarda **Red-Black-Tree** asosida quriladi.

Map konteyneri o'zgaruvchi va o'zgarmaslari quyidagi ko'rinishlarda e'lon qilinadi:

- **`map<TKey, TValue> a;`** – kalit turi **TKey**, qiymat turi **TValue** bo'lgan map turidagi bo'sh **a** o'zgaruvchisini hosil qiladi;
- **`map<TKey, TValue> a(it1, it2)`** – **[it1, it2)** oraliqdagi elementlardan yangi map turidagi **a** o'zgaruvchisini hosil qiladi, bu yerda kalit turi **TKey** va qiymat turi **TValue** iteratorlar ko'rsatayotgan oraliqdagi elementlarning, mos ravishda, kalit va qiymat turlariga mos bo'lishi shart;
- **`map<TKey, TValue> a{{Key1, Value1}, {Key2, Value2}, ...}`** – "{}" orasida berilgan juftliklardan **yangi map** turidagi o'zgaruvchi hosil qiladi. Bunda **KeyI** va **ValueI** ning turlari **TKey** va **TValue** ga mos bo'lishi shart.

Bu e'lonlar uchun quyidagilarni misol qilish mumkin:

```
map<int, string> a;
map<int, bool> b{
{1, false},
{0, true}
};
map<int, bool> c(b.begin(), b.end());
```

Avval ko'rilgan konteynerlar kabi **map** konteyneri ham **clear()**, **size()**, **empty()** a'zo funksiyalarini taqdim etadi. Quyidagi dasturda shu a'zo funksiyalarning ishini kuzatishimiz mumkin:

Dastur	Natijasi

`#include <iostream>`
`#include <map>`
`#include <ios>`
`using namespace std;`
`int main(){`
`  map<int, string> a;`
`  map<int, bool> b{`
	`b.size() = 2`
	`a.empty() = true`
	`c.size() = 0`

```
        {1, false}, {0, true}   };
    map<int, bool> c(b.begin(), b.end());
    cout << "b.size() = " << b.size() << endl;
    cout << boolalpha;
    cout << "a.empty() = " << a.empty() << endl;
    c.clear();
    cout << "c.size() = " << c.size() << endl;
    return 0;
}
```

Konteynerning **key** kalitli elementiga murojaat qilish uchun **[ ]** operatoridan foydalanish mumkin. Bundan tashqari, bu operator yordamida ushbu element qiymatini o‘zgartirish va, umuman, agar bunday element mavjud bo‘lmasa, yangi element qo‘shish uchun ham foydalanish mumkin. Misol uchun, **a – map<int, int>** turidagi o‘zgaruvchi bo‘lsin. U holda:

a[1] = 2 – kaliti **1** ga teng bo‘lgan element qiymatini **2** ga o‘zgartiradi va agar **a** konteynerida bunday element mavjud bo‘lmasa, xuddi shu kalit va qiymat juftligidan yangi element hosil qiladi;

a[1] – bu esa kaliti **1** ga teng bo‘lgan elementga murojaat qaytaradi.

Quyidagi misol shu operator ishini izohlaydi.

Dastur	Natijasi

`#include <iostream>`
`#include <map>`
`using namespace std;`
`int main(){`
`  map<string, int> a{};`
`  a["dushanba"] = 1;`
`  cout << "Dushanba: " << a["dushanba"] << endl;`
`  cout << "size: " << a.size() << endl;`
`  a["seshanba"]; //yangi element hosil qiladi`
`  cout << "Seshanba: " << a["seshanba"] << endl;`
`  cout << "size: " << a.size() << endl;`
`  return 0;`
`}`
	`Dushanba: 1`
	`size: 1`
	`Seshanba: 0`
	`size: 2`

Shunga o‘xshash vazifani **at()** a’zo funksiyasi ham bajaradi. Ularning farqli tomoni, **at()** a’zo funksiyasiga argument sifatida berilgan kalit konteynerda albatta bo‘lishi shart. Bundan kelib chiqadiki, **at()**funksiyasi yordamida konteynerning biror-bir elementiga faqatgina **murojaat** qilish mumkin (qiymatini olish yoki o‘zgartirish), lekin yangi element qo‘shish mumkin emas. Bu a’zo funksiyasi ishini quyidagi dasturdan tushunish mumkin.

Dastur	Natijasi

#include <iostream>
#include <map>
#include <ios>
using namespace std;
int main(){
map<int, bool> a{};
a[1] = true; a[0] = false;
cout << boolalpha;
cout << "a[1] = " << a.at(1) << endl;
a.at(1) = false;
cout << "a[1] = " << a.at(1) << endl;
return 0;
}
	a[1] = true
	a[1] = false

Yuqoridagi ikki usuldan farqli ravishda **insert()** a'zo funksiyasi ham konteynerga yangi element qo'shishda foydalaniladi. Bu holda yangi element **kalit** va **qiymat** juftligi funksiyaga **pair** turida beriladi. Agar konteynerda yangi qo'shilayotgan element kalitiga teng bo'lgan kalitli element mavjud bo'lsa, u holda **map** konteyneriga yangi element qo'shilmaydi. **Multimap** konteyneri esa bu holatda yangi elementni konteynerga qo'shadi va natijada bir xil kalitli ikkita element hosil bo'ladi. Bu a'zo funksiyani quyidagi ko'rinishlarda yozish mumkin:

- **pair<iterator, bool> insert(pair<TKey, TValue>& elem)** – yangi elem elementini konteynerga qo'shadi. Agar bu amal muvaffaqiyatli yakunlansa, u holda yangi qo'shilgan elementni ko'rsatuvchi va **true** qiymatlaridan iborat **pair** turidagi qiymat qaytaradi, aks holda bu elementni qo'shishga xalal berayotgan elementni ko'rsatuvchi iterator va **false** qiymatlaridan iborat **pair** turidagi qiymat qaytaradi;
- `void insert(iterator it1, iterator it2)` – [it1, it2) iteratorlari orasidagi elementlarni birma-bir konteynerga qo'shishga harakat qiladi, qo'shishning iloji bo'lmagan elementlar "tashlab" ketiladi.

Masalan:

Dastur	Natijasi
#include <iostream> #include <map> #include <vector> using namespace std;	Key: 3 Value: 3.6 Muvaffaqiyatli: true a konteyneri elementlari

```
int main(){
    map<int, double> a{};
    a.insert(make_pair(1, 1.2));
    a.insert({2, 2.4});
    auto p = a.insert({3, 3.6});
    cout << "Key: " << p.first->first << endl;
    cout << "Value: " << p.first->second << endl;
    cout << "Muvaffaqiyatli: " << boolalpha;
    cout << p.second << endl;
    vector<pair<int, double> >;
    vek = {{4, 4.8}, {5, 6}};
    a.insert(vek.begin(), vek.end());
    cout <<"a konteyneri elementlari ro`yxati: ";
    cout << endl;
    for(auto& x: a){
       cout << x.first <<" "<< x.second << endl;
    }
    return 0;
}
```

```
ro`yxati:
1 1.2
2 2.4
3 3.6
4 4.8
5 6
```

Yuqoridagi dasturdan ko‘rish mumkinki, **vector** konteyneri singari **map** konteyneri elementlariga ketma-ket murojaat qilish uchun **for** operatorining boshqa ko‘rinishidan foydalanish mumkin. Bunda **map** konteyneri elementlari [**key**, **value**] ko‘rinishidagi **pair** turidagi qiymatlar ekanligini esdan chiqarmaslik kerak.

Bundan tashqari barcha konteynerlardagi kabi **map** konteyneri elementlariga iteratorlar orqali murojaat qilish ham mumkin. Bu imkoniyatni yuqoridagi va quyidagi dasturlarda ko‘rish mumkin.

Dastur	Natijasi

`#include <iostream>`
`#include <map>`
`using namespace std;`
`int main(){`
`  map<string, string> a{`
`  {"O'zbekiston", "Toshkent"},`
`  {"Rossiya", "Moskva"},`
`  {"Buyuk Britaniya", "London"}`
`  };`
`  cout << "Iterator orqali: " << endl;`
	`Iterator orqali:`
	`Buyuk Britaniya - London`
	`O'zbekiston - Toshkent`
	`Rossiya - Moskva`
	
	`for operatori orqali:`
	`Buyuk Britaniya - London`
	`O‘zbekiston - Toshkent`
	`Rossiya - Moskva`

```
    for(auto it=a.begin(); it != a.end();it++){
        cout << it->first << " - "
             << it->second << endl;
    }
    cout << endl;
    cout << "for operatori orqali:" << endl;
    for(auto& elem : a){
        cout << elem.first << " - "
             << elem.second << endl;
    }
    return 0;
}
```

Konteynerdan elementlarni o‘chirish **erase()** a’zo funksiyasi yordamida amalga oshiriladi:

- **`iterator erase(iterator pos)`** – **pos** iteratori ko‘rsatgan elementni o‘chiradi va undan keyin kelgan elementni ko‘rsatuvchi iteratorni qaytaradi;
- **`iterator erase(iterator it1, iterator it2)`** – [**it1**, **it2**) oraliqdagi elementlarni o‘chiradi va eng so‘nggi o‘chirilgan elementdan keyin kelgan elementni ko‘rsatuvchi iteratorni qaytaradi;
- **`size_t erase(TKey& key)`** – kaliti **key** ga teng bo‘lgan elementlarni o‘chiradi va o‘chirilgan elementlar sonini qaytaradi.

Eslatib o‘tamiz, multimap konteynerida bir xil kalitli bir nechta element mavjud bo‘lishi mumkin.

Bu a’zo funksiyasini qo‘llashga doir namunaviy dastur quyidagicha:

Dastur	Natijasi

`#include <iostream>`
`#include <map>`
`#include <vector>`
`using namespace std;`
`int main(){`
`  vector<pair<int, int> > vek1 =`
`  {{1, 2}, {2, 3}, {3, 4}};`
`  vector<pair<int, int> > vek2 =`
`  {{1, 2}, {1, 3}, {3, 4}};`
`  map<int,int> a(vek1.begin(),vek1.end());`
`  multimap<int,int>b(vek2.begin(),vek2.end());`
`  cout <<"a ning elementlari: "<<endl;`
	`a ning elementlari:`
	`1 2`
	`2 3`
	`3 4`
	`b ning elementlari:`
	`1 2`
	`1 3`
	`3 4`
	`keyingi element: 2`
	`size: 2`
	`b.erase(1) dan so'ng:`
	`3 4`

```
    for(auto& x : a){
        cout << x.first  << ' ';
        cout << x.second << endl;
    }
    cout << "b ning elementlari: "
        << endl;
    for(auto& x : b){
        cout << x.first << ' ' <<
            x.second << endl;
    }

    auto it = a.erase(a.begin());
    cout << "keyingi element: ";
    cout << it->first << endl;
    cout << ", size: "<< a.size() <<endl;

    size_t cnt = b.erase(1);
    cout << "b.erase(1) dan so'ng:" << endl;
    for(auto& x : b){
        cout << x.first << ' ';
        cout << x.second << endl;
    }

    a.erase(a.begin(), a.end());
    cout << boolalpha << a.empty() << endl;
    return 0;
}
```

```
true
```

Kaliti **key** ga teng bo‘lgan elementlar sonini aniqlashda **count()** funksiyasidan foydalaniladi:

```
size_t count(TKey& key)
```

Agar kaliti **key** ga teng bo‘lgan element qidirilayotgan bo‘lsa, u holda quyidagi funksiya shu elementni ko‘rsatuvchi iteratorni qaytaradi:

```
iterator find(TKey& key)
```

Agar **multimap** konteynerida kaliti **key** ga teng bo‘lgan bir nechta element mavjud bo‘lsa, u holda bu funksiya ularning istalgan birini ko‘rsatuvchi iteratorni qaytaradi. Quyida ushbu funksiyalar ishiga oid misol keltirilgan.

Dastur	Natijasi

#include <iostream>
#include <map>
using namespace std;
int main(){
multimap<int,int> a{{1, 2},{1, 3},{2, 4}};
cout << "count(1): "<<a.count(1) << endl;
auto it = a.find(1);
cout << "find(1): " << endl;
cout << it->first << " - " << it->second;
return 0;
}
	count(1): 2
	find(1):
	1 - 2

Multimap konteynerida kaliti **key** ga teng bo'lgan elementlar ko'p bo'lishi mumkinligi aytilgan edi. Shu sababli, bu konteynerda bitta kalitga mos kelgan barcha elementlarga murojaat qilish uchun **equal_range()** a'zo funksiyasidan foydalaniladi:

```
pair<iterator, iterator> equal_range(TKey&, key)
```

Bu funksiya kaliti **key** ga teng bo'lgan barcha elementlarni topadi va ularning boshi va oxirini ko'rsatuvchi iteratorlar juftligi (**pair**) ni qaytaradi. Buni dastur misolida ko'rish mumkin.

Dastur	Natijasi

#include <iostream>
#include <map>
using namespace std;
int main(){
multimap<string, string> shahar{
{"O'zbekiston", "Toshkent"},
{"O'zbekiston", "Buxoro"},
{"O'zbekiston", "Samarqand"},
{"Angliya", "London"},
{"Angliya", "Manchester"}
};
auto rg = shahar.equal_range("O'zbekiston");
cout << "O'zbekistondagi shaharlar: "<<endl;
for(auto it= rg.first;it !=rg.second;it++){
cout << it->second << endl;
} cout << endl;
cout << "Angliyadagi shaharlar: " << endl;
	O'zbekistondagi
	shaharlar:
	Toshkent
	Buxoro
	Samarqand
	
	Angliyadagi
	shaharlar:
	London
	Manchester

```
    rg = shahar.equal_range("Angliya");
    for(auto it= rg.first;it !=rg.second;it++){
        cout << it->second << endl;
    }
    return 0;
}
```

Set konteynerida bo‘lgani kabi **map** konteyneri ham lower_bound() va upper_bound() a’zo funksiyalarini taqdim etadi:

- **`iterator lower_bound(TKey& key)`** - kaliti **key** dan kichik bo‘lmagan birinchi elementni ko‘rsatuvchi iteratorni qaytaradi. Agar bunday element mavjud bo‘lmasa, u holda konteyner oxirini ko‘rsatuvchi iterator (**end()**) ni qaytaradi;
- **`iterator upper_bound(TKey& key)`** - kaliti **key** dan katta bo‘lgan birinchi elementni ko‘rsatuvchi iteratorni qaytaradi. Agar bunday element mavjud bo‘lmasa, u holda konteyner oxirini ko‘rsatuvchi iterator (**end()**) ni qaytaradi.

