

O'ZBEKISTON RESPUBLIKASI
OLIY VA O'RTA MAXSUS TA'LIM VAZIRLIGI

SAMARQAND DAVLAT UNIVERSITETI

S.Eshtemirov, F.M.Nazarov

ALGORITMLASH VA DASTURLASH ASOSLARI



**Kitob quyidagi ko'rsatilgan
muddatda topshirilishi shart**

**Oldingi foydalanishlar
miqdori**

--	--

ами;

В

**O'ZBEKISTON RESPUBLIKASI OLIY VA O'RTA MAXSUS
TA'LIM VAZIRLIGI**

SAMARQAND DAVLAT UNIVERSITETI

Eshtemirov Sodiq, Nazarov Fayzullo

**ALGORITMLASH VA DASTURLASH
ASOSLARI**

*(5130200-Amaliy matematika va informatika, 5140200-Matematika
yo'nalishlari uchun)*

O'quv qo'llanma

SAMARQAND - 2019

UDK: 681.14(075)

BBK 32.97

Algoritmash va dasturlash asoslari. O'quv qo'llanma – SamDU, 2019 – yil, 208- bet.

Mazkur qo'llanmada algoritmlar nazariyasi hamda dasturlash asoslari fani C++ dasturlash tili misolida qarab o'tilgan. Qo'llanmada algoritmlar uning xossalari, berilish usullari, turlari, C++ dasturlash tili tarkibida boshlang'ich tushunchalar, standart funksiyalar, chiziqli, tarmoqlanuvchi, takrorlanuvchi jarayonlar, massivlar, satrlar, funksiya proseduralar, fayllar, strukturalar hamda grafik imkoniyatlar keltirib o'tilgan. Barcha mavzular amaliy jihatdan aniq misollar yordamida atroflicha yoritib o'tilgan. Qo'llanmada C++ tilining asosiy imkoniyatlari bo'yicha nazariy tushunchalar hamda bu tushunchalarni o'zlashtirish uchun masalalarning dasturlari ham keltirilgan. Har bir mavzu bo'yicha mavzuni mustahkamlash uchun nazariy savollar hamda mustaqil ishlash uchun topshiriqlar ham berilgan. Qo'llanma asosan talabalar va mustaqil o'rganuvchilar uchun qulay vosita hisoblanadi. Mazkur qo'llanma oliy o'quv yurtlari talabalari va magistrantlari, litsey, kasb hunar kollej o'quvchilari hamda mustaqil o'rganuvchilar uchun ham mo'ljallangan. Asosan Amaliy matematika va informatika hamda matematika yo'nalishlarining o'quv dasturlariga to'liq mos keladi.

Mualliflar:

Eshtemirov Sodiq

Samarqand Davlat Universiteti

"Axborotlashtirish texnologiyalari "
kafedrasi dotsenti.

Nazarov Fayzullo Maxmadiyarovich

Samarqand Davlat Universiteti

"Axborotlashtirish texnologiyalari "
kafedrasi o'qituvchisi.

Ma'sul muharrir:

Samarqand Davlat Universiteti

"Axborotlashtirish texnologiyalari "
kafedrasi professori. t.f.d. A.Axatov

Taqrizchilar:

Nizomiy nomidagi TDPU "Informatika
o'qitish metodikasi" kafedrasi mudiri
dots.S.Tursunov

TATU Samarqand filiali "Dasturiy injiniring"
kafedrasi mudiri dots.A.Qarshiyev

ISBN-978-9943-6178-9-6

©Samarqand Davlat Universiteti, 2019

2

M u n d a r i j a

Kirish.....	6
Algoritmlar	9
1.1. Algoritm tushunchasi va uning xossalari.....	9
Qiymat berish buyrug'i	11
1.2. Algoritmning berilish usullari va turlari.....	13
Algoritmning turlari.....	14
Chiziqli algoritmlar.....	14
Mantiqiy amallar va mantiqiy ifodalar	16
Murakkab buyruqlar	18
Tarmoqlanish buyrug'i.....	19
Tanlash buyrug'i.....	21
1.3. Takrorlanish (sikl) buyrug'i.....	24
1.4. Ichma-ich joylashgan sikllar.....	27
C++ dasturlash tili	32
2.1. C++ dasturlash tili tarixi	32
C++ dasturlash tili alifbesi.....	33
Arifmetik amallar va ularning bajarilish tartibi.....	34
Standart funksiyalar	35
C++ dasturlash tilida ifodalar va ularning yozilishi.....	37
2.2. C++ dasturlash tilida yozilgan dasturning umumiy tuzilishi	40
C++ dasturlash tilida izohlardan foydalanish.....	43
Chiziqli dasturlar.....	44
2.3. C++ dasturlash tili operatorlari.....	47
O'zgaruvchilarning qiymatlarini kiritish operatori.....	47
Ma'lumotlarni chiqarish operatori.....	48
Qiymat berish buyrug'i (ta'minlash operatori).....	49
Inkrement va dekrement amallari.....	50
Qiymat berish buyrug'ining qo'shimcha ko'rinishlari....	50
Identifikatorlar va turlar.....	55
3.1. Identifikatorlar va kalit so'zlar.....	55
O'zgaruvchilar.....	56
O'zgarmlar.....	56
3.2. C++ dasturlash tilining standart(tayanch) turlari...	58
Butun son turi	59
Haqiqiy son turi.....	60

	Mantiqiy tur.....	
	Mantiqiy amallar	
	Turni boshqa turga keltirish.....	
	Belgi va satr turi.....	
	Sizeof amali	
	Sanab o'tiladigan turlar.....	
IV BOB	Murakkab buyruqlar.....	
	4.1. Tarmoqlanuvchi operatorlar(If operatori).....	
	Tarmoqlanish buyrug'ining if-else ko'rinishi.....	
	Shartsiz o'tish operatori.....	
	?: amali.....	
	break operatori. continue operatori.....	
	Tanlash buyruqlari.....	
	Tanlash buyrug'ini switch operatori yordamida tashkil qilish.....	
	4.2. Takrorlanish(sikl) buyruqlari.....	
	Takrorlanish buyrug'ini for operatori yordamida tashkil qilish.....	
	Takrorlanish buyrug'ini while operatori yordamida tashkil qilish.....	
	Takrorlanish buyrug'ini do-while operatorlari yordamida tashkil qilish.....	
	Ichma-ich joylashgan sikllar.....	
	4.3. Ko'rsatkichlar.....	
	Adresdagi qiymatlar ustida inkrement va dekrement amallarini bajarish.....	
V BOB	Massivlar va satrlarni qayta ishlash.....	
	5.1. Massiv tushunchasi	
	Bir o'lchamli massivlar.....	
	Ko'p o'lchamli massivlar.....	
	Massivlarni initsializatsiyalash.....	
	Ko'p o'lchamli massivlarni initsializatsiyalash.....	
	O'lchamsiz initsializatsiyalash.....	
	5.2. Satrli massivlar.....	
	Satrlarni qayta ishlash funksiyalari.....	
	Satr uzunligini aniqlash funksiyalari.....	
	Satrlarni nusxalash funksiyalari.....	
	Satrlarni ulash funksiyalari.....	

	Satrlarni solishtirish (taqqoslash) funksiyalari.....	119
	Satrdagi harflar registrini almashtirish funksiyalari.....	121
	Satrnı teskari tartibda yozish funksiyasi.....	122
	Satrdagi belgini yoki qismini izlash funksiyalari.....	123
	Satr turini o'zgartiruvchi funksiyalar.....	126
	5.3. String turidagi satrlar va ularni qayta ishlash funksiyalari.....	129
	Satrlarni ulash funksiyasi.....	131
	Satr tarkibiga satr joylashtirish funksiyasi.....	132
	Satr yoki uning qismini o'chirish funksiyasi.....	133
	Matn qismlarini almashtirish funksiyasi.....	134
	Satrning biror qismidan nusxa olish funksiyasi.....	135
	Satr qismlarini izlash funksiyalari.....	136
	Satrlarni solishtirish funksiyalari.....	138
	Satr xossalarini aniqlash funksiyalari.....	141
	Funksiya tushunchasi.....	144
	6.1. Funksiyani e'lon qilish va aniqlash.....	146
	Funksiya prototipidan foydalanmaslik holati.....	150
	Global va lokal o'zgaruvchilar.....	151
	Funksiya qiymatini qaytarish haqida.....	153
VI BOB	Fayl va strukturalar	162
	7.1. Fayllar va ulardan foydalanish	162
	Fayllarni e'lon qilish.....	162
	Fayllarga murojaat qilish.....	164
	Fayllar ustida amallar bajarish.....	165
	Fayldan ma'lumot o'qish.....	169
	Fayl tarkibidagi ma'lumotlarni o'chirish.....	171
	7.2. C++ dasturlash tilida struktura va ulardan foydalanish	177
	Struktura va ularni e'lon qilish.....	179
	Struktura massivi.....	182
VIII BOB	C++ dasturlash tilining grafik imkoniyatlari.....	189
	8.1. Tekislikda shakllar chizish.....	189
	8.2. Tekislikda yassi figuralar chizish.....	198
	Grafik muhitda matnlar bilan ishlash.....	202
	Foydalanilgan adabiyotlar.....	207

Kirish

O'zbekiston Respublikasi Prezidentining 2017 yil 7 fevraldagi PF-4947-son farmoni bilan tasdiqlangan 2017-2021 yillarda O'zbekiston Respublikasini rivojlantirishning beshta ustuvor yo'nalishi bo'yicha harakatlar strategiyasi mamlakatning davlat va jamiyat rivojlanishi istiqbolini strategik rejalashtirish tizimiga sifat jihatdan yangi yondashuvlarni boshlab berdi. Unda belgilangan vazifalar sirasida ta'lim va fan sohasini rivojlantirishga ham alohida e'tibor berilgan. O'zbekiston Respublikasi birinchi Prezidentining 2012 yil 21 martdagi "Zamonaviy axborot-kommunikatsiya texnologiyalarini joriy etish va rivojlantirish chora-tadbirlari to'g'risida"gi PQ-1730 qarori hamda "O'zbekiston Respublikasida "Elektron ta'lim milliy tarmog'ini yaratish" investitsion loyihasini amalga oshirish chora-tadbirlari to'g'risida" gi PQ-1740 qarori va boshqa bu sohaga taalluqli me'yoriy hujjatlar algoritmik ta'minot ishlab chiqish va joriy etishning juda keng ko'lamli ekanligidan dalolat beradi. Barcha tashkilot va muassasalarda avtomatlashtirilgan tizimlar yaratish va ulardan keng ko'lamda foydalanishga erishishda algoritmallashtirishning o'rni katta hisoblanadi.

6

Bu buyruqlar noi va birlar ketma-ketligidan iborat matnlardan tashkil topgan edi. Keyinchalik insonlar uchun tushunarli bo'lgan mashina buyruqlarini o'zida saqlovchi assembler tili yaratildi. Keyinchalik FORTRAN, BASIC, PASKAL va COBOL singari yuqori darajali tillar ham paydo bo'ldiki, bu tillar yordamida so'z va gaplarning mantiqiy konstruksiyasidan foydalanib algoritm lash imkoniyati yaratildi. Ular buyruqlarni mashina tiliga interpretatorlar va kompilyatorlar yordamida o'tkazar edi.

- quyi darajadagi;
- oʻrta darajadagi;
- yuqori darajadagi.

Quyida darajadagi algoritmlash tillari bevosita kompyuter qurilmalari bilan bog'liq bo'lib, buyruqlar ularning kodlari bilan yoziladi. Bu kabi buyruqlardan tashkil topgan algoritmlar katta hajmli bo'lib, ularni taxrirlash katta mehnat talab qiladi. Dastlabki kompyuterlar (ENIAC, MESM va boshqalar) ana shunday tillarda ishlagan.

2

Yuqori darajadagi algoritmash tillari quyidagicha bosqichlarga bo'linadi:

Algoritmik(Basic, Pascal, C va b.)

Mantiqiy(Prolog, Lisp va b.)

Obyektga mo'ljallangan(Object Pascal, C++, Java va b.)

Algoritmash tillarida yaratilgan algoritmash mashina tiliga *translyatorlar* yordamida o'tkaziladi.

Translyator(translator-tarjimon) biror bir algoritmash tilida yozilgan algoritmni mashina tiliga tarjima qiladi.

Translyatorlar ikki turda bo'ladi:

- Kompilyatorlar(*compiler*-yig'uvchi) biror bir algoritmash tilida yozilgan algoritmni mashina tiliga to'liq o'qib olib tarjima qiladi;

- Interpretatorlar(*interpreter* - izohlovchi, og'zaki tarjimon) biror bir algoritmash tilida yozilgan algoritmni mashina tiliga satrma - satr tarjima qiladi.

Translyatorlarni bu ikkala turi - bir biridan farq qiladi. Kompilyatsiya qilingan algoritmash birmuncha kam vaqt talab etadi, ya'ni tezroq ishlaydi, lekin interpretatsiya qilingan algoritmashni o'zgartirish osonroq hisoblanadi.

C++ dasturlash tili tarkibida bir nechta imkoniyatlar mavjud, ya'ni consol rejimi, forma ob'yekt rejimi, grafik muhiti va ma'lumotlar bazasi bilan ishlash imkoniyatlari keng joriy etilgan. Ushbu qo'llanmada keltirilgan misol va masalalarning yechimi dasturining *int main* funksiyasi tarkibini C++ dasturlash tilining ixtiyoriy versiyalarida ishlatib ko'rish mumkin. Qo'llanma oliy o'quv yurtlari talabalari va magistrantlari, litsey, kasb-hunar kollejlari talabalari hamda mustaqil o'rganuvchilar uchun qulay manba hisoblanadi.

I-BOB. ALGORITMLAR

1.1 Algoritm tushunchasi va uning xossalari

Algoritm aslida biror masalani yechish yoki biror ishni bajarish rejasi, lib, rejadani farqi u aniq tugallangan qadamlarga bo'linadi va buyruq usulida yoziladi.

Misol 1: Kitobdan kerakli mavzuni topish algoritmini tuzing

- 1) kitobning mundarijasi yozilgan sahifasi ochilsin;
- 2) mundarijadan izlanayotgan mavzu topilsin;
- 3) mavzu yozilgan sahifa aniqlansin;
- 4) aniqlangan sahifa ochilsin.

Misol 2: Uchburchak chizish algoritmini tuzing.

- 1) qog'ozda uchta bir to'g'ri chiziqli yotmagan uchta nuqta belgilansin;
- 2) belgilangan nuqtalar chizg'ich yordamida bir birlari bilan tutashtirilsin;
- 3) hosil bo'lgan figura uchburchak deb qabul qilinsin.

Misol 3: Choy damlash algoritmini tuzing.

- 1) suv qaynatiladigan choynakga kerakli miqdorda suv solinsin;
- 2) gaz o'chog'ida olov yoqilsin;
- 3) suv solingan choynak olovga qo'yilsin;
- 4) suv qaynaguncha kutilsin;
- 5) chinni choynakga bir choy qoshiq quruq choy solinsin;
- 6) chinni choynak qaynagan suv bilan to'ldirilsin.

Ko'rinib turibdiki, algoritm aniq tugallangan qadamlarga bo'lingan, lib, bu qadamlarning har biri algoritmning buyruqlari deyiladi.

Biz kelajakda algoritmashni ifodalashning(yozishning) uch xil usuli bilan tanishamiz:

- 1)matn usuli;
- 2)blok-sxema usuli;
- 3)dastur ko'rinishida ifodalash usuli.

Yuqorida yozilgan barcha algoritmash matn usulida yozildi. Demak, algoritmashni matn usulida yozish uchun uning har bir buyrug'i nomlanadi va nomeridan keyin qavs qo'yiladi hamda buyruqlar kichik harflar bilan yoziladi, buyruq oxirida (;) qo'yiladi. Faqat oxirgi buyruq tugagach (.) qo'yiladi.

Matematik ifodalarning qiymatlarini hisoblash algoritmashni ham matn usulida ifodalash mumkin.

Misol 4. $y=3x^2-7x+28$ funksiya qiymatini hisoblash algoritmini tuzing.

- 1) x ning qiymati kiritilsin;
- 2) 3 ni x^2 ga ko'paytirib, natija a_1 bilan belgilansin;
- 3) -7 ni x ga ko'paytirib, natija a_2 bilan belgilansin;
- 4) a_2 ga 28 qo'shib, natija a_3 bilan belgilansin;
- 5) a_1 va a_3 larning yig'indisi funksiyaning qiymati deb qabul qilinsin.

Ushbu algoritmnı quyidagi *matn ko'rinishida* ham ifodalash mumkin:

- 1) x ning qiymati kiritilsin;
- 2) $a_1 := 3 * x^2$;
- 3) $a_2 := -7 * x$;
- 4) $a_3 := a_2 + 28$;
- 5) $y := a_1 + a_3$.

Algoritmnı *har bir qadami uning buyrug'i* deyiladi. Bu algoritmda 5 ta buyruq bor. Algoritm larni ifodalashning (yozishning) bu usulini ba'zi adabiyotlarda algoritm larni ifodalashning *formula usuli* ham deyishadi.

Algoritm tuzuvchi shu algoritmnı *muallifi*, uni ijro etuvchi esa uning *ijrochisi* deyiladi. Xususiyl holda muallifning o'zi ham algoritm ijrochisi bo'lishi mumkin. Algoritm ijrochisi kitoblarda "Algoritm ijrochisi algoritm buyruqlari va ko'rsatmalarini bajara oladigan abstrakt va real (texnik yoki biologik) sistema" deb ta'riflangan.

Algoritm shunday tuzilgan bo'lishi kerakki, ijrochi uni hech qanday fikrlamasdan, *ko'r-ko'rona* ijro etganda maqsadga erishsin. Algoritmning bu xossasi *formal ijro etish mumkinlik* xossasi deyiladi.

Qachon algoritmnı formal ijro etish mumkin bo'ladi? Buning uchun algoritm quyidagi 5 ta xossaga ega bo'lishi kerak.

1°. *Diskretlik* (uzluklilik) xossasi – bu xossaga ko'ra algoritm tugallangan aniq qadamlarga bo'lingan bo'lishi kerak.

2°. *Aniqlik* xossasi – bu xossaga ko'ra qaysi buyriqdan keyin qaysi buyruqning bajarilishi aniq ko'rsatilgan bo'lishi lozim (masalan, buyruqlarni nomerlash yo'li bilan).

3°. *Tushunarlik* xossasi – bu xossaga ko'ra ijrochi ijro eta olmaydigan har qanday buyruq u uchun tushunarsiz hisoblanadi. Ko'pincha bilim yetmasligi tufayli buyruq tushunarsiz bo'ladi. Masalan, 5,6-sinf o'quvchilari kvadrat ildizni hisoblash, berilgan funksiya nollarini topish, sonning logarifmini hisoblash haqidagi buyruqlarni tushunmaydi. Chunki bunday tushunchalar ularga notanish, ya'ni bilimi etmaypdi. Ammo *algoritm lashtirish va dasturlashda nolga bo'lish, manfiy sondan kvadrat ildiz chiqarish, manfiy sonning logarifmini*

hisoblash kabi bajarib bo'lmaydigan buyruqlar ham tushunarsiz hisoblanadi. Demak tushunarsizlik faqat bilim yetmaslik emas ekan.

4°. *Natijaviylik* xossasi – bu xossaga ko'ra sikli qadamlardan keyin ijrochi maqsadiga erishishi kerak. Masalan, 3-misolning algoritmi tarkibida "choynakka suv solinsin" kabi buyruq bo'lsa natijaviylik xossasi buziladi. Chunki ijrochi choynakka abadiy suv solib o'tiraveradi.

5°. *Ommaviylik* xossasi – bu xossaga ko'ra bitta tuzilgan algoritm o'z ichida bir turga mansub xohlagancha masalalarni yechish mumkin bo'lishi kerak.

Masalan; 1- Algoritmda birinchi buyruqni "algebra kitobining undarilishi ochilsin" deb o'zgartirsak, algebra kitobidan boshqa kitobdan boshqa mavzuni topib bo'lmaydi.

"Algoritm" so'zi buyuk alloma Al-Xorazmiy nomi bilan bogliq. Al-Xorazmiy nomining o'zgartirilgan shakli. U birinchi marta 4-asrda arifmetik amallarning bajarilish tartibini bayon qilgan. Yevropada bu usulni arifmetik amallarni bajarishning Al-Xorazmiy usuli deb atashdi va boshqa ishlarni ham bunday aniq va puxta rejalashtirish jarayonini Al-Xorazmiy nomi bilan bog'lay boshlashdi. Shundan so'ng vaqt o'tishi bilan Al-Xorazmiy usuli, keyinchalik Algorizm usuli kabi o'zgarib, yillar o'tishi bilan allomaning o'z Vataniga "Algoritm" bo'lib qaytib kelgan.

Qiymat berish (ta'minlash yoki o'zlashtirish) buyrug'i

Matematikada $x=5$ yozuvini x 5 ga teng, $x=x+1$ yozuvini x $x+1$ ga teng deb o'qiymiz. Lekin algoritm lashtirishda o'zgaruvchiga beriladigan qiymat belgi orqali beriladi. Masalan, $x:=5$ yozuvi x ga 5 qiymat berilsin deb o'qiladi, $x:=x+5$ yozuvi esa x ga $x+5$ qiymat berilsin yoki x $x+5$ qiymat bilan taminlansin deb o'qiladi.

Informatikada bitta o'zgaruvchi ketma-ket bir necha qiymatlar olsa eng birinchi olgan qiymati qolib qolganlari o'chib ketadi. Buni quydagicha tasavvur qilamiz. Har bir o'zgaruvchi uchun daftarning bitta katakchasi ajratilgan va o'zgaruvchining har bir qiymati katakcha kattaligidagi bog'ozga yozib o'sha katakchaga yopishtirib qo'yiladi deb tushuning. Natijada oldingilari ko'rinmay qoladi. Masalan, quyidagi buyruqlar bajarilgach a va b lar qanday qiymatlar oladi:

$a:=4$, $b:=27$; $a:=\sqrt[3]{b}$; $b:=a*b$; $a:=b/3$; $b:=b+a$; $a:=b/a$;

Yechish:

a	b
4	27
3	81

Javob: $a=4$; $b=108$.

Mustaqil ishlash uchun topshiriqlar

Quyidagi masalalarning algoritmlarini tuzing:

1. Aylana chizish;
2. Lug'atdan so'z tarjimasini topish;
3. Kofe tayyorlash;
4. Biror taomni tayyorlash;
5. Telefondan foydalanish;
6. Dars tayyorlash;
7. Misol va masalalarni yechish algoritmlarini tuzing.
8. $(12a+b-8)/(4x^3-23/(x+1)+c)+5x-28a$ ifodaning qiymatini hisoblash algoritmlarini matn usulida tuzing va har bir algoritmda 5 ta xossa qanday namoyon bolayotganligini va ulardan hohlagan biri buzilsa algoritm qanday ko'rinishda bolishini aytib bering. olishga

Quyidagi buyruqlar bajarilgach o'zgaruvchilar qanday qiymatlar oladi?

- 1) $a:=12$; $b:=4$; $c:=(a+b)/8$; $a:=a+c$; $c:=a+c$; $b:=b-c$; $b:=4b/c$;
- 2) $x:=2$; $y:=5$; $y:=2x+y$; $y:=2x+y$; $x:=2x+y$; $y:=2x-3$; $x:=x-y$;
- 3) $b:=-18$; $c:=8$; $d:=3$; $c:=|d-c|(d+b)$; $c:=|d-c|(d+1)$; $d:=|d-c|(d-1)$; $b:=|d-c|(d+1)$;
- 4) $q:=4$; $q:=q+2$; $q:=q+2$; $q:=q+2$; $q:=q+2$; $q:=q+2$; $q:=q+2$;

Quyidagi savollarga javob bering

1. Algoritm tushunchasiga ta'rif bering.
2. Algoritmning nechta va qanday xossalari bor?
3. Algoritmning rejadani farqi nimada?
4. Algoritmning ijrochisi va muallifi kim?
5. Algoritmning ommaviylik xossasini tushuntiring.
6. Algoritmning natijaviylik xossasini tushuntiring.
7. Algoritmning diskritlik xossasini tushuntiring.
8. Algoritmning aniqlik xossasini tushuntiring.
9. Algoritmning tushunarlik xossasini tushuntiring.
10. Algoritm so'zi kimning nomi bilan bog'liq? Nima uchun?
11. Qiymat berish buyrug'i qanday yoziladi?
12. Algoritmni formal ijro etishning mohiyatini tushuntiring.
13. Algoritm qachon to'g'ri tuzilgan hisoblanadi?
14. $p:=2p+3$ yozuvi qanday o'qiladi?
15. Algoritm matn usulida qanday yoziladi?

18. Algoritmning buyrug'i deyilganda nimani tushunasiz?

1.2 Algoritmning berilish usullari va uning turlari

Yuqorida qayd qilganimizdek, algoritmlarni ifodalashning (berilishining) quyidagi 3 ta usulidan foydalaniladi:

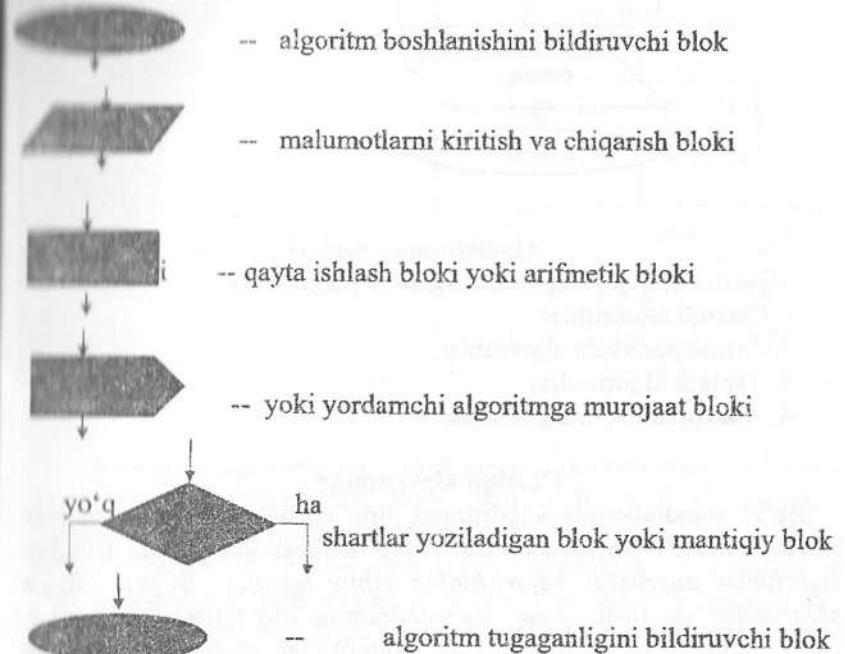
1. Matn usulida berilishi

2. Blok-sxema ko'rinishida berilishi

3. Dastur ko'rinishida berilishi (Ushbu ko'rinish ijrochi EHMga mo'ljallangan bo'lib, bu ko'rinishi bilan dasturlash asoslarini o'rganishda tanishamiz).

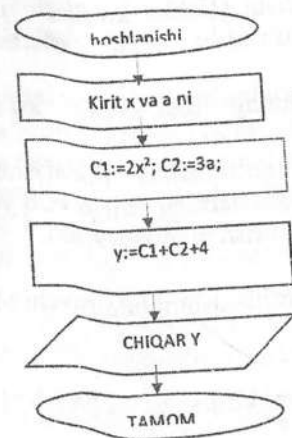
Bu algoritmlarni ifodalashning blok-sxema ko'rinishi bilan tanishamiz.

Algoritmlarni blok-sxema ko'rinishida ifodalashda buyruqlar turiga qarab, tayin geometrik figuralarning ichiga olib yoziladi va bu figuralarni boshqasi bilan almashtirish mumkin emas.



Bu bloklar strekali siniq chiziqlar bilan birlashtirilgan bo'ladi va bu siniq chiziqlar algoritmnining qaysi buyrug'idan keyin qaysi buyrug'i bajarilishi kerakligini ko'rsatadi (ya'ni algoritmnining aniqlik xossasini ifodalaydi).

Misol. $y=2x^2+3a+4$ funksiyaning qiymatini hisoblash algoritmini tuzing.



Algoritmnining turlari

Algoritmnining quyidagi 4 ta turidan foydalanamiz.

1. Chiziqli algoritmlar
2. Tarmoqlanuvchi algoritmlar
3. Tanlash algoritmlari
4. Takrorlanuvchi algoritmlar

Chiziqli algoritmlar

Ba'zi masalalarning algoritmini ijro etishda uning buyruqlari qanday tartibda berilgan bo'lsa, o'sha tartibda ijro etiladi. Bunday algoritmlar buyruqlar bajarishining tabiiy tartibiga bo'ysunadigan algoritmlar deyiladi. Agar algoritmlarning buyruqlarini bajarish tabiiy tartibga bo'ysunsa, bunday algoritmlar chiziqli algoritmlar deyiladi. Biz to hozirga qadar o'rgangan barcha algoritmlar chiziqli algoritmlardir.

Demak, algoritm chiziqli bo'lishi uchun uning tarkibida hech qanday shartlar bo'lmagani kerak. Har qanday masala algoritmini ham uch turli buyruqlar, ya'ni chiziqli, tarmoqli va takrorlanuvchi buyruqlar yordamida tasvirlash mumkin.

Misol: Quyidagi funksiyaning qiymatini hisoblash algoritmini keltiring.

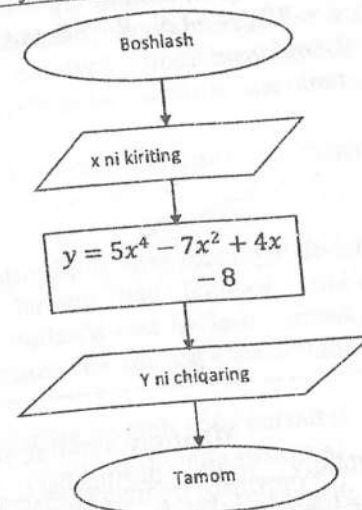
$$y = 5x^4 - 7x^2 + 4x - 8$$

Bu masalaning algoritmi quyida matn va blok-sxema ko'rinishlarida tasvirlangan.

Algoritmnı so'zlar yordamida tasvirlanishi

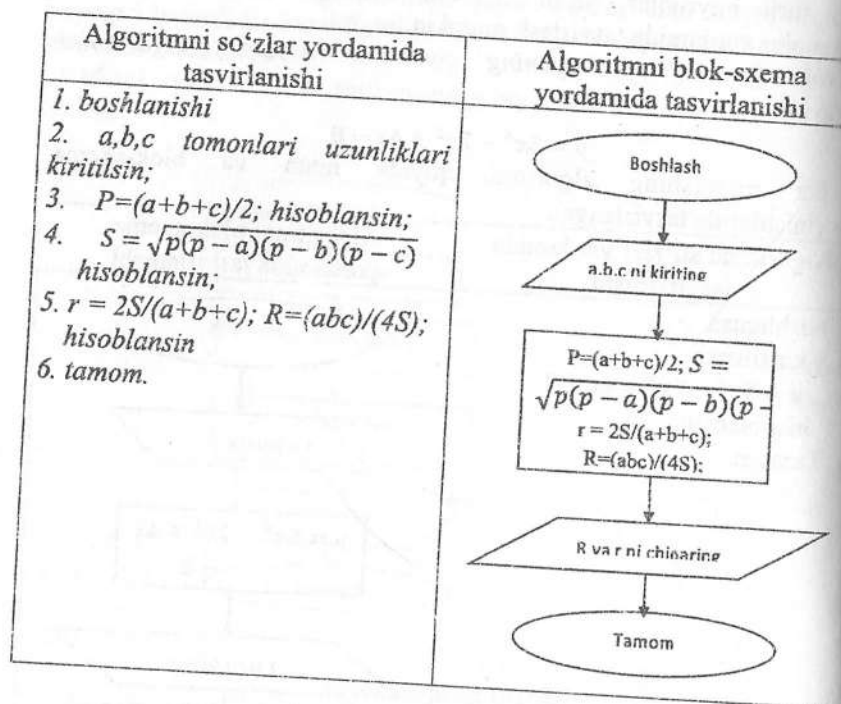
1. boshlanish
2. x kiritilsin;
3. $y = 5x^4 - 7x^2 + 4x - 8$ hisoblansin;
4. Tamom.

Algoritmnı blok-sxema yordamida tasvirlanishi



Masala algoritmlarini blok-sxema ko'rinishida ishlashni yaxshilab o'zlashtirib olish zarur, chunki bu usul algoritmlarni ifodalashning qulay vositalaridan biri bo'lib, dastur tuzishni yengillashtiradi, o'quvchilarning dasturlash qobiliyatini shakllantirishda muhim ahamiyatga ega. Dasturlash tillarda blok - sxemaning asosiy bloklariga maxsus operatorlar mos keladi. Shuni aytish kerakki, blok-sxemalardagi yozuvlar odatdagi yozuvlardan katta farq qilmaydi. Aslida dastur ham algoritmnıng bir ko'rinishi bo'lib, u ijrochi kompyuterga mo'ljallangan algoritmdir.

Misol: Uchburchak tomonlari berilganda unga ichki va tashqi chizilgan aylanalar radiuslarini hisoblash algoritmi tuzilsin.



Mantiqiy amallar va mantiqiy ifodalar

Mantiqiy ifoda tushunchasi matematik mantiq(logika) fani tushunchlariga tayanib tushuntiriladi. Matematik mantiq deganda fikrlashning shakli va qonunlari tushuniladi. Matematik mantiq tushunchasiga Al Fargʻoniy, Abu Ali ibn Sino va keyinchalik ingliz olimi Jorj Bull tomonidan asos solingan.

Har qanday mantiqiy ifoda yoki mantiqiy fikr faqat ikki xil qiymat qabul qilishi mumkin. Bular rost(true) yoki yolgʻon (false). Masalan Sobir qiz bola, poʻlat suyuq modda kabi fikrlarning qiymati yolgʻon yoki suv oʻda muzlaydi, ruchka yozish quroli kabi fikrlar rost. Matematik mantiqiy ifodalarni yozishda quyidagi munosabat belgilaridan foydalanadi:

$$=, \neq, <, >, \geq, \leq;$$

(1)

Agar ikkita ifoda (1) munosabat belgilarining biri bilan birlashtirilgan boʻlsa, bunday yozuv mantiqiy ifoda deyiladi. Masalan, 1) $3+4=20+5$; 2) $x+5 \neq 2x-1$; 3) $26/2 \cdot 3 \neq 39$; 4) $49/7+3=2 \cdot 5$;

$$5) a+b=b+c;$$

Bu yozuvlarning har biri mantiqiy ifodalar boʻlib, ulardan 1 va 3- ifodalarning qiymati yolgʻon, 4-ifodaning qiymati rost, 2 va 5- ifodalarning qiymati oʻzgaruvchilarning qiymatiga qarab rost yoki yolgʻon boʻladi. Yuqorida qayd qilganimizdek, mantiqiy ifodalar faqat ikki xil qiymatlar qabul qiladi, yaʼni **rost** yoki **yolgʻon**.

Agar ikkita ifoda *bitta munosabat belgisi orqali* birlashtirilgan boʻlsa, bunday mantiqiy ifodalar *oddiy mantiqiy ifodalar* deyiladi. Agar mantiqiy ifoda tarkibida *mantiqiy amallardan* ham foydalanilgan boʻlsa, bunday mantiqiy ifodalar *murakkab mantiqiy ifodalar* deyiladi. Murakkab mantiqiy ifodalar tarkibida quyidagicha mantiqiy amallardan foydalaniladi. ($\neg$ va $\wedge$)

1. Mantiqiy qoʻshish amali. Amalning umumiy koʻrinishi quyidagicha

$$a \vee b$$

Bunda a va b lar mantiqiy oʻzgaruvchilar yoki mantiqiy ifodalar boʻlib, bu amal natijasi a va b larning har ikkalasi yolgʻon boʻlgandagina yolgʻon, qolgan barcha hollarda rost boʻladi. Demak, $\vee$ belgisi *yoki* soʻzini almashtiradi yoki matematik mantiq elementlaridagi *konyunksiya amaliga* mos keladi.

Masalan: $(3 > 7) \vee (2 \cdot 18 = 37)$ – ifodaning qiymati yolgʻon(false)

$(25 - 14 \neq 11) \vee (45/9 = 5)$ – ifodaning qiymati rost(true)

$\text{true} \vee (29 - 19 = 10)$ – ifodaning qiymati rost(true)

Bundan tashqari mantiqiy ifodaning qiymatini mantiqiy oʻzgaruvchiga qiymat sifatida berish ham mumkin. Masalan,

$a = 13 - 5 > 12$; $b = (56/8 - 7 < 2) \vee \text{true}$; $c = \text{false} \vee a$; boʻlsa, $a \vee b \vee c \vee (64/8 > 10)$ mantiqiy ifodaning qiymati rost boʻladi.

2. Mantiqiy koʻpaytirish amali. Amalning umumiy koʻrinishi quyidagicha

$$a \wedge b$$

Bunda a va b lar mantiqiy oʻzgaruvchilar yoki mantiqiy ifodalar boʻlib, bu amal natijasi a va b larning har ikkalasi chin boʻlgandagina chin, qolgan barcha hollarda yolgʻon boʻladi. Demak, $\wedge$ belgisi *va* soʻzini almashtiradi yoki matematik mantiq elementlaridagi *konyunksiya amaliga* mos keladi. Masalan:

$(3 > 7) \wedge (2 * 18 = 37)$ ifodaning qiymati yolg'on
 $(25 - 14 = 11) \wedge (45 / 9 = 5)$ – ifodaning qiymati rost
 $(13 > -12) \wedge (2 * 18 = 37)$ ifodaning qiymati yolg'on

3. **Mantiqiy inkor amali.** Amalning umumiy ko'rinishi quyidagicha

$\neg a$

Bunda a mantiqiy o'zgaruvchi yoki mantiqiy ifoda bo'lib, bu amal natijasi a ning qiymati yolg'on bo'lganda rost, rost bo'lganda yolg'on bo'ladi, ya'ni bu amal a mulohazani inkor qiladi. Demak, $\neg$ belgisi *emas* so'zini almashtiradi.

Masalan:

- 1) $(3 > 7) \wedge \neg (2 * 18 = 37)$ ifodaning qiymati yolg'on
- 2) $\neg (25 - 14 = 11) \wedge (45 / 9 = 5)$ ifodaning qiymati rost
- 3) $(13 > -12) \wedge \neg (2 * 18 = 37)$ ifodaning qiymati rost
- 4) $\neg \text{true} \vee \neg \text{false} \wedge \neg \text{true} \wedge \neg \text{false}$ ifodaning qiymati yolg'on

Eslatma: Biz bilgan amallar ifodada aralash holda kelsa, ularning birinchi bajarilishi nuqtai nazaridan ustunlik darajasi quyidagicha:

$\neg, (*, /), \wedge, (+, -), \vee, (=, \neq), (>, <, \geq, \leq)$

Bunda qavsga olingan amallar bir bosqich amallari hisoblanadi. Tabiiyki, matematikada bo'lgani singari ifodadagi qavslar amallar bajarilish tartibina buzadi. Masalan,

14 1 2 3 11 4 5 13 10 6 7 12 8 9
 $\neg ((13 * 5 - 25 = 40) \wedge (13 - 3 = 10)) \vee \neg (5 + 7 = 12) \wedge (2 * 9 = 20));$

Murakkab buyruqlar

Tarmoqlanuvchi, tanlash va takrorlanuvchi buyruqlar tarkibida shartlar bo'ladi va shu shartlarning qiymati rost yoki yolg'onligiga qarab u yoki bu buyruqlar ketma-ketligi bajariladi. Tarkibida tarmoqlanish, tanlash yoki takrorlanish buyruqlari bo'lgan buyruqlar *murakkab buyruqlar deyiladi*. Tarkibida murakkab buyruqlar bo'lgan algoritmlar esa *murakkab algoritmlar deyiladi*.

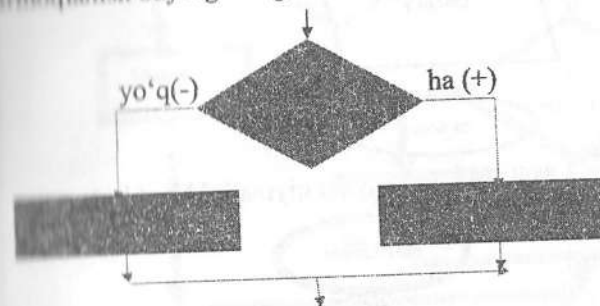
Endi algoritmlarning ikkinchi turi, ya'ni tarmoqlanuvchi algoritmlar bilan tanishamiz. O'z tarkibida tarmoqlanish buyrug'ini saqlovchi algoritmlar *tarmoqlanuvchi algoritmlar deyiladi*.

Tarmoqlanish buyrug'i

Yuqorida qayd qilganimizdek, chiziqli algoritmlar eng sodda algoritmlar bo'lib, bunday algoritmlarni tuzish, uni ijro etish algoritmi

bilan ham, ijrochidan ham alohida malaka talab qilmaydi. Lekin masalalarning algoritmlarini ham chiziqli algoritmlardan tuzish bo'lmaydi. Amalda qo'llaniladigan masalalarning algoritmlarini tuzishda tarmoqlanish, tanlash yoki takrorlanish buyruqlaridan foydalaniladi. Shunday masalalar borki, algoritmlarida qo'yilgan mantiqiy ifodaga bog'liq holda u yoki bu buyruqlar ketma-ketligini bajarishga to'g'ri keladi. Shunday holda tarmoqlanish buyrug'idan foydalaniladi. Masalan, tarmoqda elektr chirog'i yonadi, aks holda yonmaydi yoki ruchkada tug'iriladi, aks holda yozmaydi yoki matematika fani o'qitiladi, aks holda yozmaydi va hokazo.

Tarmoqlanish buyrug'ining umumiy ko'rinishi quyidagicha

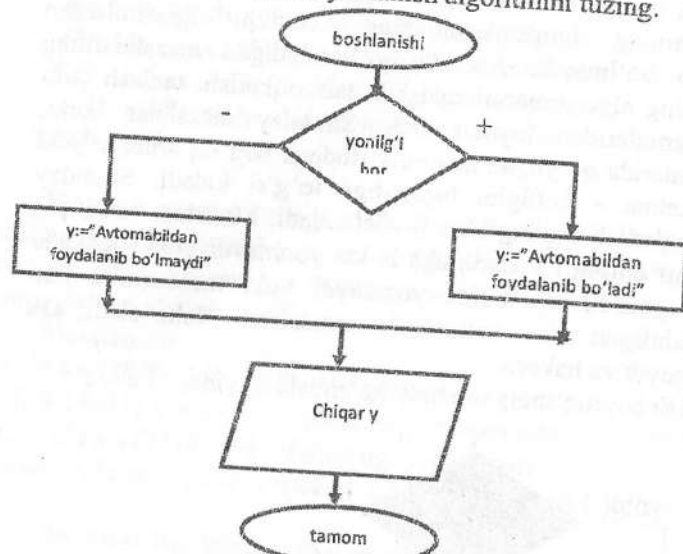


Buyruq quyidagi tartibda bajariladi:

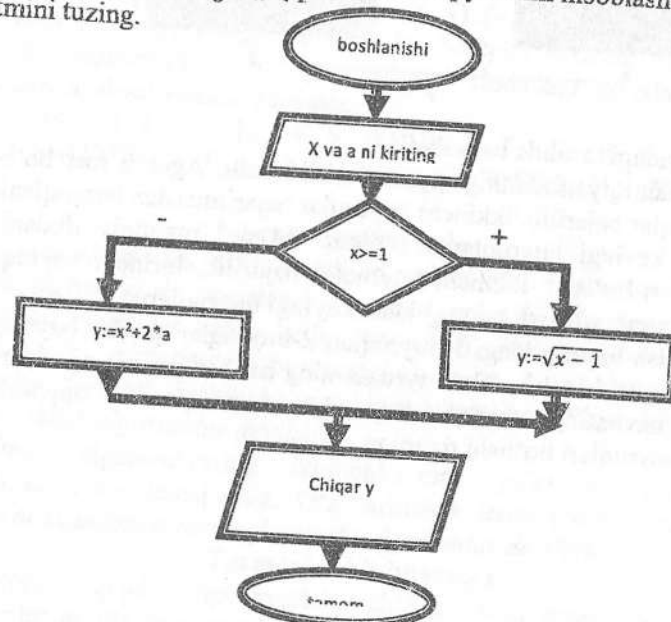
Dastlab mantiqiy ifodaning qiymati tekshiriladi. Agar u rost bo'lsa, birinchi buyruqlar bajarilib, ikkinchi buyruqlar bajarilmasdan tarmoqlanish buyrug'idan keyingi buyruqlarga o'tiladi. Agar mantiqiy ifodaning qiymati yolg'on bo'lsa, ikkinchi buyruqlar bajarilib, birinchi buyruqlar bajarilmasdan tarmoqlanish buyrug'idan keyingi buyruqlarga o'tiladi.

Tarmoqlanish buyrug'idagi 1-buyruqlar, 2-buyruqlar qanday buyruqlar ekanligi savol tug'ilishi tabiiy. Bu buyruqlarning har biri masala xarakteriga qarab, o'z navbatida chiziqli buyruqlar, tarmoqlanish buyruqlari, takrorlanish buyruqlari bo'lishi mumkin.

Misol 1: Avtomabildan foydalanish algoritmini tuzing.

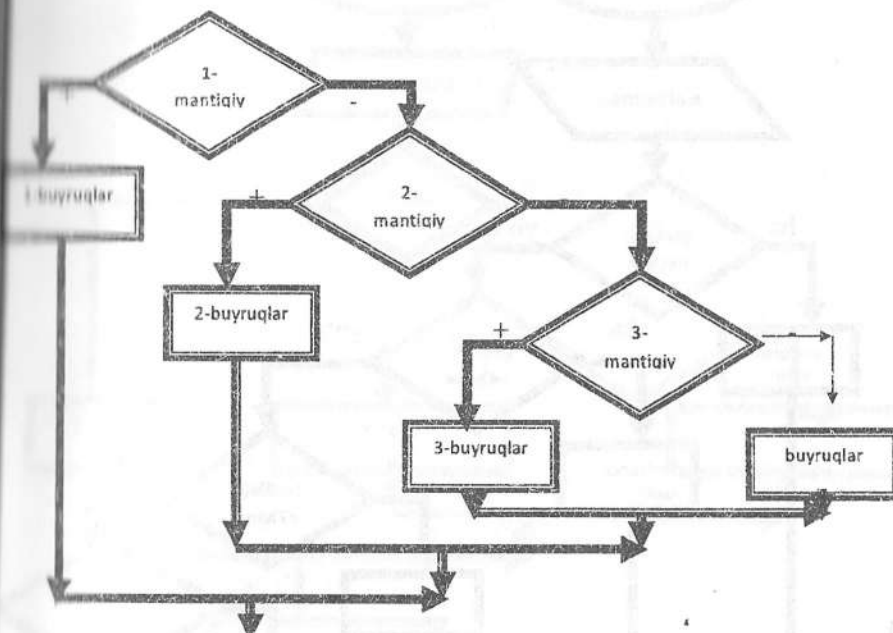


Misol 2: $y = \begin{cases} \sqrt{x-1} & \text{agar } x \geq 1 \\ x^2 + 2a & \text{agar } x < 1 \end{cases}$ funksiyani qiymatini hisoblash algoritmini tuzing.



Tanlash buyrug'i

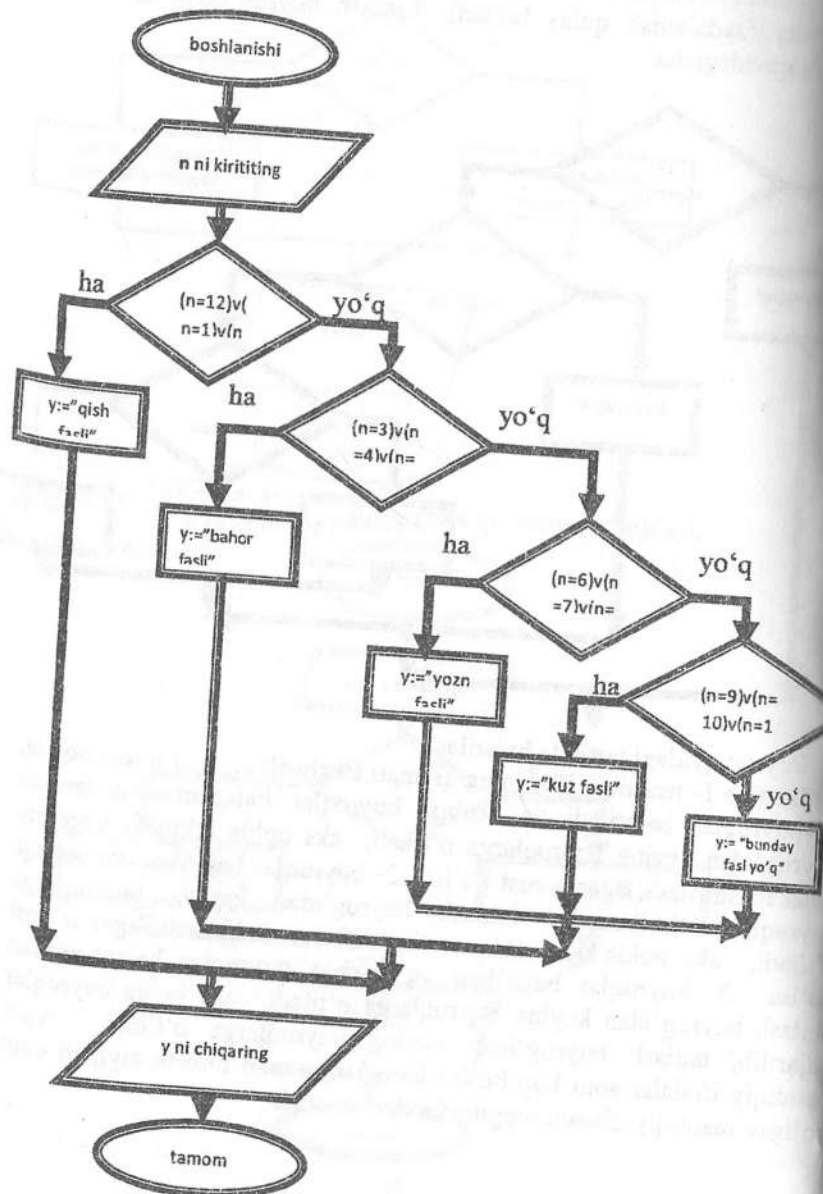
Agar masala tarkibida shartlar soni ikki yoki undan ortiq bo'lsa tanlash buyrug'idan foydalanish qulay bo'ladi. Tanlash buyrug'ining umumiy shakli quyidagicha



Buyruq quyidagi tartibda bajariladi:

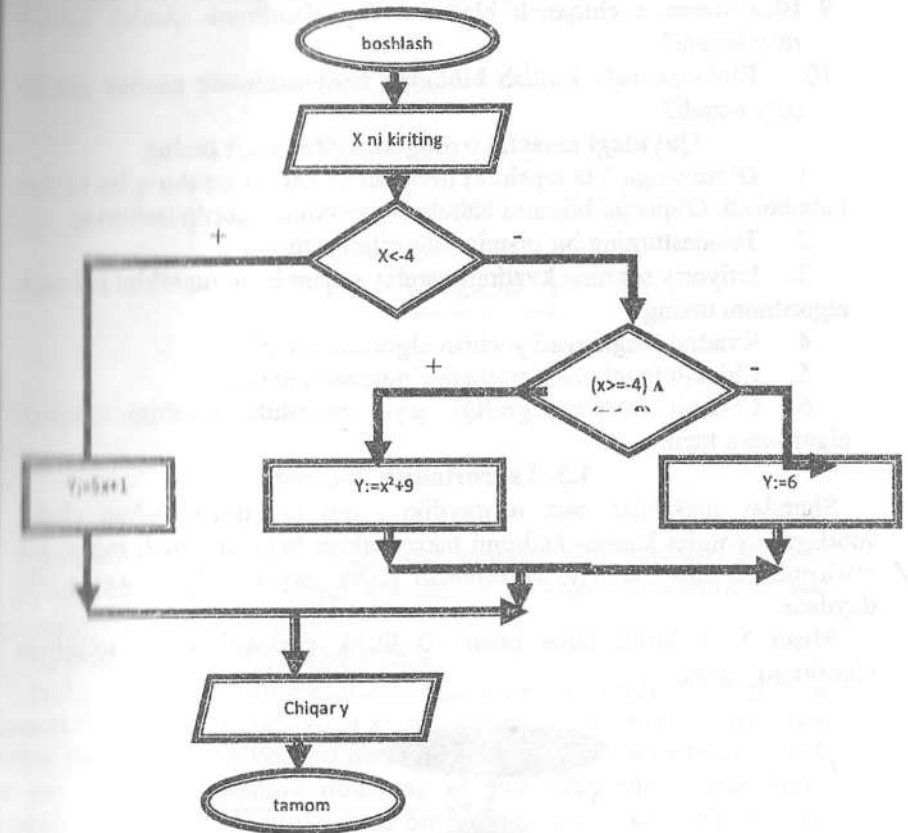
Dastlab 1- mantiqiy ifodaning qiymati tekshiriladi, agar u rost bo'lsa, 1- buyruqlar bajariladi va boshqa buyruqlar bajarilmasdan tanlash buyrug'idan keying buyruqlarga o'tiladi, aks holda ikkinchi mantiqiy ifoda tekshiriladi, agar u rost bo'lsa 2- buyruqlar bajariladi va boshqa buyruqlar bajarilmasdan tanlash buyrug'idan keying buyruqlarga o'tiladi, aks holda uchinchi mantiqiy ifoda tekshiriladi, agar u rost bo'lsa 3- buyruqlar bajariladi va boshqa buyruqlar bajarilmasdan tanlash buyrug'idan keying buyruqlarga o'tiladi, aks holda buyruqlar bajarilib, tanlash buyrug'idan keying buyruqlarga o'tiladi. Agar mantiqiy ifodalar soni kop bo'lsa ham bu jarayon birorta qiymati chin bo'lgan mantiqiy ifodani topguncha davom etadi.

Misol 1. Yil oylarining qaysi faslga tegishli ekanligini aniqlash algoritmini tuzing.



Misol 2.
$$y = \begin{cases} 5x + 1 & \text{agar } x < -4 \\ x^2 + 9 & \text{agar } -4 \leq x \leq 8 \\ 6 & \text{agar } x > 8 \end{cases}$$
 funksiyani tuzing

funksiyaning qiymatini



Quyidagi savollarga javob bering

1. Qanday masalalarning algoritmlarini tuzishda tarmoqlanish buyrug'dan foydalaniladi?
2. Qanday masalalarning algoritmlarini tuzishda tanlash buyrug'idan foydalaniladi?
3. Qanday algoritmlar tarmoqlanish algoritmlari deyiladi?
4. Tarmoqlanish buyrug'i qanday bajariladi?
5. Tanlash buyrug'i qanday bajariladi?
6. Tarmoqlanish va tanlash buyruqlari o'rtasida qanday bog'lanish bor?

7. Blok-sxemada strekali siniq chiziqlardan qanday maqsad foydalaniladi?
8. Blok-sxemaning har bir blokinini chizib, uuning vazifasi tushuntiring.
9. Blok-sxemada chiqarish blokidan foydalanilmasa qanday xatol ro'y beradi?
10. Blok-sxemada kiritish blokidan foydalanilmasa qanday xatol ro'y beradi?

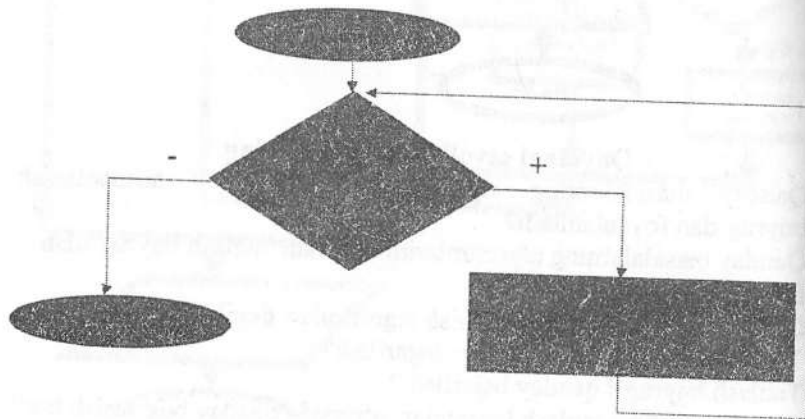
Quyidagi masalalarning algoritmlarini tuzing

1. O'quvchiga 5 ta topshiriq berilgan va har bir topshiriq bir balda baholanadi. O'quvchi bilimni baholash mezonini algoritmlashtiring
2. Teledasturning bir qismini algoritmlashtiring
3. Ixtiyoriy sonning kvadrati qanday raqam bilan tugashini aniqlash algoritmini tuzing
4. Kvadrat tenglamani yechish algoritmini tuzing
5. Chiziqli tenglamani yechish algoritmini tuzing
6. Chiziqli funksiya grafigi qaysi chorakda yotishini aniqlash algoritmini tuzing

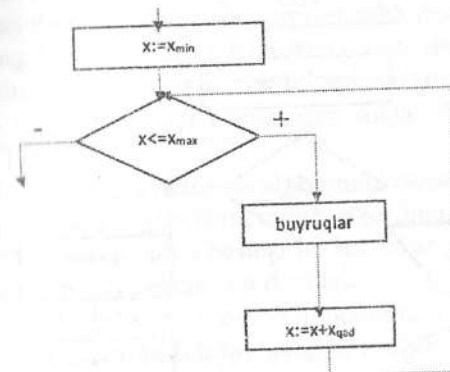
1.3. Takrorlanish buyrug'i

Shunday masalalar ham uchraydiki, uni hal qilish uchun chekl sonidagi buyruqlar ketma-ketligini takror-takror bajarish orqali maqsadga erishamiz. Bunday davriy takrorlanish siklik jaryon yoki qisqacha sikl deyiladi.

Misol 1. 1 litrlik idish bilan 10 litrlik chelakni suvga to'ldirish algoritmini tuzing.



Topshiriq 4-5 ta hayotiy masalalar tanlab algoritm tuzing. Bu amalda ko'proq matematik masalalarni yechishda sikl buyrug'idan foydalaniladi. Buyruqning umumiy ko'rinishi quyidagicha:

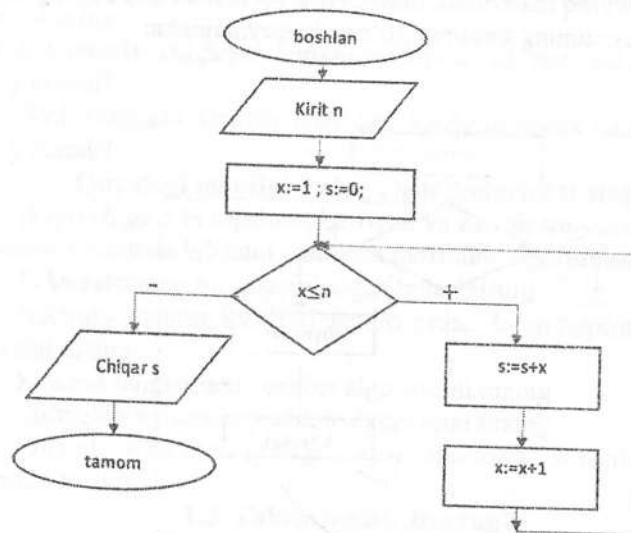


Hunda x -sikl parametri, $x_{\min}$ -sikl parametrining dastlabki qiymati, $x_{\max}$ -sikl parametrining oxirgi qiymati, x_{qad} -har takrorlanishda sikl parametri o'zgaradigan qadam.

Buyruq quyidagi tartibda bajariladi:

Sikl parametri o'zining dastlabki $x_{\min}$ qiymatini oladi va oxirgi $x_{\max}$ qiymati bilan solishtiriladi. Agar $x \leq x_{\max}$ mantiqiy ifodaning qiymati rost bo'lsa, buyruqlar bajarilib, sikl parametrining qiymati bir qadamga ortadi va yana $x \leq x_{\max}$ mantiqiy ifodaning qiymati tekshiriladi, agar uning qiymati rost bo'lsa, buyruqlar yana bir bor bajarilib, sikl parametri bir qadamga ortadi va $x \leq x_{\max}$ mantiqiy ifodaning qiymati tekshiriladi va shunday qilib bu jarayon $x \leq x_{\max}$ mantiqiy ifodaning qiymati yolg'on, ya'ni $x > x_{\max}$ bo'lguncha davom etadi va shundan so'ng sikl buyrug'idan keyingi buyruqlarga o'tiladi.

Misol 2. 1 dan n gacha bo'lgan natural sonlar yig'indisini topish algoritmini tuzing.



Quyidagi savollarga javob bering

- 1) Sikl buyrug'ining umumiy ko'rinishini yozing va bajarilish tartibini tushuntiring
- 2) Sikl buyrug'ining parametri, boshlang'ich va oxirgi qiymatlari, qadami deyilganda nimani tushunasiz?
- 3) Yuqoridagi algoritmda $s:=0$ buyrug'i nega berilgan?
- 4) Yuqoridagi algoritmda qanday o'zgarish qilsak u toq yoki juft sonlar yig'indisini hisoblaydi?
- 5) Yuqoridagi algoritmda qanday o'zgarish qilsak u juft sonlar ko'paytmasini hisoblaydi?
- 6) Yuqoridagi algoritmda qanday o'zgarish qilsak u 7 ga karrali sonlar ko'paytmasini hisoblaydi?
- 7) Yuqoridagi algoritmda qanday o'zgarish qilsak, cheksiz takrorlanish sodir bo'ladi?
- 8) Yuqoridagi algoritmda $s:=0$ buyrug'i olib tashlansa natija qanday bo'ladi?
- 9) Yuqoridagi algoritmda x ning oxirgi qiymatini toping.

Quyidagi masalalarning algoritmlarini tuzing

- 1) 1 dan n gacha bo'lgan juft sonlar yig'indisini topish.
- 2) 1 dan n gacha bo'lgan 6 karrali sonlar yig'indisini topish.
- 3) n faktorialni hisoblash.
- 4) $n!!$ ni hisoblash

$a = (-1)^n(2n-8)$, $n \in [-4;6]$, $\Delta n=2$ bo'lsa, nechta manfiy had hosil bo'lishini aniqlang.

1) Ixtiyoriy natural sonning bo'luvchilari sonini topish algoritmini tuzing.

2) Raqamlar yig'indisi 9 ga teng bo'lgan uch xonali sonlarni va ularning yig'indisini toping.

3) K ta raqamdan iborat butun son raqamlarning k darajalari yig'indisi son o'ziga teng bo'lsa, bu son Armstrong soni deyiladi. Masalan, $1^3+5^3+3^3=153$. 4 xonali Armstrong sonlarini topish algoritmini tuzing.

4) Chapdan va o'ngdan bir xil o'qiladigan sonlar Palindrom sonlari deyiladi. Masalan, 121, 434,...

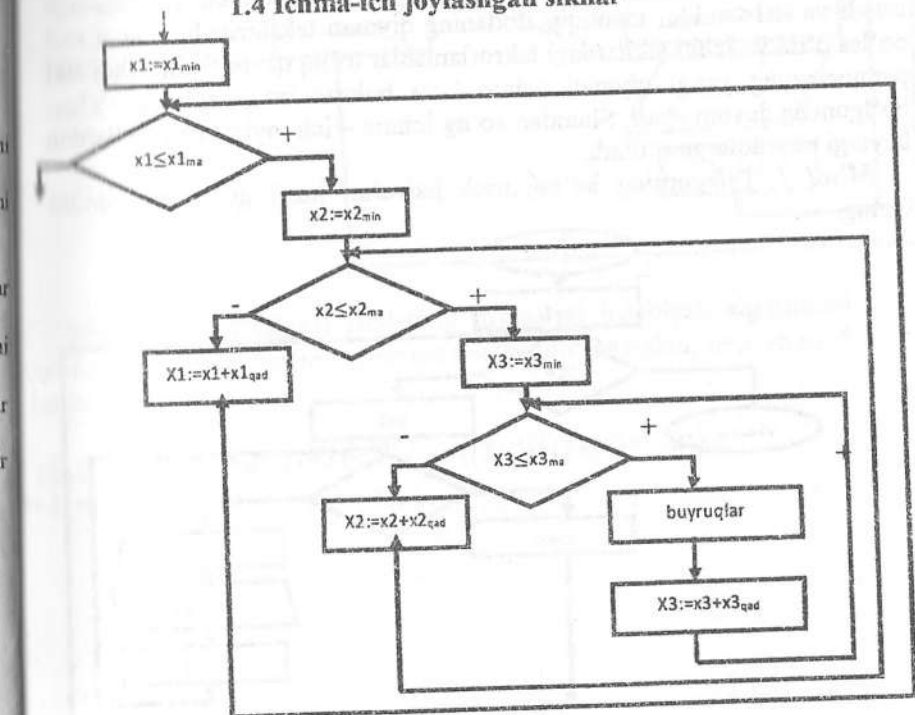
5) Uch xonali Palindrom sonlarni topish algoritmini tuzing.

6) Sonning raqamlar yig'indisini topish algoritmini tuzing.

7) Agar natural son o'zidan boshqa bo'luvchilar yig'indisiga teng bo'lsa, bunday son mukammal son deyiladi.

8) Masalan, $28=1+2+4+7+14$. Uch xonali mukammal sonlarni toping.

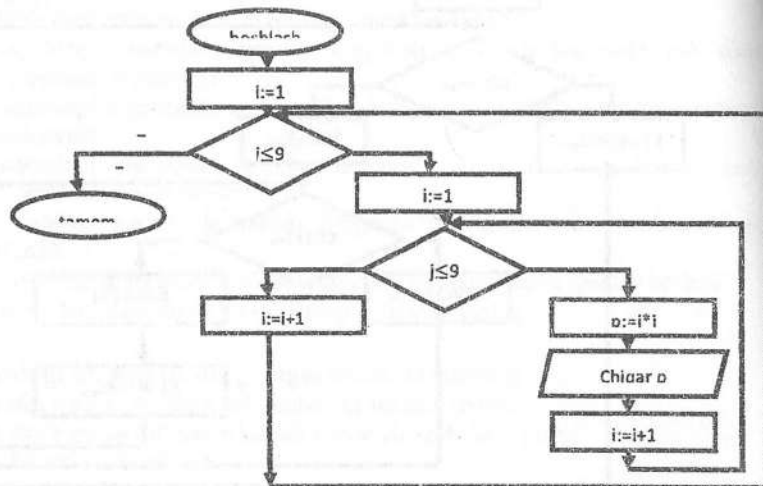
1.4 Ichma-ich joylashgan sikllar



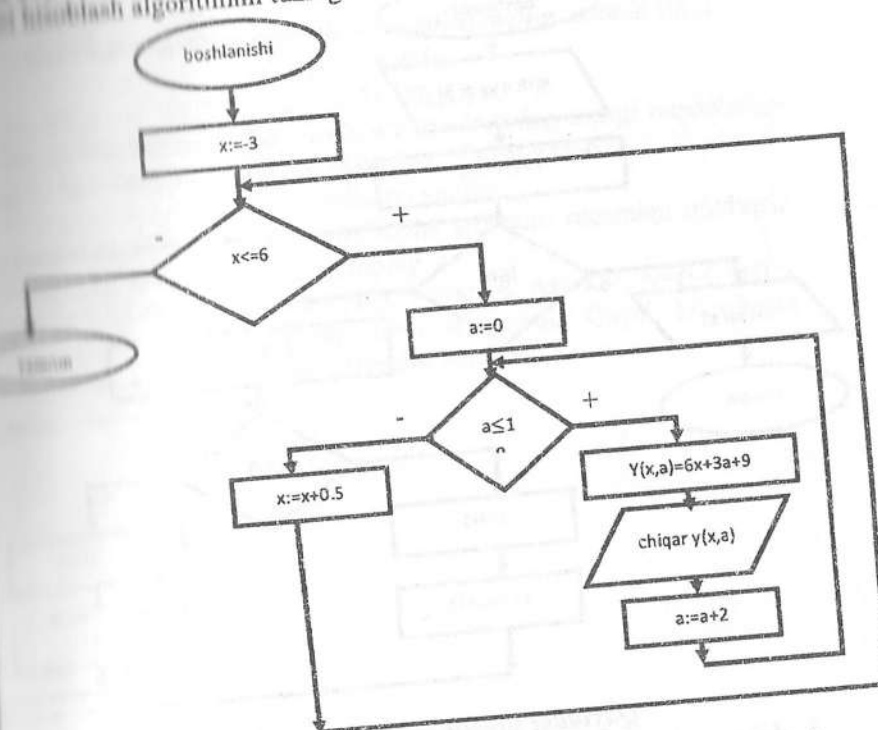
Buyruq quyidagi tartibda bajariladi:

Tashqi sikl parametri o'zining dastlabki qiymatni oladi va $x_1 \leq x_{1\max}$ mantiqiy ifodaning qiymatini bilan solishtiradi. Agar $x_1 \leq x_{1\max}$ mantiqiy ifodaning qiymati rost bo'lsa, o'rta sikl parametri o'zining dastlabki qiymatini oladi va $x_2 \leq x_{2\max}$ mantiqiy ifodaning qiymatini bilan solishtiradi. Agar $x_2 \leq x_{2\max}$ mantiqiy ifodaning qiymati rost bo'lsa, ichki sikl parametri o'zining dastlabki qiymatini oladi va $x_3 \leq x_{3\max}$ mantiqiy ifodaning qiymatini bilan solishtiradi. Agar $x_3 \leq x_{3\max}$ mantiqiy ifodaning qiymati rost bo'lsa, buyruqlar bir marta bajarilib, ichki sikl parametri bir qadamga ortadi va $x_3 \leq x_{3\max}$ mantiqiy ifodaning qiymati yana tekshiriladi. Agar rost bo'lsa buyruqlar yana bir bor bajarilib, ichki sikl parametri yana bir qadamga ortadi va hokazo ichki sikldagi takrorlanishlar $x_3 > x_{3\max}$ bo'lguncha davom etadi. Shundan so'ng o'rta sikl parametri bir qadamga ortadi va $x_2 \leq x_{2\max}$ mantiqiy ifodaning qiymati tekshiriladi. Agar u rost bo'lsa ichki sikldagi takrorlanishlar to'liq qaytariladi (o'rta sikl parametrining yangi qiymati uchun). Shundan so'ng o'rta sikl parametri yana bir qadamga ortadi va hokazo o'rta sikldagi takrorlanishlar $x_2 > x_{2\max}$ bo'lguncha davom etadi. Shundan so'ng tashqi sikl parametri bir qadamga ortadi va $x_1 \leq x_{1\max}$ mantiqiy ifodaning qiymati tekshiriladi, agar u rost bo'lsa o'rta va ichki sikllardagi takrorlanishlar to'liq qaytariladi (tashqi sikl parametrining yangi qiymati uchun) va hokazo bu jarayon $x_1 > x_{1\max}$ bo'lguncha davom etadi. Shundan so'ng ichma - ich joylashgan sikllardan keyingi buyruqlarga o'tiladi.

Misol 1. Pifagorning ko'paytirish jadvalini hosil qilish algoritmini tuzing.



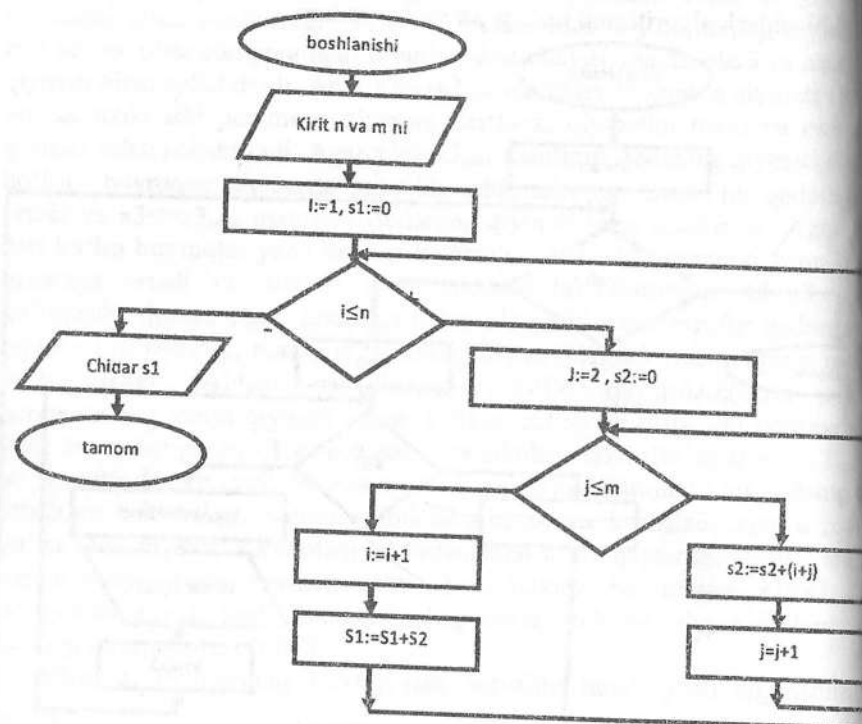
Misol 2. $y=6x+3a+9$ $x \in [-3;6]$, $\Delta x=0.5$, $a \in [0;10]$, $\Delta a=2$ funksiyaning hisoblash algoritmini tuzing. Δx va Δa -qadam



Misol 3. $\sum_{i=1}^n \sum_{j=2}^m (i+j)$ ifodaning qiymatini hisoblash algoritmini tuzing. Σ belgi yig'indi (summa) belgisidir. Masalan, $n=3$ va $m=4$ bo'lsin.

$$\sum_{i=1}^3 \sum_{j=2}^4 (i+j) = ((1+2)+(1+3)+(1+4)) + ((2+2)+(2+3)+(2+4)) + ((3+2)+(3+3)+(3+4))$$

Endi ushbu ifodaning qiymatini hisoblash algoritmini keltiramiz:



Quyidagi savollarga javob bering:

1. Ichma-ich joylashgan sikllarning umumiy ko'rinishini yozib, bajarilish tartibini tushuntiring.
2. $\prod_{i=1}^n \prod_{j=2}^m (i+j)$ ifodaning yoyilmasini yozing.
3. $\prod_{i=1}^n \sum_{j=2}^m (i+j)$ ifodaning yoyilmasini yozing.
4. Uchta ichma-ich joylashgan sikllarning bajarilish tartibi nimaga o'xshaydi?
5. To'rtta ichma-ich joylashgan sikllarning tashqi ikkitasida 12 martadan takrorlanish, ichki ikkitasida 5 martadan takrorlanish ro'y bersa, buyruqlar nacha marta bajariladi? (j: 3600)
6. To'rtta ichma-ich joylashgan sikllarning tashqi ikkitasida 12 martadan takrorlanish, ichki ikkitasida 5 martadan takrorlanish ro'y bersa va oxirida sikllar parametrlari, ya'ni $x+y+z+t$ ifodaning qiymati chiqarilsa, algoritm natijasini aniqlang. Barcha sikl

parametrlarining dastlabki qiymatlari va qadamlari 1 deb hisoblansin. (j: 38)

Quyidagi masalalarning algoritmlarini tuzing. Bunda $\prod_{j=2}^m j$ ko'paytma belgisi.

- 1) $\prod_{i=1}^n \prod_{j=2}^m (i+j)$ 2) $\prod_{i=1}^n \sum_{j=2}^m (i+j)$ 3) $\sum_{i=1}^n \prod_{j=2}^m (i+j)$
- 4) Bir va ikki xonali sonlar ichida o'z kvadratining oxirgi raqamlariga teng bo'lgan sonlarni aniqlang. Masalan, $5^2=25$; $25^2=625$.
- 5) Berilgan sonning tub bo'luvchilarini aniqlang.
- 6) Dastlabki ikkita raqamlari yig'indisi uchinchi raqamiga qoldiqsiz bo'linadigan uch xonali sonlarni toping.
- 7) Nuqta koordinatalari $x=(-1)^i(2i-j)$, $1 \leq x \leq 8$, $\Delta x=0.8$; $y=(i-(-1)^j)$, $-4 \leq y \leq 4$, $\Delta y=0.6$; qonuniyat bilan o'zgaradi. Qaysi koordinata choraklariga nechtdan nuqta tushishini aniqlang.

II-BOB. C++ DASTURLASH TILI

2.1. C++ dasturlash tili tarixi

Axborot texnologiyalarining jadal sur'atlar bilan rivojlanishi, navbatida dasturlash tillarining rivojlanishiga olib keldi. Dasturlash tillari yaratilgan davrdan boshlab dasturlash tillari ham to'xtovsiz rivojlanib kelmoqda. Birinchi avlod mashinalari bilan ishlash dasturchilar eng sodda mashina tilini o'zida ifodalovchi kompyuter buyruqlari bilan ishlaganlar. Bu buyruqlar uzun qatorli nol va bitta ketma-ketligidan iborat bo'lgan. Keyinchalik foydalanuvchi uchun nisbatan qulay va tushunarli bo'lgan assembler tili yaratiladi. Yillar o'tishi bilan yuqori darajali dasturlash tillari yaratila boshlandi. Yuqori darajali algoritmik tillarda dastur foydalanuvchi tushunadigan ma'lum bir standartlar yordamida yaratiladi. Yuqori darajali algoritmik tillarda tuzilgan dastur kompyuter tiliga maxsus interpretator va komplyatorlar deb ataluvchi dasturlar yordamida o'tkaziladi.

Elektron xisoblash mashinalari avlodlari rivojlanib borishi bilan bir vaqtda dasturlash tillari ham rivojlandi. Dasturlash tillarida tuzilgan dasturlar tarkibidagi buyruqlarning soddaligi (hammabopligi) uni bajarish uchun kompyuter sarflaydigan vaqt va kompyuter xotirasidan kam joy egallashi bilan bir-biridan farq qiladi. Shuning uchun dasturlashga ham talablar o'zgarib bordi va bu holat nafaqat dasturlashni balki dastur yozish texnologiyalarining ham o'zgarishiga olib keldi. Hozirga qadar bir necha o'nlab dasturlash tillari yaratilgan. Lekin keyingi yaratilgan dasturlash tillari o'zidan oldingilarining kamchiliklarini to'ldirib borganligi uchun oldingilaridan dasturchilar foydalanishmaydi. Bugungi kunda amalda foydalaniladigan, ayniqsa ta'lim jarayonida o'rganiladigan zamonaviy dasturlash tillari Paskal, C++, Java, PHP, Python tillardir. So'ngi yillarda Java, C++ va Python dasturlash tillari tobora ommalashib bormoqda. Mazkur tillardagi imkoniyatlar zamonaviy axborot texnologiyalarining hamma talablariga javob beradi va unda dastur tuzuvchi uchun juda katta qulayliklar mavjud.

Mavjud tillar katta 2 ta guruhga bo'linadi:

- 1) Tabiiy tillar. Bu odamlar o'zaro muloqotda bo'ladigan tillar (o'zbek, qozoq, ingliz va hokazo).
- 2) Sun'iy tillar. Sun'iy tillar yaxshigina kodlangan bo'ladi. Shuning uchun masala dasturini masala kodi ham deyishadi.

Kodlash 2 xil maqsadda amalga oshiriladi:

1) katta hajmdagi ma'lumotlarni qisqacha ifodalash maqsadida. Masalan, Samarqand davlat universitetining kodi - SamDU, test markazning kodi - DTM, oziq-ovqat mahsulotlari, yengil sanoat, og'ir sanoat, ta'lim dargohlarining nomlari ham ularning shaxsiy Odamlarning ismlari, hatto laqablari, masalan, xo'jayin, lampo shisha, novcha va hokazolarning barchasi kodlangan ma'lumotlardir.

2) maxfiylikni saqlash maqsadida. Bunday usulda kodlangan ma'lumotlarni hamma ham tushunavermaydi. Faqat tor doiradagi odamlarгина tushunadi. Maxfiy holda kodlashning ham har xil usullaridan foydalaniladi. Masalan, raqamlar orqali, hayvonlar nomlari orqali, qisqacha so'zlar orqali kodlanadi va hokazo. Bunday usulda kodlashdan josuslar, maxfiy xizmat vakillari ko'p foydalanishadi. Masalan, 16-obyekt yo'qoldi, chiroq yonmadi, mashina portladi, ikkinchini olib tashlang va hokazo bunday hodisalarning har biri qandaydir obyekt, voqea va hodisalarning kodlaridir.

Barcha dasturlash tillari ham sun'iy tillar jumlasiga kiradi va ularning ijrochisi kompyuterlardir. Ularda ham har bir buyruq qandaydir fikrning kodi hisoblanadi. Masalan, $a \% 2 == 0$ yozuvini C++ dasturlash tilida kompyuter "a juft son" deb, $\text{abs}(x)$ yozuvini esa $|x|$ deb tushunadi va hokazo. Shuning uchun dasturlash tillarini yaratish uchun qilingan dastlabki harakatlardayoq keng omma bu tillarni tezroq o'zlashtirib olishi uchun biror tabiiy tilning so'zlaridan mumkin qadar kam foydalanishga harakat qilingan.

Biz yuqori darajali dasturlash tillaridan biri bo'lgan C++ dasturlash tili bilan tanishamiz. C++ dasturlash tili 1980 yillar boshida Byarne Straustrup tomonidan C tiliga asoslangan tarzda yaratildi. C dasturlash tili 1972-yil Denis Pitch va Brayan Kornegilar tomonidan yaratilgan edi. Dasturlarni tez va sifatli yozish hozirgi kunda katta ahamiyatga ega. Grafik interfeysga ega dasturlarni yaratish uchun obyektga mo'ljallangan dasturlash tayanch baza hisoblanadi.

C++ dasturlash tili alifbesi

Har bir dasturlash tili alfavitiga ushbu tilda dastur tuzish jarayonida foydalaniladigan barcha belgilar kiradi. U holda C++ dasturlash tili alifbesiga quyidagilar kiradi:

1. Lotin alifbesi harflari;
2. Raqamlar: 0; 1; 2; 3; 4; 5; 6; 7; 8; 9;
3. Tinish belgilari: (.); (;); (.); (:);
4. Qavslar: (); {}; [];
5. Amal belgilari: +; -; *; /;
6. Maxsus belgilar: !; %; #; ?; &; ^;

Arifmetik amallar va ularning bajarilish tartibi

Matematikada bo'lgani kabi C++ dasturlash tilida ham 4 ta arifmetik amal(+; -; *;/)lardan foydalaniladi. Ularning bosqichlarga bo'linishi, bajarilish tartibi, qavslar ishtirokidagi bajarilish tartibi matematikadagi tartibidan farq qilmaydi. Masalan:

$$\begin{array}{cc} 1 & 2 & 3 & 4 & & 1 & 2 & 3 & 4 & 5 \\ 1) & 26-3+18-76+53; & & & & 2) & 38/2*x/12*45/y \end{array}$$

$$\begin{array}{ccccccc} 7 & 1 & 8 & 2 & 9 & 3 & 4 & 10 & 5 & 6 \\ 3) & 14-155/4+3*x+24/8*5-98*4/16; \end{array}$$

$$\begin{array}{ccccccccccc} 14 & 11 & 12 & 13 & 5 & 9 & 6 & 7 & 10 & 8 & 1 & 2 & 4 & 3 \\ 4) & 24+(96/24*6-(18/5+2*c/7+6*(216/3/4+3*2))) \end{array}$$

Eslatma: matematikada ba'zi hollarda, ayniqsa, harfiy va qavslar ifodalarda ko'paytirish amali oshkor yozilmaydi. C++ dasturlash tilida barcha amallar oshkor yozilishi shart. Masalan,

$$\begin{array}{l} 1) 2x+5y-2*x+5*y \\ 2) 3(7a-5)-3*(7*a-5) \end{array}$$

kabi yozuvlar xato hisoblanadi. Ularning to'g'ri yozuvi quyidagicha

$$\begin{array}{l} 1) 2*x+5*y-2*x+5*y \\ 2) 3*(7*a-5)-3*(7*a-5) \end{array}$$

Bundan tashqari C++ dasturlash tilida quyidagi amallardan ham foydalaniladi:

1. Darajaga ko'tarish amali $\text{pow}(x,n)$ funksiyasi yordamida bajariladi:

$$\text{pow}(x,n)=x^n; \text{pow}(3,4)=81;$$

2. $n\%m$ – amali n butun sonini m butun soniga bo'lganda qoldiqni hisoblaydi va qoldiqqa bo'linuvchi ishorasi beriladi.

$$\text{Masalan, } 37\%32=5; \quad -43\%8=-3; \quad 64\%(-12)=-4;$$

3. Bir butun sonni ikkinchi butun songa bo'lganda to'liqsiz bo'linmani hisoblash uchun, natijani qiymat sifatida oluvchi o'zgaruvchining turi **int** bo'lishi yetarli. Masalan:

$$\begin{array}{l} \text{int } a, b, c, d, e, f; \\ a=47; b=23; c=6; d=a/c; e=a/d; \\ \text{cout} << "d=" << d << " e=" << e; \\ \text{Dastur natijasi } d=7 \quad e=6 \text{ kabi bo'ladi.} \end{array}$$

4. $\text{pow10}(\text{int } p)$ funksiyasi 10^p ni hisoblaydi. Bunda p – butun son bo'lishi shart.
 $\text{pow10}(3)=1000; \quad \text{pow10}(1)=10$

Standart funksiyalar

Barcha dasturlash tillarida matematikada bizga tanish bo'lgan funksiyalardan foydalaniladi. Ularning ichida nomini uchratgach, uni dasturlash tili tanisa, ya'ni ko'rsatilgan argument bo'yicha funksiyaning qiymatini hisoblovchi dastur bo'lsa, bunday funksiyalar **standart funksiyalar** deyiladi. Boshqa dasturlash tillarida bo'lgani kabi C++ dasturlash tilida ham funksiya argumenti unung nomidan keyin qavs ichiga olib yoziladi. Quyida standart funksiyalar ro'yxati keltirilgan:

$$\begin{array}{l} \text{abs}(x) \rightarrow |x|, \text{ bunda } x - \text{butun son} \\ \text{fabs}(x) \rightarrow |x|, \text{ bunda } x - \text{haqiqiy son} \\ \sin(x) \rightarrow \sin x, \text{ bunda } x - \text{radianda beriladi} \\ \cos(x) \rightarrow \cos x, \text{ bunda } x - \text{radianda beriladi} \\ \tan(x) \rightarrow \tan x, \text{ bunda } x - \text{radianda beriladi} \\ \text{sqrt}(x) \rightarrow \sqrt{x}, \text{ bunda } x - \text{haqiqiy son} \\ \exp(x) \rightarrow e^x, \text{ bunda } x - \text{haqiqiy son} \\ \log(x) \rightarrow \ln x, \text{ bunda } x - \text{haqiqiy musbat son} \\ \log_{10}(x) \rightarrow \lg x, \text{ bunda } x - \text{haqiqiy musbat son} \\ \text{asin}(x) \rightarrow \arcsin x, \text{ bunda } x - \text{radianda beriladi} \\ \text{acos}(x) \rightarrow \arccos x, \text{ bunda } x - \text{radianda beriladi} \\ \text{atan}(x) \rightarrow \arctg x, \text{ bunda } x - \text{radianda beriladi} \\ \text{atan2}(x,y) \rightarrow \arctg \frac{x}{y}, \text{ bunda } x, y \text{ lar radianda beriladi.} \end{array}$$

$\text{ceil}(x) \rightarrow x$ haqiqiy sonini undan katta, lekin unga eng yaqin butun son bilan almashtiradi. Natijaga haqiqiy tur maqomi beriladi.

floor(x) → x haqiqiy sonini undan kichik, lekin unga eng yaqin butun son bilan almashtiradi. Natijaga haqiqiy tur maqomi beriladi.