Bu funksiyalar bajaradigan ish quyidagi dasturda yoritilgan.

Dastur	Natijasi

`#include <iostream>`
`#include <map>`
`using namespace std;`
`int main(){`
`  multimap<int, int> a{`
`  {1, 2}, {2, 3},`
`  {2, 3}, {4, 5}`
`  };`
`  auto it1 = a.lower_bound(2);`
`  cout << it1->first << " - " << it1->second << endl;`
`  auto it2 = a.upper_bound(2);`
`  cout << it2->first << " - " << it2->second << endl;`
`  return 0;`
`}`
	`2 - 3`
	`4 - 5`

TAVSIYA: Albatta, qo‘llanma C++ tili haqidagi barcha ma’lumot yoki imkoniyatlarni, masalalar yechish algoritmlarini qamrab olmagan. Buning iloji ham yo‘q. Shu sababli o‘quvchiga bilimni kengaytirish va chuqurlashtirish uchun turli qo‘shimcha adabiyotlar o‘qishni maslahat beramiz!

9-§. MASALALAR YECHISH NAMUNALARI

Yuqorida ta'kidlab o'tilganidek, masala yechishda qo'llanayotgan konteynerlar faqat masala mazmuniga emas, balki dasturchining xohishi va imkoniyatlariga ham bog'liq. Bir masalani bir dasturchi massiv yordamida yechsa, boshqasi vector yordamida yechishi mumkin. Bunda asosiy muammo qo'llanayotgan tur yoki konteynerga bog'liq amallar masala shartida belgilangan vaqt chegarasida bajarib bo'linishidir. Bu muammo dastur ishlash vaqti chegaralanmagan masalalarda emas, balki nufuzli musobaqalarda dastur ishlash vaqti chegaralangan bo'lgani uchun aksariyat hollarda ko'zga tashlanadi. Qo'llanmaning maqsadidan kelib chiqib, qulay va sodda usuli mavjud bo'lgan masalaning yechimini ham biror usul yoki funksiya yoki tur xossalarini bayon etish uchun nisbatan murakkab usulda taklif qilishimiz mumkin.

Namuna 1. Yangi ishlab chiqilgan BRA-OS operatsion sistemasida juda qulay imkoniyat kiritildi – katalogga yo'lni ko'rsatishda ikkita katalog o'rtasida ajratuvchi sifatida yagona slesh (/) emas, istalgan sonda ishlatish mumkin bo'ldi. Masalan, //usr///local//nginx/sbin// va /usr/local/nginx///sbin yozuvlari teng kuchli bo'lib qoldi. Avvalgi operatsion sistemalarda katalogga yo'l yozuvidagi '/' belgisi (yoki bir nechta shunday belgi) ildiz katalogga yo'lni yozishda zarur bo'lib, bittasi yetarli edi. Yo'lning yozuvini muvozanatda deymiz, agar yo'l yozuvida eng kam sondagi kerakli '/' belgisi bo'lsa. Berilgan S satrda ($|S|\le 2\cdot 10^5$) katalogga yo'l yozilgan. Shu yozuvni muvozanatga keltiring.

K:	//usr///local//nginx/sbin	////a//b/////g	/a/a/a/a/a/a/a	////	/a//aa/a//
CH:	/usr/local/nginx/sbin	/a/b/g	/a/a/a/a/a/a/a	/	/a/aa/a

Yechish. Masalani turli usullar bilan yechish mumkin, ularning ba'zilari quyidagicha:

1-usul. Berilgan **S** satr o'qib olingach **find()** a'zo funksiyasi yordamida birinchi uchragan "//" satr o'rni aniqlanadi va shu o'rindagi '/' belgisi **erase()** a'zo funksiyasi yordamida o'chiriladi. Bu vazifani bajartirish uchun quyidagi dastur lavhasini yozamiz:

```
while(s.find("//")!=string::npos)
        {t=s.find("//"); s.erase(t,1);}
```

Dasturning bu qismi bajarilgandan keyin satrda ikki katalog orasidagi ajratuvchi '/' belgisi faqat bittadan qoladi. Endi 5-testdagi kabi holat, ya'ni oxirgi belgi '/' bo'lib qolishi mumkinligini e'tiborga olish zarur. Buning uchun quyidagi dastur lavhasini yozamiz:

```
if ((s[s.size()-1]=='/')&&(s!="/")) s.erase(s.size()-1,1);
```

Endi shu ikki lavhada bajarilayotgan amallar hisobini olishga harakat qilamiz. Ikkinchi lavha **1** ta (juft) tekshirish orqali bajariladi. Lekin birinchi lavhadagi **find()** a'zo funksiyasi satrni bittalab qarab chiqishini e'tiborga olsak, u holda 1-test uchun **find()** a'zo funksiyasining

o‘zi “//” satri o‘rinlarini aniqlashga **1+5+5+9=20** va “//” satri qolmagan bo‘lsa ham, S satrni oxirigacha qarab chiqish uchun yana **20** ta, jami **41** ta tekshirish bajaradi. Agar 1-testdagi satr uzunligi 25 ga teng ekanligini hisobga olsak, bu juda ko‘p.

2-usul. Berilgan **S** satr o‘qib olingach satrni 0- va1-belgilarini, 1- va 2-belgilarini va shu kabi satr oxirigacha ‘/’ belgiga tengligini tekshiramiz. Agar biror belgilar juftligida ikkalasi shu belgiga teng bo‘lsa, u holda ikkinchi o‘rindagi ‘/’ belgisi **erase()** a’zo funksiyasi yordamida o‘chiriladi hamda yana shu o‘rindagi belgilar juftligi tekshiriladi. Bunda ko‘pi bilan **S.size()-1** ta tekshirish bajariladi. Agar 1-testdagi satr uzunligi 25 ga teng ekanligini hisobga olsak, u holda tekshirishlar soni bor-yo‘g‘i **25-1=24** dan kichik bo‘ladi (chunki kamida **1** ta o‘chirish bajariladi). Shu vazifani bajaradigan dastur lavhasi quyidagicha bo‘ladi:

```
for(t=0;t<s.size()-1;t++)
        if((s[t]=='/')&&(s[t+1]=='/')){s.erase(t+1,1);t--;}
```

Agar oxirgi belgi ‘/’ bo‘lib qolishi mumkinligini tekshiruvchi yuqoridagi lavha bilan birlashtirsak, 1-usul dasturiga nisbatan tezroq ishlaydigan dasturga ega bo‘lamiz.

Namuna 2. Sizga N ($1<N\le 2\cdot 10^5$) xonadan iborat o‘nlik sanoq sistemasidagi S natural son berilgan bo‘lib, bu sonning har bir raqami 0 yoki 1 ga teng. Sizga yana butun X va Y ($0\le Y<X<N$) sonlari berilgan. Natural S soni ustida quyidagi amallarni istalgancha (balki 0 ta) bajarish mumkin: 0 raqamini 1 ga yoki 1 raqamini 0 ga almashtirish mumkin. Eng kamida nechta almashtirish bajarilganda S sonini 10^X bo‘lgandagi qoldiq 10^Y ga teng bo‘ladi?

K:	11 5 2 11010100101	11 5 1 11010100101	2 1 0 10	7 5 2 1000100	4 2 1 1011	13 10 0 1000001101100
CH:	1	3	1	0	1	5

Yechish. Masaladagi S soni juda uzun bo‘lgani uchun massiv yoki konteyner yordamida o‘qib olinishi kerakligini tushunish mumkin. Masala dasturini sodda va tushunishga oson ko‘rinishda yozishni maqsad qilib olamiz. Shu sababli avval masalaning shartini tahlil qilamiz.

Masala shartidan ko‘rinadiki, qoldiq uchun oxirgi X ta raqamni qarash yetarli ekan, demak, oldinda turgan (N-X) ta raqamni o‘chirish kerak bo‘ladi. Bu amalni **string** yoki **vector** konteynerlarida bajarish qulay, chunki ikkala konteynerda ham **erase()** a’zo funksiyasini qo‘llash mumkin. Masala yechimi o‘chirishdan so‘ng qolgan sonning Y-o‘rindagi raqami 1 ga teng, qolganlari 0 ga teng bo‘lishi kerakligiga asoslanadi. Yechimda: Y-o‘rindagi raqami 1 ga teng bo‘lsa – uni o‘chiramiz, agar Y-o‘rindagi raqami 0 ga teng bo‘lsa – sanagichimizni bittaga oshiramiz. Endi faqat qolgan sondagi 1 raqamlari sonini hisoblash qoldi, xolos. Dasturi quyidagicha:

1-usul. **S** soni satr sifatida o‘qib olingan holda:

Dastur String

```
#include <iostream>
#include <algorithm>
using namespace std;

int main(){
    int n,x,y,t,m=0;
    string s;
    cin>>n>>x>>y>>s;
    s.erase(0,n-x);
    if(s[x-y-1]=='1')s[x-y-1]='0';
        else m++;
    m=m+count(s.begin(), s.end(), '1');
    cout<<m;
    return 0;
}
```

2-usul. S soni **vector** sifatida o‘qib olingan holda:

Dastur Vector

```
#include <iostream>
#include <algorithm>
#include < vector >
using namespace std;

int main(){
    int n,x,y,t,m=0;
    char h;
    vector<char>s;
    cin>>n>>x>>y;
    for(t=0;t<n;t++){cin>>h;s.push_back(h);}
    s.erase(s.begin(), s.begin()+n-x);
    if(s[x-y-1]=='1')s[x-y-1]='0';
        else m++;
    m=m+count(s.begin(), s.end(), '1');
    cout<<m;
    return 0;
}
```

Dasturda o‘chirish bajarmasdan ham (amallar soni kamayadi) masalani yechish mumkin. Buni mustaqil vazifa sifatida qoldiramiz.

Namuna 3. Faqat yuqori registrdagi lotin hafrlaridan iborat S (|S|<101) satr berilgan. Satrdagi harflar tartibini buzmasdan nechta usulda "QAQ" satri belgilarini tanlab olish mumkinligini aniqlang. Masalan, S= "QAQAQ" bo'lsa, javob 4 ta, ya'ni "**QAQ**AQ", "**QA**QAQ", "**Q**AQ**AQ**", "QA**QAQ**".

K:	QAQAQYSYIOIWIN	QAQQQZZYNOIWIN	QA	RQAWNACASAAKAGAAAAQ
CH:	4	3	0	10

Yechish. Masaladagi S satrni avvalgi masaladagi kabi string va vector konteyneri yordamida o'qib olish mumkin. Masala shartidan shunday xulosa qilamiz:

- 1-qadamda avval birinchi kelgan 'Q' harfi o'rnini aniqlaymiz, keyin undan keyin kelgan birinchi 'A' harfi o'rnini aniqlaymiz, so'ng 'A' harfidan keyin kelgan barcha 'Q' harflarini sanab chiqamiz, bunda agar birortasi topilmasa, sanagich qiymati 0 bo'lgancha qoladi;
- 2-qadamda birinchi kelgan 'Q' harfidan keyin kelgan ikkinchi 'A' harfi o'rnini aniqlaymiz, so'ng 'A' harfidan keyin kelgan barcha 'Q' harflarini sanab chiqamiz;
- ..., birinchi kelgan 'Q' harfidan keyin kelgan oxirgi 'A' harfi o'rnini aniqlaymiz, so'ng 'A' harfidan keyin kelgan barcha 'Q' harflarini sanab chiqamiz;
- ..., ikkinchi kelgan 'Q' harfi o'rnini aniqlaymiz, keyin undan keyin kelgan birinchi 'A' harfi o'rnini aniqlaymiz, so'ng 'A' harfidan keyin kelgan barcha 'Q' harflarini sanab chiqamiz;
- va hokazo.

Bundan ko'rinadiki, har bir harf uchun bitta **for** operatori, ya'ni 3 ta **for** operatori kerak ekan. Bu masala dasturida S satrni **string** konteyneriga o'qib olish qulay, chunki satrdagi belgilar soni avvaldan berilmagan.

Dastur String

```
#include <iostream>
using namespace std;

int main(){
    int t,k,m,uz,soni=0;
    string s;
    cin>>s;
      uz=s.size();
      for(t=0;t<uz;t++)
            for(k=t+1;k<uz;k++)
                  for(m=k+1;m<uz;m++)
                        if(s[t]=='Q'&&s[k]=='A'&&s[m]=='Q')soni++;
      cout<<soni;
    return 0;
}
```

Agar kiruvchi ma'lumot **vector**ga biror usul bilan o'qib olinsa, dasturning qolgan qismi o'zgarishsiz qoladi.

Namuna 4. Natural N (N<101) ta haddan iborat A1, A2, ..., An ketma-ketlik berilgan. Ketma-ketlik hadlari natural sonlar bo'lib, qiymati 100 dan ortmaydi. Biror butun D (D≥0) son tanlab ketma-ketlikning ixtiyoriy hadi ustida quyidagi amallardan birini bajarish mumkin:

- hadni o'zgarishsiz qoldirish;
- haddan D ni (faqat bir marta) ayirish: Ak=Ak-D;
- hadga D ni (faqat bir marta) qo'shish: Ak=Ak+D.

Shunday eng kichik D sonni aniqlash kerakki, har bir had ustida yuqoridagi amallardan birini bajarib ketma-ketlik hadlarini tenglashtirish kerak, ya'ni A1=A2=...=An. Agar shunday son mavjud bo'lsa D sonini, aks holda -1 ni chiqaring. Masalan, A1=2 va A2=8 ketma-ketlik uchun D=3 eng kichik shunday son bo'ladi, chunki A1=A1+3=5 va A2=A2-3=5 bo'ladi. A1=1, A2=4, A3=7, A4=7 ketma-ketlik uchun ham D=3 mavjud.