Bu ikkita funksiya yaxlitlash amalini bajarmasligini unutmaganing kerak. Masalan,

ceil(4.235601) = 5; **floor**(4.235601) = 4

ceil(4.9907) = 5; **floor**(4.9907) = 4

fmod(x,y) – x butun sonni y butun songa bo'lgandagi qoldiqni hisoblaydi va natijaga haqiqiy tur beradi. % amali ham shu vazifani bajaradi, faqat bu amal natijasi butun turli bo'ladi.

hypot(x, y) – x va y lar haqiqiy sonlar bo'lib, funksiya x va y katetlar bo'yicha to'g'ri burchakli uchburchakning gipotenuzasini hisoblaydi.

pow(x,y) → x^y

pow 10(x) → x – butun son bo'lib, **pow 10(x)** → 10^x

Eslatma: Standart funksiyalar kutubxonasidan foydalanish uchun **int main()** funksiyasidan oldin **#include <math.h>** sarlavha faylning ham yozish kerak. Quyida keltirilgan dastur va uning natijalarini tahlil qiling:

```
#include <iostream.h>
#include <math.h>
using namespace std;
int main()
{ double n, m, c, s, g, d, k, x, y, z, x1, y1, z1;
s=log(9); cout << "s=" << s;
c=log10(1000); cout << "c=" << c << "\n";
n=pow(2.2,2.2); cout << "n=" << n;
m=fmod(45,7); cout << "m=" << m << "\n";
g=tan(3.14/4); cout << "tg(3.14/4)=" << g << "\n";
k=1/tan(3.14/4); cout << "ctg(3.14/4)=" << k << "\n";
x=ceil(14.235601); y=floor(7.235601); cout << "x=" << x << "\n";
y=floor(7.235601);
x1=ceil(-14.235601); y1=floor(-7.235601); cout << "x1=" << x1 << "\n";
y1=floor(-7.235601);
z=fmod(-23,7); cout << "z=" << z; z1=pow(3,3); cout << "z1=" << z1 << "\n";
```

qilish ("pause");

return 0;

Dastur natijasi quyidagicha bo'ladi:

s=2.19722 c=3

m=3

tg(3.14/4)=0.999204

ctg(3.14/4)=1.0008

x=14 y=7

x1=-14 y1=-8

z=3 z1=27

Bo'ysatda bo'lmagan standart funksiyalar matematik formulalardan foydalanib, borlari orqali ifodalanadi. Masalan,

$\cos(x) \rightarrow \cos(x)/\sin(x)$ yoki $1/\tan(x)$

$\log_2 x \rightarrow \log(x)/\log(2)$

C++ dasturlash tilida ifodalar va ularning yozilishi

Ifodalarni C++ dasturlash tilida yozishda quyidagi qoidalarga amal qilish lozim:

- ifodani kasr ko'rinishida yozib bo'lmaydi

$\frac{2x+23}{19-7a} \rightarrow (2*x+23)/(19-7*a);$

- ifadada grek alifbosi harflari bo'lsa ularni o'ziga o'xshash lotin alifbosi harflari bilan almashtiriladi yoki ularning nomlari to'liq yoki qisqacha yoziladi

$(2\alpha+3\alpha)-8\beta\sin\gamma+4\delta-\mu \rightarrow (2*a+3*alfa) - 8*beta*\sin(gamma)+4*b-m;$

- ikkita amal ketma-ket yozilmaydi

$\frac{58}{-9} \rightarrow 58/-9$ bu noto'g'ri, to'g'ri varianti $58/(-9)$ kabi bo'ladi;

- agar satr biror amal belgisi bilan tugasa, yangi satr matematikadagidek o'sha amal bilan boshlanmaydi. Chunki ekranda biz ko'rib turgan satrlar fizik satrlar bo'lib, mantiqiy satr bir nechta fizik satrlardan tashkil topadi. Natijada yana ikkita amal ketma-ket yozilgan bo'lib qoladi;

- ochilgan qavslar soni yopilgan qavslar soniga teng bo'lishi kerak;

- dastur tarkibida nol raqami (0) va o harfini farqlashga odatlan lozim.

Quyidagi ifodani C++ dasturlash tilida yozing.

$$\frac{x^{n+1} + e^{k-1}}{1 + x/y - \lg z} \left(\operatorname{Ctg} \frac{x}{y} - \sqrt{z} \right) \frac{|x-1|}{|y+1|}$$

$$(\operatorname{pow}(x, y+1) + \exp(k-1)) / (1 + x/y - \tan(z)) * (1/\tan(x/y) - \operatorname{sqr}(t) * \operatorname{fabs}(\operatorname{fabs}(x-1)/(y+1)))$$

Eslatma: Qavs ichidagi amallar birinchi bajariladi, keyin funksiyalarning qiymatlari hisoblanadi. Undan keyin bizga hozirgacha ma'lum bolgan amallarning birinchi bajarilish nuqtayi nazarida bajarilish tartibi quyidagicha (Qavs ichiga olingan amallar bir bosqich amallari hisoblanadi, ya'ni ulardan qaysi biri ifodada chapda yozilgan bo'lsa, o'shasi birinchi bajariladi):

(*, /, %), (+, -)

1. Quyidagi ifodalarni C++ dasturlash tilida yozing:

$$1) \sqrt[3]{\frac{x}{x+1}} + \operatorname{arctg}(y+3x)2 - x - \ln |\sin 2x|;$$

$$2) \frac{\cos^3(x) + \sqrt{a^{2x}} - e^{x+i}}{x + \frac{y}{z + \frac{y}{x}} + (3a^2 + 5b - 1)^4}$$

$$3) \frac{19,32 \cdot 3\pi - \sqrt[3]{x} \cdot \ln y}{b \cdot \sqrt{c - (e^{-x} - e^y)}} \operatorname{Ctg} \frac{\pi}{13}$$

$$4) \sqrt[3]{x \lg 3 - \sqrt{y} \cdot e^{3x}} - \sqrt[5]{\frac{|x|}{y^3}}$$

2. Quyidagi ifoda C++ dasturlash tilida yozilgan. Oddiy matematik ko'rinishini tiklang:

$$1) \operatorname{sqr}(t)(\operatorname{pow}(x, 2) + \operatorname{pow}(y, 2) * \log(z) + \log 10(y+z)) * (\operatorname{sqr}(t)(\operatorname{pow}(x, 2) - \operatorname{pow}(y, 2)) + 1);$$

$$2) \exp(\operatorname{abs}(x-y) * \operatorname{abs}(\operatorname{pow}(x-y, x+y) / (a * \sin(x)) + a * \cos(x))) + \operatorname{pow}((\operatorname{pow}(x, 6) + \operatorname{pow}(\log(y), 2)), 1/3);$$

$$1) (\operatorname{atan}(1)/4 * \operatorname{pow}(\cos(y), 3) - \operatorname{pow}(a, 3 * \sin(1 * x))) / \operatorname{atan}(2 + (x/y));$$

$$4) \exp(\operatorname{pow}(\sin(x), 3) + \operatorname{pow}(x - 1, 1/3) - \log(\operatorname{sqr}(t)(x + \operatorname{pow}(y, 2))));$$

3. Quyidagi ifodalar C++ dasturlash tilida yozilgan.

Natural sonlarni aniqlang

$$1) j / ((1 - \operatorname{sqr}(t)(z)/(t)) - 1 + \operatorname{pow}(x+3, 5) - \operatorname{sqr}(t)(2 * x+3 * a));$$

$$1) \operatorname{atan}(1) * \sin x / \cos x + \ln(x) * \operatorname{pow}(\operatorname{sqr}(t)(x), 1/3) - 85,4;$$

$$2) (28,5 * 64,0 - 0,19 + 5 * e^4) / (\log 10(1001 + x) -$$

$$\operatorname{sqr}(t)(x));$$

$$3) x + \exp(\operatorname{ciel}(x) + \operatorname{floor}(4 \sin(x)) + m \% (n) - \operatorname{pow}(\sin x, \operatorname{ceil}(x))).$$

Quyidagi savollarga javob bering

1. Ma'lumotlarni kodlash deyilganda nimani tushunasiz?
2. Ma'lumotlar qanday maqsadlarda kodlanadi? Misollar keltiring.
3. Dasturlashning kodlashga qanday aloqasi bor?
4. C++ dasturlash tili alifbesiga qaysi belgilar kiradi?
5. Sonlarning oddiy matematik yozuvi bilan C++ dasturlash tilida yozilgan yozuvda qanday farqlari bor?
6. C va C++ dasturlash tillarini kimlar va qachon yaratishgan?
7. $n \cdot 10^m$ soni C++ da qanday yoziladi?
8. C++ dasturlash tilida arifmetik amallarning bajarilish tartibini tushuntiring (misollar yordamida).
9. C++ dasturlash tilida arifmetik amallardan tashqari yana qanday amallardan foydalaniladi?
10. Darajaga ko'tarish amali qanday bajariladi?
11. C++ da natural sonlarni bo'lishda to'liqsiz bo'linma va qoldiq qanday hisoblanadi?

12. C++ dasturlash tilida qanday funksiyalar standart funksiyalar deyiladi?

13. Standart funksiyalarni C++ dasturlash tilida yozib chiqing.

14. a) `abs(x)` va `fabs(x)`; b) `log(x)` va `log10(x)`; c) `pow(x,y)` va `pow10(x)` funksiyalarning hamda `fmod(x,y)` funksiyasi bilan (%) amalining farqini tushuntiring (misollar keltirish bilan).

15. C++ dasturlash tilida ifodalarni yozish uchun mustaqil 3 turli masalalar tanlab, bajaring.

16. Standart funksiyalar kutubxonasidan foydalanish uchun qaysi sarlavha faylidan foydalaniladi?

17. C++ dasturlash tilida arifmetik ifodalarni yozishda grek alifbesi harflari, oddiy kasrlar qanday yoziladi?

18. Matematik ifodalarni C++ da yozishda qanday cheklovlar qo'yiladi? Nol raqami va O harfini qanday farqlaysiz?

Quyidagi ifodalarning qiymatlarini hisoblang

- $\log((37\%8*0.2)*\text{pow}(4,3))-\text{fmod}(93,20)$
- $(\text{ceil}(3.75)+\text{floor}(-3.75))/\text{abs}(9*(-3))+\sin(\text{atan}(1)*4)$
- $\cos(\text{atan}(1)*2)-\text{pow}(5,2)+\log(32)/\log(2)*12$
- $36/(3+1/(2-\text{ceil}(0.3)))*91\%83$
- $72\%3*7/3*5+\text{floor}(56)/\text{ceil}(6)$
- $\cos(\text{atan}(\text{floor}(1.8965))*4/3+\text{atan}(\text{ceil}(0.723)))*2/3+41)\%14\%7*4$

2.2 C++ dasturlash tilida yozilgan dasturning umumiy tuzilishi

C++ dasturlash tilida dasturning umumiy tuzilishini quyidagi oddiy dastur yordamida tushuntiramiz:

```
#include <iostream.h>
using namespace std;
```

```
int main()
{
    int a=12, s=10;
    while (a>0)
    {
        cout<<"a="<<a<<" b="<<b<<" s="<<s<<endl;
        a=a/10;
        s=s+a;
    }
    return 0;
}
```

C++ dasturlash tilida har qanday dastur 1-satr bilan boshlanadi. `main` funksiyasi (o'z ichiga olmoq) so'zi ma'lumotlarni kiritish va chiqarish (o'qish va yozish) funksiyalari va ularning o'z-atuvchilari bilan boshlanadi. `main` funksiyasi `<iostream.h>` sarlavha faylini dasturga qo'shadi. `main` funksiyasi yozuvi quyidagicha ma'noni bildiradi: `i` (input – kiritish), `o` (output – chiqarish (yozish)), `stream` – oqim (ya'ni oqimlar oqimi), `h` (head – sarlavha).

Bu ko'pchilik yangi kompyutorlar `#include <iostream.h>` o'rniga `#include <iostream>` sarlavha faylidan foydalanish imkonini beradi. Faqat undan keyin `using namespace std;` satrini qo'shish kerak.

1-satrdan `main( )` funksiyasi berilgan bo'lib, bu funksiya ham `main` – bosh, asosiy demakdir, albatta, bo'lishi shart va dastur shu funksiyani bajarish bilan o'z ishini boshlaydi. Uning oldidagi `int` so'zi dastur ijro etilgach, natija butun son sifatida qaytarilishini bildiradi.

C++ dasturlash tilining ba'zi kompyutorlari `int main( )` funksiya prototipi e'lon qilingan bo'lsa ham bu funksiya prototipi e'lon qilinmaganligi haqida xabar beradi. Bunday holda `int main( )` dan oldin `int main( )`; funksiya e'lonini ham qo'shish kerak:

```
#include <iostream.h>;
int main( );
int main
{ ... }
```

Dasturdagi boshqa funksiyalarga dasturning qayeridadir murojaat buyrug'i bo'lgandan so'ng ular ishga tushadi. Lekin `main( )` asosiy funksiyasiga esa dastur ishga tushgan zahoti murojaat bo'ladi.

4 va 9 - satrlarda `{ va }` qavslari keltirilgan bo'lib, ular `main` funksiyasining blok belgilaridir. C++ dasturlash tilida, tarmoqlanish,

takrorlanish kabi murakkab buyruqlarga talluqli buyruqlar ikki undan ortiq bo'lganda ham bu buyruqlar shunday blok belgilari olib yoziladi. Bu yerda esa bu belgilar dastur tanasining boshlanish tugashini bildiradi desak ham bo'ladi yoki dastur tanasining chegarasi yoki `main()` funksiyasi blok (tanasi) deyish ham mumkin. Ammo belgilersiz dastur ishlamaydi.

4-satrdagi `int a, b, c, s;` yozuvi `a, b, c, s` o'zgaruvchilar e'lon deyiladi. Bunda, albatta, ularning turi ko'rsatilgan bo'lishi shart va albatta, bir-birlaridan `(,)` belgisi bilan ajratib yoziladi va oxirigisi so'ng nuqtali vergul qo'yiladi.

5-satrdagi qiymat berish buyrug'i `(=)` yordamida `a, b, c` o'zgaruvchilarga dastlabki qiymatlar berilgan. Demak, C++ dasturlash tilida qiymat berish buyrug'i `(=)` belgisidan foydalanib yoziladi, tenglik belgisi esa matematikadagi ikkita tenglik belgisi orqali `(==)` ko'rsatiladi.

`a, b, s` larga ularni e'lon qilishda ham `int a=5; b=12; s=a+b;` ko'rsatib dastlabki qiymatlarini berish, ya'ni initsializatsiyalash mumkin edi.

Ko'rinib turibdiki, C++ dasturlash tilidagi buyruqlar bir-birlaridan `(;)` belgisi bilan ajratib yoziladi. Faqat sarlavha faylini e'lon qilgandan keyin va bosh funksiya `main()` dan keyin `(;)` qo'yilmaydi.

6-satrga `<iostream.h>` sarlavha faylida aniqlangan `cout` operatoridan foydalanilgan bo'lib, undan o'zgaruvchilarning kompyuter xotirasidagi qiymatlarini ekranga chiqarish maqsadida foydalaniladi. `cout` operatoridan keyin qiymati ekranga chiqarilayotgan barcha o'zgaruvchilar hech qanday tinish belgilarisiz `<<<` belgilari bilan ajratib yoziladi. Agar matn(satr) ekranga chiqarilishi kerak bo'lsa, u qo'shtirnoq ichiga olib yoziladi va C++ tomonidan qayta ishlanmasdan, qanday yozilgan bo'lsa, shundayligicha chop etiladi. Bizning misolimizda `"a="`, `"b="`, `"s="` yozuvlari matnlardir.

8-satrdagi `return 0;` (`return`-qaytish) yozuvi `main()` funksiyasiga qiymat qaytarish buyrug'i bo'lib, dastur o'z ishini tugatganligini bildiruvchi operatoridir. Ushbu dastur o'z ishini muvaffaqiyatli yakunlasa, funksiyaga nol qiymati qaytariladi.

Dev C++ variantida esa `return 0;` buyrug'idan oldin `System` ("pause"); buyrug'i qo'shilmasa, dastur natijasini ko'rib ulgurmasdan

to'xtatadi. Ushbu kutish buyrug'i `Esc` tugmasi bosilguncha natijani ko'rib olish mumkin.

C++ dasturlash tilida tuzilgan dastur kerak bo'landa yangi fayl yaratib olinadi va bu faylning kengaytmasi `.cpp` bo'ladi.

C++ dasturlash tilida izohlardan foydalanish

C++ dasturlash tilida tuzilgan yetarlicha katta hajmdagi dasturni o'qish uchun bir tomonidan, ayniqsa boshlovchilar tomonidan tushunib olinishi kerak bo'ladi. Agar dastur yetarlicha murakkab tuzilishga ega bo'lsa, ma'lum vaqtdan keyin shu dasturning muallifi ham tushunishga qiyinchiliklarni his qilishi mumkin. Bunday holatlar uchun C++ dasturlash tilida dasturning ayrim satrlarini tushuntirish matnlari bilan ta'minlash zaruriyati mavjud. Tushuntirish matnlari esa izohlar tarzida yoki chop qilish operatorida qo'shtirnoq `(//)` ichiga olib yoziladi.

C++ dasturlash tilida izohlarning ikki xil ko'rinishidan foydalaniladi:

1. Ikki bo'lish belgisidan foydalanilsa `(//)`, bitta fizik satr bo'lib, barcha izohga aylanadi;

2. Dasturning ixtiyoriy sondagi satrlari `/*` va `*/` belgilari ichiga olinib, o'sha satrlari izohga aylanadi.

```
#include <iostream.h>
using namespace std;
int main()
{
    int a, b, c; // a, b, c lar butun turli o'zgaruvchilar
    cout<<"a va b larning qiymatlarini kiriting";
    cin>>a>>b; // a va b larning qiymatlari kiritilishi shart
    c=a+b; // c o'zgaruvchiga a va b o'zgaruvchilar qiymatlari
    yig'indisining qiymati beriladi
    cout << "c=" << c; // c ning qiymati chop etiladi
    // a ning va b ning qiymatlarini kiritishni unutmadingizmi?
    /* cout << "a ning va b ning qiymatlari chop etilmaydi";
    cout << "\n Chunki bunga hojat yo'q"; */
    return 0;
```

}
Bunda to'rt marta satrli izohlardan, bir marta umumiy izoh foydalanildi, ikki marta chop etish operatoridan foydalanish tushuntirish matnlari berildi.

Izohlar C++ dasturlash tili tomonidan ijro etilmaydi, u xohlagan tushuntirish matnlarini yozish mumkin. Ular faqat dastur tarkibida dasturchiga ko'rinib turadi. Izohlarning ekrandagi ko'rinishi ham dasturlarning asosiy qismidan rangsizligi bilan ajralib turadi. Ammo chop etish operatoridan foydalanib tushuntirish matnlari berilgan chop etish operatori(cout) ijro etiladi. Faqat izohlar ijro etilmaydi.

Izohlardan nafaqat tushuntirish matnlari sifatida, balki dastur ma'lum sondagi buyruq yoki satrlarini ijro etmaslik maqsadida ham ko'p foydalaniladi.

Chiziqli dasturlar

Dastur tarkibidagi buyruqlar qanday berilgan bo'lsa, shunda tartibda, ya'ni yuqoridan pastga qarab tartib bilan bajarilsa, bunda holda buyruqlar bajarilishi tabiiy tartibga bo'ysinadi deyiladi. Bunda buyruqlar oddiy buyruqlar yoki oddiy buyruqlar ketma-ketligi bo'lib deyiladi.

Agar dastur tarkibidagi buyruqlar tabiiy tartibda bajarilsa, bunda dasturlar *chiziqli dasturlar* deyiladi.

Misol: $x = 5$; $y = 4,5$; $z = 0,5$ bo'lsa, $\sqrt{x^2 + y^2} \ln z + \frac{\lg(y+z)}{\sqrt{x^2 - y^2} + 1}$ ifodaning qiymatini hisoblash dasturini tuzing:

```
#include <iostream.h>
#include <math.h>
using namespace std;
int main()
{ float x=5, y=4.5, z=0.5, p, q, m;
  p=sqrt(pow(x,2)+pow(y,2));
  q=sqrt(pow(x,2)-pow(y,2));
  m=p*log(z)+log10(y+z)/q;
  cout << "m=" << m << endl;
```

return 0;

Ushbu dastur chiziqli dastur hisoblanadi. Chunki uning buyruqlari berilgan tartibda bajariladi va dastur natijasi $m = -4.34196$ kabi bo'ladi.

Elatma: Butun sonlar ustida bo'lish amalini bajarayotganda amal natijasi qiymat sifatida berilayotgan o'zgaruvchining turi *haqiqiy* bo'lsa ham o'zgaruvchi to'liqsiz bo'linmani oladi. Bo'linmaning qiymatini haqiqiy son shaklida to'liq olish uchun bo'linuvchi yoki bo'luvchini haqiqiy son sifatida yozish kerak. Masalan, $8/3=2; 8/3=2.66667$ kabi bajariladi, garchi *float* $a=8/3$ yoki *double* $a=8/3$ ko'rinishda aniqlangan bo'lsa ham, natija 2 bo'ladi. Lekin funktsiyalarning qiymatlarini bo'lishda to'g'ri bajariladi. Chunki funktsiyalarning o'zlarini emas, ularning qiymatlarini bo'ladi. Bu fikrlarga quyidagi dasturni tahlil qilish yo'li bilan ishonch hosil qilish mumkin.

```
#include <iostream.h>
#include <math.h>
using namespace std;
int main()
{ float x, y, z, p, q, m;
  x=38/3; y=38./3; z=38./3; cout << "x=" << x << "y=" << y << "z=" << z << "\n";
  p=cos(atan(1)*4)/3;
  q=8.24/3.12; cout << "p=" << p << "q=" << q << "\n";
  system("pause");
  return 0;
}
```

Dastur natijasi quyidagicha bo'ladi

$x=12$ $y=12.6667$ $z=126.667$
 $p=-0.333333$ $q=2.64103$

Endi son va uning raqamlarini qayta ishlash bilan bog'liq ikkita masalalarning dasturlarini tuzamiz:

1-masala. Ixtiyoriy to'rt xonali sonning raqamlari yig'indisini topish dasturini tuzing.


```
#include <iostream.h>
using namespace std;
int main( )
{
    int n,a1,a10,a100,a1000,s;
    cout <<"to'rt xonali sonni kiriting \n";
    cin>>n;
    a1 = n%10; a10 = n/10% 10;
    a100 = n/100% 10; a1000=n/1000;
    s = a1 + a10 + a100 + a1000;
    cout <<s<<"\n sonning raqamlari yig'indisi = "<<s<<endl;
    system("pause");
    return 0;
}
```

2-masala. Besh xonali sonning birinchi va oxirgi raqamlar almashtirilganda uning qiymati qanday o'zgarishini aniqlash dasturini tuzing:

```
#include <iostream.h>
using namespace std;
int main( )
{ int n, a1, a5, a234, at, f;
  //cout <<"besh xonali sonni kiriting\n";
  cin >>n;
  a1=n%10;
  a5=n/10000;
  a234=n/10-a5*1000;
  at=a1*10000+a5+a234*10;
  if (n>at) {f=n-at; cout <<"n berilgan "<<n<<" coning "<< f <<" ga kamaydi\n";}
  else {f=at-n; cout <<"n berilgan "<<n<<" soni "<<f<<" ga ortdi\n";}
  system("pause");
  return 0;
}
```

2.3 C++ dasturlash tili operatorlari

Operator so'zining ma'nosi xizmatchi, yordamchi demakdir. Dasturlash tillarining operatorlari – bu dasturni tashkil qilishda yoki boshqa o'z fikrini kompyuterga tushunarli tarzda ifodalash uchun zarur bo'ladigan yordamchi (xizmatchi) so'zlardir. Bu so'zlar ba'zan dasturlash tilining kalit so'zlari ham deyiladi.

C++ dasturlash tilining operatorlari soni anchagina. Biz ular bilan mavzudan mavzuga o'tgan sayin ketma-ket tanishib boramiz. Ular dasturda bir-birlaridan (;) belgisi bilan ajratilgan holda yoziladi.

O'zgaruvchilarning qiymatlarini kiritish operatori

Ba'zi masalalarda ba'zi parametrlarning qiymatlari oldindan ma'lum bo'lmaydi. Ularning qiymatlarini dasturni bajarish jarayonida kiritib ketishga to'g'ri keladi. Bu hol tuzilgan algoritmnin amaliyotligini ta'minlaydi. Masalan, kvadrat tenglamani yechish dasturini tuzishda **a**, **b**, **c** koeffitsientlarning qiymatlarini har ijro qilganda o'zgartirib yurmasdan, dasturni ijro etish jarayonida kiritib ketish qulay.

Dasturni bajarish jarayonida parametrlarning qiymatlari **cin** operatori yordamida kiritiladi. Qiymati kiritiladigan o'zgaruvchilar **cin** operatoridan keyin bir-birlaridan ">>" belgilari bilan ajratilgan holda yoziladi. Masalan: **cin >> a >>b>>c;** buyrug'i dastur ijrosi davomida **a**, **b** va **c** o'zgaruvchi parametrlarning qiymatlari kiritilishi kerakligini bildiradi va kompyuter ushbu buyruqni uchratgach dastur bajarilishini to'xtatib, o'zgaruvchilar qiymatlari kiritilishini kutadi. Shundan so'ng o'zgaruvchilarning qiymatlari bir-birlaridan probel bilan ajratilgan holda 5 -4 12 kabi klaviaturadan terilib, ENTER tugmachasi bosilsa, nooshkor holda **a=5;** **b=-4;** **c=12** kabi qiymat berish buyruqlari bajariladi. Bunda quyidagilarga alohida e'tibor berish kerak:

- o'zgaruvchilar kiritayotgan qiymatlarni kiritish tartibidagi nomeri bo'yicha qabul qiladi, ya'ni **cin** operatoridagi **n**- o'zgaruvchi kiritilgan **n**- qiymatni oladi;
- kiritilayotgan qiymatlar o'zgaruvchi uchun e'lon qilingan turga mos kelishi shart, aks holda xatolik sodir bo'ladi;

- cin operatori dasturning istalgan joyida bo'lishi mumkin. Ammo bu operator tarkibidagi o'zgaruvchilardan foydalanilgan buyruqlar oldin bo'lishi kerak, aks holda dastur ijro etilayotganda buyruqlar o'zgaruvchilarning qiymatlari ma'lum bo'lmay qoladi.

Ma'lumotlarni chiqarish operatori

C++ dasturlash tili tuzilgan dasturni ijro etgach, dastur natijalar ekranga chiqarish buyrug'i bo'lmasa, ekranga hech qanday natija chiqarilmaydi. Dastur natijalari cout operatori yordamida ekranga chiqariladi.

Qiymati chiqarilayotgan o'zgaruvchilar cout operatoridan keyin birlaridan "<<" belglar bilan ajratilgan holda cout <<a<<b<<c; kiritilgan sanab ko'rsatiladi. Bunda quyidagi holatlarga e'tibor berish lozim:

- yuqorida qayd etilganidek cout operatorida ("") ichiga o'zgaruvchi yozilgan matn (satr) qayta ishlanmasdan qanday yozilgan bo'lsa, shundayligicha chop etiladi. Bu holat natijalarini chiqarish tushuntirish matnlaridan foydalanish imkoniyatini beradi;

- 't' tabulyatsiya belgilari bo'lib, o'zgaruvchilarning qiymatlarini o'ralig'ida 7 ta bo'sh joy qoldirib chop etish maqsadida foydalaniladi;

- '\n' belgilaridan foydalanilganda undan keyin chop etilayotgan ma'lumotlar alohida satrdan chop etiladi. Lekin qo'shtirnoq ichida belgilardan foydalanishda apostrof (tutuq belgisi) qo'yilmaydi;

- cout operatorida qiymatlari aniqlangan o'zgaruvchilarni ifodalovchi yozilgan bo'lsa, ularning qiymatlari hisoblanib, natijasi chop etiladi. Shuningdek, son ham o'zgarishsiz chop etiladi;

- probel (bo'sh joy) yoki tabulyatsiyalardan foydalanilmasa, bitta son nechta sonlar ketma-ket yozilib bitta son kabi chop etiladi. Yuqoridagi holatlarni quyidagi dastur misolida namoyish qilamiz.

Misol: $y = 4ax + bc$ funksiya qiymatini hisoblash dasturini tuzing:

```
#include <iostream.h>
using namespace std;
int main()
{
    int a, b, c, x, y; cin >> a >> b >> c >> x;
    y = 4*a*x + b*c; cout << c;
```

```
cout << "\n javoblarni tahlil qiling";
cout << " " << a << "\n " << b << " " << "\n" << y << "\n";
cout << a << b << c;
cout << "\n" << (2*a - b)*4 << " " << 200 << "\n";
system("pause");
return 0;
```

Agar cin >> a >> b >> c >> x; buyrug'i bo'yicha 2 3 4 5 qiymatlari kiritilsa, ekranda quyidagicha natijalar chop etiladi.

```

4
Javoblarni tahlil qiling
2
3
52
234
4 200
```

Qiymat berish buyrug'i (ta'minlash operatori)

Qiymat berish buyrug'i deyilganda qandaydir o'zgaruvchiga biror qiymatni berish haqidagi buyruq tushuniladi. C++ dasturlash tilida qiymat berish buyrug'i matematikadagi tenglik belgisi (=) yordamida yoziladi. Masalan: $x=6$; yozuvi matematikada "*x oltiga teng*" deb o'qilsa, C++ dasturlash tilida "*x ga olti qiymat berilsin*" deb o'qiladi.

Ma'lumki, dasturlash tillarida har bir e'lon qilingan o'zgaruvchining qiymatini yozib qo'yish uchun xotiradan joy ajratiladi. Qiymat berish buyrug'ining mohiyatini to'g'ri tushunish uchun kompyuterning xotirasini daftarning bir varag'i deb va varaqning har bir katakchasini xotira yacheykasi deb tushuning. Natijada o'zgaruvchiga berilgan har bir qiymatni go'yo o'sha katakcha o'lchamidagi qog'ozga yozib, katakchaga yopishtirib qo'yiladi deb tushunasiz. Demak, bitta o'zgaruvchi bir nechta qiymatlar olsa, oxirigisi ko'rinib, oldingilari ko'rinmasdan qoladi.

Misol. $x = 5$; $x = x + 3$; $y = 10$; $x = x - y$; $x = x + y$; $y = x + y$. Buyruqlar bajarilgach, x va y lar qanday qiymatlar oladi?

Javob: $x = 8$; $y = 18$.

Eslatma: Qiymat berish buyrug'i ba'zi adabiyotlarda "o'zlashtirish operatori" yoki "o'zlashtirish amali" yoki "ta'minlash operatori" ham yuritiladi.

Inkrement va dekrement amallari

C++ dasturlash tilida o'zgaruvchi qiymatini 1 ga oshirish va kamaytirish amallari ham mavjud.

$x++$; ko'rinishidagi buyruq bo'yicha x o'zgaruvchining qiymati 1 ga ortadi. Bunda " $++$ " amali **inkrement** amali deyiladi.

$x--$; ko'rinishidagi buyruq bo'yicha x o'zgaruvchining qiymati 1 ga kamayadi. " $--$ " amali **dekrement** amali deyiladi.

Misol:

```
x=10; x++; cout << "x =" << x << '\n';
```

```
y=16; y--; cout << y;
```

Buyruqlari natijasi

```
x= 11
```

15 kabi bo'ladi.

Shuni unutmaslik kerakki, **inkrement** va **dekrement** amallari faqat butun turli o'zgaruvchilar ustida bajariladi.

Qiymat berish buyrug'ining qo'shimcha ko'rinishlari

Oldingi mavzudagi dasturda $x++$ buyrug'i o'rinda $x++$ buyrug'idan, $y--$ buyrug'i o'rinda $--y$ buyrug'idan foydalanish mumkin. Chunki bu buyruqlar o'zgaruvchining qiymatini 1 ga oshirish yoki kamaytirish ma'nosida teng kuchli.

Ammo qiymat berish buyrug'ida **inkrement** va **dekrement** amallaridan foydalanishga to'g'ri kelsa, " $++$ ", " $--$ " belgilarning o'zgaruvchi oldidan yoki orqasidan qo'yishning farqi bor. Quyida qiymat berish buyrug'ining qo'shimcha ko'rinishlari bilan tanishamiz:

- $y=x++$; buyrug'i bo'yicha x ning qiymati y ga berilib, so'ngra x ning qiymati 1 ga ortadi. Bu buyruq $y=x$; $x=x+1$; buyruqlariga teng kuchli;

- $y=++x$; buyrug'i bo'yicha oldin x ning qiymati 1 ga orttirilib, hosil bo'lgan yangi qiymati y ga beriladi. Bu buyruq $x=x+1$; $y=x$; buyruqlariga teng kuchli;

- $y=a++$; buyrug'i bo'yicha oldin y ga a ning qiymati berilib, a ning qiymati 1 ga kamaytiriladi. Bu buyruq $y=a$; $a=a-1$; buyruqlariga teng kuchli;

- $y=a--$; buyrug'i bo'yicha oldin a ning qiymati 1 ga kamaytirilib, kamaytirilgan qiymati y ga beriladi. Bu buyruq $a=a-1$; $y=a$; buyruqlariga teng kuchli;

- $y=a++$; buyrug'i bo'yicha y ga a ni qo'shib, natija y ga beriladi. Bu buyruq $y=y+a$; buyrug'iga teng kuchli;

- $y=a--$; buyrug'i bo'yicha y dan a ni ayirib, natija y ga beriladi. Bu buyruq $y=y-a$; buyrug'iga teng kuchli;

- $y=a*$; buyrug'i bo'yicha y ni a ga ko'paytirib, natija y ga beriladi. Ya'ni bu buyruq $y=y*a$; buyrug'iga teng kuchli;

- $y=a/$; buyrug'i bo'yicha y ni a ga bo'lib, natija y ga beriladi. Ya'ni bu buyruq $y=y/a$; buyrug'iga teng kuchli;

- $y=a\%$; buyrug'i bo'yicha y ni a ga bo'lganda qoldiqni hisoblab, natija y ga beriladi. Ya'ni bu buyruq $y=y\%a$ buyrug'iga teng kuchli;

- $y+=a$; buyrug'i bo'yicha y ni $a+3$ ga ko'paytirib, natija y ga beriladi. Ya'ni bu buyruq $y=y*(a+3)$; buyrug'iga teng kuchli;

- $y+=++a$; buyrug'i bo'yicha a ni 1 ga orttirib, natija y ga beriladi va bu yig'indining qiymati yana y ga beriladi. Ya'ni bu buyruq $a=a+1$; $y=y+a$; buyruqlariga teng kuchli;

- $y\%+=a$; buyrug'i bo'yicha y ni a ga bo'lganda qoldiqni hisoblab, natija y ga beriladi va a ning qiymati 1 ga orttiriladi. Ya'ni bu buyruq $y=y\%a$; $a=a+1$; buyruqlariga teng kuchli;

- $y*=-a$; buyrug'i bo'yicha a ning qiymati 1 ga kamaytiriladi va natija y ga ko'paytirilib, natija y ga beriladi. Ya'ni bu buyruq $a=a-1$; $y=y*a$; buyruqlariga teng kuchli;

- $y-=a--$; buyrug'i bo'yicha y dan a ni ayirib, natija y ga beriladi va a ning qiymati 1 ga kamaytiriladi. Ya'ni bu buyruq $y=y-a$; $a=a-1$; buyruqlariga teng kuchli;

- $y+=-y$; buyrug'i bo'yicha y 1 ga kamaytirilib, natija y ga beriladi va yig'indining qiymati y ga beriladi. Ya'ni bu buyruq $y=y-y$; buyruqlariga teng kuchli.

Qiymat berish buyrug'ining $y=++a$; va $y=-a$; ko'rinishi *prefiks* ko'rinishi, $y=a++$; va $y=a--$; ko'rinishlari *postfiks* ko'rinishlari deyiladi.

Misol tariqasida quyidagi dasturni va uning natijalarini tahlil qiling.

```
#include <iostream.h>
int main()
{
    int a, b, c, d, x, y, z, n, m;
    a = 5; b = 6; c = 7; d = 8;
    x = ++a; y = ++a; x += x++; cout << "x=" << x << " y=" << y << endl;
    x = b--; y += --y; y %= d--; c *= --a; cout << "nx=" << x << " y=" << y << " c=" << c;
    n = --d; n = --n; c = --x; cout << "n" << "n=" << n << "nb=" << n << "tc=" << c << "n";
    system("pause");
    return 0;
}
```

Dastur natijasi quyidagicha chop etiladi:

```
x = 13 y = 7 a = 7
x = 7
y = 4 c = 42
n = 0
b = 5 c = 6
```

Eslatma: Qavs ichidagi amallar birinchi bajariladi, keyin funksiyalarning qiymatlari hisoblanadi. Undan keyin bizga hozirgi ma'lum bolgan amallarning birinchi bajarilish nuqtayi nazarida bajariish tartibi quyidagicha (Qavs ichiga olingan amallar bir bosqich amallari hisoblanadi):

$(i++, i--), (++i, --i), !, (*, /, \%), (+, -), (<, >, <=, >=), (=, !=), \&\&, \parallel, ?:, (=, *=, /=, +=, -=, \%)$.

Quyidagi savollarga javob bering

1. Dasturda foydalaniladigan operator deyilganda nimaning tushunasiz?

2. $y=++x$ va $y=x++$ (postfiks va prefiks) operatorini tushuntiring, farqlarini ko'rsatib (misollar keltiring).

3. $y=y-5$; $y=x-3$; yozuvlari qanday o'qiladi?

4. Inkrement va dekrement amallari qanday amallar?

5. $a++$ va $b--$ amallarining qanday farqlari bor?

6. Quyidagi ikkita buyruqni bitta buyruq qilib yozing: a)

b) $x=x-1$; $y=y\%x$; c) $n=n+c$; $c++$;

7. $x++$ va $y--$ amallarining mohiyatini tushuntiring.

8. 0 va o belgilardan qanday farqlaysiz?

9. $\backslash n$ va $\backslash t$ belgilari qanday farqlaysiz?

10. Ma'lumotlarni kiritish operatoridan qanday maqsadda foydalaniladi?

11. Dastur tarkibida qiymati kiritilayotgan o'zgaruvchilar qayerda va qanday yoziladi?

12. Ma'lumotlarni kiritish operatoridan dasturning qayerida, qanday maqsadda foydalanish mumkin?

13. $a++$ va $b--$ amallarining qanday farqlari bor?

14. $x=x-1$; $y=y\%x$; $n=n+c$; $c++$; buyrug'iga binoan -23 6 9 0.3 sonlari bir-biridan probel bilan ajratib yozildi va Enter tugmachasi bosildi.

15. Ma'lumotlarni kiritish operatoridan dasturning qayerida, qanday maqsadda foydalanish mumkin?

16. Ma'lumotlarni chiqarish operatoridan qanday maqsadda foydalaniladi?

17. Ma'lumotlarni chiqarish operatorining qanday ko'rinishlari mavjud?

18. Ekranda 7 ta bo'sh satr chop etish buyrug'i qanday yoziladi?

19. Matnlarni qanday chop etiladi?

20. $\text{cout} << 2 << "[<<<4<<"; "<<5<<"] = "<<18<<"\n"$ buyrug'i qanday ma'lumotni chop etadi?

21. $\text{cout} << "salom" << "\n" << 4 << "\t" << "ta kitob\n"$; buyruq natijasini yozing.

Qiyamat berish buyrug'ining $y=++a$; va $y=-a$; ko'rinishi
 prefiks ko'rinishi, $y=a++$; va $y=a-$; ko'rinishlari
 ko'rinishlari deyiladi.

Misol tariqasida quyidagi dasturni va uning natijalarini tahlil qiling.

```
int main()
{
    int a, b, c, d, x, y, z, n, m;
    a = 5; b = 6; c = 7; d = 8;
    x++ + a; y++ + a; x++ + x++ +; cout << "x=" << x << " y=" << y
    == << a;
    x--b--; y++--y; y%--d--; c*--a; cout << "\nx =" << x << "\n"
    << y << " c=" << c;
    n--d; n--n; c--x; cout << "\n'<< "n=" << n << "\nb = "
    "\tc=" << c << "\n';
    system("pause");
    return 0;
}
```

Dastur natijasi quyidagicha chop etiladi:

$$x = 13 \quad y = 7 \quad a = 7$$
$$x = 7$$
$$y = 4 \quad c = 42$$
$$n = 0$$
$$b = 5 \quad c = 6$$

Eslatma: Qavs ichidagi amallar birinchi bajariladi, keyin funksiyalarning qiymatlari hisoblanadi. Undan keyin bizga hozirgi ma'lum bolgan amallarning birinchi bajarilish nuqtayi nazaridagi bajarilish tartibi quyidagicha (Qavs ichiga olingan amallar bir bosqichda bajariladi):

(i++, i--), (++i, --i), !, (*, /, %), (+, -), (<, >, <=, >=), (==, !=),
&&, ||, ?:, (=, *=, /=, +=, -=, %=).

Quyidagi savollarga javob bering

1. Dasturda foydalaniladigan operator deyilganda niman

1) $\forall x \in A, \exists y \in B$ (har bir x uchun B da y mavjud bo'lsa).

...dekrement amallari qanday amallar?

... va b^- - amallarining qanday farqlari bor?

buyruqni bitta buyruq qilib yozing: a)

b) $x=x-1$; $y=y \% x$; c) $n=n+c$; $c++$;

8) postfika amallarining mohiyatini tushuntiring.

0 va o harfini qanday farqlaysiz? 0 va o belgilaridan

operatoridan qanday maqsadda

... kiritish operatoridan qanday maqsadga

... tarkibida qiymati kiritilayotgan o'zgaruvchilar qayerda va

...dasturling gaverida, ganday

Amotlarni kiritish operatoridan dasturning qayerida, qanday
... qilish mumkin?

bu yug'ri qanday ijro etilishini tushuntiring.

$y \gg z \gg h$; buyrug'iga binoan $-23 \ 6 \ 9 \ 0.3$ sonlari bir

probel bilan ajratib yozildi va Enter tugmachasi bosildi

qanday holda qanday buyruqlar bajariladi?

...shu kerak?

bu' va 't' buyruqlaridan dastur tarkibida qanday maqsadd

...nilaradi? ...chegarish operatoridan qanday maqsadda

Ma'lumotlarni chiqarish operatoridan qanday foydalaniladi?

Ma'lumotlarni chiqarish operatorining qanday ko'rinishla

4) ... qanday?

Ukranda 7 ta bo'sh satr chop etish buyrug'i qanday?

[illegible]

notni chop etadi?

cout<<"salom"<<'\\n'<<4<<'\\t'<<"ta kitob\\n"; buyruq natijas

21. Natijalarni bir-birlaridan ajratib, ya'ni probel (bo'sh o'rin) ajraib chop etish usullarini bayon qiling.

22. Qiymat berish buyrug'i belgisi, tenglik belgisi, tenglik belgisi C++ dasturlash tilida qanday yoziladi?

23. Hozirgacha bizga ma'lum bo'lgan amallarning bajarilish tartibini ayting. Ulardan qaysilari bir bosqich amallari hisoblanadi?

Quyidagi topshiriqlarni bajaring

1. Quyidagi buyruqlar natijasini aniqlang

- a) $a=12; b=4; c=(a+b)/8; a=a+c; c=a+c; b=b-c; b=4*b/c;$
`cout<<a<<b<<c<<"\n";`
- b) $x=2; y=5; y=2*x+y; y=2*x+y; x=2*x+y; y=2*x-3; x=x-y;$
`cout<<"x="<<x<<" "; y="<<y<<"\n";`
- c) $q=4; q=q+2; q=q+2; q=q+2; q=q+2; q=q+2; q=q+2;$
`cout<<"q="<<q<<"\n";`
- d) $s=4; d=24; c=-5; d=-c; c*=s-10; s+=c++; c--; d--;$
`cout<<"s="<<s<<" "; d="<<d<<" "; c="<<c<<"\n";`
- e) $n=15; m=20; k=12; k/=6; n\%=k+2; m+=m++; m-=m--; n+=m;$
`cout<<m<<k<<n<<"\n";`
- f) $x=5; y=8; x+=++y; x-=x++; y=y+x; y\%=-8; cout<<2*x+3*y<<"\n";$
- g) $a=12; b=12; a-=++a; b-=b++;$ `cout<<b<<a<<"\n";`

IDENTIFIKATORLAR VA TURLAR

3.1. Identifikatorlar va kalit so'zlar

Matematikada bo'lgani kabi C++ dasturlash tilida ham, dastur tashkil topadigan o'zgaruvchilar va o'zgarishlar o'z nomiga ega bo'lishi kerak. Bu nomlar matematikada lotin alifbesining bir, ikki yoki uch harfidan tashkil topadi. C++ dasturlash tilida esa o'zgaruvchi nomlari ketma-ket 255 tagacha bo'lan belgilardan iborat bo'lishi va ular **identifikatorlar** deb yuritiladi.

Identifikator tarkibida foydalaniladigan belgilar deyilganda faqat katta va kichik harflari, raqamlar va tag chiziqli (_) belgilar bo'lishi kerak.

Identifikatorlarni yozishga katta va kichik harflar farqlanadi.

Masalan: `romb` va `Romb`; `n12` va `N12`; `aylana_raduisi` va `aylana_Raduisi` kabi identifikatorlar boshqa-boshqa identifikatorlardir.

Shundan tashqari C++ dasturlash tilining kalit so'zlari, funksiya nomlari va obyektlar nomlari ham identifikatorlar deb atalishi mumkin.

Lekin C++ dasturlash tilining operatorlaridan o'zgaruvchilar va o'zgarishlarning nomlari sifatida foydalanib bo'lmaydi.

Agar identifikator ikki yoki undan ortiq so'zlardan tashkil topgan bo'lsa ham ular bir-birlaridan probel bilan ajratilmaydi. Ular qo'shib yoziladi. Agar ajratib yozish kerak bo'lsa, tagchiziqli (_) dan foydalaniladi.

Masalan: `tenglamaning_yechimi` yozuvi identifikatordir. C++ dasturlash tilining kalit so'zlari deyilganda buyruqlarni tashkil qilishda foydalaniladigan ingliz tilida yozilgan operatorlar (yordamchi so'zlar) ham tushuniladi.