K:	6 1 4 4 7 4 1	5 2 2 5 2 5	4 1 3 3 7	2 2 8	4 8 2 8 5	5 1 2 2 1 1	3 1 2 4
CH:	3	3	-1	3	3	1	-1

Yechish. Masala shartini tahlil etib, agar ketma-ketlik har xil sonlar soni 3 tadan ortiq bo'lsa, D soni mavjud emasligini aniqlaymiz. Chunki shartda berilgan amallar 3 ta, ya'ni qo'shish yoki ayirish yoki o'zgarishsiz qoldirishdir. Ketma-ketlikni o'sish yo'nalishida tartiblaymiz. Shundan keyin quyidagi hollar bo'ladi:

1. har xil sonlar soni 3 tadan ortiq, javob -1;
2. har xil sonlar soni 3 ta va:
   a) A3-A2=A2-A1 bo'lsa, u holda javob D=A2-A1;
   b) A3-A2≠A2-A1 bo'lsa, u holda javob -1;
3. har xil sonlar soni 2 ta va:
   a) A2-A1 juft son bo'lsa, u holda javob D=(A2-A1)/2;
   b) A2-A1 toq son bo'lsa, u holda javob D=A2-A1;
4. har xil sonlar soni 1 ta bo'lsa D=0.

Tahlildan ko'rinadiki, ketma-ketlik uchun dasturda **vector** konteyneridan foydalanish (**unique()** funksiyasi borligi uchun) qulay ekan. Endi yuqorida aytilganlarni birma-bir tartib bilan yozib chiqamiz. Bunda dastur matni tushunarli va sodda bo'lishi uchun C++ tili beradigan ba'zi imkoniyatlardan foydalanamiz hamda hadlar 0 dan boshlanishini e'tiborga olamiz.

Dastur Vector

```
#include <iostream>
#include <algorithm>
#include <vector>
using namespace std;

int main(){
    int n,t;
    cin>>n;
    vector<int> a(n);
    for (t=0; t<=n-1; t++)cin>>a[t];
    sort(a.begin(), a.end());
    auto ox=unique(a.begin(), a.end());//ox - turli elementlar
    a.erase(ox,a.end());//joylashgan oxirgi iteratordan keyingisi
    if(a.size()>=4)return cout<<-1,0;//dastur ishini tugatadi
    if(a.size()==1)return cout<<0,0; //dastur ishini tugatadi
    if(a.size()==2){
        if((a[1]-a[0])%2==0)cout<<(a[1]-a[0])/2;
            else cout<<a[1]-a[0];
    return 0;  //dastur ishini tugatadi
    }
    if(a[2]-a[1]==a[1]-a[0])cout<<a[1]-a[0];
            else cout<<-1;
      return 0;
}
```

Namuna 5. Ali, Vali va Soli vertikal N ta ($3 \leq N \leq 10^5$) taxtadan yig'ilgan devorni bo'yashi kerak. Avval Ali A[k]=3·k+1 (A[k] ≤ N bo'lgan barcha natural k uchun) tartib raqamli taxtalarni bo'yadi, keyin esa Vali B[m]=2·m+1 (B[m] ≤ N bo'lgan barcha natural m uchun) tartib raqamli taxtalarni bo'yadi, lekin Ali bo'yagan taxtalarni bo'yashi shart emas edi. Devorning bo'yalmay qolgan taxtalarini Soli bo'yashi kerak. N soni berilganda Soli bo'yagan taxtalar sonini aniqlang.

K:	100	3	100000	98765	15	81	51	56789
CH:	34	2	33334	32922	6	28	18	18930

Yechish. Masaladan Ali va Vali bo'yashi kerak bo'lgan taxtalarning ba'zilari bir xil tartib raqamga egaligini ko'rish mumkin, masalan, k=2 va m=3 da A[2]=B[3]=7 bo'ladi. Shu sababli Ali va Vali bo'yagan taxtalar sonini 3·k+1≤N va 2·m+1≤N tengsizliklardan aniqlash hamda ularning yig'indisini N sonidan ayirish amali masala yechimini bermaydi. Yechimga barcha bo'yalgan taxtalar sonini aniqlashda (ikkita bir xil sonni saqlamaydigan) **set** kontey-

neridan foydalanish va N sonidan shu konteyner o'lchamini ayirish orqali kelish mumkin. Dasturda $3 \cdot k+1 \leq N$ va $2 \cdot m+1 \leq N$ tengsizliklarni bajaruvchi sonlarni aniqlash uchun **while** operatoridan foydalanamiz.

Dastur Set

```
#include <iostream>
#include <set>
using namespace std;
int main(){
    int n,t;
    set<int>st;
    cin>>n;
    t=1;
    while(3*t+1<=n){
        st.insert(3*t+1);
        t++;
    }
    t=1;
    while(2*t+1<=n){
        st.insert(2*t+1);
        t++;
    }
    cout<<n-st.size();
    return 0;
}
```

Namuna 6. Birinchi satrda N ($0<N<10^6$) natural soni va keyingi N ta satrda [-15000; 15000] oralig'ida yotgan butun sonlar berilgan. Shu sonlarni kamaymaydigan tartibda chiqaring.

K:	6	6	6	6	6
	10	-15000	-11111	-15000	1
	6	-1	-1000	15000	1
	5	-15000	-15000	-15000	2
	1	15000	15000	15000	2
	3	0	11111	-15000	3
	1	0	0	15	4
CH:	1	-15000	-15000	-15000	1
	1	-15000	-11111	-15000	1
	3	-1	-1000	-15000	2
	5	0	0	15	2
	6	0	11111	15000	3
	10	15000	15000	15000	4

Yechish. Hadlari soni yetarlicha ko'p bo'lgan ketma-ketlikni tartiblash va chiqarish dastur ishiga ajratilgan vaqt bo'yicha chegaralashga ham bog'liq. Shu sababli vaqt bo'yicha chegaralashga ega masalalar dasturida turli usullar bilan vaqtdan "yutishga" harakat qilinadi. Bu masala yechimida ham tartiblashga ketadigan vaqtni tejash mumkin. Masala shartiga e'tibor berilsa, hadlari 1000000 ta bo'lgan ketma-ketlikda turli qiymatlar soni bor-yo'g'i 30001 ta bo'lishi mumkin ekan. Demak, har bir qiymatning ketma-ketlikda ishtirok etishlari soni sanab olsak va qiymatlar bo'yicha chiqarishni tashkil etsak bo'ladi. Ketma-ketlikda manfiy qiymatlar borligi va massiv yoki **vector**da indekslar nomanfiy butun son bo'lishi shartligini hisobga olsak, u holda bu ikki turdan foydalana olmaymiz. Lekin **map** konteyneridan foydalanishimiz mumkin, chunki unda kalitlar manfiy bo'la oladi. **Map** konteynerining bu masalada foyda beradigan yana bir xususiyati – **map** konteyneri kalit bo'yicha tartiblab oladi. Shu xususiyatlardan foydalanib quyidagi sodda ko'rinishdagi dasturni tuzamiz.

Dastur Map

```
#include <iostream>
#include <map>
using namespace std;

int main(){
    map<int,int> mp;
    int n,t,m;
    cin>>n;
    for(t=1;t<=n;t++){
        cin>>m; mp[m]++;
    }
    for(auto it=mp.begin();it != mp.end(); it++){
        t=(it->second);
        while(t>0){cout<<it->first<<endl;t--;}
    }
    return 0;
}
```

10-§. SATRLARGA DOIR VAZIFALAR

Quyida satrlarga oid turli mazmundagi vazifalar tavsiya etilmoqda. Vazifalar **string** turning a'zo funksiyalaridan foydalanib yoki boshqa usullarda bajarilishi ham mumkin. Agar boshqa talab qo'yilmagan bo'lsa, u holda hosil qilingan yoki qayta ishlangan satrlar chiqarilishi kerak.

= A =

Satr 1. Berilgan S matnning uzunligini ifodalovchi Uz qiymatini, [Uz/2]-belgisini, shu belgi va birinchi belgi o'nlik ASCII kodlarini chiqaring.

Satr 2. S matn, N va M natural sonlar berilgan. S matnning birinchi N ta belgili va oxirgi M ta belgili qismi nusxalarini alohida satrlarda chiqaring.

Satr 3. Berilgan S matnning (satr boshidan hisoblaganda) toq o'rinlardagi belgilarini va satrdagi indekslarini yonma-yon chiqaring. Masalan, S="dastur" bo'lsa, u holda javob "d0s2u4".

Satr 4. Berilgan S matnning yuqori va quyi registrdagi belgilarini alohida satrlarda chiqaring.

Satr 5. Berilgan S matnda (satr boshidan hisoblaganda) birinchi marta uchragan 'A' harfi o'rnini, bunday belgi bo'lmasa '?' belgisini chiqaring.

Satr 6. S matn faqat raqam va lotin harflaridan iborat. Matndagi belgilarni va o'ziga mos "raqam" va "harf" izohi bilan alohida satrlarda chiqaring. Masalan, S="A21" bo'lsa, u holda javob "A-harf", "2-raqam" va "1-raqam".

Satr 7. S satrga "21 19 11 10 5" matnini shu ko'rinishda kiriting va chiqaring. So'ng matnni tashkil etgan sonlarni alohida satrlarda chiqaring.

Satr 8. Berilgan S matnga teskari tarkibli (ya'ni belgilari teskari tartibda joylashtirilgan) matnni chiqaring. Masalan, S="dastur" bo'lsa, u holda javob "rutsad".

Satr 9. S va S1 matnlar berilgan. S1 matn va S1 matnga teskari tarkibli matnlarni S satrda birinchi marta uchragan pozitsiyalari bilan birlashtirib alohida satrlarda chiqaring. Agar bunday qism satrlarning birortasi bo'lmasa, shu satrni pozitsiyasi o'rniga '!' belgisini chiqaring. Masalan, S="dastur" va S1="ast"bo'lsa, u holda javob "ast1" va "tsa!".

Satr 10. Berilgan S matnning belgilari orasiga orachiq (probel) belgisi joylashtirib hosil qilingan satrni chiqaring. Masalan, S="dastur" bo'lsa, u holda javob "d a s t u r".

Satr 11. N natural son va X, Y, Z belgilar berilgan. Uzunligi N ga teng bo'lguncha S satrga X, Y va Z belgilarni shu tartibni saqlagan holda joylashtirib chiqaring. Masalan, N=11, X='j', X='r', Y='d' bo'lsa, u holda javob "jrdjrdjrdjr".

Satr 12. S1 va S2 satrlar berilgan. Katta satrni chiqaring. Agar satrlar teng bo'lsa, u holda "Equal" matnini chiqaring. Masalan, agar S1="BA" va S2="AAA" bo'lsa, u holda javob "AAA", agar S1="ABc" va S2="ABC" bo'lsa, u holda javob "ABc" bo'ladi.

Satr 13. S1 va S2 matnlar berilgan. S1 matn tarkibida S2 matnning uchrashlar sonini chiqaring. Masalan, S1="matematika" va S2="at"bo'lsa, u holda javob 2 bo'ladi.

Satr 14. N natural son va S matn berilgan. S satrning ketma-ket joylashgan belgilari soni N ta bo'lgan barcha qism satrlarini chiqaring. Masalan, N=3 va S="dastur" bo'lsa, u holda javobga "das", "ast", "stu" va "tur" chiqariladi.

Satr 15. S satr berilgan. Satr belgilaridan "ona" so'zini hosil qilish mumkin bo'lsa "Ha", aks holda "Yo'q" chiqaring.

Satr 16. N natural son va S[1..N] satrli massiv berilgan. Massiv elementlari ichidan "m" harfidan boshlanadiganlarini ekranga chiqaring.

Satr 17. N natural son va S[1..N] satrli massiv berilgan. Massiv elementlari ichidan eng qisqa satrni va uning uzunligini ekranga chiqaring. Masalan, N=3, S[1]= "Informatika", S[2]= "DASTUR" va S[3]= "kompyuter" bo'lsa, u holda javobga "DASTUR" va 6 chiqariladi.

Satr 18. N natural son va S[1..N] satrli massiv berilgan. Massiv elementlari ichidan birinchi va oxirgi belgilari bir xil bo'lgan satrlarni ekranga chiqaring. Agar bunday elementlar bo'lmasa, u holda "-1" chiqaring. Masalan, N=3, S[1]= "Informatik7", S[2]= "RegistR" va S[3]= "kompyuter" bo'lsa, u holda javob: "RegistR".

Satr 19. N natural son va S[1..N] satrli massiv berilgan. Massiv elementlari ichidan birinchi va oxirgi belgilari bir xil o'qiladigan satrlarni ekranga chiqaring. Agar bunday elementlar bo'lmasa, u holda "-1" chiqaring. Masalan, N=3, S[1]= "Informatika", S[2]= "Registr" va S[3]= "7kompyuter" bo'lsa, u holda javob: "Registr".

Satr 20. S satr va H belgi berilgan. Satrda qatnashgan barcha H belgilarini satrda turgan o'rinlari yig'indisini hisoblab chiqaring. Agar bunday belgi bo'lmasa, u holda "-1" chiqaring.

Satr 21. S satr va H belgi berilgan. Satrda qatnashgan barcha H belgilarini o'chirib chiqaring.

Satr 22. Lotin harflaridan iborat S satr berilgan. Satrda barcha unli ("a", "i", "u", "e", "o") belgilarini o'chirib chiqaring.

Satr 23. Lotin harflaridan iborat S satr berilgan. Satrda barcha "ota" so'zlarini "ona" so'ziga almashtirib chiqaring.

Satr 24. S, S1 va S2 satrlar berilgan. S satrda qatnashgan barcha S1 matnini S2 matniga almashtirib chiqaring.

Satr 25. S satr, H1 va H2 belgilari berilgan. S satrdagi har ikkita H1 va H2 belgilari orasidagi belgilarni shu oraliqdagi belgilar sonicha "*" belgisi bilan almashtirib chiqaring.