Masalan, `if`, `else`, `return`, `while`, kabi so'zlar operatorlar yoki kalit so'zlardir.

Yuqorida qyd qilganimizdek, C++ dasturlash tilining operatorlaridan identifikator sifatida foydalanib bo'lmaydi.

`x`, `a1`, `b819`, `doira`, `maktab`, `ptk_meb`, `Rels89` larning har biri identifikator bo'la oladi. Lekin,

<code>float</code>	3ka
<code>dars bo'ldi</code>	qovun, tarvuz
<code>while</code>	48

if for a/bc
lar identifikator bo'la olmaydi.

O'zgaruvchilar

C++ dasturlash tilida o'zgaruvchi tushunchasi matematika ma'nosiga mos keladi. Ya'ni dasturni ijro etish jarayonida qiymatini o'zgartirishi mumkin bo'lgan kattaliklar (identifikatorlar) o'zgaruvchilar deyiladi.

Dastur tarkibida qandaydir o'zgaruvchilardan foydalanish uchun ular, albatta, e'lon qilinishi kerak. *E'lon qilishdan maqsad o'zgaruvchi uchun EHM xotirasidan joy ajratishdir.*

O'zgaruvchini e'lon qilishda, albatta, uning *turi, nomi, dastlabki qiymati ko'rsatiladi*, ya'ni initsializatsiyalanadi. Amaliy initsializatsiyalanmasdan ham e'lon qilinishi mumkin. Ya'ni initsializatsiyalashni o'zgaruvchi e'lonida emas, balki dastur ichida ham amalga oshirish mumkin.

Masalan: `int a=5, b=10, c=43;` initsializatsiyalangan holda e'lon qilish;

`int a, b, c;` initsializatsiyalamasdan e'lon qilish;

Bunda `int` – tur nomi bo'lib, *integer* (butun) so'zining qisqartirilgan shakli. Demak, `a, b, c` lar faqat butun qiymatlarnigina qabul qilish mumkin. Boshqa turli qiymatlarni qabul qila olmaydi. Masalan, haqiqiy qiymat berilsa, haqiqiy sonning kasr qismini tashlab yuborib, butun qismini oladi.

Har xil tur uchun xotirada har xil joy ajratiladi. Hatto bitta tur uchun ham har xil razryadli mashinalarda har xil hajmdagi joylar ajratilishi mumkin.

Yuqoridagi misollarda ko'rsatilganidek, bitta turdan foydalanib, nechta o'zgaruvchilarni e'lon qilish mumkin va ular bir-birlaridan belgisi bilan ajratib yoziladi hamda `(;)` belgisi bilan yakunlanadi.

O'zgarmaslar

Dasturni ijro etish jarayonida o'z qiymatini o'zgartirmaydigan kattaliklar (identifikatorlar) o'zgarmaslar deyiladi. O'zgaruvchilarni initsializatsiya qilmasdan (ya'ni boshlang'ich qiymatini bermasdan) e'lon qilish mumkin. Lekin o'zgarmaslar, albatta, initsializatsiyalanishi shart va bu qiymatni, dasturni bajarish davomida o'zgartirib bo'lmaydi.

O'zgarmaslar ham o'zgaruvchilar kabi oldindan e'lon qilinadi. E'lon qilishdan maqsad ham bu o'zgarmaslar uchun xotiradan joy ajratishdir.

O'zgarmaslarni e'lon qilish uchun `const` kalit so'zidan foydalaniladi, undan keyin o'zgarmas turi va nomi yozilib, yuqorida ko'rsatilganidek, dastlabki qiymati ko'rsatiladi va bu qiymat boshlang'ich qiymat hisoblanadi. Sonlardan tashqari belgili o'zgarmaslar tutuq belgisi (apostrof) ichiga olib yoziladi. Misol sifatida quyidagi dasturni tahlil qiling:

```
#include <iostream.h>
using namespace std;
int main()
{
    int n,m,k=5; int const x=3,y=12;
    n=k*2;m=x+n;k=n+y;
    cout <<"n="<<n<<" m="<<m<<" k="<<k<<"\n";
    system("pause");
    return 0;
}
```

Dastur natijasi quyidagicha bo'ladi:

`n=10 m=13 k=22`

Manfiy o'zgarmaslarni ham aslida ishorasiz o'zgarmas deb qaraymiz, faqat unga ishorani o'zgartirish amali qo'llangan deb tushunamiz. Masalan, `-3` o'zgarmasni ishorasiz `3` soniga minus ishora qo'yilgan yoki `3` ishorasiz soni `-1` ga ko'paytirilgan deb tushunish kerak.

Quyidagi savollarga javob bering

1. Kalit so'zlar qanday so'zlar?
2. O'zgaruvchi va o'zgarmaslar qanday miqdorlar, ular qayerda va qanday e'lon qilinadi?
3. O'zgaruvchi va o'zgarmaslarni e'lon qilishdan maqsad nima?
4. C++ dasturlash tilida o'zgarmaslar va o'zgaruvchilar qanday farq qiladi?
5. O'zgarmaslar qanday kalit so'z yordamida qanday e'lon qilinadi?
6. O'zgarmaslarning qiymatlari faqat son ko'rinishida bo'ladimi?
7. Identifikator nima va u qanday yoziladi?

8. Identifikatorlarni yozishda qanday va nechta belgilardan foydalanish mumkin?
 9. for; 25calom; олма; uraa!; yosh bola; 423; larning qaysi identifikator bo'la oladi?

Quyidagi topshiriqlarni bajaring

1. Tomonlari a, b va c bo'lgan uchburchakning yuzi, perimetri yarim perimetrini to'g'ri burchakli uchburchakning uchlarida chop etish dasturini tuzing.
2. 1 dan 16 gacha bo'lgan sonlarni ekranning o'rtasidan to'rtta ustunda va to'rtta ustunda kvadrat matritsa shaklida chop etish dasturini tuzing.
3. $z=2*x-3/y-a/b$, $a=5$ funksiya qiymatini hisoblash dasturini tuzing.
4. 34 sonini trapetsiyaning uchlarida chop etish dasturini tuzing.
5. Besh xonali son kiritiladi. Shu sonning raqamlarini to'rt burchakning uchlarida va o'rtasida chop etish dasturini tuzing.
6. Olti xonali son kiritiladi. Shu sonning raqamlari o'rtta arifmetik topish dasturini tuzing.

3.2. C++ dasturlash tilining standart (tayanch) turlari

C++ dasturlash tilining tayanch turlari nomi, baytlardagi o'lcham va oladigan qiymatlari chegarasi quyidagi jadvalda berilgan.

C++ dasturlash tilining tayanch turlari		
Tur nomi	Baytlardagi o'lchami	Qiymat chegarasi
Bool	1	true yoki false
unsigned short int	2	0..65535
short int	2	-32768..32767
unsigned long int	4	0..42949667295
long int	4	-2147483648..2147483647
int (16 razriyadi)	2	-32768..32767
int (32 razriyadi)	4	-2147483648..2147483647
unsigned int (16 razriyadi)	2	0..65535
unsigned int (32 razriyadi)	4	0..42949667295
unsigned char	1	0..255

Char	1	-128..127
Float	4	1.2E-38..3.4E38
Double	8	2.2E-308..1.8E308
long double (32 razriyadi)	10	3.4e-4932..3.4e4932
Void	2 yoki 4	

Butun son turi

Butun son turi **int** (butun), **long** (uzun), **short** (qisqa) kabi kalit so'zlari yordamida e'lon qilinadi. Shuningdek, **unsigned** (ishorasiz) kalit so'zidan ham foydalanilsa, o'zgaruvchi yoki o'zgarmaslar faqat 0 yoki undan katta qiymatlarni qabul qilishi mumkin.

Masalan, butun turli o'zgaruvchilar

```
int a,b,c;
unsigned long int a,b,c;
short int a,b,c;
long int a,b,c;
unsigned short int a,b,c;
```

kabi ko'rinishlarda e'lon qilinishi mumkin.

Agar butun turli o'zgaruvchiga haqiqiy qiymat berilsa, kompilyator xatolik haqida xabar bermaydi. Lekin xatolik sodir bo'ladi. Chunki butun turli o'zgaruvchi haqiqiy sonning faqat butun qismini qiymat sifatida oladi. Shuningdek, ishorasiz butun o'zgaruvchiga manfiy qiymat berilsa ham kompilyator xatolik haqida xabar bermaydi. Ammo bu o'zgaruvchi kutilmagan boshqa qiymatni oladi. Lekin haqiqiy turli o'zgaruvchi ishorasiz sonlarni ham, butun sonlarni ham qiymat sifatida oladi va xatolik sodir bo'lmaydi. **Faqat jadvalda ko'rsatilgan chegaradan chiqib ketmaslik kerak.** Yuqoridagi kabi kompilyator sezmaydigan xatoliklar **mantiqiy xatoliklar** deyiladi. Mantiqiy xatoliklarni ko'pincha dastur tuzayotgan odam ham sezmay qolishi mumkin bo'lganligi uchun xavfli xatolik hisoblanadi. Chunki dastur bajarilib, natijasi e'lon qilinadi, ammo bu natija mantiqan boshqa masalaning natijasi bo'ladi.

Demak, C++ dasturlash tilida yozilgan dasturlardagi xatoliklarni ikki guruhga bo'lamiz:

1. Kompilyator sezadigan **sintaktik** xatolar, ya'ni til qoidalariga zid holda yozilgan yozuvlar, noto'g'ri foydalanilgan tinish belgilari, qavslar, kalit so'zlar va boshqalar.

2. Mantiqiy xatolar, ular kompilyator tomonidan qayd etilgan. Lekin masala natijasi xato bo'ladi.

Haqiqiy son turi

Haqiqiy turli o'zgaruvchilarni e'lon qilish uchun **float** kalit so'z foydalaniladi. Agar o'zgaruvchilarning qiymati juda katta va kichik, ya'ni $5.18 \cdot 10^{18}$ yoki $9.15 \cdot 10^{-12}$ ko'rinishlarda bo'lsa **double** yoki **long double** kalit so'zlaridan foydalanib e'lon qilinadi.

$\alpha \cdot 10^{\beta}$ ko'rinishdagi sonlarda α -sonning mantissasi, β sonning tartibi deyiladi.

Bayon qilingan fikrlar mohiyatini amalda quyidagi dastur misol tushunish mumkin:

```
#include <iostream.h>
using namespace std;
int main( )
{
    short int a,b,c;
    float x,y,z;
    double n,m;
    unsigned p,q;
    a=17; b=20.96; cout<<"a="<<a<<"b="<<b<<"\n";
    x=a+5; y=b; z=10.231; n=x; m=3.5E4;
    p=-78; q=96;
    cout<<"x="<<x<<"y="<<y<<"z="<<z<<"\n";
    cout <<"n="<<n<<"m="<<m<<"\n";
    cout <<"p="<<p<<"q="<<q<<"\n";
    system("pause");
    return 0;
} (ishlaydi)
```

Dastur natijasi quyidagicha bo'ladi.

```
a=17 b=20
x=22 y=20.96 z=10.231
n=22 m=35000
p=-78 q=96
```

Dastur natijalarini mustaqil tahlil qiling.

Mantiqiy tur

Mantiqiy turlar bilan ishlash jarayoni matematik mantiq fanining qonuniyatlariga asoslanadi. Matematik mantiq fani fikrlashning shakli

qonuniyatlarini o'rganuvchi fandır. Matematik mantiq o'zi mulohazalarga asoslanadi. Mulohaza esa ixtiyoriy fikr bo'lib, u rost yoki yolg'on (**false**) bo'la oladi.

Masalan, "Olma daraxti kuzda gullaydi", "Kompyuter yuk mashina", "Hafta yetti kundan iborat", "Oshxonada xom pishiriladi". Ko'rinish turidagi, dastlabki ikkita mulohaza yolg'on (**false**), oxirgi ikkita mulohaza esa rost (**true**).

Bunday mulohazalar ham borki, ularning qiymatlarini aniqlab bo'lmaydi. Masalan, "Hozir qish fasli", "0.00001 juda kichik son", "Bu yil talaba bo'ladi", Birinchi mulohazaning rost yoki yolg'onligi uning qachon aytilganligiga bog'liq, ikkinchi mulohazada 0.00001 soni juda kichik son deyilgan. Ammo qo'shimcha shartlar qo'yilgan bo'lmasa juda kichik son deb ataluvchi son mavjud emas, uchinchi mulohazaning rost yoki yolg'onligini ham aniqlab bo'lmaydi.

Dasturlashda asosan matematik mulohazalar bilan ish ko'riladi. Matematik mulohazalarni yozishda quyidagi munosabat belgilaridan foydalaniladi. Bu belgilardan ikkita son yoki ifodalarning qiymatlarini solishtirish (taqqoslash) maqsadida foydalaniladi.

- tenglik belgisi;
- katta yoki tenglik (kichik emaslik) belgisi;
- kichik yoki tenglik (katta emaslik) belgisi;
- teng emaslik belgisi;
- kattalik belgisi;
- kichiklik belgisi.

Masalan: $3+2 \cdot 4 > 36/9-1$; $29-4 = 5 \cdot 5$; $29! = 5 \cdot 6$ mantiqiy ifodalarning qiymatlari **true** (rost). $56/7! = 2 \cdot 4$; $243/81! = 24/8$ mantiqiy ifodalarning qiymatlari esa **false** (yolg'on).

Buningdek, $x+6 = 21$ ifodaning qiymati x ning 15 ga teng va undan katta qiymatlarida **true**, boshqa barcha qiymatlarida **false** bo'ladi.

Tarkibida bitta munosabat belgisi bo'lgan mantiqiy ifodalar oddiy mantiqiy ifodalar deyiladi.

Mantiqiy amallar ishtirok etgan mantiqiy ifodalar esa murakkab mantiqiy ifodalar deyiladi.

Dasturlashda asosan quyidagi 3 ta mantiqiy amallardan foydalaniladi.

Mantiqiy amallar

1. Mantiqiy ko'paytirish amali

Amalning umumiy ko'rinishi quyidagicha:

a&& b

Bunda **a** va **b** lar mantiqiy ifodalar yoki mantiqiy o'zgaruvchi bo'lib, amal natijasi faqat **a** va **b** larning har ikkalasi rost bo'lganda rost, qolgan barcha hollarda yolg'on bo'ladi. Bu amal matematik **konyunksiya** mantiqiy amali bilan teng kuchli, ya'ni ikkita amalni (&&) belgisi "**va**" so'zini almashtiradi.

Misollar:

(37-21)!=13&&49/7==4+3 ifodaning qiymati rost.

x=15; x>20 && x!=16 ifodaning qiymati yolg'on.

2. Mantiqiy qo'shish amali

Amalning umumiy ko'rinishi quyidagicha:

a || b

Bunda **a** va **b** lar mantiqiy ifodalar yoki mantiqiy o'zgaruvchi bo'lib, amal natijasi **a** va **b** larning har ikkilasi yolg'on bo'lganda yolg'on, boshqa barcha hollarda rost bo'ladi. Bu amal matematik **dizyunksiya** amaliga teng kuchli, ya'ni || belgilari "**yoki**" so'ziga teng keladi.

Masalan,

a=-16; b=32; 2*a-b<0||b+a<1; (52+a>5+b)||true ifodalarning qiymatlari rost

0||a-b>0 ifodaning qiymati esa yolg'on.

3. Mantiqiy inkor amali

Amalning umumiy ko'rinishi qo'yidagicha:

!a

Bunda **a** mantiqiy ifoda yoki mantiqiy o'zgaruvchi bo'lib, agar uning qiymati **true** bo'lsa, amal natijasi **false**, amal natijasi **false** bo'lsa, amal natijasi **true** bo'ladi.

Masalan,

!true; !(48/4+6==3*6); !3 ifodalar qiymatlari false

!(9*8==64); !(3*8==20 && 100%4==2); 0! ifodalarning qiymatlari esa true bo'ladi.

C++ dasturlash tili mantiqiy ifoda o'rnida 0 soni yoki qiymati nolga teng bo'lgan ifoda bo'sa, uni false deb, 0 dan farqli ixtiyoriy son yoki qiymati noldan farqli ifodani true deb qabul qiladi.

Mantiqiy ifodani mantiqiy o'zgaruvchiga qiymat sifatida berish ham mumkin.

o'zgaruvchilar **bool** kalit so'z bilan e'lon qilinadi. Bu kalit mantiqiy ilmi asos solgan ingliz matematigi Jorj Bui shunday tanlangan.

Quyidagi dasturni tahlil qiling.

```
#include <iostream.h>
using namespace std;
int main()
```

```
{ int x, y, z, x1, y1, z1;
```

```
float a, b, c;
```

```
int n, m, k, p, q;
```

```
x=10; y=10; z=21; a=19.25; b=0.75; c=40.16;
```

```
x1=10; y1=10; z1=21; a=19.25; b=0.75; c=40.16;
```

```
n=(x+y)/2; m=(x+y)/2; k=y1-x==z+13;
```

```
p=(a-b)>b-c; m=(x+y)/2; k=y1-x==z+13;
```

```
q=(a-b)>b-c; m=(x+y)/2; k=y1-x==z+13;
```

```
if (n==m && n==k) {
```

```
cout << "n=" << n << " m=" << m << " k=" << k;
```

```
if (p==q && p==k) {
```

```
cout << "p=" << p << " q=" << q << endl;
```

```
system("pause");
```

```
return 0;
```

Bu dasturda n, m, k, p, q o'zgaruvchilarning turi mantiqiy tur

bo'lib, ular mantiqiy ifodalarni qiymat sifatida olmoqda.

Dastur javobi quyidagicha bo'ladi.

```
n=1 m=1 k=0
```

```
p=1 q=1
```

Shunday qilib, murakkab mantiqiy ifodalarni yozishda amalning ikki tarafidan joy qoldirmasa ham bo'ladi. Shuningdek, amalning ikki tarafidagi mantiqiy ifodalarni qavsga olish yoki olmaslikning ham ahamiyati yo'q. Lekin dastur tushunarli bo'lishi uchun mantiqiy amallarning har ikkala tarafidan bittadan (agar kerak bo'lsa xohlagancha) bo'sh joy qoldirish, zarurat bo'lganda oddiy mantiqiy ifodalarni qavsga olish maqsadga muvofiq.

Masalan, quyidagi to'rtta mantiqiy ifodani ham kompilyator xato demaydi.

```
p=k+3>0||x+y-1<=16 && 1;
```

```
p=k+3>0 || x+y-1 <= 16&& 1;
```

```
p=(k+3>0)||(x+y-1<=16) && 1;
```

$p=(k+3>0) \parallel (x+y-1 \leq 16) \&\& 1;$

Eslatma: hatto arifmetik amallarning ikkala tarafidan bittadan joy (probel) qoldirilsa, dasturni tushunish yengil bo'ladi. Lekin aytilganlar qat'iy talab emas.

Masalan, $x+3*y+11/3$ ni $x + 3 * y + 11/3$ kabi yozish ma'qul.

Biz hozirgacha tanishgan amallarning bir-birlariga nisbatan bajarilish nuqtayi nazaridan ustunlik darajasi va bosqichlarga bo'linishi quyida keltirilgan.

$(i++, i--), (++i, --i), !, (*, /, \%), (+, -), (<, <=, >, >=), (==, !=), \&\&, \parallel, (=, *=, /=, +=, -=, \%=)$

Bunda qavsga olingan amallar bir bosqich amallari hisoblanadi. Bosqich amallari o'rtasida ustunlik daraja bo'lmaydi. Demak, amallar ifodada aralash holda kelsa, eng birinchi $i++$, $i--$ amallar, oxirida $=, *=, /=, +=, -=, \%=$ amallar bajariladi.

Misollar: Quyidagi shartlar bajarilsa **true**, aks holda **false** qiymat oladigan mantiqiy ifodalarni tuzing.

1. a-juft, b esa toq son.
2. a,b,c sonlarning hech bo'lmaganda bittasi musbat.
3. a,b,c mantiqiy o'zgaruvchilardan hech bo'lmaganda bittasi true.

Javoblar:

1. $a\%2==0\&\& b\%2==1;$
2. $a>0 \parallel b>0 \parallel c>0$
3. $a \parallel b \parallel c = true$

ASCII belgilari (kodi)

ASCII belgilari deyilganda 0 dan 255 gacha kodlangan klaviatura mavjud bo'lgan barcha belgilar va ASCII ning qo'shimcha belgilari tushuniladi. Bu belgilar klaviaturadagi harflar, raqamlar, qavslar, tinchlik belgilari va boshqa belgilardir.

ASCII – bu American Standart Code for Information Interchange (Axborot almashishga mo'ljallangan Amerika standart kodi (Американский стандартный код для обмена информацией)).

Turni boshqa turga keltirish

Bunda dastur tanasida o'zgaruvchining turini boshqa turga o'zgartirish haqida gap boradi va bunday imkoniyat C++ dasturlash tilida mavjud.

O'zgaruvchi turini boshqa turga keltirishning oshkor va oshkor bo'lmagan usullari mavjud.

O'zgaruvchi turini boshqa turga oshkor o'tkazish uchun yangi tur oshkor yozib ko'rsatiladi. Masalan, quyidagi dasturda bu hol ko'rsatib qilingan:

```
#include <iostream.h>
using namespace std;
int main()
{
    int a=19, b;
    float x=36.035, y=7.41;
    cout << "x=" << x << "\n";
    cout << "y=" << y << endl;
    x=(int)x; // oshkor holda haqiqiy turli x o'zgaruvchining turi butun
    turiga o'tkazilmoqda;
    b=y; // nooshkor holda y o'zgaruvchi haqiqiy turdan butun turga
    o'tkazilmoqda;
    cout << "x=" << x << "\n";
    cout << "b=" << b << endl;
    system("pause >> void");
    return 0;
} Dastur natijasi quyidagicha bo'ldi:
x = 36.035
y = 7.41
x=36
b=7
cout operatorida endl; kalit so'zidan foydalanildi. Bu kalit so'z,
albatta, shart bo'lmagan operator bo'lib, satrni chop etish tugaganligini
bildiradi. Ya'ni endl – end line(satr tugashi) demakdir. Turni
o'zgartirishni quyidagicha ham amalga oshirish mumkin.
{
    int a = 19, b;
    float x = 36.035, y = 7.41;
    cout << "a=" << (int) x << "\n";
    cout << "b=" << (int) y;
    system("pause");
    return 0;
}
```



```

}
Dastur natijasi quyidagicha bo'ldi:
a = 36
b = 7

```

Demak, turni boshqa turga keltirish uchun dastlab o'zgaruvchi initsializatsiya qilinadi, so'ngra o'tkazilayotgan turli o'zgaruvchi tomonidan o'zlashtiriladi. Yoki ikkinchi dasturda ko'rsatilganidek, o'zgaruvchi o'zlashtirish jarayonida ham amalga oshirish mumkin. Faqat o'zgartirishda bitta muammo paydo bo'lishi mumkin. Agar initsializatsiya qilingan sonning xotirada egallagan o'zlashtirilayotgan o'zgaruvchi uchun ajratilgan joydan katta bo'lsa, noto'g'ri natijaga ega bo'lish mumkin.

Belgi va satr turi

Belgi turi **char** kalit so'zi yordamida e'lon qilinadi va bu tur mansub o'zgaruvchilar klaviaturadagi ixtiyoriy belgini qiymat sifatida olishi mumkin. Faqat qiymat sifatida berilayotgan belgilar tutuq belgi (apostrof) ichiga olib yoziladi. Shuning uchun belgi turiga mansub o'zgaruvchi tutuq belgisini qiymat sifatida ololmaydi. Masalan, 'b', '6', 'R', '6', '@', '*', '+', '!', 'z' larning har birini **char** tur o'zgaruvchi qiymat sifatida olinishi mumkin.

Agar **char** turli o'zgaruvchiga beriladigan qiymat ikki yoki undan ortiq belgilardan iborat bo'lsa, o'zgaruvchi o'sha belgilar ketma-ketligidagi oxirgi belgini oladi. Masalan, agar **a char** turli o'zgaruvchi bo'lsa, **a = 's';** va **a = 'olmos'** buyruqlari teng kuchli bo'ladi.

Shuni unutmaslik kerakki, 'a', 'b', 'n', 'g', 'c' lar tarkibida harflar bo'lsa ham ular o'z qiymatlarini o'zgartira olmaganligi uchun **char** turli o'zgaruvchilarning qiymatlari o'zgarmaslar bo'ladi.

Shuningdek, ko'p belgili qiymatlar oluvchi o'zgaruvchilarning turli **string** ko'rinishida e'lon qilinadi va uning qiymatlari qo'shtirnoq ichiga olib yoziladi.

String turli o'zgaruvchilarning qiymatlari dasturni yozish jarayonida bitta satrga sig'masa, har bir satr (\n) belgisi bilan tugashi kerak. Aks holda keyingi satrdagi belgilar davomini o'zgaruvchiga taalluqli deb tushunmaydi va xatolik sodir bo'ladi. Ammo satrlar fizik satr oxirida (\n) belgisi bilan ajratilgan bo'lsa ham ularni ekranga chop etishda qo'shib, bitta satr shaklida chop etiladi. Aytilganlarni quyidagi dastur misolida tahlil qilib ko'ring:

```

#include <iostream.h>
using namespace std;
int main( )
{
    char n, s; string m, d, k;
    n="90"; m="8"; m="Hosila";
    // Olishning barcha qoidalarini\
    // bilishdan oldin hosilalar jadvalini\
    // yod oling";
    cout << d + n + s;
    cout << k << "\n";
    system ("pause");
    return 0;
}

```

Dastur natijasi quyidagicha chop etiladi:
Hosila olishning barcha qoidalarini bilishdan oldin hosilalar jadvalini yod oling80.

Demak, **string** turli o'zgaruvchilarning qiymatlari **matn(satr)**lar bo'ladi. Lekin, **string** turli o'zgaruvchilar bitta belgini ham qiymat sifatida olishi mumkin. Faqat bu belgi tutuq belgisi ichiga emas, balki qo'shtirnoq ichiga olib yozilishi kerak. Masalan, "+", "[", "!", "M", "!", "#", ...

sizeof amali

Har xil turli o'zgaruvchilar, o'zgarmaslar, turning o'zi (ya'ni turning nomi) EHM xotirasida har xil joylar egallaydi. Hatto bir turli o'zgaruvchilarning o'zi har xil kompyuterlarda va har xil operatsion tizimlarda har xil joy egallashi mumkin. O'zgaruvchilar, o'zgarmaslar va turlarning xotiradan qancha joy egallashini aniqlash maqsadida **sizeof** operatori (kalit so'zi)dan foydalaniladi. Buni quyidagi dastur misolida ko'ramiz:

```

#include <iostream.h>
using namespace std;
int main( )
{
    int a, b, c; float x, y, z;
    cout << "\n butun =" << sizeof (int);
    cout << "\n haqiqiy =" << sizeof (float);
    cout << "\n bul turi =" << sizeof (bool);
    cout << "\n a =" << sizeof (a);
}

```

```
cout << "n x =" << sizeof(x) << endl;
return 0;
}
```

Dastur natijasi quyidagicha bo'ladi.

```
butun =4
haqiqiy =4
bul turi =1
a =4
x =4
```

Demak, butun, haqiqiy turlar va butun, haqiqiy qiymatlar xotira bayt joyini, bul turi esa 1 bayt joyini egallaydi.

Sanab o'tiladigan turlar

Turning sanab o'tiladigan deyilishiga sabab, yaratiladigan mansub o'zgaruvchilar olishi mumkin bo'lgan qiymatlar ko'rsatiladi. O'sha sanab ko'rsatiladigan qiymatlar esa o'zgaruvchilarga iborat bo'ladi va ular noldan boshlab butun sonlar bilan kodlanadi. Bunday o'zgaruvchilar sanab ko'rsatilgan o'zgaruvchilardan boshlab qiymatlarni qabul qila olmaydi va aslida o'sha o'zgaruvchilarning kodlarini oladi. Bunday turni e'lon qilish uchun **enum** (enumeration) sanamoq) kalit so'zidan foydalaniladi. Masalan,

```
enum meva{olma, nok, o'rik, olcha, anor, uzum};
meva x,y;
```

kabi e'lon berilgan bo'lsa, x va y o'zgaruvchilar faqat { } belgisi ichidagi qiymatlardan birinigina qabul qilishi mumkin. Boshqa qiymatlarni qabul qila olmaydi.

Demak, sanab o'tiladigan turlarni yaratish jarayoni **enum** kalit so'zidan boshlanib, undan keyin tur nomi yoziladi va undan keyin { } belgilari ichida ushbu turga mansub o'zgaruvchilar qabul qilinishi mumkin bo'lgan o'zgaruvchilarning qiymatlari bir-birlaridan vergul bilan ajratilgan holda sanab ko'rsatiladi va oxirida } belgidan so'ngi qo'yiladi.

Aslida sanab o'tilgan turlar ham butun tur bilan bir xil, ya'ni bu turga mansub o'zgaruvchilar ham butun qiymatlarnigina oladi. Chunki yuqoridagi **olma**, **nok**, **o'rik**, ...larga oshkor holda ko'rsatilgan bo'lmasa ham mos ravishda 0, 1, 2, ... sonlar bilan kodlanadi. Yuqoridagi keltirilgan misolda keyingi buyruqlar

```
x = o'rik; y = olcha; cout << "x =" << x << "y =" << y;
```

kabi bo'lsa, dastur natijasi quyidagicha bo'ladi:

```
x = 1 y = 3 kabi bo'ladi.
```

Ushbu amalda { } qavslari ichidagi so'z sifatida beriladigan har bir ma'no jihatidan ularga mos sonlarga to'g'ri kelishi kerak.

enum hafta{dush, sesh, chor, pay, juma, shanba, yaksh};
kabi e'lon sintaktik jihatdan to'g'ri bo'lsa ham, mantiqan to'g'ri bo'lmaydi. Chunki dushanbaga nol qiymati to'g'ri kelmoqda. Haftaning nolinchisi kuni esa yo'q.

Agar ushbu tur

```
enum hafta{dush=1, sesh, chor, pay, juma, shan, yaksh};
```

kabi e'lon qilinsa, to'g'ri bo'ladi. Natijada, haftaning har bir kuni o'zgaruvchilarga mos keladi.

Sanab o'tiladigan turlarda qaysidir o'zgaruvchilarning initsiyalizatsiya qilinishi, avtomatik tarzda keyingisi undan keyingi butun qiymatni oladi. Masalan,

```
enum talaba {Olim=8, Sobir, Botir=15, Karim, Anvar};
```

Ushbu e'loni Olimning kodi 8, Sobirniki 9, Anvarniki esa 17 bo'lishini bildiradi. Ikkita qiymat ham bir xil initsiyalizatsiya qilinishi mumkin.

Masalan, **enum tur1{a1 = 8; a2 = 8; a3, a4, a5 = 42; a6, a7};** kabi e'lon bo'lsa, a3 ning qiymati 9, a6 niki 43 bo'ladi. Sanab o'tilgan o'zgaruvchilarning ifodalar va hatto manfiy qiymatlar ham oshishi mumkin.

```
enum misol {uch=3; to'qqiz=uch * 5; x=-4; y, z};
```

kabi e'lon bo'lsa, to'qqizga 15 qiymati, z ga esa -2 qiymati mos keladi.

Xulosa qilib aytadigan bo'lsak, { } qavslari ichida sanab ko'rsatilgan o'zgaruvchilarning qaysi biridan dasturning qayeridadir foydalansak, uning o'rniga kodi qo'yilib hisoblanadi. Buni quyidagi dastur misolida tushunish mumkin:

```
#include <iostream.h>
using namespace std;
int main( )
{ enum sabzavot {kart=500, sabzi=300, piyoz=400, sholgom=200,
karam=700};
int a, b, c;
a=3*kart+5 * sabzi + 6 * karam;
cout << "Jami xarid puli =" << a;
system ("pause");
```

```
return 0;
```

```
}
```

Dastur natijasi quyidagicha bo'ladi:

Jami xarid puli = 7200 kabi bo'ladi.

Quyidagi savollarga javob bering

1. Qanday butun turlarni bilasiz va ularning farqi nimada?
2. Qanday haqiqiy turlarni bilasiz va ularning farqi nimada?
3. Mantiqiy turni e'lon qilishda qaysi so'zdan foydalaniladi?
4. Matn turini e'lon qilishda qaysi so'zdan foydalaniladi?
5. Belgi turini e'lon qilishda qaysi so'zdan foydalaniladi?
6. Matematik mantiq fani nimani o'rganadi? Mulohaza nima?
7. C++ dasturlash tilida tenglik va teng emaslik belgilarini yozing.
8. Mantiqiy ko'paytirish, mantiqiy qo'shish va mantiqiy inkor amallarining umumiy ko'rinishini yozing va ularning natijasini ayting.
9. Siz bilgan arifmetik va mantiqiy amallarning bajarilish tartibini ayting.
10. O'zgaruvchi turini o'zgartirish usullarini ayting.
11. C++ dasturlash tilida izohlardan nima maqsadda va qanday foydalaniladi?
12. Belgi, satr turlari qanday turlar va bu turlardan qachon foydalaniladi?
13. Sanab o'tiladigan turlar qanday turlar va ular qanday, qayerda e'lon qilinadi?
14. enum meva{olma, o'rik, anor=5, yongoq, behi=2, anjuruzum=25, tut, shaftoli}; ko'rinishda meva nomli tur e'lon qilingan. Har bir meva nomining kodini aniqlang.

Quyidagi fikrlarni C++ dasturlash tilida yozing

1. a juft, b esa toq son
2. x 3 ga karrali, y esa 7 ga karrali
3. Kvadrat tenglama yagona ildizga ega
4. Chiziqli tenglama yechimga ega emas
5. x, y sonlarining qiymatlarini almashtiring
6. Tomonlari a, b, c bo'lgan uchburchakning yuzi S ga berilsin
7. Ikkita natural sonning o'rta arifmetigi ularning o'rta geometrigidan kichik emas

1. To'rt xonali sonning oxirgi raqami 3 ga teng
2. Uch xonali sonning oxirgi 2 ta raqamlari yig'indisi 9 ga teng
3. Uch xonali sonning birinchi va oxirgi raqamlarining o'rni almashtirilsa, u 99 ga kamayadi
4. Qo'shimcha o'zgaruvchilardan foydalanmasdan a va b larning qiymatlarini almashtiring.
5. Ikkita qanday ko'p xonali sondan uning raqamlari yig'indisi 10 ga ayirma 9 ga karrali bo'ladi.
6. x, y, z mantiqiy o'zgaruvchilarning kamida bittasi chin.
7. x, y, z mantiqiy o'zgaruvchilarning hammasi chin.
8. Natural a va b sonlari toq sonlar va x va y sonlarining hech birida bittasi toq son.
9. Butun a, b, c sonlarning hech bo'lmaganda bittasi musbat yoki a, b, c sonlarining faqat bittasi musbat son.
10. Butun a, b, c sonlardan faqat ikkitasi musbat son.
11. Berilgan butun n son ikki xonali, m son esa uch xonali.
12. Berilgan butun x son ikki xonali yoki uch xonali.
13. Berilgan uchta sonlardan hech bo'lmaganda 2 tasi bir biriga teng.
14. Berilgan uchta sonlardan hech bo'lmaganda bir jufti o'zaro qarama-qarshi sonlar.
15. To'rt xonali sonning barcha raqamlari xar xil.

Javoblar:

- 1) $(a \% 2 == 0) \&\& (b \% 2 == 1);$ 3) $b * b - 4 * a * c == 0;$ 4) $(a == 0) \&\& (b != 0);$ 5) $x1 = x; x = y; y = x1;$
- 7) $\text{sqrt}(a * b) <= (a + b) / 2;$ 8) $a \% 10 == 3;$ 9) $a3 = n \% 10; a2 = n / 10 \% 10;$
- 10) $a2 + a3 == 9;$ 15) $(a \% 2 == 1 \&\& b \% 2 == 1) \&\& (x \% 2 == 1 || y \% 2 == 1)$
- 20) $(x == y) \&\& (x != z) || (x == z) \&\& (x != y) || (z == y) \&\& (x != z)$

Quyidagi ifodalarning qiymatlarini aniqlang

- a) $n = 35; !(n \% 2 == 1);$ b) $p = 78; (\text{int}(10 * p) >= 1000) || 19 - 3 > 3 * 4;$
- c) $x = 3; y = 2; a = \text{true}; b = \text{false}; !((x < 4) \&\& (y > -1)) || a \&\& b \&\& (x > y);$
- d) $(37 / 8 == 5) || \text{true} \&\& \text{fmod}(6, 3) == 1;$
- e) $x = 5; y = 2; !(x - y <= 3) \&\& !((x + y) != 7);$
- f) $a = 19; b = 8; !(a / 3 + b / 5 == 7) || \text{false};$
- g) $(82 \% 30 \% 8 == 129 \% 40 - 3) \&\& \text{true} || \text{false};$
- h) $\text{int } x, y, z, t; (10 * x + y - (x + y)) \% 9 == 0 \&\& \sin(\text{atan}(1) * 2) == 1;$

IV BOB. MURAKKAB BUYRUQLAR

Biz yuqorida tanishgan ko'pchilik dasturlar chiziqli dasturlar. Lekin amalda uchraydigan masalalarning juda ko'p qismini chiziqli dasturlar yordamida dasturlashtirib bo'lmaydi. Ya'ni berilgan shart bog'liq holda buyruqlar bajarilishining tabiiy tartibi buzilib, bajarilish qaysidir satrga uzatadi. Bunday holat tarmoqlanish, tanlash, takrorlanish(sikl) buyruqlarida ro'y beradi. O'z tarkibida tarmoqlanish, takrorlanish buyruqlarini saqlovchi dasturlar murakkab buyruqli dasturlar deyiladi.

Ushbu buyruqlarning har biri bilan tanishib chiqamiz.

4.1 Tarmoqlanish buyrug'i (If shartli o'tish operatori)

Operatorning umumiy ko'rinishi quyidagicha:

if (shart) buyruqlar ketma-ketligi.

Buyruq quyidagi tartibda bajariladi:

Agar shartning qiymati **true** (chin) yoki **false** (yolg'on) bo'lsa, buyruqlar ketma-ketligi bajariladi, aks holda, ya'ni shartning qiymati **false** (yolg'on) yoki **0** bo'lsa, if operatorining buyruqlari bajarilmasdan keying buyruqlarga o'tiladi. Buyruqlarning bajarilishi if operatoridan keyin turgan shartga bog'liq bo'lganligi uchun bu operator shartli o'tish operatori deyiladi.

Misol: Tanlangan sonning juft yoki toqligini aniqlash dasturini tuzing:

```
#include <iostream.h>
using namespace std;
int main()
{ int a; cout << "Tanlangan sonni kiriting\n";
  cin >> a;
  if(a%2 == 0) cout << "\n Bu son juft" << endl;
  if(a%2 != 0) cout << "\n Bu son toq" << endl;
  system("pause");
  return 0;
}
```

Bu dasturda if operatoriga taalluqli buyruq bitta. Agar buyruqlar soni ikki yoki undan ortiq bo'lsa, ular bitta blokni tashkil qiladi va blok "va" belgilari orasiga olib yoziladi hamda kompilyator tomonidan bitta, yaxlit operator deb qaraladi. Agar buyruqlar soni ikki yoki undan ortiq bo'lib, blok belgisi orasiga olinmagan bo'lsa, kompilyator ulardan

birinчисini if operatoriga taalluqli deb tushunadi va kutilmagan natijani kiritish sodir bo'ladi.

Agar blok ichida o'zgaruvchi va o'zgarmaslarni ham e'lon qilish kerak bo'lsa, Lekin bu o'zgaruvchi va o'zgarmaslardan faqat blok ichida foydalanish mumkin, ammo blokdan tashqarida foydalanib bo'lmaydi.

Blok ichidagi har bir buyruq (;) bilan yakunlanishi shart, ammo blokdan keyin (;) qo'yilmasligi ham mumkin.

Misol: Tanlangan to'rt xonali son musbat bo'lsa, uning raqamlari yig'indisini, manfiy bo'lsa, raqamlari ko'paytmasini topish dasturini tuzing.

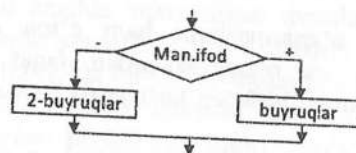
```
#include <iostream.h>
#include <math.h>
using namespace std;
int main()
{
    int x, a; int const d=36;
    cout << "\n Tanlangan sonni kiriting" << endl;
    cin >> x;
    if(x > 0){ int a, b, c, d; a=x/1000; b=x%1000/100;
               c=x/10%10; d=x%10;
               cout << "\n Bu son musbat bo'lganligi uchun raqamlari yig'indisi=" << a+b+c+d << endl;}
    if(x < 0){ int x1, y, z, t; x1=x/1000; y=x%1000/100;
               z=x/10%10; t=x%10;
               cout << "\n Bu son manfiy bo'lganligi uchun raqamlari ko'paytmasi=" << x1*y*z*t << endl;}
    system("pause");
    return 0;
}
```

Agar 2345 soni kiritilsa, dastur natijasi quyidagicha bo'ladi:
Bu son musbat bo'lganligi uchun raqamlari yig'indisi = 14 kabi bo'ladi.

Tarmoqlanish buyrug'ining if-else ko'rinishi

Berilgan shart (mantiqiy ifoda) ning qiymati **true** (rost) yoki **false** (yolg'on) ligiga bog'liq holda u yoki bu buyruqlar ketma-ketligini bajarish zarur bo'lgan holda tarmoqlanish buyrug'ining if-else ko'rinishidan foydalaniladi.

Tarmoqlanish algoritmining blok sxema ko'rinishi va bajarilish tartibi bizga algoritmlash amaliyoti bobi-dan ma'lum.



Tarmoqlanish buyrug'ini C++ dasturlash tilidagi unumdorlik ko'rinishi quyidagicha:

if(mantiqiy ifoda) {1-buyruqlar}; else {2-buyruqlar};

Buyruq quyidagi tartibda bajariladi:

Dastlab mantiqiy ifodaning qiymati tekshiriladi. Agar u chin (true) yoki 0 dan farqli bo'lsa, birinchi buyruqlar bajarilib, ikkinchi buyruqlar bajarilmasdan tarmoqlanish buyrug'idan keyingi buyruqlar o'tiladi. Agar mantiqiy ifodaning qiymati yolg'on (false) yoki 0 bo'lsa, ikkinchi buyruqlar bajarilib, birinchi buyruqlar bajarilmasdan tarmoqlanish buyrug'idan keyingi buyruqlarga o'tiladi.

1-misol. $y = \begin{cases} \sqrt{2x-5} + 0.4x - 29, & x \geq 2.5 \\ (8x-4)(x+2), & x < 2.5 \end{cases}$ funksiyaning qiymatini hisoblash dasturini tuzing.

```

#include <iostream.h>
#include <math.h>
using namespace std;
int main()
{ float x, y, z, t, p, q;
  cout << "x ning qiymatini kiriting\n";
  cin >> x;
  if (x >= 2.5) {z=sqrt(2*x-5); t=0.4*x-29; y=z+t;} else {p=8*x-4; q=x+2; y=p*q;}
  cout << "\n y =" << y << endl;
  system("pause");
  return 0;
}
  
```

Ushbu dasturda tarmoqlanish buyrug'ini if (x>=2.5) y=sqrt(2*x-5)+0.4*x-29; else y=(8*x-4)*(x+2);

if sharti bilan tashkil qilish ham mumkin edi. Bitta buyruqni mayda bir buyruqlarga bo'lishdan maqsad, murakkab buyruqlar tarkibida blok tarmoqlanishdan foydalanishni namoyish qilishdir.

Misol. if shartli o'tish operatorining ichma-ich joylashgan buyruqlaridan ham foydalaniladi. Bu hol bilan quyidagi funksiyaning qiymatini hisoblash dasturi misolida tanishamiz.

$y = \begin{cases} \sqrt{x-1}, & x \geq 1 \\ 2x+9, & -1 \leq x < 1 \\ x^2+19, & x < -1 \end{cases}$ funksiya qiymatini hisoblash dasturini tuzing.

```

#include <iostream.h>
#include <math.h>
using namespace std;
int main()
{ float x, y; cin >> x;
  if (x >= 1) y=sqrt(x-1); else if (x >=-1 && x < 1) y=2*x+9; else
  y=x^2+19;
  cout << "y=" << y << "\n";
  system("pause");
  return 0;
}
  
```

if shartli o'tish operatorida o'zgaruvchilarni e'lon qilib bo'lmaydi. Ammo bloklarning ichida e'lon qilish mumkin, lekin oldin qayd qilganimizdek, bunday o'zgaruvchilardan blokdan tashqarida foydalanib bo'lmaydi. Bu holat bilan quyidagi dastur orqali tanishib, natijalarni tahlil qiling.

```

#include <iostream.h>
using namespace std;
int main()
{
  int x=2, y, i=2, j=4;
  if (x >= 3) { int i, j; i=5; j=8; y=i+j; }
  y=i+j;
  cout << "\n y =" << y << "\n";
  system("pause");
  return 0;
}
  
```

Dastur natijasi $y=6$ kabi bo'ladi. Bir qarashda dasturni $x \geq 3$ mantiqiy ifodaning qiymati yolg'on bo'lganligi uchun shunday chiqqandek tuyuladi. Ammo 5 - satrni `int x=5, y, i;` kabi o'zgartirsak ham natija o'zgarmaydi. Chunki i, j lar ichidagi qiymatlari blokdan tashqarida unutiladi.

Shartsiz o'tish operatori

Shartsiz o'tish operatorining umumiy ko'rinishi quyidagicha:

`goto` belgi;

Bunda `goto` kalit so'zining ma'nosi "ga o'tish" demakdir. Belg ixtiyoriy identifikator. Belgi o'rinda son va raqamlardan foydalanib bo'lmaydi.

C++ `goto` operatorini uchratgach, undan keyingi belgi dastur qaysi satrida ko'rsatilgan bo'lsa, dastur bajarilishini o'sha satrdan davom ettiradi.

Misol: 1 dan n gacha bo'lgan sonlar yig'indisini topish dasturini tuzing.

```
#include <iostream.h>
using namespace std;
int main()
{ int x, n, s;
  x=1; s=0; cin >> n;
  sum: s+=x; x++;
  if (x<=n) goto sum;
  cout << "s=" << s << "\n";
  system("pause");
  return 0;
}
```

Dasturni ijro etish jarayonida n ga 6 qiymati kiritilsa, javob $s=21$ kabi bo'ladi.

Demak, `goto` operatori yordamida dasturning istalgan satridan oldingi yoki keyingi satrlaridan biriga sakrab o'tib, dastur ijrosini davom ettirish mumkin. Bu barcha dasturlash tillarida bo'lgan yaxshi imkoniyat bo'lishi bilan birgalikda quyidagicha muammolarni ham keltirib chiqaradi:

1) dastur tarkibida `goto` operatoridan ko'p foydalanilganda dasturni yoki dastur ijrosini tushunish qiyinlashadi. Shuningdek,

dasturning qanday masalani hal qilishini ma'lum vaqt o'tgach dasturning o'zi ham aniqlashi uchun ancha vaqt talab etilishi

tarmoqlanish, takrorlanish buyruqlarining ichidan sakrab chiqish va aksincha tashqaridan buyruq ichiga kirish holatlariga olib kelishi mumkin. Shuning uchun `goto` operatoridan mumkin qadar kam foydalanishga harakat qilishadi.

?: amali

Dasturning umumiy ko'rinishi quyidagicha:

mantiqiy ifoda? 1-buyruq:2-buyruq;

Buyruqlarning bajarilish tartibi tarmoqlanish buyrug'ining bajarilish tartibiga o'xshaydi. Ya'ni mantiqiy ifodaning qiymati chin yoki 0 bo'lsa, birinchi buyruq, yolg'on (false) yoki 0 bo'lsa, ikkinchi buyruq bajariladi. Tarmoqlanish buyrug'idan farqi shundaki, amaliyotda biror o'zgaruvchiga qiymat sifatida berilishi mumkin. Birinchi va ikkinchi buyruqlar o'rinda buyruqlar soni ikki yoki undan ortiq bo'la olmaydi.

Misol. Berilgan sonning juft yoki toqligini aniqlash dasturini tuzing:

```
#include <iostream.h>
using namespace std;
int main()
{ int x, y; string a,b,s;
  cout << "Bitta son kiriting\n";
  cin >> x;
  s=x%2 ? "Bu son toq": "Bu son juft";
  cout << s << "\n";
  system("pause");
  return 0;
}
```

Dastur bajarilish jarayonida x ga juft son kiritilsa, $x\%2$ ifodaning qiymati 0 bo'ladi va nooshkor holda $s="Bu son juft";$ buyrug'i bajariladi, chunki mantiqiy ifoda o'rinda kelayotgan ifodaning qiymati 0 ga teng. Agar x ga toq son kiritilsa, $x\%2$ ifodaning qiymati 1 ga teng bo'ladi va demak, mantiqiy ifodaning qiymati rost bolib, nooshkor holda $s="Bu son toq";$ buyrug'i bajariladi.

Bu dastur quyidagi variantlarda ham ishlaydi:

```
s = x % 2 ? a = "Bu son toq": b = "Bu son juft";  
s = ! (x % 2) ? a = "Bu son toq": b = "Bu son juft";
```

Bunda a, b, s o'zgaruvchilar turi **string** ekanligiga e'tibor beriladi.
Bu amal quyidagi hollarda ham ishlaydi:

```
x=3; y=8;      x>=y?      n=x+y: k=x-y;  
x=5; y=15;     x<=y?      n=x-y: 16;  
x=12; y=10     x!=y?      16:13;
```

Eslatma: birinchi va ikkinchi buyruqlar oralig'ida qo'yilayotganligiga e'tibor bering.

break operatori

Ba'zan dastur yoki qaysidir buyruqlar to'liq ijro etilmasdan qo'qqisidan ishni to'xtatib (avariya holati) dastur yoki buyruqdan chiqib ketishga to'g'ri keladi. Ayniqsa, bunday holat takrorlanish va tanlash buyruqlarini tashkil qilishda ko'p uchraydi.

Bunday vaziyatlarda **break** (uzish) operatoridan foydalaniladi. **break** operatorini uchratgach, dasturni bajarishni to'xtatadi va **break** operatori joylashgan blokdan chiqib, undan keyingi buyruqlar bajarishga o'tadi. Demak, unutmaslik kerakki, **break** operatori faqat o'zi joylashgan blokdan chiqib ketish imkonini beradi. Lekin tanlash sikl buyruqlari ichida **break** operatoridan foydalanilsa, o'zgaruvchi buyruqning o'zini blok deb tushunadi.

Bu operatoridan zarurat bo'lganda dasturning ixtiyoriy joyida foydalanish mumkin. Ammo chuqur o'ylamasdan foydalanmasdan tavsiya etiladi. Masalan, **int main()** funksiyasi blokida foydalanilgan **break;** operatoridan keyingi buyruqlar bajarilmaydi. Bu holda **break** operatori **int main()** funksiyasi blokidan chiqishni ta'minlaydi va natijada dasturni bajarish to'xtatiladi. Keyingi mavzularda aniq misollar yordamida bu operator bilan yanada yaqinroq tanishamiz.

continue operatori

continue (qolish) operatorining ishlash jarayoni ham **break** operatoriga o'xshaydi. Ammo u **break** operatoriga o'xshab blokdan chiqib ketishni ta'minlamaydi, balki faqat takrorlanish tanasini bajarilishini to'xtatib, keyingi qadamga o'tishni ta'minlaydi xolos, ya'ni sikl parametrining keyingi qadamlari uchun hisoblash ishlarini ta'minlaydi.

1 dan 10 gacha bo'lgan sonlar ichida 9 va 5 sonlaridan iborat sonlar yig'indisini hisoblash dasturini tuzing.

```
#include <iostream.h>  
using namespace std;  
int main()  
{  
    int x,y,s=0;  
    for(x=1;x<=10;x++) {if((x==9)||(x==5)) continue;s+=x;};  
    cout<<"s="<<s<<"\n";  
    system("pause");  
    return 0;  
}
```

Dastur natijasi **s=41** kabi bo'ladi. Chunki sikl parametrining 9 va 5 sonlar qiymatlari **continue** operatori yordamida o'tkazib yuboriladi va 1 dan 10 gacha bo'lgan boshqa sonlar yig'indisi hisoblanadi.

Agar ushbu dasturda **{if((x==9)||(x==5)) continue;s+=x;};** blokni **{if(x==5) break;s+=x;};** kabi o'zgartirsak, natija **s=15** kabi bo'ladi. Chunki **x=6** bo'lganda **break** operatori sikl tanasining bajarilish jarayonini to'xtatadi.

break va **continue** operatorlaridan tarmoqlanish buyrug'ining blokida foydalanib b'lmaydi. Masalan,
if(x>=6) {x=x+a; beak;}; else {x=x-a; continue;}; kabi buyruqlarni kompilyator xato deb hisoblaydi. Ammo yuqorida keltirilganidek, **if(x==6) beak; else continue;** kabi buyruqlarga ruxsat etiladi.

Tanlash buyruqlari

Agar masala mazmunida shartlar soni ko'p bo'lsa, ichma-ich joylashgan tarmoqlanish buyrug'idan foydalanish mumkin. Ammo dastur matnini tushunish qiyin bo'ladi. Bunday hollar uchun dasturlash tillarida, jumladan C++ dasturlash tilida ham tanlash buyruqlaridan foydalanish maqsadga muvofiq.

Masalan, hafta kunlari tartib raqami berilganda nomini aniqlash, o'quvchilar bilimini baholash me'zonini ishlab chiqish, berilgan son kvadrati yoki kubi qanday raqam bilan tugashini aniqlash, trigonometrik funksiyalarning koordinata choraklaridagi ishorasini aniqlash, o'quvchi yoshini bilgan holda qaysi sinfda o'qishini aniqlash

masalasi kabi ko'p sonli masalalar dasturlarini tuzishda tanlash buyruqlaridan foydalanish katta qulayliklar yaratadi.

Quyida tanlash buyrug'ining ikki xil ko'rinishi bilan tanishish. Qaysi ko'rinishidan qachon foydalanishni masala xarakteridan kelib chiqqan holda dasturchi o'zi tanlaydi.

Tanlash buyrug'ini **if** va **goto** operatorlari yordamida tashkil qilish.

Misol. Hafta kunlari tartib raqami bo'yicha nomini aniqlash dasturini tuzing.

```
#include <iostream.h>
using namespace std;
int main()
{ int x; string k;
  cout << "hafta kunining tartib raqamini kiriting ";
  cin >> x;
  if (x == 1) {k = "Dushanba\n"; goto tam;}
  if (x == 2) {k = "Seshanba\n"; goto tam;}
  if (x == 3) {k = "Chorshanba\n"; goto tam;}
  if (x == 4) {k = "Payshanba\n"; goto tam;}
  if (x == 5) {k = "Juma\n"; goto tam;}
  if (x == 6) {k = "Shanba\n"; goto tam;}
  if (x == 7) {k = "Yakshanba\n"; goto tam;}
  k = "Haftaning bunday nomerli kuni mavjud emas\n";
  tam: cout << k << endl;
  system("pause");
  return 0;
}
```

Dastur tarkibida foydalanilgan operatorlar va ularning bajarilish tartibi oldingi mavzulardan tanish bo'lganligi tufayli qo'shimcha tushuntirishlar berilmadi.

Faqat quyidagi savollarga javob bering:

1. Har bir shartli o'tish operatorida blok belgisidan nima uchun foydalanildi?

2. **goto tam;** buyruqlari olib tashlansa, qanday natijalar chop etiladi?

Tanlash buyrug'ini **switch** operatori yordamida tashkil qilish

switch (kalit) operatori yordamida tashkil qilinadigan tanlash buyrug'ining umumiy ko'rinishi quyidagicha:

```
switch (x)
{
  case x1: birinchi buyruqlar; break;
  case x2: ikkinchi buyruqlar; break;
  case x3: uchinchi buyruqlar; break;
  ...
  case xn: n chi buyruqlar; break;
  default: buyruqlar;
}
```

Hunda **switch**, **case** (hol, holda), **default** (aks holda) so'zlari tanlash buyrug'ini tashkil qilishda foydalanilgan yordamchi so'zlar (operatorlar).

x – qiymati o'zgarmas bo'lgan ifoda yoki o'zgarmas son. Buyruq quyidagi tartibda bajariladi:

Dastlab **x** ning qiymati aniqlanadi va bu qiymat **x1, x2, x3, ... xn** lar bilan solishtiriladi. Ulardan qaysi biri bilan aynan bir xil bo'lsa, o'zlanishiga mos (:)dan keyingi buyruqlar bajarilib, boshqalari bilan solishtirilmasdan tanlash buyrug'idan keyingi buyruqlarga o'tiladi. Chunki har bir buyruqlarning oxirida **break;** operatori turibdi va u yuqorida qayd qilganimizdek, dastur bajarilishini **switch** operatoriga taalluqli blokdan chiqarib yuboradi.

Agar **x** ning qiymati **x1, x2, x3, ... xn** larning hech biri bilan bir xil bo'lmasa **default** operatoridan keyingi buyruqlar bajarilib, tanlash buyrug'ini bajarish tugaydi.

Aslida, **default** operatori albatta bo'lishi shart bo'lmagan operatoridir. Agar u buyruq tarkibida bo'lmasa va **x** ning qiymati **x1, x2, x3, ..., xn** lar bilan bir xil bo'lmasa, tanlash buyrug'i tarkibidagi birorta ham buyruq bajarilmasdan keyingi buyruqlarga o'tiladi.

Misol. Yil oylarining qaysi fasllarga tegishligini aniqlash dasturini tuzing.

```
#include <iostream.h>
using namespace std;
int main()
```



```

{ int n; cout << "oy nomerini kirting \n\n";
cin >> n;
switch (n)
{
case 1:
case 2:
case 12: cout << "Bu qish oyi \n\n"; break;
case 3:
case 4:
case 5: cout << "Bu bahor oyi \n\n"; break;
case 6:
case 7:
case 8: cout << "Bu yoz oyi \n\n"; break;
case 9:
case 10:
case 11: cout << "Bu kuz oyi \n\n"; break;
default: cout << "Yilning bunday raqamli oyi mavjud emas \n\n";
}
system("pause");
return 0;
}

```

Dasturdan ko'rinadiki, agar bir nechta **case** operatorlariga mos buyruqlar bo'lmasa, birinchi uchragan buyruq oldingilariga ham taalluqli bo'ladi.

Dasturlash asoslarini o'rganishda quyidagilarga e'tibor bering

Dasturlashda "qo'pol xato" yoki "arzimasa xato" degan xatolar yo'q ekanligini unutmang. Faqat xato bor, xolos. Faraz qilaylik, tarmoqlanish yoki tanlash buyrug'i noto'g'ri tashkil qilingan bo'lsa, bitta operatorning biror harfi tushib qolishi yoki boshqa harf bilan almashtirilishi (masalan, **include** yoki **includ**) yoki bitta tinish belgisining o'z o'rnida qo'yilmasligi kabi xatolar dasturning ijro etilmasligiga olib kelsa, bu xatolarni ahamiyati jihatidan yuqori yoki quyi darajadagi xatolarga bo'lish mumkinmi?

Ko'pincha dasturlashni boshlab o'rganayotgan boshlovchilar dasturda bir nechta xatolarga yo'l qo'yib, ularni topmasdan soatlab vaqt sarflashadi. Yanada aniqroq aytadigan bo'lsak, biror buyruq yoki mavzuni o'rganishga ketadigan vaqtdan unga oid dasturni kompyuterga kiritib, javobini olishga ketadigan vaqt bir necha marta ko'p bo'ladi.

Bunday kamchiliklarni bartaraf etish uchun quyidagilarga e'tibor qiling:

- har bir kalit so'zni aniq aytish va yozishda xatoga yo'l qo'ymaguncha mashq qiling;
- tinish belgilarining qayerdan va qanday qo'yilishiga e'tibor

qiling;

- figurali (sistemali), oddiy qavslarni, apostrof va qo'shtirnoqni ochganda darhol yopib, kursorni bitta chapga qaytarib, keyin ma'lumotlarni tersangiz u yopilmasdan qolmaydi;

- xatolik haqida C++ dasturlash tilining ma'lumotlari bo'yicha to'g'ri harakat qilishni tezroq o'rganing;

- dastlab biror masalaga mustaqil dastur tuzmasdan tuzilgan dasturni o'rganib chiqib, qaytadan shu dasturni mustaqil tuzish yo'li bilan o'z tajribangizni oshirib boring;

- tajribali hamkasblaringiz hamda o'rtoqlaringizdan oddiy narsalarni ham so'rashdan uyalmang.

Bularning hammasi vaqtingizning sezilarli darajada tejallishiga olib keladi.

Masalan, tanlash buyrug'ini o'rganishda nimalarga e'tibor berish kerak?

- **switch**, **case**, **default** operatorlarini to'g'ri yozish;
- **switch ()** operatoridan keyin tinish belgisi qo'yilmasligiga;
- **switch** operatori o'z blokiga egaligiga;
- **case** so'zidan keyin yoziladigan o'zgarma bilan buyruqlar oralig'ida va **default** bilan buyruqlar oralig'ida (:) borligiga;
- (:) dan keyin buyruqlar ko'p bo'lsa ham ular blok belgi orasida yozilishi shart emasligiga;
- (:) dan keying buyruqlar, albatta, **break**; operatori bilan tugashiga (**default** bundan mustasno).

Quyidagi savollarga javob bering

1. C++ dasturlash tilida tarmoqlanish buyrug'ining umumiy ko'rinishlarini yozing va bajarilish tartibini tushuntiring.
3. Tarmoqlanish buyrug'ida qachon qavs operatorlaridan foydalaniladi?
4. **sizeof** amalining vazifasini tushuntiring.
5. $n = \text{sizeof}(s)$, $m = \text{sizeof}(\text{bool})$, $k = \text{sizeof}(\text{float})$; buyruqlari nimani hisoblaydi?

6. Shartsiz o'tish operatorining umumiy ko'rinishini yozing va bajarilish tartibini tushuntiring.

7. Shartsiz o'tish operatorida foydalaniladigan belgilar qanday kattaliklar bo'ladi?

8. ?: operatorini va uning bajarilish tartibini tushuntiring.

9. Tanlash buyruqlaridan qachon foydalaniladi?

10. **break** va **continue** operatorlari qanday ijro etilishini tushuntiring.

11. Tanlash buyrug'ining qanday ko'rinishlari mavjud?

12. Tanlash buyrug'ining **case n of** ko'rinishini yozing va bajarilish tartibini tushuntiring.

13. Tanlash buyrug'ining shartli o'tish operatori yordamida tashkil qilingan ko'rinishini yozib, bajarilish tartibini tushuntiring.

14. Tanlash buyrug'ining **case n of** ko'rinishi qanday operator bilan yakunlanadi?

15. Tarmoqlanish, tanlash buyruqlari tarkibida o'zgaruvchilarni e'lon qilish mumkinmi, uning dastur boshida e'lon qilinishidan nima farqi bor?

16. Tarmoqlanish buyruqlaridan **if** va **if-else** ko'rinishlarining bajarilish tartibidagi farqlarni tushuntiring.

17. Ichma-ich joylashgan Tarmoqlanish buyruqlarining umumiy ko'rinishini yozib, bajarilish tartibini tushuntiring.

Quyidagi masalalarning dasturlarini tuzing

1. x, y, z, t sonlarining eng kattasi va eng kichigini aniqlang.

2. $M(x, y)$ nuqta koordinata tekisligining qayerida joylashganligini aniqlang.

3. Berilgan oddiy kasrning turini aniqlang.

4. x_1, x_2, x_3 sonlari berilgan va ularning faqat ikkitasi teng. Farq qiluvchi sonni toping.

5. $y=kx+b$ funksiya grafigining qaysi koordinata choragida joylashganligini aniqlang.

6. Ikki chiziqli funksiya berilgan. Ularning grafiklari o'zaro qanday joylashganligini aniqlang.

7. Teledasturning bir qismini dasturlashtiring.

8. Har qanday sonning kvadrati qanday raqam bilan tugashini aniqlang.

9. $y=\sin x$, $y=\cos x$, $y=\tan x$, $y=\cot x$ funksiyalarning koordinata choraklaridagi ishoralarini aniqlang.

10. Yil oylarining qaysi faslga tegishlilikini aniqlang.

11. Ixtiyoriy kiritilgan ko'p xonali sonning 2, 3, 4, 5, 6, 10 sonlaridan qaysi biriga karrali ekanligini aniqlang.

12. 37 dan 349 gacha bo'lgan sonlar ichida 23 ga karrali sonlarning soni, yig'indisini topish dasturini **goto** operatoridan foydalanib tuzing.

13. **kl!** ni hisoblash dasturini **goto** operatoridan foydalanib tuzing.

14. Uchburchak uchlarining koordinatalari berilgan. Uning turini aniqlang.

15. O'lchami bilan berilgan burchakning turini aniqlang.

16. O'quvchilarga 20 ta test topshirig'i beriladi. O'quvchilarni besh balli baholash tizimida baholash dasturini tuzing.

17. 40 ta son tanlanadi. Ulardan nechitasi juft, nechitasi toq, nechitasi musbat va nechitasi mahfiy ekanligini aniqlang.

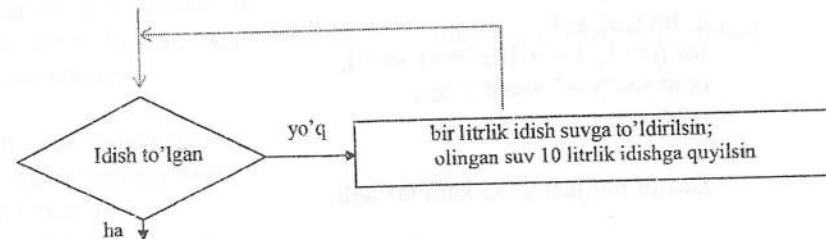
18. Tekislikda 4 ta nuqta koordinatalari bilan berilgan bo'lsa, qaysi ikki nuqta orasidagi masofa eng katta ekanligini aniqlang.

19. Ikki xonali sonlar ketma-ket yozib chiqildi. n -o'rindagi raqamni toping.

4.2 Takrorlanish(sikl) buyruqlari

Amalda shunday masalalar ham uchraydiki, ularni hal qilish dasturlarini hozirgacha tanishgan buyruqlar yordamida tuzib bo'lmaydi. Ular shunday masalalarki, natijaga erishish uchun qo'yilgan shartga bog'liq holda bir yoki bir nechta amal yoki buyruqlar ketma-ketligini takror-takror bajarishga to'g'ri keladi.

Masalan, bir litrlik idish bilan, o'n litrlik idishni to'ldirish algoritmi quyidagicha bo'ladi:



Ko'rinib turibdiki, arifmetik blokda ikkita amal(buyruq) 10 marta takror-takror bajarilgach maqsadga erishiladi. Yoki g'isht teruvchi bir nechta amallar ketma-ketligini talab etilgan balandlikdagi devor hosil bo'lguncha o'z ishini n marta takrorlaydi. Takrorlanish buyruqlaridan, ayniqsa, aniq fanlarga doir masalalarni hal qilishda ko'p foydalaniladi. Bunga keyingi mavzular bilan tanishish jarayonida ishonch hosil qilish mumkin. Quyida takrorlanish buyruqlarining uch xil ko'rinishi bilan tanishamiz.

Takrorlanish buyrug'ini *for* operatori yordamida tashkil qilish

Buyruqning umumiy ko'rinishi quyidagicha:

for ($i = i_{\min}; i \leq i_{\max}; i = i + i_{\text{qad}}$) **buyruqlar**;

Bunda **for** (uchun) – takrorlanishni tashkil qilishda foydalanilgan kalit so'z, i – sikl parametri, $i_{\min}$ – sikl parametrining boshlang'ich qiymati, $i_{\max}$ – sikl parametrining oxirgi qiymati, i_{qad} – har takrorlanishda sikl parametri o'zgaradigan qadam, buyruqlar o'rnida operator yoki operatorlar bloki bo'ladi.

Takrorlanish buyrug'i quyidagi tartibda bajariladi:

Sikl parametri o'zining dastlabki $i_{\min}$ qiymatini oladi va oxirgi $i_{\max}$ qiymati bilan solishtiradi. Agar $i \leq i_{\max}$ mantiqiy ifodaning qiymati **true** yoki **0** dan farqli bo'lsa buyruqlar bir marta bajarilib, sikl parametri bir qadamga ortadi va yana $i \leq i_{\max}$ shart tekshiriladi va hakazo bu jarayon $i > i_{\max}$ bo'lguncha davom etadi.

Misol.1 dan 10 gacha bo'lgan natural sonlar yig'indisini topish dasturini tuzing.

```
#include <iostream.h>
using namespace std;
int main()
{ int i, s; s=0;
  for (i = 1; i <= 10; i++) s+= i;
  cout << "s =" << s << "\n";
  return 0;
}
```

Dastur natijasi $s=55$ kabi bo'ladi.

Sikl parametrining qadamini sun'iy yo'l bilan 1 dan boshqa ixtiyoriy son qilish mumkin. Masalan, qadamni 2 qilish uchun quyidagi uchta usullarning biridan foydalaning:

- 1) for ($i = 1; i \leq 10; i++, i++$) $s+=i$;
- 2) for ($i = 1; i \leq 10; i=i+2$) $s+=i$;
- 3) for ($i = 1; i \leq 10; i++$) { $s+=i; i++$ };

for sikl buyrug'i sikl parametrining manfiy yoki kasr qiymatlarida ham bajariladi. Masalan, yuqoridagi dasturda

```
float i, s; s=0;
for (i = 0; i >= -10; i = i - 0.5) s+=i;
kabi o'zgarish qilinsa, natija  $s = -105$  kabi bo'ladi.
```

for sikl buyrug'idagi $i = 1; i \leq n; i++$; kabi ifodalar ikki yoki undan ortiq ifodalardan tashkil topishi mumkin. Bu holda bir-birlaridan vergul (,) bilan ajratiladi. Masalan, yuqoridagi dasturda:

```
for (i = 1, j = 1; i <= 10, j <= 10; i++, j++) s+= i+j;
kabi o'zgarish qilinsa, natija  $s=110$  kabi bo'ladi.
```

Sikldagi $i=1$ ifoda boshida faqat bir marta bajariladi. Sikl ichida o'zgaruvchi e'loni ham bo'lishi mumkin. Ammo bunday o'zgaruvchilardan sikl tanasidan tashqarida foydalanib bo'lmaydi.

Masalan, for ($int\ i=1; i \leq n; i=i+3$) kabi holatga ruxsat etiladi.

for sikl buyrug'ining parametrlar qismi bo'sh bo'lishi ham mumkin, lekin ikkita (;) bo'lishi shart.

Masalan, for(;;) $s = s + i$; buyrug'i bajariladi, ammo cheksiz takrorlanish ro'y beradi.

Eslatma: Cheksiz takrorlanishni bekor qilish uchun **выполнить** → **сброс программы** algoritmi bo'yicha harakatlaning yoki Alt+F2 tugmachalar kombinatsiyasidan foydalaning.

Shuningdek, tanasiz sikldan ham foydalanish mumkin.

Masalan,
for ($i = 1; i \leq 10; s += ++i$);
cout << "s =" << s << "\n";

Misol.2 $y = 5x - 4a$, $x \in [-3; 3]$, $\Delta x = 2$; funksiya qiymatini hisoblash dasturini tuzing:

```
#include <iostream.h>
using namespace std;
int main()
{ float x, y, a, z; cin >> a;
```

```
for (x = -3; x <= 3; x = x + 2) { y=5*x-4*a;
cout << "y("<<x<<")=" << y << "\n";}
system("pause");
return 0;
}
```

Agar a ga 5 qiymat kiritilsa, natija quyidagi ko'rinishda chop etiladi:

```
y(-3)=-35
y(-1)=-25
y(1)=-15
y(3)=-5
```

Sikl buyrug'ining bu ko'rinishi oldshartli sikl buyrug'i yoki sharti oldin kelgan sikl buyrug'i deyiladi. Chunki oldin shart tekshirilib, keyin buyruqlar bajariladi. Demak, agar birinchi qadamdayoq shartning qiymati yolg'on bo'lsa, takrorlanish buyrug'i biror marta ham bajarilmasdan dasturning keyingi buyruqlariga o'tiladi.

Takrorlanish buyrug'ini *while* kalit so'zi yordamida tashkil qilish

Yana bitta oldshartli sikl buyrug'i bilan tanishamiz. Buyruqning umumiy ko'rinishi quyidagicha:

```
i=imin; while (i<=imax) {buyruqlar; i=i+iqad};
```

Bunda *while*(toki) so'zi buyruqni tashkil qilishda foydalanilgan yordamchi (kalit) so'z. i – sikl parametri.

Buyruq quyidagi tartibda bajariladi:

Sikl parametri o'zining dastlabki i_{min} qiymatini oladi va oxirgi i_{max} qiymati bilan solishtiriladi. Agar $i \leq i_{max}$ mantiqiy ifodaning qiymati *true* yoki 0 dan farqli bo'lsa, buyruqlar bir marta bajarilib, sikl parametri bir qadamga ortadi va yana $i \leq i_{max}$ mantiqiy ifodaning qiymati tekshiriladi. Agar $i \leq i_{max}$ mantiqiy ifodaning qiymati *true* bo'lsa, buyruqlar yana bir marta bajarilib, sikl parametrining qiymati bir qadamga ortadi va hokazo bu jarayon toki $i \leq i_{max}$ mantiqiy ifodaning qiymati 0 yoki *false* bo'lguncha davom etadi.

Tabiiyki, mantiqiy ifoda o'rnida o'zgarmas son bo'lsa yoki sikl parametrining qiymati buyruqlar bajarilgandan keyin o'zgarmasa cheksiz takrorlanish sodir bo'ladi.

Misol. 1 dan 20 gacha bo'lgan natural sonlar yig'indisini topish dasturini tuzing.

```
#include <iostream.h>
using namespace std;
int main()
{ int i, s, n; s=0; i=1;
while (i<=20) s+= i++;
cout << "s=" << s << "\n";
system("pause");
return 0;
}
```

javob s=210.

while so'zidan keyin qavs ichida qiymat berish buyruqlari ham bo'lishi mumkin va bu buyruqlar mantiqiy ifodadan vergul (,) belgisi bilan ajratiladi hamda ushbu qiymat berish buyrug'i bajarilib, keyin mantiqiy ifodaning qiymati tekshiriladi. Faqat qavs ichidagi oxirgi ifoda mantiqiy ifoda bo'lishi kerak.

Sikl buyrug'i quyidagi ko'rinishda bo'lsa ham ishlaydi.

```
While (s+= i++, i<=10);
```

Qiymat berish buyrug'i ikki yoki undan ortiq bo'lishi ham mumkin.

Masalan, quyidagicha o'zgarish qilinsa juft sonlar yig'indisi hisoblanadi.

```
while (i+=1, s+=i++, i<=10);
```

Agar sikl buyrug'ida buyruqlar yoki buyruqlar bloki bo'lmasa ham (;) bo'lishi shart, aks holda keyingi buyruqni sikl tanasi deb tushunadi. Masalan, oxirgi buyruqda (;)bo'lmasa, ya'ni *while* (i+=1, s+=i++, i<=10) kabi bo'lsa dastur natijasi quyidagicha bo'ladi:

```
s = 2
s = 6
s = 12
s = 10
```

Demak, *while* sikl buyrug'ida qavsdan keyin (;) qo'yish buyruqlar blokining bo'shligini bildiradi.

Misol. Tanlangan natural sonning raqamlari ko'paytmasini topish dasturini tuzing.

```
#include <iostream.h>
```



```
using namespace std;
int main()
{ int a, a1, p; cout << "a sonini kiriting " << '\n';
  cin >> a;
  p=1; while(a1=a%10, a=a/10, p=p*a1, a);
  cout << "sonning raqamlari ko'paytmasining " << p << '\n';
  system("pause");
  return 0;
}
```

Ushbu misolda takrorlanish buyrug'ini quyidagicha ham tashkil qilish mumkin edi.

```
p=1; while (a) {a1 = a%10, a=a/10; p=p*a1};
```

bunda takrorlanish buyrug'ining bajarilish tartibi bo'yicha buyruqlar har bajarilganda tanlangan sonning o'ng tomonidan bitta raqami qirqib olinib, p ga ko'paytirilib borilaveradi, toki a=0 bo'lguncha.

Takrorlanish buyrug'ini *do-while* operatorlari yordamida tashkil qilish

do-while operatorining *while* operatoridan farqi shundaki, dastlab buyruq yoki buyruqlar bloki bajarilib, keyin shart tekshiriladi. Shuning uchun bu operator so'ng shartli sikl operatori yoki sharti keyin qo'yilgan sikl operatori deb ham nomlanadi. Demak, so'ng shartli sikl buyrug'idagi buyruqlar kamida bir marta bajariladi.

Buyruqning umumiy ko'rinishi quyidagicha:

```
do { buyruqlar bloki; } while ( mantiqiy ifoda );
```

Dastlab *do* (bajarish) va *while*(toki) kalit so'zlari oralig'idagi buyruqlar to'liq bajarilib, so'ngra mantiqiy ifodaning qiymati tekshiriladi. Agar u *true* yoki 0 dan farqli bo'lsa, yana buyruqlar bloki bajariladi va hokazo bu jarayon mantiqiy ifodaning qiymati yolg'on bo'lguncha davom etadi.

Misol. 187 va 319 sonlari oralig'ida 13 ga karrali bo'lgan natural sonlarni topish dasturini tuzing.

```
#include <iostream.h>
using namespace std;
```

```
int main()
{ int x = 187;
  do { if (!(x%13)) cout << "Navbatdagi 13 ga karrali son =" << x << '\n'; x++; } while (x <= 319);
  system("pause");
  return 0;
}
```

Bu holda ham *while* so'zidan keyin qavs ichida qiymat berish buyruqlaridan foydalanish mumkin. Faqat buyruqlar bloki bo'sh bo'lib qolsa, u yerda (;) bo'lishi shart.

Misol.1 dan 100 gacha bo'lgan natural sonlar yig'indisini hisoblash dasturini tuzing.

```
#include <iostream.h>
using namespace std;
int main()
{ int x, s; s=0; x=1;
  do ;while (s+=x, x++, x<=100);
  cout << "s =" << s << '\n';
  system("pause");
  return 0;
}
```

Ichma-ich joylashgan sikllar

Barcha dasturlash tillarida ichma-ich joylashgan sikl buyruqlaridan foydalanish imkoniyati yechilishi mumkin bo'lgan masalalar ko'lamini sezilarli darajada kengaytiradi.

1. *for* operatori yordamida tashkil qilinadigan ichma-ich joylashgan sikl buyruqlarining umumiy ko'rinishi quyidagicha:

```
for (i = i_min; i <= i_max; i = i + i_qad)
  for (j = j_min; j <= j_max; j = j + j_qad)
    for (k = k_min; k <= k_max; k = k + k_qad) buyruqlar;
```

Sikllarning uchta ichma-ich joylashgan holini tanladik. Aslida ularning soni uchtdan ko'p bo'lishi ham mumkin edi.

Buyruqning bajarilish tartibi quyidagicha:

Tashqi sikl parametri o'zining boshlang'ich i_{min} qiymatini qabul qiladi va oxirgi i_{max} qiymati bilan solishtiradi. Agar $i \leq i_{max}$ mantiqiy

ifodaning qiymati chin bo'lsa, o'rta sikl parametri o'zining boshlang'ich $j_{\min}$ qiymatini oladi va oxirgi $j_{\max}$ qiymati bilan solishtiradi. Agar $j \leq j_{\max}$ mantiqiy ifodaning qiymati chin bo'lsa, ichki sikl parametri o'zining boshlang'ich $k_{\min}$ qiymatini oladi va oxirgi $k_{\max}$ qiymati bilan solishtiradi. Agar $k \leq k_{\max}$ mantiqiy ifodaning qiymati rost bo'lsa, buyruqlar bir marta bajarilib, ichki sikl parametri bir qadamga ortadi ($k = k + k_{\text{qad}}$) va yana $k \leq k_{\max}$ mantiqiy ifodaning qiymati tekshiriladi. Agar u rost bo'lsa, buyruq yana bir marta bajariladi va hokazo bu jarayon $k > k_{\max}$ bo'lguncha davom etadi. So'ngra o'rta sikl parametri bir qadamga ortadi ($j = j + j_{\text{qad}}$) va $j \leq j_{\max}$ shart tekshiriladi. Agar u rost bo'lsa, ichki sikldagi takrorlanishlar o'rta sikl parametrining yangi qiymati uchun to'liq qaytadan amalga oshiriladi va hokazo bu jarayon $j > j_{\max}$ bo'lguncha davom etadi. Shundan so'ng tashqi sikl parametri bir qadamga ortadi ($i = i + i_{\text{qad}}$) va $i \leq i_{\max}$ mantiqiy ifodaning qiymati tekshiriladi. Agar u rost bo'lsa, tashqi sikl parametrining yangi qiymati uchun o'rta va ichki sikllardagi takrorlanishlar to'liq qaytadan amalga oshiriladi va hokazo bu jarayon $i > i_{\max}$ bo'lguncha davom etadi.

Demak, ichma-ich joylashgan uchta siklning bajarilish tartibi soatning sekund, minut va soat millari harakatiga juda o'xshaydi. Haqiqatdan, minut mili bir minutga siljiguncha sekund mili 60 ta harakat sodir qiladi, soat mili bir soatga siljishi uchun esa minut mili 60 minutga siljiydi. Ya'ni tashqi sikl parametri soat miliga, o'rta sikl parametri minut miliga, ichki sikl parametri esa sekund miliga mos keladi. Demak, tashqi siklda 5 marta, o'rta siklda 8 marta, ichki siklda 9 marta takrorlanishi sodir bo'lsa, buyruqlar jami $5 * 8 * 9 = 360$ marta bajariladi.

Ichma-ich joylashgan sikllarning umumiy ko'rinishida blok belgilari (`{ }`) yo'q. Agar bir `for` operatori bilan ikkinchisi oralig'ida kamida bitta buyruq bo'lsa ham blok belgisi ichiga olinadi.

Misol. Natural sonning tub bo'luvchilarini va ularning yig'indisini topish dasturini tuzing.

```
#include <iostream.h>
using namespace std;
int main ()
{
    int i, j, n, s, m; s=0; cin>>n;
    for (i=2; i<=n; i++){ m=i/2; for (j=2; j<=m; j++)
```

```
if (i%j) continue; else goto tam;
if (!(n%i)) { cout <<n<< " sonning tub bo'luvchisi=" <<i<< "\n";
    i++; }
tam: continue;};
cout << "tub bo'luvchilari yig'indisi=" <<s<< "\n";
system ("pause");
return 0;
}
```

Masalan, n ning qiymati sifatida 45 soni kiritilsa, dastur javobi quyidagicha chop etiladi:

```
45 sonining tub bo'luvchisi = 3
45 sonining tub bo'luvchisi = 5
Tub bo'luvchilari yig'indisi = 8
```

2. Ichma-ich joylashgan sikllarning `while` va `do-while` shakllarini quyidagi ko'rinishlarda ifodalash mumkin. Old shartli ichma-ich joylashgan sikllarning umumiy ko'rinishi quyidagicha:

```
i = i_min; while (i <= i_max)
{ j = j_min; while (j <= j_max)
{ k = k_min; while (k <= k_max)
{ buyruqlar; k = k + k_qad; }
j = j + j_qad; }
i = i + i_qad; }
```

Buyruqning bajarilish tartibi `for` ko'rinishi bilan bir xil.

Misol 1. Quyidagi ifodaning qiymatini hisoblash dasturini tuzing.

$$\sum_{i=0}^n \sum_{j=2}^m \sum_{k=0}^l (2i + 3j - 4k)$$

```
#include <iostream.h>
using namespace std;
int main()
{ int i, j, k, n, m, l, s1, s2, s3;
  cin >>n>>m>>l; s1=0;
  i=1; while (i<=n) {s2=0;
  j=2; while (j<=m) {s3=0;
  k=0; while (k<=l)
  { s3+=2*i+3*j-4*k; k++;}
  s2+=s3; j++;}
```

```

s1+=s2; i++;}
cout << "s1=" << s1 << "\n"; system("pause");
return 0;
}

```

do-while ko'rinishida ichma-ich joylashgan sikllarning umumiy ko'rinishi quyidagicha:

```

i = i_min;
do
{
j = j_min;
do
{
k = k_min;
do
{ buyruqlar;
k = k + k_qad; }
while (k <= k_max);
j = j + j_qad; }
while (j <= j_max);
i = i + i_max; }
while (i <= i_max)

```

Buyruqning bajarilish tartibini mustaqil o'rganing.

Misol 2. Quyidagi ifodaning qiymatini hisoblash dasturini tuzing.

$$\prod_{i=1}^n \prod_{j=2}^m \prod_{k=1}^l (i + j + k)$$

```

#include <iostream.h>
using namespace std;
int main( )
{ int i, j, k, n, m, l, p1, p2, p3; cin >> n >> m >> l;
p1 = 1; i = 1; do {
p2 = 1; j = 2; do {
p3 = 1; k = 1; do {
p3 *= i+j+k; k++; } while (k <= l);
p2 *= p3; j++; } while (j <= m);
p1 *= p2; i++; } while (i <= n);
cout << "p1 =" << p1 << "\n"; system("pause");
return 0;
}

```

Misol 3. $y=3x-7a+9,3$; $x \in [-3;2]$; $\Delta x=0,5$; $a \in [1;5]$; $\Delta a=0,8$; funksiya qiymatini hisoblash dasturini tuzing.

```

#include <iostream.h>
#include <math.h>
using namespace std;
int main ()
{ float x,y,a;
x=-3; do {
a=1; do { y=3*x-7*a+9.3;
cout << "y (" << x << ", " << a << ") = " << y << "\n";
a=a+0.8; } while (a <= 5); x=x+0.5; }
while (x <= 2);
system("pause");
return 0;
}

```

Quyidagi savollarga javob bering

1. Qanday masalalarning dasturlarini tuzishda sikl buyruqlaridan foydalaniladi?
2. C++ dasturlash tilida sikl buyruqlarining necha xil ko'rinishidan foydalaniladi? Ularni sanab ko'rsating.
3. Parametrli sikl buyrug'ining umumiy ko'rinishini yozing va bajarilish tartibini tushuntiring.
4. 1 dan n gacha bo'lgan 4 ga karrali sonlar yig'indisini topish dasturini parametrli sikl buyrug'idan foydalanib tuzing.
5. Parametrli sikl buyrug'ida sikl parametrining qadami qayerda ko'rsatiladi?
6. Parametrli sikl buyrug'ini bajarish tugaganda sikl parametrining qadami nimaga teng bo'ladi?
7. 9 dan 41 gacha bolgan sonlar ko'paytmasini topish dasturini do-while sikl buyrug'idan foydalanib tuzing.
8. Oldshartli sikl buyrug'ining umumiy ko'rinishini yozing va bajarilish tartibini tushuntiring.
9. Sikl buyruqlarida qachon blok belgilaridan foydalaniladi?
10. So'ngshartli sikl buyrug'ining umumiy ko'rinishini, blok-sxemasini yozing va bajarilish tartibini tushuntiring.
11. Sikl buyrug'ining **do-while** ko'rinishida nima uchun blok belgilaridan foydalanilmaydi?

12. Nima uchun sikl buyruqlari old shartli va so'ng shartli nomlanadi.

13. Parametrli ichma-ich joylashgan sikllarning umumiy ko'rinishini yozib, bajarilish tartibini ayting.

14. So'ngshartli ichma-ich joylashgan sikllarning umumiy ko'rinishini yozib, bajarilish tartibini ayting.

15. Oldshartli ichma-ich joylashgan sikllarning umumiy ko'rinishini yozib, bajarilish tartibini ayting.

16. Agar tashqi sikl parametri x_1 2 dan 4 gacha, o'rta sikl parametri x_2 4 dan 7 gacha, ichki sikl parametri x_3 11 dan 13 gacha qiymatlarni qabul qila oladi. Ichma-ich joylashgan sikllardagi buyruq $t = x_1 + 2 * x_2 * x_3$; kabi bo'lsa t ning qiymatlarini toping.

17. So'ngshartli ichma-ich joylashgan sikllarda 1-mantiqiy ifodaning qiymati yolg'on bo'lsa, u qanday bajariladi?

18. Oldshartli ichma-ich joylashgan sikllarda: a) birinchi, b) ikkinchi, c) uchinchi mantiqiy ifodaning qiymati yolg'on bo'lsa, u qanday bajariladi?

19. Ichma-ich joylashgan sikllarni goto operatori yordamida tashkil qiling.

4.3 Ko'rsatkichlar

Ko'rsatkichlar C++ dasturlash tilining zarur qurollaridan biridir. Tushunchaning nomi nega ko'rsatkich, u nimani ko'rsatadi degan savollar tug'ilishi tabiiy.

Ko'rsatkich ham o'zgaruvchi bo'lib, u ham e'lon qilinadi va uning oddiy o'zgaruvchilar e'lonidan farqi shundaki, ularni e'lon qilishga oldidan ko'paytirish amali belgisi, ya'ni (*) yulduzcha qo'yiladi.

Ko'rsatkich o'zgaruvchi boshqa o'zgaruvchining EHM xotirasidagi adresiga teng qiymatni oladi, ya'ni o'sha o'zgaruvchining adresini ko'rsatadi. Ko'rsatkichni e'lon qilishning umumiy ko'rinishi qo'yidagicha

tur * o'zgaruvchi nomi;

Masalan, **int *x**; yozuvi butun turli **x** nomli ko'rsatkich o'zgaruvchi, **float *a**; yozuvi esa **a** nomli haqiqiy turli ko'rsatkich o'zgaruvchi e'lon qilinganligini bildiradi.