Satr 26. S satr berilgan. Satrdagi 4 ga karrali bo'lmagan o'rinlarda joylashgan barcha belgilarni o'chirib chiqaring. Masalan, S= "Informatika" bo'lsa, u holda javob: S= "Iri".

Satr 27. Natural N son berilgan. Bir o'lchamli A[1..N] satrlar massivini hosil qiling, bunda massiv elementlari "A", "AA", ... , "AAA...A" ketma-ketlik hosil qilsin.

Satr 28. Natural N son berilgan. Bir o'lchamli A[1..N] satrlar massivini hosil qiling, bunda massiv elementlari "AAA...A",..., "AA", "A" ketma-ketlik hosil qilsin.

Satr 29. Natural N (<27) son berilgan. Bir o'lchamli A[1..N] satrlar massivini hosil qiling, bunda massiv elementlari "A", "AB", "ABC",... , "ABC...Z" ketma-ketlik hosil qilsin.

Satr 30. Natural N (<27) son berilgan. Bir o'lchamli A[1..N] satrlar massivini hosil qiling, bunda massiv elementlari "ABC...Z",..., "ABC", "AB", "A" ketma-ketlik hosil qilsin.

Satr 31. Faqat "0", "1", "2" belgilaridan iborat A satr berilgan. A satrni B satrga quyidagicha ko'chiring: agar A satrning k-o'rindagi belgisi "0" bo'lsa, u holda B satrda k-o'rinda "1", A satrning k-o'rindagi belgisi "1" bo'lsa, u holda B satrda k-o'rinda "2" va A satrning k-o'rindagi belgisi "2" bo'lsa, u holda B satrda k-o'rinda "0" bo'lsin. A va B satrlarni chiqaring.

Satr 32. Natural N son berilgan. A[1..N] belgilar massivini "A", "B" va "G" harflari bilan tasodifiy sonlar generatori yordamida to'ldiring. A[1..N] belgilar massivini B[1..N] belgilar massiviga quyidagicha ko'chiring: agar A[k]="A" bo'lsa, u holda B[k]="B", agar A[k]="B" bo'lsa, u holda B[k]="G" va agar A[k]="G" bo'lsa, u holda B[k]="A" bo'lsin. A va B massivlarni yonma-yon chiqaring.

Satr 33. S satr berilgan. S satrda ketma-ket kelgan orachiqlarni bitta orachiq bilan almashtirib chiqaring.

Satr 33. S satr va H belgi berilgan. S satrda ketma-ket kelgan barcha H belgisini bitta H belgisi bilan almashtirib chiqaring.

= B =

Satr 34. S va S0 satrlar berilgan. S satrda qatnashgan S0 bilan bir xil birinchi qismini o'chirib chiqaring. Agar S satrda S0 bilan bir xil qism bo'lmasa, u holda "-1" chiqaring.

Satr 35. S va S0 satrlar berilgan. S satrda qatnashgan S0 bilan bir xil oxirgi qismini o'chirib chiqaring. Agar S satrda S0 bilan bir xil qism bo'lmasa, u holda "-1" chiqaring.

Satr 36. S va S0 satrlar berilgan. S satrda qatnashgan barcha S0 bilan bir xil qismlarini o'chirib chiqaring. Agar S satrda S0 bilan bir xil qism bo'lmasa, u holda "-1" chiqaring.

Satr 37. S satr berilgan. Satrdagi so'z deb orachiq belgisi qatnashmagan belgilar ketma-ketligi tushunilsa, u holda satrda qatnashgan so'zlar sonini chiqaring. Masalan, S= " Ali,, vali!!12321 SOLI " bo'lsa, u holda javobga "Ali,,", "vali!!12321" va "SOLI" chiqariladi.

Satr 38. A va B satrlar berilgan. Satrlarni o'xshash yoki o'xshash emasligini aniqlang. Satrlar o'xshash deyiladi, agar ular bir xil belgilardan tashkil topgan bo'lsa. Masalan, A= "lola" va B= "olalla" bo'lsa, u holda javob: "O'xshash".

Satr 39. N natural son berilgan. S satrga 1 dan N gacha bo'lgan sonlarni joylashtirib chiqaring. Masalan, N=11 bo'lsa, u holda javob: "1234567891011".

Satr 40. S matn va N natural son berilgan. S matnning belgilari orasiga N tadan "5" raqamini joylashtirib hosil qilingan satrni chiqaring. Masalan, S="dastur" va N=2 bo'lsa, u holda javob: "d55a55s55t55u55r".

Satr 41. Berilgan S matnning har bir belgisidan keyin belgining pozitsiyasicha '*' belgisini joylashtirib hosil qilingan satrni chiqaring. Masalan, S="dastur" bo'lsa, u holda javob (birinchi 'd' belgisining pozitsiyasi 0 ga teng bo'lgani uchun) "da*s**t***u ****ı*****".

Satr 42. Berilgan S matnning har bir belgisidan keyin belgining o'nlik ASCII kodicha '+' belgisini joylashtirib hosil qilingan satrni chiqaring.

Satr 43. S matn faqat raqam va lotin harflaridan iborat. Matndagi raqamlar sonini chiqaring. Masalan, S="R11A02D" bo'lsa, u holda javob: 4.

Satr 44. S matn faqat raqam va lotin harflaridan iborat. Matndagi harflar registrini almashtirib hosil qilingan satrni chiqaring. Masalan, S="R11a02d" bo'lsa, u holda javob: "r11A02D".

Satr 45. S matn faqat raqam va lotin harflaridan iborat. Matndagi raqamlar yig'indisini chiqaring. Masalan, S="A10D02t" bo'lsa, u holda javob: 3.

Satr 46. S matn faqat raqam va lotin harflaridan iborat. Matndagi harflar ajratgan sonlar yig'indisini chiqaring. Masalan, S="A10D02t" bo'lsa, u holda javob: 12.

Satr 47. S matn "raqam±raqam±...±raqam" ko'rinishidagi ifodadan iborat (ifodada ± belgilari o'rnida yoki + yoki – uchraydi). Ifoda natijasini chiqaring. Masalan, S="1+9–0+2" da javob: 12.

Satr 48. S matn faqat 0 va 1 raqamlaridan iborat, ya'ni ikkilik sanoq sistemasidagi son berilgan. Shu sonni o'nlik sanoq sistemasidagi ko'rinishini chiqaring. Masalan, S="0110101" bo'lsa, u holda javob: 53.

Satr 49. O'nlik sanoq sistemasidagi N manfiymas butun son berilgan. N sonining ikkilik sanoq sistemasidagi ifodasi bo'lgan S matnni chiqaring. Masalan, N=53 bo'lsa, u holda javob: S="0110101".

Satr 50. Faqat lotin harflaridan iborat ikkita matn berilgan. Birinchi matndan ikkinchi matnni faqat harflarning o'rnini almashtirib hosil qilish mumkin bo'lsa "Ha", aks holda "-1" chiqaring.

Satr 51. S1 va S2 satrlar berilgan. S1 satr belgilaridan S2 satrini hosil qilish mumkin bo'lsa "Ha", aks holda "Yo'q" chiqaring.

Satr 52. S satr berilgan. Satrdagi belgilarni alifbo bo'yicha saralab, avval belgi, keyin qavs ichida shu belgini satrda qatnashishlar soni bilan birga chiqaring. Masalan, S= "informatika" bo'lsa, u holda javob: "a(2)f(1)i(2)k(1)m(1)n(1)o(1)r(1)t(1)".

Satr 53. N natural son va A[1..N] satrlar massivi berilgan. Shu massivdagi palindrom satrlarni aniqlang. Satr palindrom deyiladi, agar chapdan ham, o'ngdan ham bir xil o'qilsa.

Satr 54. S satr berilgan. Satrdagi so'zlar orachiq bilan ajratilgan. Satrdagi palindrom so'zlarni aniqlang. So'z palindrom deyiladi, agar chapdan ham, o'ngdan ham bir xil o'qilsa.

Satr 55. S satr berilgan. Satrda 16 lik sanoq sistemasidagi sonlar orachiq bilan ajratilgan. Satrdagi sonlarning 10 lik sanoq sistemasidagi ko'rinishini aniqlang.

= C =

Satr 56. S satr faqat raqamlardan iborat. Faqat S satr belgilari qatnashgan eng katta sonni chiqaring. Masalan, S="2107196323011963" bo'lsa, u holda javob "9976633322111100" bo'ladi.

Satr 57. Faqat raqamlardan iborat S matn berilgan. Shu matndagi raqamlar bir martadan qatnashgan eng katta sonni chiqaring. Masalan, S="21071963" bo'lsa, javob 9763210 bo'ladi.

Satr 58. S satr faqat raqamlardan iborat. Faqat S satr belgilari qatnashgan eng kichik sonni chiqaring. Masalan, S="2107196323011963" bo'lsa, u holda javob: "1001112233366799".

Satr 59. Faqat raqamlardan iborat S matn berilgan. Shu matndagi raqamlar bir martadan qatnashgan eng kichik sonni chiqaring. Masalan, S="21071963" bo'lsa, javob: 1023679.

Satr 60. Berilgan (((1?2)?3)?4)?5)?6 ifodada har bir ? belgisi o'rniga to'rt arifmetik amal +, -, *, / dan birini shunday qo'yingki, natija 35 ga teng bo'lsin (bu yerda bo'lish amali butun bo'lishga teng kuchli). Shartni qanoatlantiuvchi barcha yechimlarni chiqaring.

Satr 61. Faqat lotin harflari va orachiqlardan iborat S matn berilgan. Matndagi so'z – bu orachiq bilan chegaralangan va tarkibida orachiq bo'lmagan belgilar ketma-ketligidir. Berilgan matnda faqat bir marta qatnashgan so'zlarni alifbo tartibida joylashtirib chiqaring. Masalan, S= "matn Das tur" bo'lsa, u holda javob: "Das matn tur".

Satr 62. Faqat lotin harflaridan iborat N ta so'z berilgan. Shu so'zlar ketma-ketligidagi barcha anagrammalarni chiqaring (anagramma – bir-biridan harflarning o'rni bilan farqlanuvchi so'zlar). Masalan, N=5 va ketma-ketlikda "olma", "olam", "asar", "sara", "lola" berilgan bo'lsa, u holda javob: "olma", "olam", "asar", "sara".

Satr 63. N natural son berilgan. Kvadrat ildizi butun bo'lgan sonlar ketma-ket yozilsa, quyidagicha satr hosil bo'ladi: 149162536496481... Shu satrning N-o'rindagi raqamini chiqaring.

Satr 64. N natural son berilgan. Raqamlarining yig'indisi toq bo'lgan sonlar ketma-ket yozilsa, quyidagicha satr hosil bo'ladi: 1357910121416182123... Shu satrning N-o'rindagi raqamini chiqaring.

Satr 65. N natural son berilgan. Raqamlarining yig'indisi juft bo'lgan sonlar ketma-ket yozilsa, quyidagicha satr hosil bo'ladi: 24681113151719202224... Shu satrning N-o'rindagi raqamini chiqaring.

Satr 66. S satr berilgan. Satrdagi bittadan ortiq ketma-ket kelgan belgilarni bir marta yozib keyin qavs ichida shu belgini ketma-ket kelish sonini chiqaring. Masalan, S= "Aaa22223" bo'lsa, u holda javob: "Aa(2)2(4)3".

Satr 67. Qavslardan iborat S satr berilgan. Qavslar matematik jihatdan to'g'ri yoki noto'g'ri qo'yilganligini tekshirib javob chiqaruvchi dastur tuzing. Masalan, agar S= "((( ))( ))" bo'lsa, javob "To'g'ri", agar S= "(( )))(( )" bo'lsa, u holda javob: "Noto'g'ri".

Satr 68. Berilgan S matn lotin harflari va orachiqlardan iborat. Berilgan A va B so'zlarning ikkalasini ham S matndagi harflarning o'rnini almashtirmasdan faqatgina o'chirish yordamida hosil qilish mumkin bo'lsa "Mumkin", aks holda "Mumkin emas" javobini chiqaring. Masalan, agar S= "hkloanxzbjsdjujskksa" bo'lib, A= "jussa" va B= "kosa" bo'lsa, u holda javob "Mumkin", agar A= "hola" va B= "kosa" bo'lsa, u holda javob: "Mumkin emas".

Satr 69. Berilgan natural N (10^{21}<N) sonning juft o'rindagi raqamlari yig'indisi Sj, toq o'rindagi raqamlari yig'indisi St bo'lsin. S1–S2 ayirmani chiqaring.

Satr 70. Lotin harflaridan iborat ketma-ketlik shunday tuziladi: avval ketma-ketlik bo'sh. Keyingi har bir qadamda avvalgi ketma-ketlik ulanadi, so'ng unga chapdan lotin alifbosining navbatdagi harfi ulanadi. Quyida ketma-ketlikni hosil qilishning birinchi 4 ta qadami keltirilgan. Avval ketma-ketlik bo'sh.

Qadam 1. a.

Qadam 2. baa.

Qadam 3. cbaabaa

Qadam 4. dcbaabaacbaabaa.

Berilgan N (N<2^{26}-1) natural son uchun 26-qadamda hosil bo'lgan ketma-ketlikda N-o'rinda turgan belgini chiqaring.

Satr 71. Faqat 0 va 1 dan iborat 1001011001101001... satr quyidagicha hosil qilinadi: avval 1 yoziladi, keyin quyidagicha amallar takrorlanadi: hosil qilingan qismning o'ng tomoniga shu qismdagi 0 larni 1 lar va 1 larni 0 lar bilan almashtirib yoziladi, ya'ni 1→10→1001→10010110→... Berilgan N (N ≤ 2147483647) natural son uchun satrning N-belgisini chiqaring.

= D =

Satr 72. S va T satrlar berilgan. Ali satrlar ustida shunday amal bajarishni yaxshi ko'radiki, natijada hosil bo'lgan yangi satrning o'qilishi eski satrni teskari tartibda o'qilishi bilan bir xil bo'ladi. Ali yaxshi ko'rgan amalini bajarib S satrni T satr ko'rinishiga keltirdi. Agar u yaxshi ko'rgan amalini to'g'ri bajargan bo'lsa "Yes", aks holda "No" javobini chiqaring.

K:	code edoc	abb aba	code code	abacaba abacaba	asrgdfngfnmfgnhweratgjkk asrgdfngfnmfgnhweratgjkk	a a	1234 4321	asd dsa	a b
CH:	Yes	No	No	Yes	No	Yes	Yes	Yes	No

Satr 73. N (N<101) natural son va N ta satr berilgan. Ali futbol o'yinini juda yaxshi ko'radi. U Gol.uz saytidan 1963 yildagi futbol bo'yicha jahon chempionati final o'yini natijasini

ko‘rib o‘tirar edi. Buni qarangki, bu o‘yinning asosiy vaqtida durang bo‘lmagan ekan. Ali N ta goldan iborat jadvalni ko‘rib chiqdi. Jadvalning har bir alohida satrida gol urgan komandaning nomi yozib borilgan, lekin o‘yinda g‘olib bo‘lgan komanda haqida ma’lumot berilmagan edi. Jahon chempionati finalida g‘olib bo‘lgan komandaning nomini chiqaring.

K:	1 ABC	5 A ABA ABA A A	2 XTSJEP XTSJEP	3 XZYDJAEDZ XZYDJAEDZ XZYDJAEDZ	3 QCCYXL QCCYXL AXGLFQDD	3 AZID EERWBC EERWBC	5 PKUZYTFYWN PKUZYTFYWN STC PKUZYTFYWN PKUZYTFYWN
CH:	ABC	A	XTSJEP	XZYDJAEDZ	QCCYXL	EERWBC	PKUZYTFYWN

Satr 74. S satr (|S|<2001, bu yerda |S| – S satr uzunligi) berilgan. Jajji Ali kompyuterdagi chatda ishlashni o‘rganib oldi. Lekin u klaviaturada biroz qiynalib va adashib yozadi. U chatga kirib barcha bilan ingliz tilida salomlashmoqchi bo‘ldi. Odatda, u yozgan matndagi ba’zi belgilarni o‘chirishdan “hello” so‘zini hosil qilish mumkin bo‘lsa yoki “hello” so‘zini yozsa, u holda chatdagilar bu matnni jajji Alining salomlashishi deb tushunishadi. Agar Ali “hlelo” deb yozsa, chatdagilar bu matnni salomlashish deb tushunishmaydi. Agar jajji Ali S satrni yozgan bo‘lsa va bu matn salomlashish bo‘lsa “Hi”, aks holda “Uh” javobini chiqaring.

K:	ahhellllloou	hlelo	helhcludoo	tymbzjyqhymedasloqbq	hehwelloho	yehluhlkwo
CH:	Hi	Uh	Hi	Uh	Hi	Uh

Satr 75. S1 va S2 satrlar (|S1|<2001, |S2|<2001) berilgan. Ali do‘sti Valiga S2 mazmundagi xat yozish uchun gazetaning sarlavhasidagi harflarni kesib olib foydalanmoqchi bo‘ldi. Gazetaning sarlavhasi esa S1 mazmunda edi. Tushunish qiyin emaski, Aliga orachiq uchun belgi kesib olish shart emas. Agar Ali do‘sti Valiga o‘zi xohlagan mazmundagi xatni yoza olsa “Ha”, aks holda “Yo‘q” javobini chiqaring.

K:	abcdefg hijk k j i h g f e d c b a	HpOKgo eAtAVB	HpOKgo ogK	GR Z Gc ZG	NwcGaIe SkOva aIa	GRZ Gc LPzD
CH:	Ha	Yo‘q	Ha	Ha	Ha	Yo‘q

Satr 76. S satr (|S|<2001) berilgan. Satr quyi va yuqori registrdagi lotin harflaridan iborat. Bir xil registrga o‘tkazilganda eng kam sondagi harflarning registri almashtirishdan hosil bo‘lgan satrni chiqaring.

K:	HoUse	ViP	maTRIx	BNHWpnpawg	VTYGP	dastur	KSXBXWpebh
CH:	house	VIP	matrix	bnhwpnpawg	VTYGP	dastur	KSXBXWPEBH

Satr 77. Kema aysbergga urilib halokatga uchramoqda. Kemada N ta (N<2001) yo‘lovchi bo‘lib, ularning darajasi bo‘yicha farqlanishi quyidagicha: captain (kema kapitani), man (erkak yo‘lovchi), woman (ayol yo‘lovchi), child (bola yo‘lovchi) va rat (sichqon).

Kemadagilar bitta safda turgancha chapdan o'ngga 1 dan N gacha tartiblanib qutqaruv qayiqlariga chiqishmoqchi. Lekin ular darajasiga mos quyidagi qat'iy qoida asosida qutqaruv qayiqlariga chiqishadi:

- Avval sichqonlar;
- Keyin ayol yoki bolalar;
- So'ng erkaklar;
- Oxirida kapitan.

Agar biror qadamda ikkita yo'lovchidan qayiqqa avval qaysi biri chiqishini aniqlab bo'lmasa, u holda qayiqqa avval safda chaproqda turgan yo'lovchi chiqadi.

Kiritishda N natural son va keyingi N ta satrda orachiq bilan ajratilgan satrlar juftligi berilgan. Satrlar juftligidan birinchisi ismni, ikkinchisi esa darajani bildiradi. Javobga yuqoridagi qoida asosida qayiqqa chiquvchilarni har birining ismini yangi satrdan chiqaring.

K:	6 Jack captain Alice woman Charlie man Teddy rat Bob child Julia woman	1 A captain	5 A captain B man D woman C child E rat	5 Joyxnkypf captain Dxssgr woman Keojmnpd rat Gdv man Hnw man	6 Nokb man Rwyg man Oxkxp man W woman Gztc rat V captain
CH:	Teddy Alice Bob Julia Charlie Jack	A	E D C B A	Keojmnpd Dxssgr Gdv Hnw Joyxnkypf	Gztc W Nokb Rwyg Oxkxp V

Satr 78. N (N<2001) natural son va keyingi N ta satrda lotin harflaridan iborat so'zlar berilgan. Agar so'zdagi belgilar soni 10 tadan ortiq bo'lsa, u holda so'zning avval birinchi belgisini, keyin birinchi va oxirgi belgilari orasidagi belgilar sonini, so'ng oxirgi belgini chiqaring. Aks holda, ya'ni agar so'zdagi belgilar soni 10 tadan ortiq bo'lmasa, u holda so'zni to'liqligicha chiqaring.

K:	3 word localization internationalization	5 abcdefgh abcdefghi abcdefghij abcdefghijk abcdefghijklm	3 njfngnrurunrgunrunvurn jfvnjfdnvjdbfvsbdubruvbubvkdb ksdnvidnviudbvibd

CH:	word l10n i18n	abcdefgh abcdefghi abcdefghij a9k a11m	n20n j27b k15d

Satr 79. Yaponlarda Xayku nomli lirik nazm turi bor bo'lib, u 17 ta bo'g'indan iborat. Bu 17 ta bo'g'in 3 ta iboraga taqsimlanib, 1-ibora 5 ta, 2-ibora 7 ta va 3-ibora 5 ta bo'g'indan iborat bo'ladi. Soddalik uchun bu masalada iboradagi bo'g'inlar soni unli (ya'ni: a, e, i, o, u) harflar soniga teng deb hisoblaymiz. Sizga 3 ta ibora 3 ta satrda berilgan bo'lib, ular faqat quyi registrdagi lotin harflari va orachiqlardan iborat bo'lishi mumkin. Agar nazm Xayku bo'lsa "YES", aks holda "NO"chiqaring.

K:	on codeforces beta round is running a rustling of keys	hatsu shigure saru mo komino wo hoshige nari	o vetus stagnum rana de ripa salit ac sonant aquae
CH:	YES	YES	NO

Satr 80. N natural son va 1, 2, ..., 9 raqamlaridan iborat S satr ($|S| = N$, $N < 65001$) berilgan. Satrning uzluksiz S[m..k] qism satri deb S[m]S[m+1]S[m+2]...S[k] qism satriga aytamiz. S[m..k] satrni juft deb ataymiz, agar satrga mos son juft bo'lsa. Berilgan S satrning barcha juft uzluksiz qism satrlari sonini chiqaring. Masalan,

K:	4 1234	4 2244	1 3	1 6	10 9572683145	2 13	2 18	2 68	3 112	3 122	3 212
CH:	6	10	0	1	24	0	2	3	3	5	4

Satr 81. Afrika krossvordi NxM ($0<N<101$, $0<M<101$) to'g'ri birchakli jadvaldan iborat bo'lib, har bir katakka bitta kichik lotin harfi yozilgan bo'ladi. Jadvalda bir mahfiy so'z yozilgan bo'lib, shu so'zni aniqlash kerak. Krossvordni yechish uchun biror harf jadval satrida yoki ustunida bittadan ortiq takrorlangan bo'lsa, u holda shu harflarning barchasini bir vaqtda o'chirish kerak. Mahfiy so'zni aniqlash uchun jadvalda o'chmay qolgan harflarni chapdan o'ngga va yuqoridan pastga yurgan holda birlashtirish kifoya. Berilgan N va M natural sonlar hamda mos jadvalga ko'ra mahfiy so'zni chiqaring.

K:	3 3 cba bcd cbc	5 5 fcofd ooedo afaoa rdcdf eofsf	4 4 usah usha hasu suha	1 1 a	2 2 zx xz	1 3 ifi	2 1 h j	2 3 mhw bfq	3 2 xe er wb
CH:	abcd	codeforces	ahhasusu	a	zxxz	f	hj	mhwbfq	xeerwb

Satr 82. Ali futbolni yoqtiradi. Bir kuni futbol ko'rayotib o'yinchilar joylashuvini qog'ozga yozib oldi. Bir komanda o'yinchilarini 0 bilan, ikkinchi komanda o'yinchilarini 1 bilan belgiladi. Ali biror komandadan 7 ta o'yinchining ketma-ket joylashishini xavfli deb hisoblaydi. Faqat 0 va 1 lardan iborat Ali yozgan S satr (|S|<2001) berilgan. Joylashuv xavfli bo'lsa "YES", aks holda "NO" javobni chiqaring.

K:	001001	1000000001	00100110111111101	11110111011101	00000001
CH:	NO	YES	YES	NO	YES

Satr 83. Faqat < va > belgilari qatnashishi mumkin bo'lgan S satr (|S|<2001) berilgan. Satrni "yaxshi" deb ataymiz, agar satrda faqat bir xil belgilar bo'lsa. Masalan, "<<<<" satr yaxshi, "<>>" satr esa yaxshi emas. Yaxshi bo'lmagan satrdagi belgi yoki belgilarni o'chirib yaxshi satrga aylantirish mumkin. Lekin satrda belgi o'chirishdan avval quyidagicha amalni keragicha, faqat satr bo'sh bo'lib qolmasa bo'ldi, bajarish mumkin: < belgisidan oldingi belgini o'chirish (agar biror belgi bo'lsa) yoki > belgisidan keyingi belgini o'chirish (agar biror belgi bo'lsa).

Berilgan satrni yaxshi ko'rinishga keltirish uchun o'chirish shart bo'lgan belgilar sonining eng kam qiymatini chiqaring.

K:	3 2 <> 3 >< 1 >	2 5 >>>>> 6 <<<<< 	3 1 > 1 < 6 <<<>>>	3 10 >>>><<<< 6 <<<<>> 6 <<>>>>	3 10 >>>>>>><< 10 <<<<<<<> 10 <<<<>>>>>	3 10 <<<>>><<< 10 <<<<<>> 10 ><<<<>>>><
CH:	1 0 0	0 0 	0 0 3	0 2 2	0 2 5	0 1 0

Satr 84. S satr "soat:minut" formatida berilgan bo'lib, shu vaqtdan keyingi birinchi palindrom vaqtni aniqlang. Soat hisobi 00 dan 23 gacha, minut hisobi esa 00 dan 59 gacha qo'llaniladi. Chapdan ham o'ngdan ham bir xil o'qiladigan vaqt palindrom vaqt deyiladi.

K:	12:21	23:59	15:51	10:44	04:02	12:15	07:07	23:31
CH:	13:31	00:00	20:02	11:11	04:40	12:21	10:01	23:32

Satr 85. Sonni go'zal deymiz, agar son faqat 4 va 7 raqamlaridan iborat bo'lsa. Masalan, 4, 7, 44, 47 yoki 744 sonlari go'zal, 5 va 100 sonlari esa go'zal emas. Ali quyi registrdagi lotin harflaridan tashkil topgan satrning go'zal bo'lishini o'rganib oldi. Avval har bir harf uchun ularning satrdagi tartiblari o'sish tartibida yozib olinadi. Natijada 26 ta sonlar ro'yxati

hosil bo‘ladi, bunda ba’zilari bo‘sh bo‘lishi ham mumkin. Satr go‘zal deyiladi, shunda va faqat shunda, agar har bir harfning ro‘yxatida ixtiyoriy yonma-yon turgan sonlari ayirmasi moduli go‘zal son bo‘lsa. Agar harf bir marta qatnashgan bo‘lsa, u holda bu harf satrning go‘zalligiga ta’sir etmaydi, deb hisoblanadi, ya’ni belgilari bittadan qatnashgan satr ham go‘zaldir. Masalan, “abcda” satrida a: 1, 5; b: 2; c: 3; d: 4 bo‘lgani uchun go‘zal satr, ya’ni 5-1=4. Uzunligi berilgan N (N<100001) natural songa teng bo‘lgan Ali uchun go‘zal bo‘lgan leksikografik eng kichik satrni chiqaring.