Yoki boshqacha qilib aytganda o'zgaruvchi e'lon qilinishi chog'ida uning oldiga (*) belgisi qo'yilsa, uni ko'rsatkichga aylantiradi. Bundan buyon «ko'rsatkich o'zgaruvchi» o'miga «ko'rsatkich» so'zidan foydalanamiz.

Ushbu maslik kerakki, ko'rsatkich qaysi turga mansub bo'lsa, faqat shu turga taalluqli o'zgaruvchilarning adreslarini aniqlay oladi.

Ko'rsatkichlar bilan ikkita, ya'ni «*» va «&» operatorlari ishlatiladi. Bunda «&» operatori o'zgaruvchining EHM xotirasidagi adresini aniqlaydi.

«*» operatori esa «&» aniqlagan o'zgaruvchining o'sha adresda joylashgan qiymatini aniqlaydi. Bu fikrlarni quyidagi dastur yordamida tushunib olish mumkin.

```
#include <iostream.h>
using namespace std;
int main( )
{
    int n, m, *x; n=123;
    x=&n; cout <<x<<"\n";
    m=*x; cout <<m<<"\n";
    system("pause");
    return 0;
}
```

Dastur natijasi quyidagicha bo'ladi:

0x28ff44

123

Dasturda quyidagicha o'zgarishlar qilinsa ham natijasida hech qanday o'zgarish bo'lmaydi.

```
int n=123;
int *x=&n; cout<<x<<"\n": m=*x;
yoki
int=123;
int*x (&n); cout<<x<<"\n"; m=*x;
```

Har ikkala holda ham **x** ga boshlang'ich qiymat sifatida **n** o'zgaruvchining xotiradagi adresi, **m** o'zgaruvchiga esa **n** o'zgaruvchi ko'rsatilgan adresdagi qiymati boshlang'ich qiymat sifatida berilmoqda.

Adresdagi qiymatlar ustida inkrement va dekrement amallarini bajarish

Ba'zan o'zgaruvchilarning xotirada egallangan joyini hisobga olishga ham to'g'ri keladi. 32 razryadli mashinalarda bitta belgi 1 bayt,

butun turli (int) va haqiqiy turli (float) o'zgaruvchilar to'rt bayt suzuvchi vergulli (double) turli o'zgaruvchilar sakkiz bayt joy egallaydi.

Ko'rsatkichdan qiymat berish buyrug'ining chap tarafidan foydalanish mumkin. Bu holda u qandaydir qiymatni oladi va shu qiymatni o'sha adresga yozib qo'yadi. Ushbu holatni quyidagi dastur tahlil qilish bilan tushunish mumkin.

```
#include <iostream.h>
using namespace std;
int main()
{
    int *p, x=100; cout <<"x ning eski qiymati=" <<x<<"\n";
    p=&x; cout<<"x o'zgaruvchining adresi=" <<p<<"\n";
    *p=213;
    cout<<"x ning yangi qiymati=" <<x<<"\n";
    system("pause");
    return 0;
}
```

Dastur natijasi quyidagicha bo'ladi:

x ning eski qiymati= 100
x o'zgaruvchining adresi= 0x28ff44
x ning yangi qiymati= 213

Shuningdek, ko'rsatkich yordamida u ko'rsatgan adresdagi qiymatning ustida inkrement va dekrement amallarini bajarish mumkin. Bunda ko'rsatkich qavs ichida yozilishi shart. Chunki inkrement va dekrement amallari ko'rsatkichga nisbatan oldin bajariladi. Buni quyidagi dastur misolida tushunish mumkin.

```
#include <iostream.h>
using namespace std;
int main()
{
    float x=54.25, *p1; cout <<"x=" <<x<<"\n";
    p1=&x; cout <<"x ning adresi=" <<p1<<"\n";
    (*p1)++; cout <<"x=" <<x<<"\n";
    (*p1)++; cout <<"x=" <<x<<"\n";
}
```

```
(*p1)++; cout <<"x=" <<x<<"\n";
(*p1)++; cout <<"x=" <<x<<"\n";
(*p1)++; cout <<"x=" <<x<<"\n";
(*p1)++; cout <<"x=" <<x<<"\n";
(*p1)++; cout <<"x=" <<x<<"\n";
system("pause");
return 0;}
```

Dastur natijasi quyidagicha bo'ladi:

x=54.25
x ning adresi= x28ff44
x=55.25
x=56.25
x=55.25
x=13
x=14
x=15
x=14

Endi ko'rsatkichlar ustida arifmetik amallarni bajarish bilan tanishamiz:

Bizga ma'lumki, float *a, *b; buyrug'i bo'yicha a va b nomli ko'rsatkichlarni aniqlaymiz. Shuningdek, a=&n; buyrug'i bo'yicha a ko'rsatkichning qiymati n ning xotiradagi adresiga teng bo'ladi. Shundan so'ng (*a) ko'rsatkich n ning EHM xotirasidagi qiymatiga teng qiymatni oladi. Lekin, a=&n; buyrug'i bo'lmasa (*a) ham hech qanday qiymatni olmaydi. Chunki adressiz ko'rsatkich qaysi qiymatni olishi mumkin? Agar a=&n; buyrug'idan keyin (*a)++ buyrug'i o'rniga a++ buyrug'idan foydalansak, bu inkrement amali n ning EHM xotirasidagi adresiga birni qo'shmaydi, balki shu adresdan keyingi adres qiymatiga teng qiymatni oladi. Masalan, n ning xotiradagi adresi 0x28ff3c bo'lsa, a++ ning qiymati 0x28ff40 bo'ladi. Bunda 0x28ff3c ning ustiga 4 ni qo'shdi. Chunki yuqorida qayd qilganimizdek, float turli o'zgaruvchilar xotirada 4 bayt joyni egallaydi.

Tabiiyki, a=&n; cout<<- - a <<"\n"; buyrug'i natijasi 0x288ff38 bo'ladi. Bunda a ning dastlabki qiymati hisobga olinaydi, ya'ni 0x288ff3c dan 4 ayrilyapdi (16lik sanoq sistemasida).

Demak, biror o'zgaruvchining xotiradagi adresini 1ga oshirmoqchi bo'lsak, uning qiymati 1 ga emas, balki o'sha o'zgaruvchining xotirada egallaydigan joyining hajmi qadar oshadi.

Xuddi shuningdek, ko'rsatkichga qandaydir butun sonni qo'shsak yoki ayirsak o'sha ko'rsatkich turi xotirada egallaydigan joyi hajmini berilgan butun songa ko'paytirilgani qadar sonni qo'shish yoki ayirish natijasiga teng bo'ladi. Aytilganlar bo'yicha quyidagi dasturni va uning natijalarini tahlil qiling.

```
#include <iostream.h>
using namespace std;
int main()
{
    int *y,a,n=-3;
    y=&n; cout <<"y="<<*y<<"\n";
    cout <<"z="<<y<<"\n";
    y=y+5; cout <<"y+5="<<y<<"\n";
    y=y-8; cout <<"y-8="<<y<<"\n";
    y=&n-5; cout <<"&n-5="<<y<<"\n";
    system("pause");
    return 0;}
```

Dastur natijasi quyidagicha bo'ladi.

```
*y=-3
y=0x28ff3c
y+5=0x28ff50
y-8=0x28gg30
&n-5=0x28ff28
```

Bunda amallar 16 lik sanoq sistemasida bajarilmoqda. Ko'rsatkichlar ustida qo'shish amalini bajarib bo'lmaydi. Lekin, ikkita ko'rsatkich ustida ayirish amalini bajarish mumkin. Bu holni quyidagi dastur misolida namoyish qilamiz.

```
#include <iostream.h>
#include <math.h>
using namespace std;
int main()
{
```

```
int *y, *z,a,n=-3; a=14;
y=&n; cout <<"y="<<y<<"\n";
z=&a; cout <<"z="<<z<<"\n";
cout <<"z-y="<<z-y<<"\n";
cout <<"*z-*y="<<*z-*y<<"\n";
cout <<"*z+*y="<<*z+*y<<"\n";
system("pause");
return 0;}
```

Dastur natijasi quyidagicha bo'ladi.

```
y=0x28ff34
z=0x28ff38
z-y=1
*z-*y=17
*z+*y=11
```

Dasturni va uning natijalarini mustaqil tahlil qiling.

Quyidagi savollarga javob bering

1. Ko'rsatkich tushunchasining mohiyatini tushuntiring.
2. Ko'rsatkich bilan o'zgaruvchining farqini tushuntiring.
3. float *z, *g; yozuvini qanday tushunasiz?
4. c=*f; va d=&j; yozuvlarini qanday tushunasiz?
5. Ko'rsatkichlar bilan ishlashda adresdagi qiymatlar ustida inkrement va dekrement amallari qanday bajariladi?
6. Ko'rsatkichlar ustida arifmetik amallar qanday bajariladi?

Quyidagi masalalarning dasturlarini tuzing

1. k va n natural sonlari berilgan. k sonini n marta chop etuvchi dastur tuzing.
2. a va b butun sonlari berilgan ($a < b$). a va b sonlar orasidagi barcha butun sonlarni (a va b ni ham) va ularning sonini aniqlovchi dastur tuzing.
3. n ($n > 0$) butun son berilgan. Quyidagi yig'indini hisoblovchi dastur tuzilsin:

$$s = n^2 + (n+1)^2 + (n+2)^2 + \dots + (2n)^2.$$

4. n ($n > 0$) butun son berilgan. Quyidagi ko'paytmani hisoblovchi dastur tuzilsin: $P = 1.1 * 1.2 * 1.3 * \dots$ (n ta ko'paytuvchi).
5. N ($N > 0$) butun son berilgan. Quyidagi ifodani hisoblovchi dastur tuzilsin:

$$S = 1.1 - 1.2 + 1.3 - \dots$$
6. a xaqiqiy va n ($n > 0$) butun sonlari berilgan. a ning n -darajasini aniqlovchi algoritmi tuzilsin: $a^n = a * a * \dots * a$
7. a xaqiqiy va n ($n > 0$) butun sonlari berilgan. Bitta sikldan foydalanib a ning 1 dan n gacha bo'lgan barcha darajasini chiqaruvchi algoritmi tuzilsin.
8. Berilgan n ta $A(x, y)$ nuqtalardan qaysilari $y = kx + b$ to'g'ri chiziqli grafigiga tegishli ekanligini aniqlang.
9. Berilgan n ta $B(x, y)$ nuqtalardan qaysilari R radusli doiraga tegishli ekanligini aniqlang.
10. Berilgan n ta sondan nechitasi butun son ekanligini aniqlang.
11. Berilgan sonning bo'luvchilari sonini aniqlang.
12. Berilgan sonni tub yoki murakkabligini aniqlang.
13. O'zidan oldingi natural sonlar yig'indisidan 5 marta kichik bo'lgan ikki xonali sonlarni aniqlang.
14. Shunday uch xonali sonlarni topingki, ularning dastlabki ikkita raqamidan tuzilgan qismi 3 raqamiga qoldiqsiz bo'linsin.
15. Raqamlar yig'indisi 27 ga teng bo'lgan 4 xonali sonlarni toping.
16. Ketma-ket n marta uchburchak tomonlari kiritiladi. Qaysi urinishda kiritilgan tomonlar bo'yicha uchburchak yasash mumkinligini aniqlang.
17. n xonali son kiritiladi. Unda nechta 0 va 1 raqamlari borligini aniqlang.
18. n butun sonning necha xonali ekanligini aniqlang.
19. Uch xonali armstrong sonlarni toping.
20. Ikki va uch xonali Palindrom sonlarini toping.
21. Bir va ikki xonali sonlar ichida o'z kvadrating oxirgi raqamlariga teng sonlarni toping.
22. $(1 + 1/2^2) * (1 + 1/3^2) * (1 + 1/4^2) * \dots * (1 + 1/n^2)$ ifodaning qiymatini hisoblang.
23. 1) $\sum_{i=2}^n (5 * i - 8a)$; 2) $\prod_{k=1}^m \prod_{j=2}^n (4 * k - j)$ ifodalarning qiymatini hisoblang.
24. n dan m gacha bo'lgan sonlar yozib chiqildi. Necha xonali son hosil bo'lganligini aniqlang?

25. O'zidan oldingi natural sonlar yig'indisidan 5 marta kichik bo'lgan bir va ikki xonali sonlarni toping.

26. 2 dan 50 gacha bo'lgan sonlar ichida 6 bilan o'zaro tub bo'lgan sonlarni toping.

27. 1 dan 100 gacha bo'lgan sonlar ichida dastlabki nechtasining o'zidan boshqa bo'luvchilari yig'indisi 163 dan katta bo'lishini aniqlang.

28. 1 dan n gacha bo'lgan sonlar ichida qaysilari o'zidan boshqa bo'luvchilari yig'indisiga teng bo'lishini aniqlang?

29. Ikki va uch o'lchovli kvadrat matritsalarining determinantlarini hisoblash dasturlarini tuzing.

30. Uch o'lchovli ikkita kvadrat matritsalarining ko'paytmasini topib, kvadrat matritsa ko'rinishida chop etish dasturini tuzing.

V BOB. MASSIV VA SATRLARNI QAYTA ISHLASH

5.1. Massiv tushunchasi

Dasturlash tillarida ro'yxat yoki jadval ko'rinishidagi ma'lumotlarni massiv deb atashadi. Massiv so'zining ma'nosi o'lcham, o'lcham demakdir.

Jurnaldagi o'quvchilar ro'yxati, paradda to'g'ri to'rtburchak hosil qilib turgan soldatlar, bog'da tartib bilan ekilgan daraxtlar, talabalarning reytingini aniqlovchi qaydnoma, determinantlar, matritsalar, Pifagorning ko'paytirish jadvali massivga misol bo'la oladi.

Massiv o'z ichiga qamrab olgan har bir obyekt, ya'ni har bir o'quvchi, soldat, daraxt yoki determinant, matritsani hosil qiluvchi har bir son o'sha massivning elementi yoki massiv komponentasi deyiladi. Keltirilgan misollardan ko'rinib turibdiki, massivning barcha elementlari bitta turga mansub bo'lib, ular bitta nom bilan nomlanadi va bir-birlaridan nomlari (indekslari) bilan farq qiladi.

Demak, bitta nom bilan nomlanuvchi, bitta turli mansub va bir-birlaridan faqat nomlari bilan farq qiluvchi chekli sondagi obyektlar to'plami massiv deyiladi.

Bir o'lchamli massivlar

Bir o'lchamli massivning har bir elementi bitta nomerga ega bo'ladi. Bunday massivlar asosan obyektlar ro'yxati yoki bir o'zgaruvchili funktsiya grafigini chizish uchun tuzilgan jadval kabi ma'lumotlardan tashkil topgan bo'ladi.

Bir o'lchamli massivni e'lon qilishning umumiy ko'rinishi quyidagicha

Tur nomi massiv nomi [o'lcham];

Masalan, `float a [20];`. Bunda `float` - massiv turi, `a` - massiv nomi. Massiv nomi faqat bitta harfdan iborat bo'lmasdan ixtiyoriy identifikator ham bo'lishi mumkin, `[20]` - massiv o'lchami, ya'ni massiv elementlari soni. Ushbu massiv quyidagicha o'qiladi. «Elementlari haqiqiy sonlardan iborat bo'lgan `a` nomli massiv e'lon qilingan bo'lib, elementlari soni 20 ta va ular 0 dan 19 gacha nomerlangan».

Massivni bu tarzda e'lon qilishdan maqsad EHM xotirasida uning elementlari uchun joy ajratishdir. Massivni e'lon qilmasdan uning

elementlaridan dastur tarkibida foydalanish har xil xatoliklarga olib kelishi mumkin.

1-1 dasturlash tilida massiv elementlari noldan boshlab nomerlanadi. Masalan, massiv o'lsami `n` bo'lsa uning elementlari 0 dan `n-1` gacha nomerlanadi.

Massiv elementlari nomerli kvadrat qavsga olib ko'rsatiladi. Demak, yuqorida e'lon qilingan massivning birinchi element `a[0]`, oxirgi elementi esa `a[19]` bo'ladi.

Massiv elementlari nomeri albatta butun son bo'lishi shart. Shuningdek, massiv indeks qiymati butun son bo'ladigan algebrik ifoda bo'lishi ham mumkin. Yoki massiv indeksi butun turli o'zgaruvchi ham bo'lishi mumkin. Shuning uchun agar massiv indeksini sikl parametri kabi aniqlash yo'li bilan massivning barcha elementlari ustida amallar bajarish mumkin.

`float` va `double` kabi haqiqiy turga mansub o'zgaruvchilarning qiymatlari har qanday kichik kesmada cheksiz ko'p bo'lganligi tufayli bunday o'zgaruvchilardan indeks sifatida foydalanib bo'lmaydi, ya'ni bunday o'zgaruvchilar yordamida massiv elementlarini tartiblab sanab bo'lmaydi.

Massiv uzunligi, ya'ni elementlari sonini aniqlash uchun

`sizeof` (massiv nomi)

funktsiyasidan foydalaniladi. Aytilganlarni quyidagi dastur misolida tushunish qiyin emas.

```
#include <iostream.h>
using namespace std;
int main()
{int n=53, m=23; n=n-m;
int s[n+42], j, i; j=sizeof(s);
char satr[n+42]; i=sizeof(satr);
cout << " j=" << j << "\n";
cout << " i=" << i << "\n";
system("pause"); }
Dastur natijasi
j=288
i=72 kabi bo'ladi.
```

Chunki har bir butun turli o'zgaruvchi xotirada 4 bayt joyni, char turi har bir belgi esa bir bayt joyni egallaydi.

Misol 1. Tasodifiy tanlangan 6 ta son hosil qilish dasturini tuzing.

```
#include <iostream.h>
#include <time.h>
using namespace std;
int main( )
{
    int a [10],i; srand(time(NULL));
    for (i=0; i<=5; i++) { a [i]=rand ();
    cout <<"a["<<i<<"]="<<a[i]<<"\n";}
    system ("pause");
    return 0;
}
```

Dastur ijro etilishi natijasida quyidagicha natijalar chop etiladi.

```
a [0]=41
a [1]=18467
a [2]=63345
a [3]=26500
a [4]=119169
a [5]=15724
```

Bunda **rand()** funksiyasi 6 ta sonni tasodifiy tanladi. Massivni massivga qiymat sifatida berib bo'lmaydi.

Masalan, **float x[15], y[15];** kabi **x** va **y** massivlar e'lon qilingan bo'lsa, **x=y** yoki **y=x** kabi buyruqlarga ruxsat etilmaydi. Lekin bu massiv elementi qiymatini ikkinchi massiv elementi olishi mumkin. Masalan, **a[0]=b[10]** yoki **b[5]=a[3]** kabi buyruqlarga ruxsat etiladi.

Massiv o'lchami e'lon qilingandan keyin uning chegarasidan chiqib ketish kutilmagan xatoliklarga olib kelishi mumkin. Masalan, massiv **int b[40]** kabi e'lon qilingan bo'lsa, uning elementlari soni qirqta bo'lib, ular 0 dan 39 gacha nomerlangan bo'lishi bizga ma'lum. Lekin **b[41], b[46], b[55]** elementlardan foydalanishga yoki ularga qiymat berishga ruxsat etilmaydi.

Misol 2. $y=5x^2+4x-12$, $x \in [-3;15]$, $\Delta x=2$ funksiya qiymatlarini hisoblash dasturini tuzing.

```
#include <iostream.h>
using namespace std;
int main( )
```

```
{
    int i,j;
    for (i=1; i<=15; i++) {
        for (j=1; j<=5; j++) { y [i]=5*x*x+4*x-12;
        cout <<"y["<<i<<"]="<<y[i]<<"\n";}
        system ("pause");
    }
}
```

Ko'p o'lchamli massivlar

Ko'p o'lchamli massivlar quyidagicha e'lon qilinadi.

Fur nomi massiv nomi [1-o'lcham] [2-o'lcham].....[n-o'lcham]

Masalan, **float x[5] [4]** yozuvi quyidagicha o'qiladi. Elementlari haqiqiy turli ikki o'lchamli **x** nomli massiv e'lon qilingan bo'lib, uning elementlari beshta satr va to'rtta ustunda joylashgan. Elementlarining soni 20 ta. Demak, ikki o'lchamli massivda birinchi indeks satr nomerini, ikkinchi indeks esa ustun nomerini bildiradi. Berilgan massiv elementlarini quyidagi matritsa ko'rinishida tasavvir qilish mumkin.

```
x [0] [0] x [0] [1].....x [0] [3]
x [1] [0] x [1] [1].....x [1] [3]
.....
x [4] [0] x [4] [1].....x [4] [3]
```

Ko'p o'lchamli massivlar ichida ikki o'lchamli massivlardan eng ko'p foydalaniladi. Bunday massiv elementlarini chop etishda ham aynan matritsa ko'rinishda chop etishga e'tibor qilinsa, natijalarini tahlil qilish jarayoni ancha yengillashadi.

Misol. Pifagorning ko'paytirish jadvalini hosil qilish dasturini tuzing.

```
#include <iostream.h>
using namespace std;
int main ()
{
    int p[8][8], i, j;
    i=0; do {
        j=0; do { p[i][j]=(i+1)*(j+1);
```

```

if (p[i][j] / 10 == 0) cout << "" << p[i][j];
else cout << "" << p[i][j]; j++;} while (j <= 8);
i++; cout << '\n'; cout << '\n';} while (i <= 8);
system("pause");
return 0;
}

```

Massiv o'lchami kattalashgan sayin uning elementlari uchun xotiradan ajratiladigan joy hajmi juda tez o'sadi. Masalan, 32 razryadli mashinada uch o'lchamli float b [10] [8] [10] massivi e'lon qilinsa bo'lsa, u xotirada $10 \cdot 10 \cdot 8 \cdot 4 = 3200$ bayt joyni egallaydi.

Agar shu massiv int b [10] [10] [10] [10] kabi to'rt o'lchamli ko'rinishda e'lon qilinsa, xotiradan 40000 baytlik joy band qilinadi.

Demak, n o'lchamli massivning xotiradan egallaydigan joyi formulasi quyidagicha.

Baytlar soni = 1-o'lcham * 2-o'lcham * * n-o'lcham * turning baytlardagi o'lchami.

Amalda uch o'lchamli massivlardan ham juda kam foydalaniladi. Uch o'lchamdan yuqori o'lchamli massivlardan juda kam foydalaniladi.

Massivlarni initsializatsiyalash

C++ dasturlash tilida massiv elementlarini initsializatsiyalash (dastlabki qiymatlarini berish) masalasi ham alohida imkoniyat sifatida qaraladi.

Massiv elementlarini initsializatsiyalashning umumiy ko'rinishi quyidagicha:

Tur nomi massiv nomi [o'lcham]={dastlabki qiymatlari ro'yxati};

Dastlab qiymatlari ro'yxati bir-birlaridan vergul bilan ajratilib, oxirgi qiymatdan keyin hech qanday tinish belgilari qo'yilmaydi va blok belgisi ichiga olib yoziladi. Blok belgisidan keyin (;) qo'yilishiga e'tibor qiling. Chunki murakkab buyruqlarda figurali qavsdan oldin (;) belgisi qo'yilar edi.

Masalan, elementlari 8 ta bo'lgan butun turli c nomli massiv quyidagicha initsializatsiyalanadi.

```
int c[8]={3,16,-5,9,29,36,65,1};
```

Har bir massivning har bir elementi o'z nomeriga mos nomerda elementni qiymat sifatida oladi, ya'ni $c[0]=3$; $c[4]=29$; kabi qiymat berish buyruqlari nooshkor holda bajariladi. Har bir massivlar initsializatsiyalanayotganda esa satrning (matnning) har bir belgisini tartib nomeri bo'yicha massivning bitta elementi sifatida oladi. Ammo satrda n ta element bo'lsa, massiv elementini n+1 deb tanlash kerak. Chunki satrning oxiriga bitta '\0' element avtomatik tarzda qo'shiladi.

Masalan, char s[6]="salom"; ko'rinishdagi initsializatsiya char s[6]={ 's', 'a', 'l', 'o', 'm', '\0' }; kabi initsializatsiyalashga teng kuchlidir.

Hunda nooshkor holda s [0]='s'; s [1]='a'; s[2]='l';..... kabi initsializatsiyalash jarayoni amalga oshiriladi.

Misol. 1 dan 10 gacha bo'lgan sonlarning kvadratlarini chop etish dasturini tuzing.

```

#include <iostream.h>
using namespace std;
int main()
{
    int kod[10]={1, 2, 3, 4, 5, 6, 7, 8, 9, 10}, i=1;
    while (i<=10) { kod [i]= i*i;
    cout << "n [" << i << "]=" << kod [i] << "\n"; i++;}
    system("pause");
    return 0;
}

```

Misol. Berilgan so'zning harflari o'rtasida ikkitadan probel qoldirib chop etish dasturini tuzing.

```

#include <iostream.h>
using namespace std;
int main ()
{
    char satr[11]="matematika";
    int i=0;
    while (i<=10) { cout << " " << satr [i] ; i++;}
    cout << "\n";
    system("pause");
}

```

```
return 0;
}
```

Ko'p o'lchamli massivlarni initsializatsiyalash

Ko'p o'lchamli massivlar ham bir o'lchamli massivlar kabi initsializatsiyalanadi. Masalan, quyidagi uch holda ham ikki o'lchamli massiv 1 dan 5 gacha bo'lgan sonlarga ularning kublarini mos keltiradigan qilib initsializatsiyalangan. Ya'ni uchala hol ham teng kuchli bo'lib, dastur tarkibida ularning ixtiyoriy bir ko'rinishida foydalanish mumkin.

```
1) int kub [5] [2]={
    1,1,
    2,8,
    3,27,
    4,64,
    5,125
};
```

```
2) int kub [5] [2]={
    { 1,1},
    { 2,8 },
    { 3,27},
    { 4,64},
    { 5,125}
};
```

```
3) int kub [5] [2]={ 1,1,2,8,3,27,4,64,5,125};
```

Har uchala holda nooshkor holda quyidagicha qiymat berish buyruqlari bajariladi.

```
kub [0] [0]=1 ; kub [0] [1]=1 ; kub [1,0]=2 ; kub [1,1]=8 ;
```

```
kub [2] [0]=3 ; ... , kub [4] [0]=5 ; kub [4] [1]=125 ;
```

Endi 1 dan 5 gacha bo'lgan sonlarga ularning kublarini mos keltiruvchi dasturni tuzamiz:

```
#include <iostream.h>
using namespace std;
int main ()
{
    int i,j;
    int kub[5] [2]={ 1,1,2,8,3,27,4,64,5,125};
    i=0; do {
        j=0; do {
            cout <<" " <<kub[i][j] ; j++; } while (j<=1);
            i++; cout <<"\n" ; } while (i<=4);
```

```
system ("pause");
return 0 ;
```

Dastur natijasi quyidagicha chop etiladi.

```
1 1
2 8
3 27
4 64
5 125
```

Ikki dan ortiq o'lchamli massivlar initsializatsiyasi ham xuddi shu kabi amalga oshiriladi.

Masalan, quyidagi dasturda uch o'lchamli massivning har bir elementiga uning indeksleri yig'indisi mos keltirilgan.

```
#include< iostream.h >
using namespace std;
int main ()
{
    int i, j, k;
    int n[4][2][3]={
        0,1,2,
        1,2,3,
        1,2,3,
        2,3,4,
        2,3,4,
        3,4,5,
        3,4,5,
        4,5,6};
    i=0; do {
        j=0; do {
            k=0; do {
                cout <<" " << i<< j << k << " " << n[i][j][k] ; k++; } while
                (k<=2);
                j++; cout <<"\n" ; } while (j<=1);
                i++; while (i<=3);
            return 0;
        }
```

Dastur va uning natijalarini mustaqil tahlil qiling

Eslatma: Massiv elementlarini initsializatsiyalashda ko'rsatma (berilgan) qiymatlar yetarli bo'lmasa, qiymatlar yetmasa elementlariga C++ dasturlash tili avtomatik tarzda nol qiymatini beradi.

O'lchamsiz initsializatsiyalash

Massivlarni initsializatsiyalashda ular e'lon qilinishidayoq massiv elementlari soni `int kvad[10]`, `char satr[11]`, `int kub[5][2]`, `int n[4][2][3]` kabi ko'rsatilgan edi. Ammo elementlar sonini ko'paytirish yoki kamatirishga to'g'ri kelib qolsa, har safar o'lchamini ham o'zgartirib chiqish kerak bo'ladi. Agar massiv o'lchami bo'sh holda ko'rsatilsa, C++ dasturlash tili o'lchamni avtomatik tarzda massiv elementlari soniga mos holda tanlaydi.

Masalan, `int kvad[] = {5, -6, 9, 13, 28, 91, 131, 41, 35};` buyrug'i bo'yicha o'lcham 9, `char h[] = "kutubxonalar";` buyrug'i bo'yicha o'lcham 13 ga teng deb avtomatik tarzda tanlanadi.

Ko'p o'lchamli massivlarga esa faqat birinchi o'lchamni ko'rsatmaslik mumkin, qolganlarini aniq ko'rsatish shart. Masalan, oldingi mavzudagi dasturlarga `int kub[][2]` va `int n[][2][3]` kabi o'zgarish qilib ishlatib ko'ring va o'zingiz ishonch hosil qiling.

Quyidagi savollarga javob bering

1. Massiv deganda nimani tushunasiz va u qanday ma'noni anglatadi? Misollar keltiring.
2. Massivlar qanday va qayerda e'lon qilihadi? Massiv nima maqsadda e'lon qilinadi?
3. `int son[8];` yozuvini tushuntiring va u qanday o'qiladi?
4. Necha xil massivlardan foydalaniladi?
5. $M(n, m)$ massivda nechta ustun va nechta satr bor?
6. Massiv elementlari boshqa kattaliklar majmuasidan qanday farq qiladi?
7. Bir, ikki, uch o'lchamli massiv elementlari qanday yoziladi? Yosib ko'rsating.
8. Matritsa nima? Kvadrat matritsa qanday matritsa?
9. Kvadrat matritsaning:
 - a) bosh diagonalida;
 - b) bosh diagonalidan yuqorida;
 - c) bosh diagonalidan pastda joylashgan elementlari qanday xususiyatlarga ega?

10. `float c[30];` massivi e'lon qilingan bo'lib, uning elementlari qonuniyat bo'yicha o'zgarsa, uning n-elementi nachaga teng bo'ladi?

11. Elementlari butun turli D nomli ikki o'lchamli massiv elementlari 6 satr va 23 ustunda joylashgan. Shu massivni e'lon qiling.

12. Bir o'lchamli massiv qanday initsializatsialanadi? Misollar keltiring.

13. Ko'p o'lchamli massiv qanday initsializatsialanadi? Misollar keltiring.

14. Massivlarni o'lchamsiz initsializatsialashni tushuntiring. Misollar keltiring.

15. `char f[8] = "g'hty+e";` kabi initsializatsialashni yana qanday usulga oshirish mumkin?

Quyidagi masalalarning dasturlarini tuzing

1. $y(x) = 2x^3 + 4x + 12$; $-5 \leq x \leq 29$, $x_{\text{qad}} = 1$ bolsa, $y(0) + y(-4) - y(8) + y(13)$ ifoda qiymatini topish;
2. $p(x, y) = 2xy - 4ay + 9(y - ax)$ $2 \leq x \leq 20$, $-2 \leq y \leq 15$; $x_{\text{qad}} = 1$, $y_{\text{qad}} = 3$ bolsa, $p(0, 13) + p(4, -2) * p(18, 10) - p(p(-4, 4), p(1, 7))$ ifoda qiymatini topish dasturlarini tuzing.
3. berilgan 8 xonali sonning chapdan nechanchi o'rinda turgan raqamlari tengligini aniqlang.
4. A nomli kvadrat matritsa berilgan bo'lib, uning elementlari $a_{ij} = 4 * i - 5 * j$ qonuniyat bo'yicha o'zgaradi:
 - a) uning elementlarini matritsa ko'rinishida chop eting;
 - b) uni birlik matritsaga aylantiring;
 - c) matritsaning indekslari yig'indisi 10 ga teng elementlarini toping;
 - d) musbat va manfiy elementlari yig'indisini toping;
 - e) bosh diagonalidan yuqoridagi elementlarini 5 bilan, quyidagi elementlarini -5 bilan almashtiring;
 - f) juft va toq elementlari sonini toping.
5. $z(x, y) = 7 * x^2 - 3 * y + 9 * a$, $-3 \leq x \leq 8$, $\Delta x = 1$, $0 \leq y \leq 12$, $\Delta y = 2$ funksiyaning qiymatlarini hisoblash dasturini tuzing.

5.2. Satrli massivlar

C++ dasturlash tilida satrli massivlarni e'lon qiluvchi alohida mavjud emas, ularni e'lon qilish uchun ham `char` so'zidan foydalaniladi.

Agar matn hajmi katta bo'lsa, uni bir nechta satrlarga yozish to'g'ri keladi va har bir satrni bir o'lchamli massiv tarzida yaratib, initsializatsiyalash kerak bo'ladi. Bunday massivlarni yaratish noqulayliklarni keltirib chiqaradi. Shuning uchun C++ dasturlash tilida ikki o'lchamli satrli massivlardan foydalaniladi. Matnni bir necha satrlarga yozish imkoniyati mavjud. Masalan, quyidagi massiv

```
char massatr[10][80];
```

kabi aniqlangan bo'lsa, C++ dasturlash tilida xotiradan 10 ta satr va har bir satrda 80 tadan belgi uchun joy ajratadi. Demak, massiv bu tarzda berilsa, birinchi indeks satrlar sonini, ikkinchi indeks esa har bir satrda yozilishi mumkin bo'lgan belgilar sonini bildiradi.

Yuqorida aniqlangan `massatr[10][80]` massivning nechanchida satridagi ma'lumotlarni qayta ishlash uchun, uning birinchi indeksini ko'rsatish yetarli va o'sha ixtiyoriy satrida 80 tadan belgi bo'lishi mumkin.

Yuqorida keltirilgan fikrlarni quyidagi dastur misolida tushunish mumkin.

```
#include <iostream.h>
#include<cstdio.h>
int main ()
{int i;
char t[10][80];
for (i=0; i<=4; i++) gets (t [i]); cout <<"\n";
cout << t [2]<<"\n";
cout << t [1]<<"\n";
cout << t [0]<<"\n";
system ("pause");
return 0;
}
```

Dastur ijro etilishi jarayonida beshta qiymat so'raydi. Agar ularni quyidagi tartibda kiritsak,
0 maktab

```
1 maktab
2 maktab
3 maktab
4 maktab
5 natija quyidagicha
6 maktab
7 maktab
8 maktab
```

Bunda `gets( )` funksiyasi satrli massiv qiymatini kiritishga mo'ljallangan funksiya bo'lib, ulardan `<cstdio.h>` sarlavha fayli ishlatilida foydalanish mumkin.

Satrlarni qayta ishlashda funksiyalari

`Char` turiga mansub o'zgaruvchilarning qiymatlarini kiritish uchun ham `cin` kiritish operatorlaridan foydalanish mumkin. Ammo bunda satr tarkibida probel bo'lsa, probeldan keyingi qismini o'qimaydi. Shuning uchun bunday o'zgaruvchilarning qiymatlarini kiritishda

```
cin.getline(s, n)
```

funksiyasidan foydalanish maqsadga muvofiq.

Bunda `s`-satrli o'zgaruvchining nomi, `n`-kiritilishi kerak bo'lgan belgilar soni. Masalan, `getline(x,10)` buyrug'i bo'yicha 9 ta belgidan iborat belgilar ketma-ketligini kiritish mumkin. Chunki yuqorida qayd qilganimizdek satrli ma'lumotlarni kiritishda C++ dasturlash tili oxirgi belgi sifatida avtomatik tarzda `'\0'`, ya'ni nol belgisini o'qimaydi. Agar kiritilayotgan satrdagi belgilar soni `n` dan ko'p bo'lsa, ortiqchasini tashlab yuboradi.

```
Masalan, char t[20], t1[20];
```

```
cin.getline (t,8); cout <<"t="<<t<<"\n";
```

```
cin.getline (t1,5); cout <<"t1="<<t1<<"\n";
```

dastur ijro etilishi jarayonida `"maktab"` va `"sinf"` so'zlari kiritilsa, natija

```
maktab
```

```
sinf
```

kabi chop etiladi.

Eslatma: satrlar bilan ishlash funksiyalarining aksariyat qismi `<string.h>` sarlavha faylida bo'ladi.

Satr uzunligini aniqlash funksiyalari

Satrlarni qayta ishlash jarayonida ularning uzunligi (ya'ni unda belgilar sonini) bilish katta ahamiyatga ega. Probel belgisi ham son uzunligini hisoblashda bitta belgi sifatida inobatga olinadi. Satr uzunligini aniqlash uchun `<string.h>` funksiyalar kutubxonasidan `strlen (satr)` funksiyasidan foydalaniladi. `strlen(satr)` funksiyasi satrning haqiqiy uzunligidan bitta kam uzunlikni ko'rsatadi. Ya'ni oxirgi belgi '\0' ni inobatga olmaydi. Yoki boshqacha qilib aytganda nechta belgi kiritgan bo'lsangiz o'sha belgilar soniga teng qiymatni oladi. Agar satrning haqiqiy uzunligini aniqlash kerak bo'lsa, `sizeof (satr)` funksiyasidan foydalaniladi.

Masalan,

```
char s[] = "universitet";
n= strlen(s); cout << "n=" << n << "\n";
m= sizeof(s); cout << "m=" << m << "\n";
buyruqlari natijasi
```

n=11

m=12

kabi bo'ladi.

Agar satrli massiv `char s[50]`; kabi aniqlangan bo'lsa, `s` massiv nechta belgilar ketma-ketligini qiymat sifatida olishidan qat'iy nazar `sizeof(s)` funksiyaning qiymati 50 ga teng bo'laveradi, ya'ni satrning xotiradagi uzunligiga teng bo'lgan qiymatni oladi.

Masalan, `char s[50]="matematika";`

```
n= strlen(s); cout << "n=" << n << "\n";
m= sizeof(s); cout << "m=" << m << "\n";
buyruqlar natijasi
```

n=10

m=50

kabi bo'ladi.

Satrlarni nusxalash funksiyalari

Satr qiymatini biridan ikkinchisiga nusxalashga mo'ljallangan bir nechta funksiyalar mavjud va ular `<string.h>` funksiyalar kutubxonasiga joylashtirilgan. Ularning ba'zilar bilan tanishamiz.

1. `strcpy(satr1,satr2)` funksiyasi. Bunda `satr2` ning qiymatini `satr1` ga baytma-bayt nusxalaydi. Bunda `satr2` ning uzunligi `satr1` uzunligidan katta bo'lmasligi kerak. Chunki C++ dasturlash tilining

ba'zi versiyalarida ortib qolgan qismi `satr1` ning ustidan nusxalanishi mumkin.

1. `strcpy (st1, st2, n)` funksiyasi. Bunda funksiya `st1` satrining dastlabki `n` ta belgisi ustiga `st2` ning `n` dastlabki `n` ta belgisini nusxalaydi.

2. `strdup (satr)` funksiyasi. Funksiya xotiradan `satr` uchun joy ajratib, o'sha joyga `satr` ni nusxalaydi.

Agar zarurat bo'lsa ko'rsatkichlardan foydalanib, satrning nechanchidir belgisidan keyingilarini ham nusxalash mumkin. Quyidagi dastur bu funksiyalar mohiyatini ochishga yordam beradi.

```
1 #include <iostream.h>
2 #include <string.h>
3 using namespace std;
4 int main()
5 { int i,j,n,m;
6 char*star1="Satr nusxasini olish";
7 char*satr2;
8 char str3[30]="Assalom", st1[20]="123456789 soni",
9 st2[26];
10 char*korsat=st1; cout<<korsat<<"\n";
11 korsat+=4; cout<<korsat<<"\n";
12 strcpy(st2, "nusxalashga misol"); cout<<st2<<"\n";
13 strcpy(st2, "soniga qarab nusxalash", strlen (str3));
cout<<st2<<"\n";
14 strcpy(st1, "dastlabki 4 belgini nusxalash",4); cout<<st1<<"\n";
15 satr2=strdup(satr1); cout<<satr2<<"\n";
16 system("pause");
17 return 0;
18 }
```

Dastur natijasi quyidagicha bo'ladi:

```
123456789 soni
56789 soni
nusxalashga misol
soniga shga misol
dast 56789 soni
satr nusxasini olish
```

10-satrdagi «ko'rsat» nomli ko'rsatkichga `st1` massivi qiymati initsializatsiyalangan. 11-satrdagi `korsar+=4`; buyrug'i bilan ko'rsatkich o'zgaruvchining qiymati 4 bayt o'ngga siljitiildi. 12-satrdagi «nuxxalashga misol» matni `strep( )` funksiyasi yordamida `st1` massiviga nusxalandi. 13-satrdagi buyruq bo'yicha «soniga qarab nusxalash» matnining dastlabki 7 ta belgisi `st2` massivining dastlabki 7 ta belgisi o'rniga nusxalandi. Chunki `strlen(str3)` funksiyaning qiymati 7 ga tengligini tushunish qiyin emas. 15-satrdagi buyruq bo'yicha `satr1` ko'rsatkichning qiymati `satr2` ko'rsatkichga nusxalandi.

Satrlarni ulash funksiyalari

Satrlarni ulash amalini bajarishga mo'ljallangan `strcat(satr1,satr2)` va `strncat(satr1, satr2, n)` funksiyalari mavjud bo'lib, ular `<string.h>` funksiyalar kutubxonasida aniqlangan.

Bunda `satr1` – tarkibiga qo'shimcha satr ulanayotgan satr, `satr2` – esa boshqa satrga ulanadigan satr.

`strcat(satr1, satr2)` funksiyasi `satr1` ning orqasidan `satr2` ni ulaydi.

`strncat(satr1, satr2, n)` funksiyasi esa `satr1` ning orqasidan `satr2` ning dastlabki `n` ta belgisini ulaydi.

Bu fikrlarni quyidagi dasturni tahlil qilish yo'li bilan tushunish mumkin.

```
#include <iostream.h>
#include <string.h>
int main ()
{
    char satr1 [60]="men 26-“, satr6[30]="men vrach";
    char satr2[30]="maktabda o'qiyman";
    char satr4 [60]="men 36-“, satr5[60]="men 16-“;
    char satr3 [30]="kollej talabasiman“
    strcat (satr1, satr2);
    strcat (satr4, satr3); strncat (satr5, satr3,15);
    cout <<satr1<<"\n"<<satr4<<"\n"<<satr5<<"\n";
    strncat (satr6," bo'lgan“,4); cout<<"\n"<<satr6<<"\n";
    strncat (satr6,"dimki“,3); cout<<satr6<<"\n";
    system ("pause");
    return 0;
```

Dastur natijasi quyidagicha bo'ladi. Uni tahlil qiling.
 men 26-maktabda o'qiyman
 men 36-kollej talabasiman
 men 16-kollej talabasi
 men vrach bo'
 men vrach bo'ldim

Satrlarni solishtirish (taqqoslash) funksiyalari

C++ da satrlarni solishtirish uchun quyidagi funksiyalardan foydalaniladi.

1. `strcmp(satr1, satr2)` funksiyasi. Bu funksiya `satr1` satrining har bir belgisi `satr2` satrining mos nomerlaridagi belgilarini ASCII kodlari bo'yicha solishtiradi.

Eslatma: Lotin alifbesining kichik harflari kodi katta harflari kodidan 32 ga ortiq qilib kodlangan va solishtirish natijasi qandaydir `n` butun soniga qiymat sifatida berilishi kerak. Masalan, `n=strcmp(s1,s2)`

Agar

`s1>s2` bo'lsa `n>0`

`s1=s2` bo'lsa `n=0`

`s1<s2` bo'lsa `n<0` bo'ladi

`n` ning qiymati birinchi farq qiluvchi belgilar kodlarini solishtirish bilan aniqlanadi.

```
#include <iostream.h>
#include <string.h>
using namespace std;
int main()
{
    int j,i,s,n,m; char k1[20]="",k2[20]="";
    char satr1[60]="Dasturlash ";char *kor1=satr1;
    char satr2[60]="dasturlash ";char *kor2=satr2;
    m=strcmp(kor1,kor2); cout<<" m="<<m;
    n=strcmp(kor2,kor1); cout<<" n="<<n<<"\n";
    system("pause");
    return 0;
```

}
satr1 va **satr2** larning birinchi belgilari **P** va **p** larning kodlari farq qilayapti. Shuning uchun dastur natijasi **m=-1**; **n=1** kabi bo'lishi mumkin.
 2. Agar satrlarni solishtirishda harflarning registrlarini (ya'ni katta yoki kichikligini) farqlash kerak bo'lmasa, u holda **strcmp(satr1, satr2)** funksiyasidan foydalaniladi. Bu hol quyidagi dasturda o'z aksini topgan.

```
#include <iostream.h>
#include <string.h>
using namespace std;
int main()
{
    int j,i,s,n,m;
    char satr1[60]="Dasturlash ";char *kor1=satr1;
    char satr2[60]="dasturlash ";char *kor2=satr2;
    char satr3[60]="pragrammalash ";char *kor3=satr3;
    m=strcmp(kor1,kor2); cout<<" m="<<m;
    n=strcmp(kor2,kor3); cout<<" n="<<n<<"\n";
    system("pause");
    return 0;
}
```

Dastur natijasi **m=0**, **n=1** kabi bo'lishini tushunish qiyin emas.

3. Agar ikkita satrning faqat dastlabki **n** ta belgilarini solishtirish talab etilgan bo'lsa, **strncmp(satr1,satr2,n)** funksiyasidan foydalaniladi. Quyidagi dasturni tahlil qilib bunga ishonch hosil qilish mumkin.

```
#include <iostream.h>
#include <string.h>
using namespace std;
int main()
{
    int j,i,s,n,m;
    char satr1[60]="dasturlash ";char *kor1=satr1;
    char satr2[60]="dasturlash ";char *kor2=satr2;
    m=strncmp(kor1,kor2,6); cout<<" m="<<m;
    n=strncmp(kor1,kor2,7); cout<<" n="<<n<<"\n";
    system("pause");
}
```

return 0;

Dastur natijasi **m=0**, **n=32** kabi bo'ladi.

Misol. Ikkita satr berilgan. Ularning farq qiluvchi belgilarini topish dasturini tuzing.

```
#include <iostream.h>
#include <string.h>
using namespace std;
int main()
{
    int j, i, s, n, m; char k1[20]="", k2[20]="";
    char satr1[60]="Progoamalash tillarini o'rganamiz"; char *kor1=satr1;
    char satr2[60]="Programalash tillarini o'rganamiz"; char *kor2=satr2;
    for(i=1; i<=32; i++) {
        strncpy(k1, kor1, 1); strncpy(k2, kor2, 1); kor1+=1; kor2+=1;
        m=strcmp(k1, k2);
        if(n!=0) cout<<i<<"-chi belgisi har xil, ular: "<<k1<<" va "<<k2<<"\n";
    }
    system("pause");
    return 0;
}
```

Dastur natijasi quyida keltirilgan. Mustaqil tahlil qiling.

1-chi belgisi har xil, ular: p va P
 5-chi belgisi har xil, ular: o va r
 8-chi belgisi har xil, ular: a va f
 25-chi belgisi har xil, ular: R va r
 26-chi belgisi har xil, ular: c va g
 32-chi belgisi har xil, ular: Z va z

Satrdagi harflar registrini almashtirish funksiyalari

Agar satrdagi belgilari ichida harflar bo'lsa, ularni bosh harflardan kichik harflarga almashtirish uchun **strlwr(satr)** funksiyasidan, aksincha kichik harflarni bosh harflarga almashtirish uchun esa **strupr(satr)** funksiyasidan foydalaniladi. Masalan,


```
#include <iostream.h>
#include <string.h>
using namespace std;
int main()
{
    char satr2[30]="maktabda o'qiyman";
   strupr(satr2);
    cout<<satr2<<"\n";
    char satr3[30]="MEN TALABAMAN";
    strlwr(satr3);
    cout<<satr3<<"\n";
    system("pause");
    return 0;
}
```

Dastur natijasi quyidagicha bo'ladi
 MAKTABDA O'QIYMAN
 men talabaman
 kabi bo'ladi

Satrni teskari tartibda yozish funksiyasi

Berilgan satrni teskari tartibda yozish uchun **strrev(satr)** funksiyasidan foydalaniladi. Masalan,

```
#include <iostream.h>
#include <string.h>
using namespace std;
int main()
{
    char satr3[30]="TALABAMAN";
    strrev(satr3);
    cout<<satr3<<"\n";
    system("pause");
    return 0;
}
```

Dastur natijasi
 NAMABALAT
 kabi bo'ladi

Satrdagi belgini yoki uning qismini izlash funksiyalari

C++ da berilgan satrdan biror belgini yoki satrning biror qismini izlashga mo'ljallangan bir nechta funksiyalar mavjud.

1. Satrdan tayin belgini izlash uchun **strchr(satr, 'belgi')** funksiyasi aniqlangan bo'lib, agar belgini satr tarkibidan topsa, funksiya satrning o'sha belgisini va undan keyingi belgilarini qiymat sifatida oladi. Masalan,

```
#include <iostream>
#include <string.h>
using namespace std;
int main()
{
    char satr3[30]="134567389";
    char satr4[30]="gfdrcx98mjhk";
    char *satr,*satr1;
    satr=strchr(satr3,'3');cout<<satr<<"\n";
    satr1=strchr(satr4,'m');cout<<satr1<<"\n";
    system("pause");
    return 0;
}
```

Dastur natijasi quyidagicha bo'ladi
 34567389
 mjhk

2. Satrlardan birinchi farq qiluvchi belgini, ya'ni **satr1** dan **satr2** da bo'lmagan birinchi belgini topish uchun **strspn(satr1, satr2)** funksiyasidan foydalaniladi. Bu funksiya **satr1** ning har bir belgisini **satr2** dan izlab chiqadi. Qaysi belgisini topmasa, o'sha belgi indeksiga teng qiymatni oladi. Masalan, quyidagi dasturni tahlil qilish bilan bunga ishonch hosil qilish mumkin.

```
#include <iostream.h>
#include <string.h>
using namespace std;
int main()
{ int i,j,k;
    char satr1[30]="...";
```

```

j=strspn(satr1,satr2);
cout<<"satr1 ning satr2 dan farq qiluvchi birinchi belgisi
nomeri="<<j<<"\n";
system("pause");
return 0;
}

```

Dastur natijasi quyidagicha bo'ladi.

satr1 ning satr2 dan farq qiluvchi birinchi belgisi nomeri=10 kabi bo'ladi

3. Birinchi satrdan ikkinchi sairda bo'lgan birinchi belgini topish uchun **strcspn (satr1,satr2)** funksiyasidan foydalaniladi.

```

#include <iostream.h>
#include <string.h>
using namespace std;
int main()
{ int i,j,k;
  char satr1[30]="asalkning yechish";
  char satr2[30]="dfghjkhjlk nkdfiggjkk";
  j=strcspn(satr1,satr2);
  cout<<"satr1 ning satr2 da bo'lgan birinchi belgisi
nomeri="<<j<<"\n";
  system("pause");
  return 0;
}

```

Dastur natijasi quyidagicha bo'ladi.

satr1 ning satr2 da bo'lgan birinchi belgisi nomeri = 3

4. **Strpbrk (satr1, satr2)** funksiyasi **satr1** tarkibidagi belgilarni **satr2** dan izlaydi. Agar topsa, **satr1** ning topilgan belgidan o'ngdagi qismini qiymat sifatida oladi. Topmasa funksiya NULL qiymatni oladi.

```

#include <iostream.h>
#include <string.h>
using namespace std;
int main()
{ int i,j,k;
  char satr1[30]="asan5ng yechish";
  char satr2[30]="df3hjhkjmkdffijk";
  char*n;

```

```

n=stpbrk(satr1,satr2);
cout<<"satr1 ning satr2 da bo'lgan birinchi belgisidan o'ngdagi
qismi="<<n<<"\n";
system("pause");
return 0;
}

```

Dastur natijasi quyidagicha bo'ladi.

satr1ning satr2 da bo'lgan birinchi belgisidan o'ngdagi qismi = hish

5. Bir satr tarkibida ikkinchi satrning bor yoki yo'qligi aniqlash uchun **strstr(satr1, satr2)** funksiyasidan foydalaniladi. Bu funksiya **satr1** ning tartibidan **satr2** ni izlaydi. Agar topsa, **satr1**ning o'sha topilgan joyidan o'ng qismiga teng qiymatni oladi. Agar topmasa NULL qiymatni oladi.

Misol. Dastlabki bir nechta natural sonlar ro'yxati berilgan. Ixtiyoriy tanlangan son nomining ro'yxatda bor yoki yo'qligini aniqlash dasturini tuzing.

```

#include <iostream.h>
#include <string.h>
using namespace std;
int main()
{ int i,j,k;
  char satr1[]="bi, ikki, uch, to'rt, besh, olti";
  char satr2[30];
  char*n;
  cout<<"son nomini kiriting\n";cin>>satr2;
  n=strstr(satr1,satr2); cout<<"kipitilgan son ro'yxatda ";
  if (n==NULL) cout<<"yo'q\n"; else cout<<"bor\n";
  system("pause");
  return 0;
}

```

Dastur natijasini mustaqil tahlil qiling.

Satr turini o'zgartiruvchi funksiyalar

Ko'pincha sonlarni qayta ishlashni talab qiluvchi masalalarda sonni matematik almashtirishlardan foydalanib qayta ishlagandan ko'ra uni satr ko'rinishda qayta ishlab, keyin satrni songa aylantirib qo'yish ancha qulay bo'ladi. Buning uchun S++ dagi `<stdlib.h>` funksiyalar kutubxonasidagi quyidagi funksiyalardan foydalaniladi.

1. **atoi(satr)** funksiyasi. Bu funksiya turi butun (**int**) bo'lib, satr ko'rinishida berilgan sonni butun turga aylantiradi. Satrni to'liq songa aylantirish uchun satrning belgilari to'liq raqamlar bo'lishi kerak. Agar satrning boshidagi bir nechta belgilarigina raqamlardan iborat bo'lsa, **atoi()** funksiyasi faqat shu qismini songa aylantiradi va qolgan qismini tashlab yuboradi. Agar satr raqam bilan boshlanmagan bo'lsa, funksiya 0(nol) ga teng qiymatni oladi. Yuqorida keltirilgan fikrlarni quyidagi dastur misolida tahlil qiling.

```
#include <iostream.h>
#include <stdlib.h>
using namespace std;
int main()
{
    int i,j,k,n,m;
    char satr1[30]="215625", satr2[30]="25";
    char satr3[30]="2154254254698",
    satr4[30]="7458TALA215AN", satr5[]="assalom";
    k=atoi(satr1); n=atoi(satr2); i=atoi(satr3); j=atoi(satr4);
    m=atoi(satr5);
    cout<<"i="<<i<<"\n";
    cout<<"k="<<k<<"\n";
    cout<<"m="<<m<<"\n";
    system("pause");
    return 0;
}
Dastur natijasi quyidagicha bo'ladi
i=-1819327894
j=7458
k=215625
n=25
m=0
```

Satrni songa aylantirishda son tur chegarasidan chiqib ketsa C++ kutilmagan natijani chop etadi. Masalan, dastur **i** parametri uchun shunday qiymat tanlangan

2. **atof(satr)** funksiyasi. Bu funksiya ham **atoi()** funksiyasi kabi ishlaydi. Faqat bu funksiyaning qiymati **long** turiga mansub bo'ladi. Bundan tashqari funksiyaga **float** yoki **double** turi berilsa, satr ko'rinishidagi sonni **double** ko'rinishidagi, ya'ni sonning mantissasi va tartibini ko'tatish bilan yoki o'nli kasr ko'rinishiga o'tkazadi. Agar songa aylantirilayotgan satr chegaradan chiqib ketsa, funksiya kutilmagan qiymatni oladi.

```
#include <iostream.h>
#include <stdlib.h>
using namespace std;
int main()
{
    int k,n,m; double i,j;
    char satr1[]="3.14D15E+2";
    char satr3[30]="21.34254254E15", satr4[30]="745.8215";
    n=atoi(satr1); i=atof(satr3); j=atof(satr4);
    cout<<"i="<<i<<"\n";
    cout<<"j="<<j<<"\n";
    system("pause");
    return 0;
}
```

Dastur natijasi quyidagicha bo'ladi
i=2.13425e+016
j=745.822
n=3

3. **itoa(n,satr,m)** funksiyasi. Bu funksiya **int** turiga mansub bo'lgan **n** sonini **m** lik sanoq sistemasiga o'tkazib, satr turiga o'tkazadi va hosil bo'lgan qiymatni **satr** massivga beradi.

itoa(n,satr,m) funksiyasi vazifasi ham **itoa()** funksiyasi vazifasi kabi. Faqat **itoa()** funksiyasi **long** turiga mansub sonlarni ham satrga aylantiradi. Quyidagi dasturni tahlil qilib, bunga ishonch hosil qilish mumkin.

```
#include <iostream.h>
#include <stdlib.h>
```

```

#include <string.h>
using namespace std;
int main()
{
    int n=3245,m; long int i,j=568324557; long double k=5.6e4;
    char satr1[20],satr2[20],
    satr3[20],satr4[20],satr5[20],satr6[20],*satr7;
    itoa(n,satr1,2); itoa(n,satr2,8);itoa(n,satr3,16);itoa(n,satr4,10);
    cout<<"satr1="<<satr1<<"\nsatr2="<<satr2<<"\nsatr3="<<satr3<<"\nsatr4="<<satr4<<"\n";
    ltoa(j,satr1,2); ltoa(k,satr2,8);ltoa(j,satr3,16);ltoa(j,satr4,10);
    cout<<"satr1="<<satr1<<"\nsatr2="<<satr2<<"\nsatr3="<<satr3<<"\nsatr4="<<satr4<<"\n";
    itoa(n,satr5,10);ltoa(j,satr6,10);satr7=strstr(satr6,satr5);cout<<"
    satr7="<<satr7<<"\n";
    system("pause");
    return 0;
}
Dastur natijasi quyidagicha bo'ladi.
satr1=110010101101
satr2=6255
satr3=cad
satr4=3245
satr1=100001110111111111000111001101
satr2=155300
satr3=21dfflcd
satr4=568324557
satr7=324557

```

4. **gevt (son,raqamlar soni, satr)** funksiyasi. Bu funksiya **float** yoki **doudle** turidan sonni satrga aylantiradi. Bunda son **float** yoki **double** turiga mansub son bo'lilshi kerak, **raqamlar soni** satrga aylantirilgan sonning raqamlar soni. Agar sonning raqamlari soni ko'rsatilgan sondan ortiq bo'lsa, quyidagi ikki hol bo'lishi mumkin.

- 1) Butun son berilgan bo'lsa, u **nEm** ko'rinishida ifodalanadi;
- 2) O'nli kasar ko'rinishida bo'lsa, ortiqcha raqamlari yaxlitlanadi.

Quyidagi dasturda bu imkoniyatlar o'z aksini topgan

```

#include <iostream.h>
#include <stdlib.h>
using namespace std;
int main()
{
    int n=4; double i=4562, m=1234567,k=12.455476;
    char satr[20];
    gevt(i,n,satr);cout<<"i ning satr korinishi="<<satr<<"\n";
    gevt(m,n,satr);cout<<"m ning satr korinishi="<<satr<<"\n";
    gevt(k,n,satr);cout<<"k ning satr korinishi="<<satr<<"\n";
    system("pause");
    return 0;
}

```

Dastur natijasi quyidagicha bo'ladi.

i ning satr korinishi=4562
m ning satr korinishi=1.235e+006
k ning satr korinishi=12.46

5.3. String turidagi satrlar va ularni qayta ishlash funksiyalari

Satrlar bilan ishlash jarayonida **string** turidan foydalanish satrlarni qayta ishlashda katta imkoniyatlar yaratadi. Bu turga mansub o'zgaruvchilarning qiymatlari **'0'** belgisi bilan tugamaydi. Demak, satr uzunligi undagi belgilar soniga teng bo'ladi.

String turiga mansub o'zgaruvchilarning qiymati ikki yoki undan ortiq belgilar ketma-ketligidan iborat bo'lsa, qo'shtirnoq ichida olib yoziladi. Agar bitta belgidan iborat bo'lsa tutuq belgisi ichiga olib yozilishi ham mumkin

Srting turiga mansub o'zgaruvchilarga boshlang'ich qiymatlarini quyidagi usullarda berish (initsializatsiyalash)mumkin:

string x1="kitob", x2 ("daftar");
string x3 (x1), x4=x1, x5='m', x6='';

string turli o'zgaruvchining **n**- belgisidan nusxa olish uchun **s.at(n)** funksiyasidan foydalaniladi. Bunda **s** satrli o'zgaruvchining nomi. Shuningdek, bu amalni bajarish uchun o'zgaruvchi nomidan keyin kvadrat qavs ichida belgining tartib nomerini ko'rsatish ham mumkin. Bir nechta satrlarni ulash uchun **(+)** belgisidan foydalaniladi. Masalan, quyidagi dasturni tahlil qiling.


```

#include <iostream.h>
#include <string.h>
using namespace std;
int main()
{
    long int j,i,s,n,m;
    string s1="men",s2(" axborotlashtirish"),s3(" "+s1),s4=s2,s5,s6;
    s5='M';s6=s1+s2+s3+s4+" "+s5;
    cout<<s1<<s2<<s3<<s4<<"\n";
    cout<<' '+s6<<"\n";
    cout<<"s2.at(3)=="<<s2.at(3)<<"\n";
    n=strlen("axborotlashtirish");cout<<"n=="<<n<<"\n";
    s=0;for(i=1;i<=n;i++) if(s2.at(i)=='o')
    s+=1;cout<<"s=="<<s<<"\n";
    cout<<"s2[10]=="<<s2[10]<<"\n";
    system("pause");
    return 0;
}

```

Dastur natijasi quyidagicha bo'ladi:

```

men axborotlashtirish men axborotlashtirish
men axborotlashtirish men axborotlashtirish M
s2.at(3)=b
n=17
s=2
s2[10]=s

```

Endi satrlarni qayta ishlashda foydalaniladigan asosiy funksiyalar bilan tanishamiz:

1. **satr2. assign (satr1, n, m)** funksiyasi

Funksiya **satr1** ni yoki uning qismini **satr2** ga nusxalaydi. Bu funksiyaning quyidagi ko'rinishlaridan foydalaniladi.

a) **satr2. assign (satr1);** ko'rinishi. Bu holda funksiya **satr1** ni to'liq **satr2** ga nusxalaydi. Agar **satr2** initsializatsiyalangan bo'lsa ham uning oldingi qiymati unutiladi.

b) **satr2. assign (satr1, n, m);** ko'rinishi. Ushbu variantida funksiya **satr1** ning **n**- belgisidan keyingi **m** ta belgisini **satr2** ga nusxalaydi, aniqrog'i **satr2** ning ustidan yozib yuboradi, **n**-belgisi kirmaydi.

c) **satr2. assign (satr1, n);** ko'rinishi. Bu holda **satr1** char turli satrli massiv bo'lishi kerak. Funksiya char turli massivni **string** turiga aylantirib, uning dastlabki **n** ta belgisini **satr2** ga nusxalaydi. Barcha variantlaridan foydalanish usullari quyidagi dasturda o'z aksini topgan.

```

#include <iostream.h>
#include <string.h>
using namespace std;
int main()
{
    int j,i,s,n,m;
    string s1="satr nusxasini olish",s3,s4,s5,s7,s2="matn";
    char s6[]="assalom"; s2.assign(s1);cout<<"s2=="<<s2<<"\n";
    s3.assign(s1);cout<<"s3=="<<s3<<"\n";
    s4.assign(s1,4,6);cout<<"s4=="<<s4<<"\n";
    s5.assign(s6,4);cout<<"s5=="<<s5<<"\n";
    s7.assign("maktabdosh",6);cout<<"s7=="<<s7<<"\n";
    system("pause");
    return 0;
}

```

Dastur natijasi quyidagicha bo'ladi

```

s2=satr nusxasini olish
s3=satr nusxasini olish
s4= nusxa
s5=assa
s7=maktab

```

Satrlarni ulash funksiyasi

Funksiyaning umumiy ko'rinishi

satr2. append (satr1, n, m); (**append** –ulash, qo'shish demakdir) kabi bo'lib, xuddi nusxalash funksiyasida bo'lgani kabi uning quyidagi ko'rinishlaridan foydalaniladi.

a) **satr2. append (satr1);** ko'rinishidan foydalanilganda **satr2** ning oxiridan **satr1** ulanadi. Agar **satr2** initsializatsiya qilinmagan bo'lsa, **satr2** o'zgaruvchi **satr1** o'zgaruvchining qiymatini oladi.

b) **satr2. append (satr1, n, m);** ko'rinishidan foydalanilganda **satr1** ning **n** –belgisidan keyingi **m** ta belgisi **satr2** ning oxiridan ulanadi.

c) `satr2.append(satr1,n)`; ko'rinishidan foydalanilganda `satr1` cha turli massiv bo'lishi kerak. Funksiya `char` turli `satr1` ni `string` turla aylantirib, uning dastlabki `n` ta belgisini `satr2` oxiridan ulydi.

Funksiyaning vazifasini o'zlashtirish uchun quyidagi dastur uning natijasini tahlil qiling.

```
#include <iostream.h>
#include <string.h>
using namespace std;
int main()
{ int j,i,s,n,m;
  string s1="matnni ulash",s3,s4="bo'lakni",s5,s7="bizning",s2="matnga ";
  char s6[]="assalom"; s2.append(s1);cout<<"s2="<<s2<<"\n";
  s3.append(s1);cout<<"s3="<<s3<<"\n";
  s4.append(s1,6,6);cout<<"s4="<<s4<<"\n";
  s5.append(s6,4);cout<<"s5="<<s5<<"\n";
  s7.append("maktabdosh",6);cout<<"s7="<<s7<<"\n";
  system("pause");
  return 0;
}
Dastur natijasi quyidagicha bo'ladi
s2=matnga matnni ulash
s3=matnni ulash
s4=bo'lakni ulash
s5=assa
s7=bizning maktab
```

Satr tarkibiga satr joylashtirish funksiyasi

Funksiyaning umumiy ko'rinishi quyidagicha:

`satr2.insert(n,satr1,m,k)`;

Funksiyaning quyidagi ko'rinishlaridan foydalaniladi:

a) `satr2.insert(n,satr1)`; ko'rinishidan foydalanilganda funksiya `satr2` ning `n`-o'rnidan boshlab `satr1` ni joylashtiradi.

b) `satr2.insert(n,satr1,m,k)`; ko'rinishidan foydalanilganda funksiya `satr2` ning `n`- belgisidan keyin `satr1` ning `m`-belgisidan keyingi `k`-ta belgisini joylashtiradi. `m` - belgisi kirmaydi.

c) `satr2.insert(n,satr1,m)`; ko'rinishidan foydalanilganda funksiya `satr2` ning `n`-belgisidan keyin `satr1` ning dastlabki `m` ta belgisini joylashtiradi. Masalan,

```
#include <iostream.h>
#include <string.h>
using namespace std;
int main()
{ int j,i,s,n,m;
  string s1=" matnni joylash",s3,s4="bo'lakni",s5="tezroq",s7="bizning",s2="matnga ";
  char s6[]="bo'ladi"; s2.insert(6,s1);cout<<"s2="<<s2<<"\n";
  s4.insert(6,s1,10,4);cout<<"s4="<<s4<<"\n";
  s5.insert(3,s6,4);cout<<"s5="<<s5<<"\n";
  s7.insert(3," maktabdosh",7);cout<<"s7="<<s7<<"\n";
  system("pause");
  return 0;
}
```

Dastur natijasi quyidagicha bo'ladi
s2=matnga matnni joylash
s4=bo'lakylasni
s5=tezbo'lroq
s7=biz maktabning

Satr yoki uning qismini o'chirish funksiyasi

Satr yoki uning qismini o'chirish uchun `satr.erase(n,m)`; (erase-o'chirish) funksiyasidan foydalaniladi. Bu funksiyaning quyidagicha ko'rinishlari mavjud:

a) `satr1.erase()`; ko'rinishidan foydalanilganda `satr1` ning barcha belgilari o'chiriladi.

b) `satr1.erase(n,m)`; ko'rinishidan foydalanilganda `satr1` ning `n`-belgisidan keyingi `m` ta belgisi o'chirilib, qolgan qismi `satr1`ga beriladi.

s) `satr1.erase(n)`; ko'rinishidan foydalanilganda `satr1` ning dastlabki `n` ta belgisi qoldirilib, qolgan qismi o'chiriladi.

```
#include <iostream.h>
```

```
#include <string.h>
using namespace std;
int main()
{ string s1="matn bo'lagini o'chirish",s2="bugun dars bo'ladi";
  string s3="informatika";
  s1.erase();cout<<"s1="<<s1<<"\n";
  s2.erase(3,4);cout<<"s2="<<s2<<"\n";
  s3.erase(5);cout<<"s3="<<s3<<"\n";
  system("pause");
  return 0;
}
```

Dastur natijasi quyidagicha bo'ladi
s1=namoyishi
s2=bugars bo'ladi
s3=infor

Matn qismlarini almashtirish funksiyasi

Bu amalni bajarish uchun `satr1.replace(n,m,satr2,i,j)` funksiyasidan foydalaniladi. Funksiyaning quyidagicha ko'rinishlari mavjud.

a) `satr1.replace(n,m,satr2);` ko'rinishidan foydalanilganda `satr1` ning `n`-belgisidan keyingi `m` ta belgisi `satr2` bilan almashtiriladi.

b) `satr1.replace(n,m,satr2,i,j);` ko'rinishidan foydalanilganda `satr1` ning `n`-belgisidan keyingi `m` ta belgisi `satr2` ning `i`-belgisidan keyingi `j` ta belgisi bilan almashtiriladi.

s) `satr1.replace(n,m,satr2,k);` ko'rinishidan foydalanilganda `satr1` ning `n`-belgisidan keyingi `m` ta belgisi `satr2` char turli massivning dastlabki `k` ta belgisidan tashkil topgan qismini `string` turiga aylantirib almashtiradi. Quyidagi dasturni tahlil qilish bilan bu fikrlarga ishonch hosil qilish mumkin.

```
#include <iostream.h>
#include <string.h>
using namespace std;
int main()
{ char s4[]="bo'lmaydi";
  string s1="matn bo'lagini almashtirish",s2="bugun dars bo'ladi";
```

```
string s3="nima";
s1.replace(5,9,"qismini");cout<<"s1="<<s1<<"\n";
s1.replace(4,8,s3,0,2);cout<<"s1="<<s1<<"\n";
s2.replace(11,7,s4,9);cout<<"s2="<<s2<<"\n";
system("pause");
return 0;
}
```

Dastur natijasi quyidagicha bo'ladi
s1=matn qismini almashtirish
s1=matnni almashtirish
s2=bugun dars bo'lmaydi

Satrning biror qismidan nusxa olish funksiyasi

Bu amalni bajarish uchun `satr1.substr(n,m)` funksiyasidan foydalaniladi. Funksiyaning quyidagicha ko'rinishlari mavjud.

a) `satr1.substr(n,m)`. Bunda `satr1` qismidan nusxa olinayotgan `satr` bo'lib, funksiya shu satrning `n`-belgisidan keyingi `m` ta belgisidan nusxa oladi;

b) `satr1.substr(n)` ko'rinishida foydalanilsa `satr1` satrning `n`-belgisidan oxirigacha nusxa olinadi;

c) `satr1.substr()` ko'rinishidan foydalanilganda esa `satr1` dan to'liq nusxa olinadi. Bu holda funksiyaning qiymati qandaydir `string` turli o'zgaruvchiga berilishi kerak.

```
#include <iostream.h>
#include <string.h>
using namespace std;
int main()
{ string s1="matn bo'lagini almashtirish";
  string s5,s6,s7,s8;
  s5=s1.substr(5,5); cout<<"s5="<<s5<<"\n";
  s7=s1.substr(6);cout<<"s7="<<s7<<"\n";
  s8=s1.substr();cout<<"s8="<<s8<<"\n";
  system("pause");
  return 0;
}
```

Dastur natilasi quyidagicha bo'ladi

```
s5=bo'la
s7=o'lagini almashtirish
s8=matn bo'lagini almashtirish
```

Satr qismlarini izlash funksiyalari

Satr qismini izlashga mo'ljallangan bir nechta funksiyalar aniqlangan bo'lib, quyida ularning asosiylari bilan tanishamiz.

a) `n = satr1.find (satr2,n)` funksiyasi.

Funksiya `satr2` ni `satr1` ning `n`-belgisidan keyin izlaydi. Agar topa `n` o'sha topilgan bo'lakdan oldingi belgi nomerini qiymat sifatida oladi. Agar `satr2` bo'lak `satr1` ning tarkibida bir necha bor uchrasa ham funksiya faqat birinchi uchraganigacha izlaydi. Agar `n` parametri ko'rsatilmasa `satr2` bo'lakning o'zini `satr1` ning boshidan boshlab izlaydi. Shuningdek funksiya bitta belgini ham izlaydi. `find` so'zining tarjimai uchrash, uchratish demakdir.

b) `n=satr1.rfind (satr2, n);`

funksiyasi vazifasi `find` funksiyasining vazifasi kabi. Faqat `satr2` ni `satr1` ning boshidan emas oxiridan izlaydi.

```
#include <string.h>
using namespace std;
int main()
{ string s1="matematika va informatika",s2="mat",s3="ti";
  int s5,s6,s7,s8,s9,s10,s11,s12;
  s5=s1.find(s2); cout<<"s5="<<s5;
  s6=s1.find('a'); cout<<" s6="<<s6;
  s7=s1.find(s2,5); cout<<" s7="<<s7<<"\n";
  s8=s1.rfind(s2,8); cout<<"s8="<<s8;
  s9=s1.rfind('a',15); cout<<" s9="<<s9;
  s10=s1.rfind(s3); cout<<" s10="<<s10<<"\n";
  system("pause");
  return 0;
}
```

Dastur natijasi quyidagicha bo'ladi

```
s5=4   s6=1   s7=19
s8=4   s9=12  s10=21
```

c) `n=satr1.find-first-of ( satr2, n);` funksiyasi `satr2` ning ixtiyoriy belgisini `satr1` ning `n` - belgisidan keyin izlaydi. Agar topa olsa , `n` o'sha belgidan oldingi belgining nomerini qiymat sifatida oladi.

d) `n= satr1.find-last-of (satr2, n);` funksiyasining vazifasi ham `find-first-of ( )` funksiyasini kabi faqat `satr2` tarkibidagi ixtiyoriy belgini o'ng qismidan izlaydi.

```
#include <iostream.h>
#include <string.h>
using namespace std;
int main()
{ string s1="matematika va informatika",s2="hoja",s3="pf";
  int s5,s6,s7,s8,s9,s10,s11,s12;
  s5=s1.find_first_of(s2); cout<<"s5="<<s5;
  s6=s1.find_first_of('r'); cout<<" s6="<<s6;
  s7=s1.find_first_of(s2,11); cout<<" s7="<<s7<<"\n";
  s8=s1.find_last_of(s2); cout<<"s8="<<s8;
  s9=s1.find_last_of('a',15); cout<<" s9="<<s9;
  s10=s1.find_last_of("ti"); cout<<" s10="<<s10<<"\n";
  system("pause");
  return 0;
}
```

Dastur natijasi quyidagicha bo'ladi

```
s5=1   s6=18   s7=12
s8=24  s9=12   s10=22
```

e) `n=satr1.find-first-not-of ( satr2, n);` funksiyasi `satr1` ning tarkibidan `satr2` ning chapdan bironta ham belgisi bo'lmagan qismini izlaydi. Agar topa olsa , `n` o'sha belgidan oldingi belgining tartib nomeri qiymatini oladi .

f) `n= satr1. find-last-not-of ( satr2, m);` funksiyasi ham shu amalni bajaradi. Faqat izlash o'ng tomondan amalga oshiriladi. Har ikkala funksiyaning bajaradigan vazifasini quyidagi dasturni tahlil qilish yo'li bilan tushunish qiyin emas.

```
#include <iostream.h>
#include <string.h>
using namespace std;
int main()
```



```
{string s1="ahojamatika va informatikt",s2="ahok",s3="pt";
  int s5,s6,s7,s8,s9,s10,s11,s12;
  s5=s1.find_first_not_of(s2); cout<<"s5="<<s5;
  s6=s1.find_first_not_of('r'); cout<<" s6="<<s6;
  s7=s1.find_first_not_of(s2,11); cout<<" s7="<<s7<<"\n";
  s8=s1.find_last_not_of(s3); cout<<"s8="<<s8;
  s9=s1.find_last_not_of('a',15); cout<<" s9="<<s9;
  s10=s1.find_last_not_of("ti"); cout<<" s10="<<s10<<"\n";
  system("pause");
  return 0;
}
```

Dastur natijasi quyidagicha bo'ladi

```
s5=3   s6=0   s7=11
s8=24  s9=15  s10=24
```

Satrlarni solishtirish funksiyalari

Satrlarni solishtirish funksiyalari satrdagi belgilarning kodlar bo'yicha solishtiradi. Shuning uchun dastlab klavitudagi kodlar foydalaniladigan belgilarning kodlarini aniqlash dasturini va uning natijasini keltiramiz.

```
#include <iostream.h>
#include <string.h>
using namespace std;
int main()
{ int x,y,z,n,m,t,i;
  char
a[40]="ABCDEFGHIJKLMNOPQRSTUVWXYZ",b[30]="abcdefghijklmnopqrstuvwxyz";
  char c[40]="0123456789 !@#$%&*()+-=,?^<>V,.;:";
  for(i=0;i<=32;i++) {x=(int)a[i];y=(int)b[i];z=(int)c[i];
  if(i<=25){cout<<" "<<a[i]<<" ning kodi="<<x<<" "<<b[i]<<"
ning kodi="<<y;
  cout<<" "<<c[i]<<" ning kodi="<<z<<"\n";} else cout<<"
"<<c[i]<<" ning kodi="<<z<<"\n";}
  system("pause");
  return 0;
```

Dastur natijasi quyidagicha bo'ladi

A ning kodi=65	a ning kodi=97	0 ning kodi=48	^ ning
B ning kodi=66	b ning kodi=98	1 ning kodi=49	<ning
C ning kodi=67	c ning kodi=99	2 ning kodi=50	>ning
D ning kodi=68	d ning kodi=100	3 ning kodi=51	/ ning
E ning kodi=69	e ning kodi=101	4 ning kodi=52	, ning
F ning kodi=70	f ning kodi=102	5 ning kodi=53	. ning
G ning kodi=71	g ning kodi=103	6 ning kodi=54	: ning
H ning kodi=72	h ning kodi=104	7 ning kodi=55	
I ning kodi=73	i ning kodi=105	8 ning kodi=56	
J ning kodi=74	j ning kodi=106	9 ning kodi=57	
K ning kodi=75	k ning kodi=107	probel ning kodi=32	
L ning kodi=76	l ning kodi=108	! ning kodi=33	
M ning kodi=77	m ning kodi=109	@ ning kodi=64	
N ning kodi=78	n ning kodi=110	# ning kodi=35	
O ning kodi=79	o ning kodi=111	\$ ning kodi=36	
P ning kodi=80	p ning kodi=112	% ning kodi=37	
Q ning kodi=81	q ning kodi=113	& ning kodi=38	
R ning kodi=82	r ning kodi=114	* ning kodi=42	
S ning kodi=83	s ning kodi=115	(ning kodi=40	
T ning kodi=84	t ning kodi=116	) ning kodi=41	
U ning kodi=85	u ning kodi=117	+ ning kodi=43	
V ning kodi=86	v ning kodi=118	- ning kodi=45	
W ning kodi=87	w ning kodi=119	= ning kodi=61	
X ning kodi=88	x ning kodi=120	; ning kodi=59	
Y ning kodi=89	y ning kodi=121	, ning kodi=44	
Z ning kodi=90	z ning kodi=122	? ning kodi=63	

Satrlarni solishtirishda compare funksiyasining quyidagi bir nechta ko'rinishlaridan foydalaniladi.

```
a)s=satr1.compare(satr2);
```

Bunda `satr1` va `satr2` lar solishtiriladi. Agar

`satr1 > satr2` bo'lsa `s > 0`

`satr1 = satr2` bo'lsa `s = 0`

`satr1 < satr2` bo'lsa `s < 0`

bo'ladi. Satrlarni solishtirishda, ularning mos nomerlaridagi belgilarining ASCII kodlari solishtirilib ketilaveradi. Qaysidir nomerli belgilari bir-biridan farq qilsa va ulardan qaysi birining kodi katta bo'lsa, o'sha satr katta hisoblanadi.

b) `s=satr1.compare(n,m,satr2)`; ko'rinishidan foydalanilganda `satr1` ning `n`-nomeridan keyingi `m` ta belgilari ketma-ketligi `satr2` bilan solishtiriladi.

c) `s=satr1.compare(n,m,satr2,i,j)`; ko'rinishidan foydalanilganda `satr1` ning `n`-belgisidan keyingi `m` ta belgilari ketma-ketligi `satr2` ning `i`-belgisidan keyingi `j` ta belgilari ketma-ketligi bilan solishtiriladi.

Solishtirish funksiyasining imkoniyatlarini quyidagi dastur misolida namoyish qilamiz.

```
#include <iostream.h>
#include <string.h>
using namespace std;
int main()
{ int x,y,z,n,m,t,s5,s6,s7,s8;
  string s1="daraja ko'rsatkichida",s2="ABCDEF",s3="ABCD";
  string s4="daraja ko'rsat",s9="ab",s10="aa3-";
  s5=s2.compare(s3); cout<<"s5="<<s5;
  s6=s2.compare(s1); cout<<" s6="<<s6;
  s8=s9.compare(s10); cout<<" s8="<<s8;
  s7=s1.compare(s4); cout<<" s7="<<s7<<"\n";
  x=s1.compare(0,14,s4); cout<<" x="<<x;
  y=s2.compare(0,5,s3); cout<<" y="<<y;
  z=s1.compare(1,4,s4,1,4); cout<<" z="<<z;
  n=s1.compare(3,4,s4,2,4); cout<<" n="<<n;
  m=s1.compare(3,5,s4,3,4); cout<<" m="<<m<<"\n";
  system("pause");
  return 0;
}
```

Dastur natijasi quyidagicha bo'ladi

```
s5=2   s6=-1   s8=1   s7=7
x=0    y=1     z=0    n=-1   m=1
```

Satr xossalarini aniqlash funksiyalari

Satrning xossalarini aniqlash uchun quyidagi funksiyalardan foydalaniladi.

`x=satr.size()`; funksiyasi `satr` ning o'lchamini aniqlaydi.

`x=satr.length()`; funksiyasi `satr` ning belgilari sonini aniqlaydi.

`x=satr.max_size()`; funksiyasi `satr` ning maksimal o'lchamini aniqlaydi.

`x=satr.capacity()`; funksiyasi `satr` ning xotirada egallagan joyi hajmini aniqlaydi.

`x=satr.empty()`; funksiyasi qiymati `bool` turli bo'lib, agar `satr` bo'sh bo'lsa, funksiya qiymati `true`, aks holda `false` bo'ladi. Quyidagi dasturni va uning natijalarini tahlil qiling.

```
#include <iostream.h>
#include <string.h>
using namespace std;
int main()
{ int z,n,m,t,s6,s7,s8;bool x,y;float s5;
  string s1="daraja ko'rsatkichida",s2="ABCDEF",s3="ABCD";
  string s4="daraja ko'rsat",s9="ab87,+|",s10="";
  s5=s9.size(); cout<<"s5="<<s5;
  s6=s9.length(); cout<<" s6="<<s6;
  s8=s10.max_size(); cout<<" s8="<<s8;
  s7=s9.capacity(); cout<<" s7="<<s7<<"\n";
  x=s9.empty(); cout<<" x="<<x;
  y=s10.empty(); cout<<" y="<<y<<"\n";
  system("pause");
  return 0;
}
```

Dastur natijasi quyidagicha bo'ladi

```
s5=8   s6=8   s8=1073741820   s7=8
x=0    y=1
```

Quyidagi savollarga javob bering

1. Dasturlashda satr nima, belgi nima? Ularning turlari qanday bo'ladi?
2. Satrli massivlar qanday va qayerda e'lon qilinadi?
3. `char sum[15][30]`; satrli massiv elementlari qanday joylashadi?

4. `char m[15][30]`; massiv e'lonida m harfi, 15 va 30 sonlari nima bildiradi?
5. Satrli massiv elementlari qiymatlarini kiritish operatori qaysi?
6. `getline(satr,k)`; funksiyasi vazifasini tushuntiring.
7. `216+44` va `'216'+44` ifodalar natijasi qanday bo'ladi va nima uchun?
8. Satr uzunligini aniqlash funksiyalari qaysi?
9. Satrlarni nusxalashda foydalaniladigan barcha funksiyalarni yozing va tushuntiring?
10. Satrlarni ulashda foydalaniladigan barcha funksiyalarni yozing va tushuntiring?
11. Satrlarni taqqoslashda foydalaniladigan barcha funksiyalarni yozing va tushuntiring?
12. Satrdagi harflar registrini almashtirish funksiyalarini yozing va tushuntiring?
13. Satrni teskari tartibda yozish funksiyasini tushuntiring?
14. Satrdagi belgi yoki uning qismini izlash foydalaniladigan barcha funksiyalari yozing va tushuntiring?
15. Satr turini o'zgartirishda foydalaniladigan barcha funksiyalarni yozing va tushuntiring?
16. `string` turli o'zgaruvchilar bilan satrli massivlar o'rtasida qanday farqlar bor?
17. `string` turli matnlarni qayta ishlashga mo'ljallangan qanday funksiyalarini bilasiz? Ularni tushuntiring.
18. 1) `"gdf3546^.,,"`, 2) `"9876543 %$#@!"` satrlar uzunligini aniqlang.
19. Satrning o'zini yoki uning biror qismini o'chirish funksiyasi qaysi va u qanday bajariladi?
20. Satrni songa va sonni satrga aylantirish funksiyalarini yozib, tushuntiring.

Quyidagi masalalarning dasturlarini tuzing

1. Berilgan matnning har bir belgisidan keyin bittadan bo'sh joy qoldiring.
2. Matndagi a harflarini o harflariga, n harflarini p harfiga almashtiring.
3. Matndagi o va a harflarini o'chiring.
4. Matndagi "sen" yozuvini "siz" yozuviga almashtiring.

5. "lola", "demak", "talabalari", so'zlari berilgan "maktab talalari" jumlasini hosil qiling.
6. Berilgan so'zni teskari tartibda yozing. Masalan kino-onik.
7. 1 dan 100 gacha bo'lgan 17 ga karrali sonlarni ketma-ket yozishdan hosil bo'lgan matnni chop etish dasturini tuzing.
8. Berilgan so'zning toq o'rindagi belgilarini o'chiring.
9. Berilgan so'zning nechanchi o'rinlardagi belgilari 'a' ekanligini aniqlang.
10. "Integral" so'zi berilgan. Uning kerakli qismlaridan nusxa olib, "agent" so'zini hosil qiling.
11. "Universitet" so'zi berilgan. Uni krill alifbesiga o'tkazish dasturini tuzing.
12. Ixtiyoriy yetti xonali son satr ko'rinishida kiritiladi. Shu sonning raqamlari yig'indisini toping.
13. Matn belgilarini ASCII kodlari bilan almashtirishdan hosil bo'lgan matnni chop eting.
14. n uch xonali sondan m besh xonali songacha bo'lgan barcha natural sonlar yozib chiqilsa, necha xonali son hosil bo'lishini aniqlang.
15. Matn har bir belgisidan keyin ikkita belgi yozish usulida kodlangan bo'lsa, matnning o'zini tiklang.
16. s1 satr s2 satrda n marta uchrasa, n ning qiymatini aniqlang.

VI BOB. FUNKSIYA TUSHUNCHASI

Masalaning dasturi yetarlicha katta hajmga ega bo'lsa, uni bir butunligicha tasavvur qilib, uning mayda qismlari o'rtasidagi bog'lanishni sezish va ular o'rtasidagi bog'lanishni tashkil qilish jarayoni ko'p mehnat, aqliy zo'riqish va vaqt talab qiladi.

Tajribali dasturchilar yetarlicha murakkab masalaning dasturlarini ham aqliy zo'riqishlarsiz tuzish texnologiyalari va usullarini ishlab chiqishgan.

Shunday usullardan biri dasturlashga tuzimli (sistemali) yondashish usulidir. Dasturlashga tizimli yondashishdan quyidagi hollarda foyalaniladi.

Birinchidan, masalani mumkin bo'lgan mayda qismlarga, agar mumkin bo'lsa mayda qismlarni ham yana mayda qismlarga bo'lib ketaveramiz, toki o'sha qismlar hamma tushunadigan oddiy amallardan iborat bo'lib qolsin. Shundan so'ng o'sha qismlarning dasturlarini alohida-alohida tuzib, asosiy (bosh) dasturga biriktiriladi va yaxlit masalani yechish dasturi paydo bo'ladi.