K:	5	3	8	13	1	2	7
CH:	abcda	abc	abcdabcd	abcdabcdabcda	a	ab	abcdabc

Satr 86. Faqat quyi va yuqori registrdagi lotin harflaridan qatnashgan bir xil uzunlikdagi A va B satrlar (|A|<2001, |B|<2001) berilgan. Satrlarni o‘qilishdagi tovushlarining alifbosi bo‘yicha taqqoslang. Ya’ni masalan, “AlFa” va “alFa” satrlar bir xil o‘qiladi, ya’ni teng, “AlFa” va “alFb” satrlardan o‘qilishi bo‘yicha ikkinchi (‘a’<‘b’) satr katta. O‘qilishi bo‘yicha satrlar teng bo‘lsa 0, A satr B satrdan kichik bo‘lsa -1, aks holda 1 chiqaring.

K:	aaaa aaaA	abs Abz	abcdefg AbCdEfF	asadasdasd asdwasdawd	aslkjlkasdd asdlkjdajwi	UG ak	JZR Vae	a Z
CH:	0	-1	1	-1	1	1	-1	-1

Satr 87. Faqat quyi va yuqori registrdagi lotin harflari qatnashgan S satr (|S|<2001) berilgan. Satr ustida quyidagi amallarni bajarib chiqaring:

- Barcha unli (a, o, y, e, u, i) harflarni o‘chiring;
- Barcha undosh (a, o, y, e, u, i dan farqli) harflardan avval nuqta qo‘ying;
- Yuqori registrdagi harflarni quyi registrga o‘tkazing.

K:	tour	Codeforces	aBAcAba	ktajqhpqsvhw	ggdvq	obn	wpwl
CH:	.t.r	.c.d.f.r.c.s	.b.c.b	.k.t.j.q.h.p.q.s.v.h.w	.g.g.d.v.q	.b.n	.w.p.w.l

Satr 88. Matn terayotganda klaviaturadagi Caps Lock klavishi xato bosilib qolgan deb hisoblaymiz, agar:

- yoki barcha harflar yuqori registrda bo‘lsa;
- yoki birinchi harfdan boshqa barcha harflar yuqori registrda bo‘lsa.

S satr (|S|<2001) berilgan. Agar S satr Caps Lock klavishi xato bosilib qolganda yozilgan bo‘lsa, u holda harflarning registrini almashtirib, aks holda satrning o‘zini chiqaring.

K:	cAPS	Lock	cAPSlOCK	LoCK	OOPS	A	z	CAPs
CH:	Caps	Lock	cAPSlOCK	LoCK	oops	a	Z	CAPs

Satr 89. ProgramBank sonlar ustida quyidagicha moliyaviy formatda amallar bajaradigan dastur tuzib berishni so‘radi:

- sonni butun va kasr qismi "." belgisi bilan ajratilsin;
- o'qish oson bo'lishi uchun, kerak bo'lsa, sonning butun qismi kichik razryaddan boshlab uchtadan razryadga "," belgisi bilan ajratilsin, masalan, 12345678 o'rniga 12,345,678 kabi;
- moliyaviy formatlarda ishlatilganidek, sonning kasr qismi 2 xona aniqlikda bo'lsin, ortiqcha qism tashlab yuboriladi, yana agar kasr qism ikki xonadan kam bo'lsa, ikki xonagacha 0 bilan to'ldirilsin;
- son oldida "$" (anakonda) belgisi yozilsin;
- "-" ishora belgisi bu moliyaviy formatda yozilmay, manfiy sonlar ishorasiz holda qavs ichida yozilsin.

S satr (|S|<2001) berilgan. Satr o'nlik raqamlardan iborat bo'lib, unda yana ko'pi bilan bitta "." belgisi va "-" belgisi ishtirok etgan. Sonda "-" belgisi bo'lsa, u eng avval yozilgan. Sonda "." belgisi bo'lsa, undan avval va keyin raqam bor. Son aniq 0 ga teng (ya'ni 0 yoki 0.000) bo'lsa, unda "-" belgisi ishtirok etmaydi. S satrdagi sonni ProgramBank so'ragan formatda chiqaring.

K:	2012	0.000	-0.00987654321	-12345678.9	-3136	50117.75
CH:	$2,012.00	$0.00	($0.00)	($12,345,678.90)	($3,136.00)	$50,117.75

Satr 90. Ikkita gnomning genetik kodi ikkita satr (|satr|<100001) ko'rinishida berilgan. Ikkita gnom bir xil irqqa tegishli bo'lishi uchun birinchi gnomning genetik kodida ikkita belgining o'rnini almashtirib ikkinchi gnomning genetik kodini hosil qilish mumkin bo'lsa. Agar ikkala gnom bitta irqqa tegishli bo'lsa "YES", aks holda "NO" javobini chiqaring.

K:	ab ba	aa ab	a za	rtfabanpc atfabrnpc	qxolmbkkt aovlajmlf	aabb bbaa	aba aab	abab abcd
CH:	YES	NO	NO	YES	NO	NO	YES	NO

Satr 91. S satrning qism satri deb S satrning biror sondagi (0 ta bo'lishi ham mumkin) belgilarining o'chirilishidan hosil bo'lgan satrga aytamiz. Satr palindrom deyiladi, agar o'zini teskarisiga, ya'ni satr belgilarini o'ngdan chapga qarab yig'ib chiqilgan satrga teng bo'lsa. Faqat quyi registrdagi lotin harflaridan iborat berilgan S (|S|<2001) satrning leksikografik eng katta palindrom qism satrini chiqaring.

K:	radar	bowwowwow	codeforces	mississipp	tourist	stargates	helloworld
CH:	rr	wwwww	s	ssss	u	tt	w

Satr 92. Yuqori registrdagi lotin harflaridan iborat S satr (|S|<2001) berilgan. Satrga yanglish qo'shilib qolgan barcha "WUB" so'zlarini quyidagicha tartibda o'chirib chiqaring:

- satrning boshida ketma-ket kelgan barcha "WUB" so'zlarini o'chiring;

- satrning oxirida ketma-ket kelgan barcha "WUB" so'zlarini o'chiring;
- har ikkita belgi orasidagi barcha "WUB" so'zlarini bitta orachiq bilan almashtiring.

K:	WUBABCWUB	WUBWUBSR	RWUBWUBLWUB	A	AWUBBWUBCWUBD
CH:	ABC	SR	R L	A	A B C D

Satr 93. Biror satrni K-tartibli satr deymiz, agar biror satrning K marta ketma-ket yozilganiga teng bo'lsa. Masalan, "aabaabaabaab" satr quyidagicha tartibli satrlar bo'la oladi: "aabaabaabaab" – 1-tartibli, "aabaab"+"aabaab" – 2-tartibli, "aab"+"aab"+"aab"+"aab" – 4-tartibli. Natural K ($1 \le K \le 1000$) son va quyi registrdagi lotin harflaridan iborat S satr ($|S|<1001$) berilgan. S satr belgilarining o'rnini almashtirish orqali biror K-tartibli satr hosil qilib chiqaring. Agar bunday satr hosil qilib bo'lmasa -1 chiqaring. Masalan:

K:	2 aazz	3 abcabcabz	1 a	2 aaaabb	2 aaab	3 bbbccc	1 aabaab	3 aaaaaaaaacccddddddd
CH:	azaz	-1	a	aabaab	-1	bcbcbc	aabaab	aaacddaaacddaaacdd

Satr 94. Ali va Vali shaxmat o'ynashni yoqtirishadi. Ma'lumki, shaxmat doskasi quyidagicha bo'lishi kerak:

- shaxmat doskasi 8x8 jadval ko'rinishida bo'ladi;
- chap yuqori burchakdagi katak oq bo'ladi;
- satrdagi va ustundagi ikkita qo'shni katakdan biri oq va ikkinchisi qora rangda bo'ladi.

Ali va Valiga kataklari 8 ta katagi oq (W) yoki qora (B) rangga bo'yalgan shaxmat doskasi qatorlariga mos 8 ta satr berilgan. Ular har bir satrda istalgancha siklik surish bajarishi mumkin, ya'ni oxirgi katakni birinchi katak o'rniga qo'yib qolgan barcha katakni o'ngga surishi mumkin. Agar ikki do'st berilgan satrlardan shu usulda shaxmat doskasini hosil qila olishi mumkin bo'lsa "YES", aks holda "NO" javobini chiqaring.

K:	WBWBWBWB BWBWBWBW BWBWBWBW BWBWBWBW WBWBWBWB WBWBWBWB BWBWBWBW WBWBWBWB	BWBWBWBW WBWBWBWB BWBWBWBW BWBWBWBW WBWBWBWB WBWBWBWB WBWBWBWB WBWBWBWB	WBWBWBWB WBWBWBWB BBWBWWWB BWBWBWBW BWBWBWBW BWBWBWWW BWBWBWBW BWBWBWBW	WBWBWBWB WBWBWBWB BWBWBWBW WBWBWBWB WBWBWBWB WBWBWBWB WBWBWBWB BWWWBWBW
CH:	YES	YES	NO	NO

Satr 95. Maktabning 1-sinfida o'qiyotgan jajji Soli faqat 1, 2, 3 sonlari hamda qo'shish amali qatnashgan yig'indini hisoblay olishi uchun ifodadagi sonlar kamaymaydigan tartibda

bo‘lishi kerak. Ya’ni agar 1+2+3+1 yig‘indi berilsa, uni jajji Soli hisoblay olmaydi, lekin u 1+1+2+3 ifodani hisoblay oladi. Berilgan S satrdagi (|S|<2001) yig‘indini jajji Soli hisoblay oladigan ko‘rinishda chiqaring.

K:	3+2+1	1+1+3+1+3	2	3+3	2+2+1+1+3	2+1+2+2+2+3+1+3+1+2	1+2
CH:	1+2+3	1+1+1+3+3	2	3+3	1+1+2+2+3	1+1+1+2+2+2+2+2+3+3	1+2

Satr 96. Kompyuter paroli "vaporeon", "jolteon", "flareon", "espeon", "umbreon", "leafeon", "glaceon", "sylveon" so‘zlaridan biri bo‘lib, Sizga uzunligi natural N (5<N<9) ga teng va ba’zi harflari ma’lum bo‘lgan hamda boshqa harflari nuqta belgisi bilan almashtirilgan kodlangan satr berilgan. Kompyuter parolini aniqlang.

K:	7 j.....	7 ...feon	7 .l.r.o.	6 .s..o.	7 .mb....	7 .y.....	7 glaceon
CH:	jolteon	leafeon	flareon	espeon	umbreon	sylveon	glaceon

Satr 97. Quyi registrdagi lotin harflaridan iborat S satr (|S|<100001) berilgan. Sizdan satrga biror-bir belgini (albatta bitta belgini) shunday joylashtirish talab qilinadiki, hosil bo‘lgan satr palindrom satr bo‘lsin. Satr palindrom deyiladi, agar o‘zini teskarisiga, ya’ni satr belgilarini o‘ngdan chapga qarab yig‘ib chiqilgan satrga teng bo‘lsa. Agar berilgan satrni bitta belgi yordamida palindrom satrga aylantirish mumkin bo‘lsa palindrom satrni, aks holda “NA” javobini chiqaring. Agar palindrom satrlarning bir nechtasi hosil bo‘lsa, birortasini chiqaring. Masalan:

K:	revive	ee	kitayuta	fft	a	yutampo	afrofa	aqiqa	ghghgh
CH:	reviver	ehe	NA	tfft	aa	NA	afrorfa	aqiiqa	ghghghg

Satr 98. Pangramma deb alifboning barcha harflari qatnashgan so‘z yoki gapga aytamiz, bunda quyi va yuqori registrdagi harflar farqlanmaydi. Satr uzunligini ifodalovchi N (N<2001) natural son va satr berilgan. Agar satr pangramma bo‘lsa “YES”, aks holda “NO” javobini chiqaring.

K:	12 toosmallword	35 TheQuickBrownFoxJumpsOverTheLazyDog	6 ABCDEFghIjKLm
CH:	NO	YES	NO

Satr 99. Ali chapdan o‘ngga 1 dan N gacha (1<N<100001) tartiblangan xonaning 1-sida turibdi. Uning maqsadi N-xonaga borish bo‘lib, har bir xona orasida eshik bor. Eshiklarning turi yuqori registrdagi lotin harflari bilan ifodalanadi. Har bir eshikni eshik turidagi harfning quyi registridagi harfiga mos kalit ocha oladi. Ali biladiki, bir marta ishlatilgan kalitni boshqa ishlatib bo‘lmaydi. S satrda (|S|=2·N-2) kalit va eshiklar ketma-ketligi berilgan. Ali N-xonaga bora olishi uchun avvaldan eng kamida nechta kalitga ega bo‘lishi kerakligini chiqaring.

K:	3 aAbB	4 aBaCaB	5 xYyXzZaZ	5 aArRaRaR	2 dA	10 hNcMeXsSlHsUwYeMcA
CH:	0	3	2	2	1	7

Satr 100. Faqat raqamlardan iborat A va B satrlar ($|A|\leq 10^6$, $|B|\leq 10^6$) berilgan. Satrlar boshida 0 raqamlari kelishi ham mumkin. Agar A soni B sonidan katta bo'lsa ">", agar A soni B sonidan kichik bo'lsa "<", agar sonlar teng bo'lsa "=" chiqaring.

K:	9 10	11 10	00012345 12345	0123 9	0123 111	9 9	0 0000	10000000000000000000000000000000 10000000000000000000000000000001
CH:	<	>	=	>	>	=	=	<

Satr 101. Quyi registrdagi lotin harflaridan iborat S satr ($|S|<200001$) berilgan. Eng kam sondagi harflarni almashtirib satrni shunday o'zgartiringki, ikkita qo'shni harflar har xil bo'lsin. Agar yechimlar ko'p bo'lsa ixtiyoriysini chiqaring. Masalan:

K:	aab	caaab	zscoder	a	dtottttotd	aazz	zz	nnop	aaaaaa	qasdasd
CH:	acb	cabab	zscoder	a	dtotataotd	abzx	zx	npop	ababab	qasdasd

Satr 102. Satrning qism satrlari deb satrning bir nechta (0 ta bo'lishi ham mumkin) ketma-ket kelgan harflaridan tashkil topgan satrga aytamiz. Masalan, "aar" satrning qism satrlari quyidagilar: "" (bo'sh satr), "a", "a", "r", "aa", "ar", "aar". Natural N ($N\leq 10^5$) soni va uzunligi N ga teng quyi registrdagi lotin harflaridan iborat S satr ($|S|\leq 10^5$) berilgan. Berilgan satrning barcha qism satrlari bir-biridan farqli bo'lishi uchun satrda bajariladigan eng kam o'zgartirishlar sonini aniqlang. Agar satrning barcha qism satrlarini bir-biridan farqli qilishning imkoni bo'lmasa, u holda -1 chiqaring.

K:	2 aa	4 koko	5 murat	6 acbead	7 cdaadad	25 peoaicnbisdocqofsqdpgobpn	10 zzzzzzzzzz
CH:	1	2	0	1	4	12	9

Satr 103. Kichik lotin harflaridan iborat S satr ($|S|<100001$) berilgan. Satrni bo'sh bo'lmagan biror qism satrini tanlab bir marta quyidagicha o'zgartirish mumkin: "z" harfi "y" bilan, "y" harfi "x" bilan, ..., "b" harfi "a" bilan, "a" harfi "z" bilan. Shu o'zgartirish natijasida hosil bo'ladgan leksikografik eng kichik satrni aniqlang.

K:	codeforces	abacaba	babbbabaababbaa	aaaa	aaaabbba	bbbbb	bcdabcd
CH:	bncdenqbdr	aaacaba	aabbbabaababbaa	aaaz	aaaaaaaa	aaaaa	abcabcd

Satr 104. A va B satrlar ($|A|<10^5$, $|B|<10^5$) berilgan. Satrning qism satrlari deb satrning bir nechta (0 ta bo'lishi ham mumkin) ketma-ket kelgan harflaridan tashkil topgan satrga aytamiz. Masalan, "rad" satrning qism satrlari quyidagilar: "" (bo'sh satr), "r", "a", "d", "ra", "ad", "rad". A va B satrlarning eng uzun umumiy bo'lmagan , ya'ni birining qism satri

bo‘ladigan va ikkinchisining qism satri bo‘lmaydigan, qism satrining uzunligini aniqlang. Agar umumiy bo‘lmagan qism satrlar bo‘lmasa, u holda -1 chiqaring.

K:	abcd defgh	a a	mmmmm mnmmm	abcdefgh abdcefgh	aabb bbaa	abaa abaa	mo momo
CH:	5	-1	5	8	4	-1	4

Satr 105. N (N<1001) natural son va orachiq bilan ajratilgan faqat quyi registrdagi lotin harflaridan iborat N ta so‘zdan (|so‘z|<1001) iborat S satr berilgan. Ipmok tilida so‘zlar faqat obyektlarni ifodalaydi. Bu tildagi so‘zlar maxsus xossalarga ega:

- so‘z o‘zak bo‘la oladi, agar barcha harflari turlicha bo‘lsa;
- o‘zak va uning harflarini o‘rin almashishlaridan hosil bo‘lgan barcha so‘zlar faqat bitta obyektni ifodalaydi;
- Y so‘zining o‘zagi bo‘lib Y so‘zdagi barcha harflardan bittadan qatnashgan X so‘zi xizmat qiladi, ya’ni agar X=“a” bo‘lsa, u holda u Y=“aa” yoki Y=“aaaa” so‘zlarining, agar X=“ab” bo‘lsa, u holda u Y=“aba” yoki Y=“aabbaab” so‘zlarining o‘zagi bo‘lib xizmat qiladi;
- Ipmok tilidagi har qanday so‘z shu so‘zning o‘zagi ifodalagan obyektlarni ifodalaydi.

Berilgan N soni va S satriga ko‘ra turli obyektlar sonini chiqaring.

K:	5 a aa aaa ab abb	3 amer arem mrea	2 fhjlqs aceginpr	2 bcdfghimn efghijlmo	5 a a a a a
CH:	2	1	2	2	1

11-§. KONTEYNERLARGA OID VAZIFALAR

Quyida berilgan vazifalarning tartibida ko‘rsatilgan konteynerlarning nomlari dasturda qo‘shimcha boshqa konteynerlarni qatnashtirish imkoniyatini rad etmaydi, faqat shu turdagi konteyner albatta qatnashishi shartligini bildiradi. Quyida vector konteyneriga oid vazifalar berilmagan. Bunga sabab: massivlarga oid masalalarni **vector** konteyneri yordamida ham yechib ko‘rish mumkin.

Pair 1. Satr 77 masalasini **pair** konteyneri yordamida yeching.

Pair 2. S satr quyi registrdagi lotin harflaridan iborat (|S|<1000). Satrda uchragan harflarni alifbodagi o‘rniga mos chiqaring. Har bir harfdan keyin shu harfning satrdagi barcha o‘rinlarini chiqaring. Masalan, S= “banan” bo‘lsa, u holda javob: a:2,4 b:1 n:3,5.

Pair 3. N ($0<N<10^6$) ta elementli A massiv berilgan. Massiv elementlari kamaymaydigan tartibda tartiblanganidan so‘ng kiritilgan massivdagi indekslarning yangi joylashuvini chiqaring.

Pair 4. Cofos mamlakatida Kompa shahri qurilmoqda. Hozirgacha markazi Ox o'qida joylashgan tomonlari koordinata o'qlariga parallel bo'lgan kvadrat shaklidagi N ta uy qurib bo'lingan. Uylar faqat chegarasi orqali kesishishi mumkin.

Ali ishlayotgan arxitektura kompaniyasiga Kompa shahrida yangi uy qurish uchun buyurtma kelib tushdi. Buyurtmachi quyidagi talablarni qo'ydi:

- uy T tomonli kvadrat shaklida bo'lsin;
- uy markazi Ox o'qida joylashsin;
- uy tomonlari koordinata o'qlariga parallel bo'lsin;
- uy mavjud uylarning hech bo'lmasa bittasi bilan tutashsin;
- uy mavjud uylar bilan faqat chegarasi orqali kesishsin.

Aliga birinchi satrda mavjud uylar soni natural N ($0<N<1001$) va yangi uy tomoni natural T ($0<T<1001$), keyingi N ta satrning har birida ikkitadan butun son: biri mavjud uyning Ox o'qidagi markazi koordinatasi x_k ($-1001<x_k<1001$), ikkinchisi shu uyning tomoni a_k ($0<a_k<1001$) berilgan. Aliga yangi uyning mumkin bo'lgan holatlari sonini aniqlab bering. Yangi uyning markazi butun son bo'lishi shart emas.

K:	2 2 0 4 6 2	2 2 0 4 5 2	3 966 988 5 15 2 -992 79	1 1 1 1	2 2 0 4 7 4	4 1 -12 1 -14 1 4 1 -11 1	6 15 19 1 2 3 6 2 -21 2 -15 2 23 1	3 501 827 327 -85 480 -999 343	2 999 -999 471 530 588	4 512 -997 354 -568 216 -234 221 603 403
CH:	4	3	6	2	4	5	2	6	4	4

Set 5-11. Massiv 138-...-143, Satr 98 masalalarini set konteyneri yordamida yeching.

Set 12. Hay'at a'zolari dasturlash kontestida qatnashayotganlar orasida spamchilar (keraksiz ma'lumotlar yuborib noqulayliklar keltirib chiqaruvchilar) borligini bilib qolishdi. Kontest yakunlanishiga 10 minut qolganda spamchilar xato yechimlarini yuklab serverni band qilib qo'yishar edi. Shu sababli hay'at a'zolari spamchilarni diskvalifikatsiya (musobaqalardan chetlashtirish) qilishmoqchi bo'ldi. Hay'at a'zolari bittadan ortiq yechimini yuklagan kontest qatnashchilarini spamer deb hisobladi. Agar oxirgi 10 minutda yechimlarini yuklagan kontest qatnashchilari soni N ($0\le N\le 100$) va ularning logini ($|login|<33$) ma'lum bo'lsa, u holda spamchilar loginlarini ixtiyoriy ketma-ketlikda chiqaring. Hech bo'lmasa bitta spamchi borligi aniq.

K:	11 naucoder iceman abikbaev abikbaev petr abikbaev abikbaev x abikbaev acrush x	9 barsa coder404 ufa barsa yalyublyu alfa yalyublyu sss aha	10 traktor ekskovator buldozer tank motosikl avtomobil raketa tank vertolyot samolyot	9 abbbbbbb bad coca down abbbbbbb bad coca down coca	11 sadasdagghghgg fgfvcvbhtybfgfgnnvbnfg bckjdhfldnclksdhos ioijklkdasjkjkj kjsdfkjshkjhk ooooooooooooooo hhhhhhhhhhhhhhhh oooooooooooooooooh hooooooooooooooooo sadasdagghghgg oooooooooooooooooo
CH:	abikbaev x	barsa yalyublyu	tank	abbbbbbb bad coca down	sadasdagghghgg

Set 13. Ali va Vali shunday o'yin o'ynashmoqda. Qog'ozga biror N (|N|<10001) butun son yoziladi. O'yinni Ali boshlab biror butun son aytadi va qog'ozdagi sondan shu son ayrilib qog'ozga yoziladi. Navbat Valiga o'tib u ham biror butun son aytadi va qog'ozdagi sondan shu son ayrilib qog'ozga yoziladi. Shu yo'sinda o'yin davom etadi. Lekin agar birorta o'yinchi son aytgandan keyin qog'ozdagi biror son takrorlansa, shu o'yinchi yutqazadi. Agar ularning har biri M (M<1001) tadan son aytganda qog'ozdagi birorta ham son takrorlanmasa, o'yin durang bilan tugaydi. N va M sonlari va ular aytayotgan sonlar juftliklari M ta satrda berilgan. Agar o'yinda Ali g'olib chiqsa Ali, agar Vali g'olib chiqsa Vali, aks holda Durang so'zini chiqaring.

K:	70 7 5 21 13 -2 9 27 18 2 5 5 18 18 7 7	15 5 14 8 -10 5 4 -8 5 7 -6 -7	55 5 -7 12 9 -15 11 8 0 4 2 -6	0 6 100 58 -200 18 -5 9 15 -5 66 -56 15 45	21 4 7 7 7 7 7 7 7 7	17 5 -8 8 9 -9 11 -11 0 5 4 3	1000 6 411 567 45 -84 19 20 0 5 4 3 15 45
CH:	Durang	Ali	Vali	Ali	Durang	Ali	Ali

Set 14. Ali turli chatlarda har xil nomdagi foydalanuvchilar bilan suhbatlashardi. Bir kuni xayoliga shunday fikr keldi: agar chatdagi foydalanuvchining nomida turli harflar soni toq bo'lsa – qiz bola, aks holda o'g'il bola bo'ladi. Berilgan nomga ko'ra (|nom|<101) Alining

fikri bo'yicha qiz bola bo'lsa "CHAT WITH HER!" matnini, aks holda "IGNORE HIM!" matnini chiqaring.

K:	wjmzbmr	xiaodao	sevenkplus	tgcdptnkc
CH:	CHAT WITH HER!	IGNORE HIM!	CHAT WITH HER!	IGNORE HIM!

Set 15. Ali o'zi yasagan qurilmasida N (0<N<1001) ta tajriba o'tkazib turli butun Ak natijalar (|Ak|<1001) oldi. Natijalar tahlilida unga ikkinchi minimal natija kerak bo'ldi. Aliga ikkinchi minimal sonni topishga yordam bering. Agar buning imkoni bo'lmasa "NO" javobini chiqaring.

K:	4 1 2 2 -4	5 1 2 3 1 1	5 21 21 21 21 21	6 15 7 -3 9 15 7	6 150 149 149 149 149 149
CH:	1	2	NO	7	

MultiSet 16. O'lchamlari NxM (1≤N, M≤500) bo'lgan A va B matritsalar berilgan. Ali A matritsaga ixtiyoriy marta quyidagi amalni qo'llashi mumkin: A matritsaning ixtiyoriy kvadrat qism matritsasini tanlab transponirlashi mumkin, ya'ni qism matritsaning s-satr va u-ustunda turgan elementi transponirlash amalidan keyin u-satr va s-ustunga joylashadi, shu bilan birga, qism matritsa A matritsadagi o'z o'rnida qoladi. A matritsadan B matritsani shu amal yordamida hosil qilish mumkin bo'lsa "YES", aks holda "NO" javobini chiqaring.

Birinchi satrda orachiq bilan ajratilgan N va M sonlari, ya'ni, mos ravishda, A va B matritsalarning satrlari va ustunlari soni berilgan. Keyingi N ta satrning s-satrida A matritsaning u-ustuniga mos orachiq bilan ajratilgan butun sonlar berilgan ($1 \le A[s,u] \le 10^9$). Xuddi shu kabi, keyingi N ta satrning s-satrida A matritsaning u-ustuniga mos orachiq bilan ajratilgan butun sonlar berilgan ($1 \le B[s,u] \le 10^9$).

K:	2 2 1 1 6 1 1 6 1 1	2 2 4 4 4 5 5 4 4 4	1 1 321 371	3 3 1 2 3 4 5 6 7 8 9 1 4 7 2 5 6 3 8 9	3 2 5 3 5 5 2 3 5 5 3 5 2 3	3 4 3 4 3 3 3 7 5 5 1 1 5 3 3 3 7 3 4 1 5 5 3 1 5 3	1 1 212055293 212055293	3 3 1 2 3 2 1 2 1 2 1 1 2 3 2 3 2 1 2 1	3 3 1 2 3 4 3 6 7 8 9 1 2 3 4 7 6 7 8 9
CH:	YES	NO	NO	YES	YES	YES	YES	NO	YES

Map 17. Maktabda N (1≤ N ≤1000) ta server bo'lib, har bir server nom va IP-adresga ega (nomlar turlicha bo'lishi shart emas, lekin IP-adreslar bir-biridan farqlanadi). Ali bu nom va IP-adreslarning barchasini biladi. Soddalik uchun har qanday nginx buyrug'i "buyruq IP" ko'rinishida bo'lsin, bu yerda buyruq – quyi registrdagi lotin harflaridan iborat satr, IP – maktabdagi serverlardan birining adresi. Har bir IP-adres "a.b.c.d" ko'rinishga ega bo'lib, a,

b,c va d sonlari manfiymas butun va 255 dan ortmaydi (birlamchi nollarsiz). Ali izoh qo'shishi kerak bo'lgan nginx konfiguratsiya fayli M ($1 \leq M \leq 1000$) ta buyruqdan iborat. Boshqalar serverlarning IP-adresini eslab qolmaganligi sababli konfiguratsiya faylini o'qish qulayroq bo'lishi uchun Ali har bir buyruqdan keyin IP-adresga mos server nomini qo'shib yozishi kerak. Ya'ni agar buyruq "buyruq IP" ko'rinishida bo'lgan bo'lsa, u holda Ali uni "buyruq IP #nom" ko'rinishiga almashtirishi kerak, bu yerda nom – IP-adresga mos server nomi.

Birinchi satrda N va M berilgan. Keyingi N ta satrda orachiq bilan ajratilgan nom (belgilari soni 10 tadan oshmaydigan) va IP-adres berilgan. Keyingi M ta satrda orachiq bilan ajratilgan konfiguratsiya faylining buyrug'i (belgilari soni 10 tadan oshmaydigan) va IP-adres berilgan. Ali yuqorida aytib o'tilgan vazifani bajargandan keyingi konfiguratsiya faylining M ta buyrug'ini chiqaring.

K:	2 2 main 192.168.0.2 replica 192.168.0.1 block 192.168.0.1; proxy 192.168.0.2;	3 5 google 8.8.8.8 codeforces 212.193.33.27 server 138.197.64.57 redirect 138.197.64.57; block 8.8.8.8; cf 212.193.33.27; unblock 8.8.8.8; check 138.197.64.57;
CH:	block 192.168.0.1; #replica proxy 192.168.0.2; #main	redirect 138.197.64.57; #server block 8.8.8.8; #google cf 212.193.33.27; #codeforces unblock 8.8.8.8; #google check 138.197.64.57; #server

Map 18. Ali universitetda tarix fanidan ustoz ma'ruzalarini qisqaroq so'zlar bilan yozish uchun til o'ylab topgan edi. Ustoz gapirayotgan tildagi so'zlar ham Ali o'ylab topgan tildagi so'zlar ham yozilishi bilan farqlanadi. Ikkala tildagi so'zlar quyi registrdagi lotin harflaridan tashkil topgan. Agar ustoz aytgan so'z bilan Ali o'ylab topgan tildagi so'z uzunligi teng bo'lsa, u holda bunday so'zlarni Ali ustoz gapirgan tildagidek yozardi.

Birinchi satrda ustoz ma'ruzasidagi so'zlar soni N ($1 \leq N \leq 3000$) va har bir tildagi so'zlar soni M ($1 \leq M \leq 3000$) berilgan. Keyingi M ta satrda ustoz gapirgan tildagi va Ali o'ylab topgan tildagi so'zlar orachiq bilan ajratib berilgan. Oxirgi satrda ustoz ma'ruzasidagi N ta so'z orachiq bilan ajratib berilgan. Berilgan barcha so'zlarda belgilar soni 10 tadan oshmaydi. Ali yozib olgan ma'ruzani chiqaring.

K:	4 3 codeforces codesecrof contest round letter message codeforces contest letter contest	5 3 joll wuqrd euzf un hbnyiyc rsoqqveh hbnyiyc joll joll euzf joll	5 5 queyqj f b vn tabzvk qpfzoqx ytnyonoc hnxsd jpggvr lchinjmt queyqj jpggvr b ytnyonoc b
CH:	codeforces round letter round	hbnyiyc joll joll un joll	f jpggvr b hnxsd b

Map 19. Tarix ustozi imtihon olishni osonlashtirish uchun shunday usul tanladi: o'quvchi o'zi bilgan "tarixiy" sanalarga mos yillarni ixtiyoriy tartibda (ba'zilarini bir necha marta) ro'yxatga yozib chiqadi. Ustoz esa bu yillar ro'yxatini kamaymaydigan tartibda (ba'zilarini bir necha marta) yozib olgan bo'ladi. Ustoz o'quvchi ro'yxatidagi sonlardan o'z ro'yxatida bor sonlarni sanab chiqadi. Birinchi satrda ustozning ro'yxatidagi yillar soni N ($1 \le N \le 15000$) va keyingi N ta satrda yillar, keyingi satrda o'quvchining ro'yxatidagi yillar soni M ($1 \le M \le 10^6$) va keyingi M ta satrda shu yillar berilgan. Kiritiladigan sonlar qiymati [1; 10^9] oraliqqa tegishli. O'quvchining to'g'ri javoblari sonini aniqlang.

K:	2 1054 1492 4 1492 65536 1492 100	2 1054 1492 4 1492 1054 1492 100	3 1111 1111 1111 3 1234 3214 1101	4 1234 2345 5678 11111 2 11111 1234	1 1234 5 1234 1234 1234 1234 1234	4 1234 12345 55555 61234 4 5555 1234 6123 1022	2 1102 2107 5 2107 2107 2107 2107 2107	2 1102 2107 6 1102 1102 1102 1102 1102 1102	2 1102 2107 6 2107 1102 1102 1102 2107 1102
CH:	2	3	0	2	5	1	5	6	6

Map 20. Cofos mamlakatidagi g'aroyib kompa o'yinida ikki kishi ishtirok etib, har bir o'yinchi biror qiymatdagi ball yutadi yoki biror qiymatdagi ball yutqazadi. O'yin boshida barcha o'yinchining bali 0 ga teng bo'ladi. O'yin natijasi "name score" ko'rinishida saqlab boriladi, bu yerda name – o'yinchi nomi, score – butun son bo'lib, o'yinchi yiqqan balidir. Agar score manfiy son bo'lsa, u holda shu o'yinda o'yinchi yutqazgan bo'ladi.

O'yin g'olibi quyidagicha aniqlanadi:

- agar o'yin oxirida maksimal ball yiqqan o'yinchi yagona bo'lsa, shu g'olib bo'ladi;
- agar o'yin oxirida bittadan ortiq o'yinchi maksimal ball yiqqan bo'lsa, u holda birinchi bo'lib shu maksimal balni yiqqan o'yinchi g'olib bo'ladi.

Hech bo‘lmasa bitta o‘yinchining bali musbat bo‘lishi aniq bo‘lsa, g‘olib o‘yinchini aniqlang.

K:	3 mike 3 andrew 5 mike 2	5 kaxqybeultn -352 mgochgrmeyieyskhuourfg -910 kaxqybeultn 691 mgochgrmeyieyskhuourfg -76 kaxqybeultn -303
CH:	andrew	kaxqybeultn

Map 21. Cofos mamlakatida g‘aroyib Loto o‘yini o‘tkaziladi. O‘yinchilar o‘z fayllariga moduli bo‘yicha 100000 dan katta bo‘lmagan teng sonlar ham bo‘lishi mumkin bo‘lgan K tadan (0<K<100) butun son yozishadi va Loto serveriga jo‘natishadi. Barcha fayllar yig‘ilgach Loto serveri Lotos nomli butun sonlar to‘plamini tasodifiy sonlar generatori yordamida quyidagicha hosil qiladi:

- barcha hosil qilingan sonlar moduli bo‘yicha 100000 dan katta bo‘lmaydi;
- hosil qilingan Lotos to‘plamidagi sonlar soni M ta (0<M<100001) bo‘ladi.

Birinchi satrda orachiq bilan ajratilgan o‘yinchilar soni N (0<N<101), M va K (0<K<101) sonlari berilgan. Keyingi N ta satrning har birida o‘yinchilar yuborgan orachiq bilan ajratilgan K tadan butun son berilgan. Oxirgi satrda Loto serveri hosil qilgan M ta butun son berilgan. O‘yinchi yuborgan biror son Lotos to‘plamida necha marta uchrasa, shuncha marta hisobga olinadi. Lotos to‘plamidagi eng ko‘p sonni topgan o‘yinchini, bunday o‘yinchilar ko‘p bo‘lsa, barchasining tartib raqamlarini bitta satrda orachiq bilan ajratib chiqaring. Hech qaysi o‘yinchi yutmagan bo‘lsa, u holda 0 chiqaring.

K:	3 5 2 5 -2 0 4 100 -100 91 14 2 0 5	4 5 3 0 0 0 1 1000 -1000 1941 1945 4 1991 2019 28 1945 0 0 0 1945	4 6 4 0 1 2 3 19 10 19 10 19 91 20 19 19 19 19 19 19 10 0 0 0 0	2 6 1 5 -5 0 -2 5 -5 0 -5	2 6 1 1 2 0 -2 5 -5 0 -5
CH:	1 2	1	1 2 4	2	0

ADABIYOTLAR

Bosma adabiyotlar

1. A.R. Azamatov. Algoritmlash va dasturlash asoslari. O‘quv qo‘llanma. Toshkent, 2014 y.
2. B. Boltayev, A. Azamatov, G. Azamatova, B. Xurramov. Paskal dasturlash tili. Uslubiy qo‘llanma. Toshkent, 2013 y.
3. B. Boltayev, A. Azamatov, B. Xurramov, G. Ishanxodjayeva, S. Muminov. Kompyuterlarning arifmetik asosi. Sanoq sistemalari. Uslubiy qo‘llanma. Toshkent, 2017 y.
4. Sh. Madrahimov, S. Gaynazarov. C++ programmalash tili. Toshkent, 2008 y.
5. Sh. Madrahimov. Kompyuterda amaliyot fanidan laboratoriya ishlari. Toshkent, 2007 y.
6. B. Boltayev, A. Azamatov, Sh. Xidirov, B. Xurramov, K. Iskandarov. Algoritmlash va Paskal dasturlash tili bo‘yicha berilgan misol va masalalarni yechish usullari. Toshkent, 2012 y.
7. А. Кулаков, С. Ландо, А. Семенов, А. Шен. Алгоритмика. V – VII sinflar uchun darslik. Moskva, 1997 y.
8. М. Абрамян. Programming Taskbook. Rossiya, 2005 y.
9. Д. Кнут. Исскуство программирования. 3 tomlik. AQSh, 1998 y.
10. B. Boltayev, M. Mahkamov, A. Azamatov. Informatikadan olimpiada masalalarini yechish. Uslubiy qo‘llanma, Toshkent, 2004 y.
11. B. Boltayev, M. Mahkamov, A. Azamatov. Informatikadan olimpiada masalalarini yechish-2. Uslubiy qo‘llanma, Toshkent, 2004 y.
12. T. Azlarov, F. Zokirova. Informatika va HT dan olimpiada masalalari va mashqlari to‘plami. Toshkent, 1996 y.
13. B. Boltayev, M. Mahkamov, A. Azamatov. Informatika. 8-sinf masalalar to‘plami va ularni yechish usullari. Uslubiy qo‘llanma. Toshkent, 2005 y.
14. B. Boltayev, A. Azamatov. A. Asqarov, M. Sodiqov, G. Azamatova. Informatika va hisoblash texnikasi asoslari. 9-sinf uchun darslik. Toshkent, 2015 y.
15. B. Boltayev, M. Mahkamov, A. Azamatov, S. Rahmonqulova. Informatika va AT. 7-sinf. Toshkent: 2017 y.
16. B. Boltayev, M. Mahkamov, A. Azamatov. Paskal dasturlash tili. Uslubiy qo‘llanma. Toshkent, 2007 y.
17. Р. Грэхем, Д. Кнут, О. Паташник. Конкретная математика. Математические основы информатики. Moskva, 2010 y.
18. I. Horton, P. Van Weert. Beginning C++17: From novice to Professional. Nyu York, 2018 y.
19. A. Azamatov. Dasturlash bo‘yicha maktab o‘quvchilarining komanda chempionati. Sankt-Peterburg, 2007 y. Fizika, matematika va informatika ilmiy-uslubiy jurnal, 2008 y, 1-son.
20. A. Azamatov, H. Komilova. Dasturlash to‘garaklarida iqtidorli o‘quvchilar bilan ishlashda tanlama masalalari. Fizika, matematika va informatika ilmiy-uslubiy jurnal, 2015 y, 4-son.
21. A. Azamatov, A. Rahimov. Dasturlash texnologiyasida rekursiya. Aniq va tabiiy fanlar metodikasi uslubiy jurnali. 2018 y, 3-son.

22. A. Azamatov, A. Rahimov. C++ tilida mulohazalarga oid masalalar yechish. Aniq va tabiiy fanlar metodikasi uslubiy jurnali. 2018 y, 9-son.

Elektron adabiyotlar

1. codeforces.com
2. hackerrank.com
3. topcoder.com
4. csacademy.com
5. atcoder.jp
6. usaco.org
7. e-maxx.ru
8. cppreference.com

MUNDARIJA

Ilmiy-ommabop nashr

B.J. BOLTAYEV, A.R. AZAMATOV, A.D. RAHIMOV,
B.A. AZAMATOV, D.T. ASRAYEVA, SH.Z. QAMBARALIYEV

C++ TILI ASOSLARI

Muharrir: Dilorom MATKARIMOVA
Badiiy muharrir: Bahriddin BOZOROV
Texnik muharrir: Dilshod NAZAROV
Sahifalovchi: Inomjon O'SAROV
Musahhih: Mahfuza IMOMOVA

Nashriyot litsenziyasi: AI №134, 27.04.2009
Terishga berildi: 05.01.2020-y.
Bosishga ruxsat etildi: 24.01.2021-y.
Ofset qog'ozi. Qog'oz bichimi: 84x108 $^{1}/_{16}$.
Times garniturasi. Ofset bosma.
Hisob-nashriyot t.: 40,2. Shartli b.t.: 42,4.
Adadi: 1500 nusxa.
Buyurtma № 11

«AKADEMNASHR» nashriyotida tayyorlandi va chop etildi.
100156, Toshkent shahri Chilonzor tumani 20^{A}-mavze 42-uy.

Tel.: (+99871) 216-87-81
e-mail: info@akademnashr.uz
web: www.akademnashr.uz