Masalan, tanlangan funktsiyani tekshirish dasturini tuzish talab etilgan bo'lsin. Bunda quyidagi amallarni bajaruvchi dasturlar tuzib olinadi:

- funktsiyaning aniqlanish sohasini topish;
- funktsiyaning qiymatlar sohasini topish;
- funktsiyaning nollarini aniqlash;
- funktsiyaning OY o'qini kesish nuqtalarini aniqlash;
- funktsiyaning juft-toqligini aniqlash;
- funktsiyaning davrini aniqlash;
- funktsiyaning musbat oraliqlarini aniqlash;
- funktsiyaning manfiy oraliqlarini aniqlash;
- funktsiyaning o'sish oraliqlarini aniqlash;
- funktsiyaning kamayish oraliqlarini aniqlash;
- funktsiyaning ekstremum nuqtalarini aniqlash;
- funktsiyaning ekstremum qiymatlarini aniqlash;
- funktsiyaning grafigini chizish;

amallarini bajaruvchi dasturlar tuzib olinadi. Agar zarurat bo'lsa, bu dasturlarning har biri yana mayda dasturlarga bo'linib, har birining dasturlari tuziladi va «Funktsiyani tekshirish» nomli asosiy dasturga birlashtiriladi.

Ikkinchidan, bitta masalaning dasturini tuzishda bitta masalaning dasturidan bir necha marta foydalanishga to'g'ri keladi. Masalan, binomial koeffitsientlarni aniqlashda, ya'ni,

$$C_n^m = \frac{n!}{m!(n-m)!}$$

ifodaning qiymatini hisoblashda faktorialni hisoblash dasturidan uch marta foydalanish kerak va har safar bu dasturni qayta-qayta tuzib yurmasdan uni bir marta tuzib, kerak bo'lganda uch marta murojaat tashkil qilish kerak. Natijada dasturning hajmi kichiklashadi, uni tushunish ham oson bo'ladi. Bunday vaziyatlarda o'sha mayda dasturlarni funktsiya ko'rinishida yozib, kerak bo'lganda unga xohlagancha marta murojaat qilish yo'li bilan natijalarni olish mumkin.

Funktsiyaning o'zi nima degan savol tug'ilishi tabiiy. Funktsiya – bu biror aniq masalani yechishga mo'ljallangan, to'liq tugallangan, mustaqil dastur, yanaham aniqroq aytadigan bo'lsak, qism dastur yoki yordamchi dastur.

C++ dasturlash tilida har qanday dastur hech bo'lmaganda bitta bosh funktsiyani, ya'ni `main()` funktsiyasini o'z tarkibida saqlaydi. Shuningdek, `main()` funktsiyasidan boshqa qandaydir masalalarni yechishga mo'ljallangan ko'plab funktsiyalar bo'lishi ham mumkin.

Har bir funktsiya o'z nomiga ega bo'ladi va unga o'sha nomini ko'rsatish yo'li bilan murojaat qilinadi. C++ dastur tarkibida biror funktsiya nomini uchratgach, dastur bajarilishini o'sha funktsiya tanasida davom ettiradi va funktsiyadagi buyruqlar tugagach, o'sha buyruqlar natijasini murojaat bolgan joyiga qaytaradi va murojaat buyrug'idan keyingi buyruqlarni bajarishda davom etadi.

Dasturlashda foydalaniladigan funktsiyalar ikki guruhga bo'linadi: *standart funktsiyalar* va *standart bo'lmagan funktsiyalar*.

Standart funktsiyalar dasturlash tillari kompilyatorlarining bir qismidan iborat bo'lib, shu kompilyatorni tayyorlovchi firma tomonidan yaxlit paket sifatida dasturlash tili dasturlari guruhiga qo'shiladi. Natijada dasturlash tili to'g'ri yozilgan funktsiya nomi uchratsa, avtomatik tarzda o'sha funktsiya qiymatini hisoblash dasturiga murojaat qilib, uning qiymatini qaytaradi.

Nostandart funktsiyalar esa dasturchilar tomonidan yaratiladi. Biz quyida nostandart funktsiyalarni yaratish jarayoni va usullari bilan tanishamiz.

6.1. Funksiyani e'lon qilish va aniqlash

Yuqorida qayd qilganimizdek, funksiya - bu qism dastur yoki asosiy (bosh) dasturga yordamchi dastur hisoblanadi. Dastur tarkibida xususiy holda **main()** funksiyasi tarkibida boshqa funksiyaga (qism dasturga) murojaat qilish uchun o'sha funksiya avval e'lon qilinishi so'ngra aniqlanishi lozim.

Funksiyani e'lon qilish deyilganda **main()** funksiyasidan oldin funksiyaning turini, ya'ni qabul qiladigan qiymatlari turini, nomi qavs ichida parametrlari(argumentlari)ni turlarini ko'rsatib yozib chiqish tushuniladi va funksiyadan foydalanishda kompilyator bu ma'lumotlarni inobatga oladi.

Funksiya e'loni uning prototini yoki signaturasi ham deyiladi. Funksiyani aniqlash esa uning qanday masalani yechayotganligini buyruqlar yordamida ko'rsatish yoki funksiya sarlavhasiga uning tanasini qo'shib yozish tushuniladi.

Funksiya prototininin umumiy ko'rinishi quyidagicha

Funksiya turi funksiya nomi (turlari ko'rsatilgan argumentlar ro'yxati);

Bunda funksiya turi hozirgacha bizga ma'lum bo'lgan turlardan biri bo'ladi. Bu tur funksiya qaysi turga mansub qiymat qabul qilishiga qarab tanlanadi. Funksiya nomi u yechadigan masala mazmuniga mos holda tanlanadi.

Masalan, kvadrat tenglamani yechayotgan bo'lsa, **ktve**, yig'indini hisoblayotgan bo'lsa, **summa**, natural sonning faktorialini hisoblayotgan bo'lsa, **fak** kabi tanlanishi mumkin. Funksiya turi va nomi probel bilan ajratiladi. Funksiya nomidan keyin qavs ichida turlari ko'rsatilgan argumentlari(parametrlari) ko'rsatiladi va qavs yopilgach albatta (;) qo'yiladi.

Qaysidir dasturdan(xususiy holda qaysidir funksiyadan) biror funksiyaga murojaat qilinsa, murojaat bo'lgan funksiyaning tanasidagi hisoblash ishlari bajarilib, hisoblash natijasi o'sha murojaat bo'lgan joyga qaytariladi. Aynan qaytarilayotgan qiymat turi bilan funksiya turi bir xil bo'lishi kerak. Aks holda kompilyator xato sodir bo'lganligi haqida xabar beradi. Bundan buyon funksiya tanasi bajarilgandan keyin qaytariladigan qiymatni funksiyaga qaytariladigan qiymat yoki qisqacha qilib qaytariladigan qiymat deb ataymiz.

Endi funksiya prototini e'lon qilish usullari bilan tanishamiz.

Funksiya prototini argumentlarining nomlarini ko'rsatmasdan, faqat turlarini ko'rsatish yo'li bilan ham e'lon qilish mumkin. Chunki ular formal parametrlar. Masalan,

float bo'luvchi (int, float, char);

Bunda **bo'luvchi** nomli haqiqiy turli funksiya e'lon qilindi va u uchta argumentga ega bo'lib, ular har uchasi uch xil turli. C++ dasturlash tilida bunga ruxsat etiladi. Lekin dastur tushunarli bo'lishi uchun funksiya prototini e'lon qilishning bu usulidan foydalanmaslik tavsiya etiladi. Funksiya prototipidagi parametrlar formal parametrlar bo'lsa ham ularni quyidagicha ko'rsatib e'lon qilish dastur mohiyatini tushunishni yengillashtiradi.

Float bo'luvchi (int x, float y, char z);

Yuqorida qayd qilinganidek, funksiya qaytariladigan qiymat turi ham e'lon qilinishi kerak. Agar funksiya prototipida uning turi ko'rsatilmagan bo'lsa, C++ ning ba'zi kompilyatorlari unga avtomatik tarzda **int** turini beradi. Masalan, funksiya prototipi

fak (int n);

kabi e'lon qilingan bo'lsa, qaytariladigan qiymat butun turli deb qabul qilinadi. Ammo funksiya turini ko'rsatmagandan ko'ra ko'rsatish yaxshiroq xatto bu tur **int** bo'lsa ham.

Endi funksiyaning aniqlash jarayoni bilan tanishamiz.

Funksiyani aniqlovchi matn ikki qismdan iborat, ya'ni funksiya sarlavhasidan va tanasidan iborat bo'ladi. Funksiya sarlavhasi yozuvi uning prototipi yozuvi bilan bir xil bo'ladi. Faqat birinchidan, argumenti nomi aniq yozilishi shart, ikkinchidan, funksiya sarlavhasini yozib bolgach qavsdan keyin (;) qo'yilmaydi.

Funksiya sarlavhasidan keyin figurali qavs ({} ochiladi va shundan so'ng funksiya tarkibida foydalaniladigan qo'shimcha o'zgaruvchilar, ya'ni asosiy dasturda e'lon qilinmagan o'zgaruvchilar turlarini ko'rsatib e'lon qilinadi hamda funksiya yechayotgan masalaga taalluqli buyruqlar ketma-ketligi asosiy dasturda bo'lgani kabi bir-birlaridan ({} bilan ajratilgan holda yoziladi. Oxirida agar funksiyaning qiymati chaqiriq byurug'i bo'lgan joyga qaytarilishi kerak bo'lsa **return** operatori yozilib probel qo'yiladi hamda qiymati qaytarilayotgan o'zgaruvchi yoki ifoda yozilib, (;) quyiladi hamda figurali qavs ({} yopiladi.

Funksiyaning ushbu figurali qavslar ({} orasidagi qismi uning tanasi deyiladi.

Misol. Binomial koeffitsientni aniqlash dasturini tuzing.
Ma'lumki, binomial koeffitsient

$$C_n^m = \frac{n!}{m!(n-m)!} \quad \text{formula bilan topiladi.}$$

```

1 #include <iostream.h>
2 using namespace std;
3 int fak(int n);
4 int main()
5 {
6     int n,m,nf,mf,nm; float c;
7     cin>>n>>m;
8     nf=fak(n); mf=fak(m); nm=fak(n-m);
9     c=nf/(mf*nm);
10    cout<<"nc="<<c<<"\n";
11    system("pause");
12    return 0;
13 }
14 int fak(int n)
15 { int f=1, i=1;
16   while(i<=n) {f=f*i; i++;}
17   return f;
18 }
```

Dasturning 3-satrida butun turli fak nomli funksiya prototipi e'lon qilindi va u bitta butun turli n parametriga ega.

int main () funksiyasi tanasi tugagandan keyin fak nomli funksiya 14-satrdan 18-satrgacha aniqlandi. Demak, funksiya biror funksiya tanasidan tashqarida aniqlanishi keak. Lekin funksiya tanasidan boshqa funksiya tanasiga murojaat qilish mumkin. 14-satr fak nomli funksiya(yordamchi dastur)ning sarlavhasi deyiladi. U funksiya prototipi bilan bir xil. Faqat sarlavha oxirida, ya'ni qavsdan keyin (:) belgisi qo'yilmagan. 15 va 18 satrlardagi figurali qavslar fak funksiyasining blok belgilari. Dasturning 15- satridan 18-satrigacha bo'lgan qismi fak nomli funksiyaning tanasi deyiladi.

Bundan buyon tarkibida boshqa funksiya murojaat byurug'i bo'lgan dasturni asosiy dastur deb, murojaat bo'layotgan funksiyan esa yordamchi dastur(poddastur) deb ham ataymiz. Demak, dasturning 4-satridan 13-satrigacha bo'lgan qismi asosiy dastur, 14-satridan 18-satrigacha bo'lgan qismi esa yordamchi dastur hisoblanadi.

Asosiy dasturning 8-satrida uchta chaqiriq byuruqlari (yoki funksiyaga murojaat byurulari) berilgan. Ulardan biri, masalan, nf=fak(n); byurug'ini tahlil qilaylik. C++ bu buyruqni uchratgach, dastur bajarilishini 15-satrdan davom ettiradi, ya'ni fak nomli funksiyaning tanasini bajaradi. Bunda 15 va 16-satrlarda n! hisoblanadi. n! ning natijaviy qiymatini butun turli f o'zgaruvchi oladi.

17-satrdagi return f; byurug'i f o'zgaruvchining qiymatini murojaat bo'lgan joyga qaytarishni ta'minlaydi. Natijada nooshkor holda nf=f qiymat berish byurug'i bajarilib, keyingi mf=fak(m) byuruqni bajarishga o'tiladi.

Xulosa: demak, qism dastur tanasida bajarilgan hisob natijalarini return operatori va undan keyin yozilgan o'zgaruvchi yoki ifoda yordamilda asosiy dasturga olib chiqiladi.

Funksiya prototipi, funksiya sarlavhasi va chaqiriq buyruqlaridagi parametrlar nomlarining har xil bo'lishi dastur bajarilishiga hech qanday ta'sir qilmaydi. Chunki funksiya prototipi va aniqlanishidagi argumentlar (bizning dasturimizda n parametr) formal (nomiga turgan) parametrlar bo'lib, ularning qiymati shu funksiya murojaat bo'lguncha aniq bo'lmaydi. Demak, yuqorida keltirilgan dastur quyidagi ko'rinishda ham bexato ishlaydi.

```

#include <iostream.h>
using namespace std; int fak(int s);
int main()
{ int n, m, nf, mf, nm; float c;
  cin>>n>>m; nf=fak(n); mf=fak(m); nm=fak(n-m);
  c=nf/(mf*nm);
  cout<<"nc="<<c<<"\n"; system("pause"); return 0;
}
int fak(int y)
{ int t f=1, i=1; while(i<=y) {f=f*i; i++;}; return f;
}
```

Lekin zaruriyat bo'lmaganda bu usuldan foydalanmaslik tavsiya etiladi, chunki dasturni tushunish qiyinlashadi.

Funksiya prototipidan foydalanmaslik holati

Funksiya prototipini yozmasdan ham funksiyadan foydalanish mumkin. Buning uchun funksiya `int main()` funksiyasi tanasidan tashqarida aniqlanadi va shu funksiyani chaqiruvchi byurug'idan oldingi satrda biriga funksiyani sarlavhasi yozib qo'yilishi kerak. Bu usul quyidagi dasturdan foydalanib tahlil qilamiz.

```

1 #include <iostream.h>
2 using namespace std;
3 int main()
4 { int x, i, j, n, m, k, z, v, d, g;
5 int fun1(int y2, int y1);
6 m=fun1(i, j); k=m+10;
7 cout<<"nk="<<k<<"\n";
8 system("pause");
9 return 0;
10 }
11 int fun1(int y2, int y1)
12 {int g1, g2, p1;
13 p1=4;
14 int fun2(int a, int b);
15 g2=fun2(y1, y2); cout<<"g2="<<g2<<"\n";
16 g1=g2+p1;
17 return g1;
18 }
19 int fun2(int a, int b)
20 {int a1=6, a2=8, a3;
21 a3=a1*a2;
22 return a3;}

```

11-satrdagi `fun1`, 19-satrdagi `fun2` nomli funksiyalar aniqlangan, lekin, ularning prototipi e'lon qilinmagan. Chunki 6-satrdagi `fun1` funksiyasiga murojaat byurug'idan oldin 5-satrdagi bu funksiyani sarlavhasi yozilgan. Shuningdek, 15-satrdagi `fun2` funksiyasini chaqiruvchi byurug'i bor. Lekin 14-satrdagi shu funksiyani sarlavhasi yozilgan.

Funksiyalardan bunday usul yordamida foydalanish hech qanday mantiqiy xatolarga olib kelmaydi. Ammo, ayniqsa, funksiyadan funksiyaga murojaatlarni tashkil qilishda har xil chalkashliklar sodir bo'lishi mumkin. Shuning uchun bu usuldan ham zarurat bo'lmaganda foydalanmaslik tavsiya etiladi.

Global va lokal o'zgaruvchilar

O'zgaruvchilar funksiya tanasida yoki undan tashqarida e'lon qilinishi mumkin. Bunda funksiya tanasidan tashqarida e'lon qilingan o'zgaruvchilar global (umumiy) o'zgaruvchilar bo'lib, ulardan barchasining ixtiyoriy joyida, jumladan funksiyalar tarkibida ham foydalanish mumkin. Ammo biror funksiya tanasida e'lon qilingan o'zgaruvchilardan shu funksiya tanasidagina foydalanish mumkin, funksiya tanasidan tashqarida ulardan foydalanib bo'lmaydi. Bunday o'zgaruvchilar shu funksiya uchun lokal (mahalliy) o'zgaruvchilar hisoblanadi.

Ko'pincha adabiyotlarda mahalliy o'zgaruvchilarni funksiya tanasidan tashqarida «amal qilmaydi» yoki «ko'rinmaydi» deyishadi. Shuning uchun o'zgaruvchining amal qilish sohasi va amal qilish vaqti tushunchalari mavjud.

O'zgaruvchining amal qilish sohasi deyilganda, o'sha o'zgaruvchi e'lon qilingan funksiya bloki tushuniladi. Amal qilish vaqti (yoki yashash vaqti) deyilganda esa o'zgaruvchi amal qilishi mumkin bo'lgan dastur qismini EHM tomonidan bajarish uchun ketadigan vaqt tushuniladi. Yoki boshqacha qilib aytganda, o'zgaruvchining amal qilish vaqti — bu u e'lon qilingan funksiya tanasini bajarish uchun sarflanadigan vaqtdir.

Shuni unutmash kerakki, masalan, `f1()` funksiyadan funksiyaga murojaat qilinayotgan bo'lsa ham `f1()` funksiyasida qilingan o'zgaruvchilar shu funksiyani mahalliy o'zgaruvchi bo'lib, `f2()` funksiya uchun ham umumiy o'zgaruvchi hisoblanmaydi.

Umumiy (global) o'zgaruvchilar `main()` funksiyasidan oldin qilinishi kerak. Hatto, `main()` bosh funksiyasida e'lon qilingan o'zgaruvchilar ham mahalliy hisoblanadi.

Yana bir holatga e'tiboringizni qaratamiz. Agar umumiy o'zgaruvchining umumiy qiymati unutilib, mahalliy qiymat berilgan bo'lsa, o'sha funksiya ichida mahalliy qiymatni o'zgaruvchining qiymatini hisobga oldirish kerak. Lekin o'sha funksiya ichida mahalliy qiymat berilgan bo'lsa, umumiy o'zgaruvchining qiymatini hisobga oldirish kerak.

Shuningdek, global o'zgaruvchilar initsializatsiya bo'lsa, uning qiymati nol deb hisoblanadi. Aytilgan mohiyatini quyidagi dastur yordamida tushunish qiyin em


```

1 #include <iostream.h>
2 int fun1(int y2, int y1);
3 float a=6, b=20, c;
4 int main()
5 { int x=12, y=8, m, k;
6 m=fun1(x,y); k=m+10;
7 cout<<"nk="<<k<<" b="<<b<<" c="<<c<<"\n";
8 system("pause");
9 return 0;
10 }
11 int fun1(int y2, int y1)
12 {int b=12, p1, a=15;
13 p1=y1+y2+b+::a;
14 return p1;
15 }

```

Dastur natijasi quyidagicha bo'ladi
k=48 b=20 c=0

Dasturning 3-satrida haqiqiy turli **a**, **b**, **s** umumiy o'zgaruvchilar e'lon qilindi. Shuning uchun ulardan dasturning ixtiyoriy joyida foydalanish mumkin. **a** va **b** o'zgaruvchilar ham umumiy, fun1() funksiyasi tanasida esa mahalliy o'zgaruvchilar sifatida e'lon qilingan.

Shuning uchun 13-satrdagi hisoblarda **b** o'zgaruvchining 20 ga teng umumiy qiymati unutilib, 12 ga teng bo'lgan mahalliy qiymati hisobga olingan. Ammo **a** o'zgaruvchidan :: **a** ko'rinishda foydalanilganligi uchun uning mahalliy qiymati 15 unutilib, umumiy qiymati 6 hisobga olinmoqda. **c** umumiy parametrning dastlabki qiymati berilmaganligi uchun uning qiymati **c=0** kabi chop etilmoqda.

Birinchidan, mahalliy va umumiy o'zgaruvchilarni bir xil nomlamaslik kerak. Bu har xil mantiqiy xatolarga olib kelishi mumkin. Ikkinchidan, zarurat bo'lmasa umumiy o'zgaruvchilardan ham foydalanmaslik tavsiya etiladi. Chunki umumiy o'zgaruvchilardan barcha funksiyalar tarkibida foydalanish mumkin bo'lganligi uchun qaysidir funksiya tanasida uning qiymatini o'zgartirganligimizni unutib, eski qiymatini hisobga olgan holda keyingi buyruqlarni yozib chiqsak, mantiqiy xatoliklar sodir bo'lishi mumkin.

Biror funksiya tarkibida boshqa funksiyani aniqlashga ruxsat etilmaydi, ya'ni funksiya ixtiyoriy funksiyadan, hatto main()

funksiyasidan ham tashqarida aniqlanadi. Ammo bitta funksiyadan xohlagancha funksiyaga murojaat qilish mumkin. Hatto funksiyadan o'ziga murojaat tashkil qilish mumkin. Bunday murojaat *rekursiv murojaat* yoki qisqacha *rekursiya* deyiladi.

Yuqorida qayd qilganimizdek, funksiya argumenti turlari funksiya turiga mos kelmasligi mumkin. Masalan, funksiya argumentlari biri **int**, biri **float**, biri **bool**, biri **char** turli bo'lishi, hatto biri biror funksiyaga chaqiriq byurug'i yoki qiymati aniq bo'lgan ifoda bo'lishi mumkin. Faqat chaqiriq byurug'iga qaytariladigan qiymat turi o'sha funksiya turi bilan mos kelishi yetarli.

Misol, $y = \sqrt{|a^x|}$ funksiya qiymatini hisoblash dasturini tuzing.

Bu oddiy funksiya qiymatini hisoblash uchun **ildiz**, **modul** va **daraja** nomli uchta funksiyadan foydalanib, funksiya argumentida funksiyadan foydalanishning murakkab ko'rinishi bilan tanishamiz.

```

#include <iostream.h>
#include <math.h>
using namespace std;
int ildiz(int k2); int modul(int k3); int daraja(int k4,int k5);
int main()
{
    int n,m,nf,mf,nm; float y;
    y=ildiz(16*modul(daraja(4,3)));
    cout<<"ny="<<y<<"\n";
    system("pause");
    return 0;
}
int ildiz(int n)
{int i; i=sqrt(n); return i;}
int modul(int m)
{int j; j=abs(m); return j;}
int daraja(int a,int b)
{int p; p=pow(a,b); return p;}

```

Funksiya qiymatini qaytarish haqida

Barcha dasturlarda **int main()** asosiy funksiyasidan foydalandik va bu funksiyadan xohlagancha boshqa funksiyalarga murojaat tashkil qilish mumkin. **void** turida bo'lmagan har qanday funksiya **return**

so'zi bilan yakunlanadi. Jumladan, `main()` funksiyasi ham shu operator bilan yakunlanadi. `return` operatoridan foydalanishda quyidagilarga e'tibor berish kerak:

a) faqat `void` turli funksiyalar parametrsiz `return` operatori bilan yakunlanishi mumkin. Chunki qiymat chaqiriq buyrug'i qaytarilmaydi. Shuning uchun bu turli funksiyalar `return` operatori bilan yakunlanishi, ya'ni blok belgisi bilan yakunlanishi mumkin;

b) `void` turidan boshqa turli barcha funksiyalar `return` operatori bilan yakunlanadi, ya'ni `return` parametrlari ko'rinishida bo'lishi kerak. Agar bu operatoridan parametrsiz foydalanilsa, kompilyator qaytariladigan qiymat yo'q deb xabar beradi;

c) `void` turidan boshqa turli funksiya `returnsiz` tugasa ham, blok belgisi `{}` ni uchratgach, kompilyator funksiya tanasini bajarish tugaganligini tushunadi va xatolik haqida xabar bermaydi. Ammo funksiya tanasidagi hisob natijalari qaytarilmaganligi tufayli, chaqiriq buyrug'i qandaydir o'zgaruvchiga qiymat sifatida berilgan bo'lsa, o'sha o'zgaruvchi qiymat sifatida tasodifiy sonni oladi va tabiiyki, keyingi buyruqlarda mantiqiy xatoliklar sodir bo'ladi;

d) `return` so'zidan keyin parametrlar sifatida biror o'zgaruvchi nomidan, masalan, `return p` kabi foydalanilgan bo'lib, `p` e'lon qilingan, ammo qiymati aniqlanmagan bo'lsa ham qiymat sifatida tasodifiy son qaytariladi;

e) chaqiriq buyrug'idan foydalanilgandan keyin qandaydir qiymat qaytariladi. Lekin o'sha qiymatni zarurat bo'lmasa biror o'zgaruvchiga berib, chaqiriq buyrug'idan keyingi buyruqlarda foydalanish shart emas. Bunday holda funksiya tanasidagi amallar natijasi o'z-o'zidan yo'qolib ketadi;

f) funksiya tanasidagi hisob natijalari qanday bo'lishidan qat'iy nazar `return`da ko'rsatilgan son yoki parametrning qiymati qaytariladi. Masalan, funksiya `return 20` kabi yakunlangan bo'lsa, funksiya tanasidagi hisob natijalari turi va qiymatidan qat'iy nazar, chaqiriq buyrug'i natijasi 20 ga teng bo'ladi;

d) yuqorida funksiya turi qanday bo'lsa unga qaytariladigan qiymat turi ham shunday bo'lmasa, xatolik sodir bo'lishi haqida aytilgan edi. Bu xatolik ham mantiqiy xatolik bo'lishi mumkin. Ya'ni kompilyator xato haqida xabar bermasligi mumkin. Masalan, funksiya turi `int` bo'lsin. Lekin unga haqiqiy qiymat qaytarilsa, qaytarilgan qiymatning kasr qismi tashlab yuboriladi. Xatto, butun turli funksiyaga `return true`

bo'lsa, `bool` turli qiymat qaytarilsa, bu qiymat 1 deb, `false` qiymat qaytarilsa, u nol deb qabul qilinadi;

h) `return` operatori orqali faqat massivning qiymatini qaytarib berilmaydi. Lekin massiv elementi qiymatini qaytarish mumkin;

i) `main()` bosh funksiyasida qiymat qaytarish uchun `return 0;` buyrug'idan foydalaniladi. Bunda nol butun son bo'lganligi uchun foydalanilgan. Ammo nol o'rinda ixtiyoriy manfiy va musbat, butun va haqiqiy sonlardan, qiymati aniq bo'lgan arifmetik va mantiqiy amallardan ham foydalanish mumkin. Shuningdek, `return` operatori bo'lmasa ham dastur ijro etiladi. Ya'ni bosh funksiyaning blok belgisini uchratgach C++ dastur yakunlanganligini tushunadi.

Qiymat qaytarish bo'yicha aytilgan fikrlarni quyidagi dastur misolida tushuntiramiz.

```
1 #include <iostream.h> using namespace std;
2 int fun1 (int x, int y);
3 int main()
4 {int i,j,k,m,r; i=5; j=2;
5 m=fun1(i,j); k=10+m;
6 cout<<"\n k="<<k<<"\n";
7 system("pause");
8 return -56;
9 }
10 int fun1 (int x,int y)
11 {int q1,p; q1=x+y;
12 cout<<"\n q1="<<q1<<"\n";
13 return q1; system("pause"); }
```

Dastur natijasi

q1=7

k=14 kabi bo'ladi

j) agar dasturning 13-satrdagi `return q1` buyrug'i olib tashlansa, m ning qiymati tasodifiy tanlanadi va dastur natijasi

q1=7

k=4469706

kabi bo'ladi. Shuningdek, 8-satrdagi `return -56` buyrug'i ham olib tashlansa, natija o'zgarmaydi. Ammo 13-satrdagi `return q1` buyrug'ini qoldirib, 8-satrdagi `return -56` buyrug'ini olib tashlasak ham, dastur to'g'ri bajariladi va dastur natijasi

```
q1=7
k=17
```

kabi bo'ladi. Chunki bosh funksiyadan `return` buyrug'i o'tirish tashlangan bo'lsa ham C++ blok belgisi `()` ni uchratgach, dasturni tugatish tugaganligini tushunadi.

k) agar 13-satrdagi `return p` kabi o'zgartirsak, yana qaytariladigan qiymatni tasodifiy tanlaydi va dastur natijasi

```
q1=7
k=4246714
```

kabi bo'ladi. Shuningdek, 13-satrdagi buyruqni o'zgarishsiz qoldirib, 8-satrdagi buyruqni `return r` kabi o'zgartirsak, dastur to'g'ri bajariladi. `main()` funksiyasiga qaytariladigan qiymat uning tarkibidagi buyruqlarga ta'sir qilmaydi, chunki ular ijro etilgan bo'ladi.

l) 5-satrdagi buyruqlar `m=fin1(i,j); k=i-j+10;` kabi o'zgartirilsa, dastur natijasi

```
q1=7
k=13
```

kabi bo'ladi. Ya'ni qaytarilgan qiymat biror o'zgaruvchiga qiymat sifatida berilmaydi va undan umuman foydalanilmaydi. Natijada qaytarilgan qiymat unutiladi.

m) 13-satrdagi buyruq `return-19` kabi o'zgartirilsa, dastur natijasi

```
q1=7
k=-9
```

kabi bo'ladi. Chunki `fun1()` funksiyasi tanasida hisoblar natijasi qanday bo'lishidan qat'iy nazar chaqiriq buyrug'iga - 19 qaytariladi.

n) dasturning 4-satri `float i, j, k, m, r; i=5.54, j=2.5412;` kabi o'zgartirilsa, dastur natijasi o'zgarmaydi, ya'ni

```
q1=7
k=17
```

kabi bo'ladi. Dasturning 13-satri `return true` kabi o'zgartirilsa, dastur natijasi

```
q1=7
k=11
```

kabi bo'ladi.

Umuman, `return` operatori parametri qandaydir son bo'lmasdan qiymati aniq bo'lgan sonli yoki harfiy ifoda, hatto, mantiqiy ifoda ham bo'lishi mumkin. Masalan, agar `x` va `y` larning qiymatlari oldindan aniqlangan bo'lsa, quyidagicha buyruqlarga ruxsat etiladi:

```
return (2*x-3)*y;
return (37/7+8*3)/2;
return x+2>=y-5;
return fun2(4,5)
```

Masalan, dasturning 13-satrida `return 3*x<=y-1` kabi o'zgarish qilinsa, natija

```
q1=7
k=10
kabi bo'ladi.
```

Agar dasturning 2-satriga `int fun2(int x, int y);` yozuvini qo'shib, 3-satrida `m=fun2(i,j); k=10+m` kabi o'zgartirib, 14-18 satrlariga quyidagicha buyruqlar qo'shilsa,

```
14 int fun2 (int x, int y)
15 { int q1,p; q1=x-y;
16 cout<<"n q1="<<q1<<"n";
17 return fun1 (4,7);
}
```

```
dastur natijasi
q1=3
q1=11
k=21
kabi bo'ladi.
```

Quyidagi savollarga javob bering

1. Dasturlashda funksiya tushunchasining kiritilishi qanday imkoniyatlar yaratadi?
2. Dasturni tuzishda funksiya tushunchasidan foydalanish katta yengilliklar paydo qiladigan bir necha misollar keltiring?
3. C++ dasturlash tilida funksiya o'zi nima?
4. C++ dasturlash tilida tuzilgan dasturda kamida nechta funksiyadan foydalaniladi?
5. Funksiyaga murojaat qilish qanday amallarning (buyruqlarning) bajarilishini ta'minlaydi?
6. Funksiya tanasidagi barcha buyruqlar bajarilgandan keyin qaysi buyruqlar bajariladi?
7. C++ dasturlash tilining asosiy funksiyasi qaysi?

8. Funksiyani chaqiriq buyrug'i dasturning qayerida va qanday yoziladi?
9. Standart va nostandart funksiyalar qanday funksiyalar?
10. C++ dasturlash tilida yordamchi yoki qism dasturlar qanday dasturlar?
11. C++ dasturlash tilida funksiya qanday va qayerda e'lon qilinadi?
12. Funksiya prototipi, signaturasi tushunchalarini tushuntiring.
13. Funksiya qayerda va qanday aniqlanadi?
14. Funksiya prototipi bilan uni aniqlashning qanday farqi bor?
15. Funksiya portotipining umumiy ko'rinishini yozing.
16. Chiziqli tenglamani yechishga mo'ljallangan funksiyani signaturasini yozing.
17. Funksiyaga qaytariladigan qiymat deyilganda qanday qiymat tushuniladi va uning turi qanday bo'lishi kerak?
18. Funksiya turi bilan argumentlari turida qanday bog'lanish bo'lishi kerak?
19. Funksiyaning formal parametrlari qanday parametrlar? Nima uchun?
20. Funksiya necha qismdan iborat?
21. Funksiya tanasi qanday belgilar ichida yoziladi?
22. Funksiya prototipini aniqlashning necha xil usulini bilasiz?
23. Funksiyaga qiymat qaysi operator orqali, qanday qaytariladi?
24. Funksiyaga qaytariladigan qiymat faqat son bo'lishi shartmi?
25. `return k`; buyrug'i nimani bildiradi?
26. Funksiya prototipini, aniqlanishi va chaqiriq buyrug'idagi argumentlar har xil nomlar bilan nomlanishi mumkinmi? Nima uchun?
27. Funksiya prototipi yozmasdan ham funksiya yaratish mumkinmi? Agar mumkin bo'lsa u qanday tashkil qilinadi?
28. Qanday o'zgaruvchilardan funksiya tashqarisida foydalanib bo'lmaydi?
29. Global o'zgaruvchilar qanday o'zgaruvchilar?
30. O'zgaruvchining amal qilishi sohasi qaysi soha?
31. Global o'zgaruvchilar qayerda e'lon qilinadi?
32. `main()` funksiyasida aniqlangan o'zgaruvchilar mahalliy o'zgaruvchilarmi yoki umumiy o'zgaruvchilarmi?
32. Umumiy va mahalliy o'zgaruvchilar bir xil nom bilan nomlangan bo'lsa funksiya ichida va funksiya tashqarisida qaysi qiymat hisobga olinadi?

32. Global va lokal o'zgaruvchilar bir xil nomlangan bo'lsin. Qanday yo'l bilan funksiya ichida global o'zgaruvchining qiymatidan foydalanish mumkin?
33. Global o'zgaruvchi initsializatsiya qilinmagan bo'lsa, qanday xatolik sodir bo'ladi?
34. Funksiya tarkibida ko'pi bilan nechta buyruq bo'lishi mumkin?
35. Bitta funksiyadan ko'pi bilan nechta funksiyaga murojaat tashkil qilish mumkin?
36. Funksiyaning o'zidan o'ziga murojaat qilish mumkinmi va qanday murojaat qanday nomlanadi?
37. Har qanday funksiya tanasi qanday yakunlanadi?
38. a e'lon qilingan, ammo qiymati aniqlanmagan o'zgaruvchi bo'lib, `return a`; buyrug'i berilsa qanday xatolik ro'y beradi?
39. Bosh funksiyadan boshqa biror funksiyada `return s`; buyrug'i o'rniga `return 30`; buyrug'idan foydalanilsa qanday xatolik sodir bo'ladi va nima uchun?
40. Bosh funksiyaga qiymat qaytarishda `return 0`; buyrug'idagi nol raqami o'rnida yana qanday kattaliklardan foydalanish mumkin?

Quyidagi masalalarning dasturlarini tuzishda funksiya tushunchasidan foydalaning

1. Uzunligi a, b, c va d ga teng bo'lgan to'rtta kesma berilgan. Shu kesmaning qaysilaridan uchburchak yasash mumkin bo'lsa, ularning turini va yuzlarini hisoblash dasturini tuzing.
2. Uchburchakning tomonlari a, b, c. Uning perimetri, yuzi, balandliklari, medianalari, ichki va tashqi chizilgan aylanalari uzunliklarini aniqlang.
3. To'rtta natural sonlarning EKUK va EKUB larini toping.
4. Quyidagi ifodalarning qiymatlarini hisoblash dasturlarini tuzing:

- 1) $\sum_{i=1}^n [i + \sum_{j=2}^m (i + j)]$
- 2) $\sum_{i=1}^n (i + (i + 1)!)$
- 3) $\prod_{i=1}^n (i + c) + \prod_{j=5}^m (j + 2) / 3 + \prod_{k=1}^l (k - 5) * b$
- 4) $\sum_{n=1}^m \frac{n!(n-1)!}{(n+k)!k!}$
- 5) $Q = \begin{cases} \sum_{i=1}^4 (i! + (2i)!); & A, A \geq 4. \\ \sum_{i=1}^3 ((3i)! - i!), & A \leq 4. \end{cases}$

$$6) \frac{n!+k!}{(n+k)!(n-k)!} (n > k)$$

5. Uchburchak uchlarining koordinatalari berilgan. Uchburchak medianalarining uzunliklarini toping.
6. Uchburchak yuzini va to'g'ri to'rtburchakning yuzini topish dasturlaridan funksiya sifatida foydalanib, asoslari a va b , balandligi h bo'lgan teng yonli trapetsiyaning yuzini hisoblang.
7. Ikkita, uchta, to'rtta sondan kattasini topish dasturlaridan funksiya sifatida foydalanib, 10 ta sondan kattasini toping.
8. Berilgan satrni teskari tartibda yozish, undagi 0 va a harflari sonini aniqlash, k harfini olib tashlash, i harfini ikkita i harfi bilan almashtirish dasturini tuzing. Bu amallarning har birini bajaruvchi dasturni funksiya sifatida yarating.
9. Quyidagi ifodalarning qiymatlarini hisoblash dasturlarini tuzing va bunda albatta funksiya dan foydalaning:

$$\frac{n! + \sum_{i=1}^n (i+2)}{k! + \prod_{j=2}^m (j-1)}$$

$$\frac{\sum_{i=3}^n i + \prod_{j=1}^5 (j+a)}{\sum_{k=1}^m (k-1) + (m-a)!}$$

$$\sum_{i=1}^n \left(4 + \prod_{j=1}^i (j+1) \right) \prod_{l=2}^n \left(i + \sum_{j=1}^i \left(\frac{i}{i+j} + \prod_{k=1}^j k \right) \right)$$

$$\sum_{i=1}^n \left(\sum_{j=1}^m (i+j) + (i-1)! \right)$$

$$\sum_{i=1}^n i - \sum_{j=2}^m (2j+3) + \sum_{k=1}^n (k+9)$$

$$\frac{n!+k!}{(n+k)!(n-k)!}; (n > k)$$

10. To'rtta so'z berilgan. Bu so'zlarning har birida «a» harfi necha marta ishtirok etganligini aniqlang.
11. a, b, c, d kesmalar berilgan. Qaysi 3 ta kesmalardan uchburchak yasash mumkinligini aniqlang.
12. Talabalar bilimni baholash dasturini tuzing.
13. Uch xonali sonlar ichida nechta mukammal, armstrong, palindrom sonlari borligini aniqlang.
14. Kiritilgan 6 ta sonlardan qaysilari juft-jufti bilan o'zaro tubligini aniqlang.
15. Raqamlari yig'indisi n ga teng bo'lgan barcha K xonali sonlarni toping.
16. Raqamlari yig'indisi tub son bo'lgan to'rt xonali sonlarni toring.
17. Ikkita ko'p xonali sonlar kiritiladi. Qaysi raqam har ikkilasida ham borligini aniqlang.
18. Shunday $ne[1;50]$ sonlarini topingki, $n!$ Qaysidir sonning kvadratiga teng bo'lsin.

VII-BOB. FAYL VA STRUKTURALAR

7.1. Fayllar va ulardan foydalanish

C++ dasturlash tili muhitida fayllar bilan ishlash imkoniyati ham mavjud. Dasturlash tillarida fayllardan dastur tarkibida o'zgaruvchilar qabul qilishi lozim bo'lgan qiymatlarni fayl sifatida yozib qo'yish va kerak bo'lganda o'qib, ulardan foydalanish maqsadida foydalaniladi. Dasturlash texnologiyalari rivojlanib borishi bilan buni dasturlash tillari tarkibida fayllar bilan ishlash qismi ham rivojlanib bormoqda. Etarlicha murakkab masalalarning dasturlarini tuzishda o'zgaruvchilarning qabul qilishi mumkin bo'lgan qiymatlar soni ko'p bo'lda va har safar o'sha qiymatlarni klaviaturadan kiritish vaqt va mehnat talab qiladi. Bunday hollarda o'zgaruvchilarning qabul qilishi mumkin bo'lgan qiymatlarini oldindan fayl sifatida saqlab, kerak bo'lganda o'sha qiymatlarni fayldan o'qish maqsadga muvofiq. Fayllardan qiymatlarni o'qish jarayoni esa foydalanuvchi aralashuvisiz dastur tomonidan tezkor holda amalga oshiriladi.

Ta'rif: Kompyuterda alohida nom bilan saqlanadigan va ma'lum bir turga mansub bo'lgan ma'lumotlar majmuasi fayl deb nomlanadi.

C++ dasturlash tili tarkibida fayllarga murojaat qilish uchun fayl nomiga to'g'ridan-to'g'ri murojaat qilib bo'lmaydi. Buning uchun avval kerakli ma'lumotlarni o'zida saqlovchi faylni yaratib, keyin undagi ma'lumotlarni o'qishni tashkil qilish kerak.

Ma'lumotlarni faylga yozish va undan o'qish jarayoni C++ dasturlash tilida ma'lumotlarni **cin** va **cout** operatorlari yordamida kiritish va chiqarish jarayoni bilan mazmun jihatidan bir xil. Farqi shundaki, **cin** va **cout** operatorlari yordamida ma'lumotlarni kiritish va chiqarish jarayonida bu ma'lumotlar ekranda ko'rinib turadi, faylga yozish va undan o'qish jarayoni ekranda ko'rinmaydi, yani nooshkor holda amalga oshiriladi.

Fayllarni e'lon qilish

C++ dasturlash tilida fayllarga murojaat qilish uchun albatta oldin **.txt** kengaytmali faylni yaratib, alohida joyga saqlab quyish kerak. Dasturlash tilida ishlatiladigan fayllar nomi ikki turda bo'ladi.

- 1) Fizik nomi;
- 2) Mantiqiy nomi.

Faylning fizik nomi kompyuterda **.txt** kengaytmali bo'lib, barcha ma'lumotlar shu **.txt** kengaytmali faylda saqlanadi.

Faylning mantiqiy nomi esa dastur tarkibida fizik nomi bilan bog'langan nomi hisoblanadi.

C++ dasturlash tilida fayllarni e'lon qilishdan oldin albatta **#include <fstream>** kutubxonasini qo'shish kerak.

C++ dasturlash tilida fayllarning fizik nomiga to'g'ridan to'g'ri murojaat qilib bo'lmaydi. Buning uchun faylning fizik nomiga mantiqiy nomini bog'lab murojaat qilish mumkin. Fayllarning mantiqiy nomini fizik nomiga bog'lagandan so'ng dasturning ixtiyoriy joyida faylga murojaat qilish, albatta uning mantiqiy nomi bilan amalga oshiriladi.

C++ dasturlash tilida fayllarni e'lon qilish va ularni mantiqiy nomi bilan bog'lash uchun ixtiyoriy o'zgaruvchidan mantiqiy nom sifatida foydalanish mumkin.

C++ dasturlash tilida faylning mantiqiy nomi quyidagi ikki maqsadda e'lon qilinadi:

1) faylning fizik nomiga ma'lumot yozish maqsadida;

2) faylning fizik nomidan ma'lumot o'qish maqsadida;

Mantiqiy nomning fayllarga ma'lumotlar yozish maqsadida e'lon qilinishining umumiy ko'rinishi quyidagicha bo'ladi:

ofstream <faylni mantiqiy nomi>;

ofstream kalit so'zi orqali fayl e'lon qilinganda **.txt** kengaytmali faylga faqatgina ma'lumot yozish amalini bajarish mumkin. Ammo bunday e'lon qilish bilan **.txt** kengaytmali fayldan ma'lumot o'qib bo'lmaydi.

C++ dasturlash tilida mantiqiy nomning ma'lumot o'qish uchun e'lon qilinishining umumiy ko'rinishi quyidagicha bo'ladi:

ifstream <faylni mantiqiy nomi>;

ifstream kalit so'zi orqali fayl e'lon qilinganda faqat **.txt** kengaytmali fayldan ma'lumot o'qish amalini bajarish mumkin. Ya'ni dasturni ilro etish jarayonida agar zarurat bo'lsa **.txt** kengaytmali faylda yozilgan ma'lumotlardan ularni o'qish yo'li bilan foydalanish imkoniyati paydo bo'ladi.

Agar faylni o'zidan ma'lumot o'qib shu faylning o'ziga ma'lumot yozish talab etilsa, albatta, birinchi fayl o'qish uchun ochilib, ma'lumot o'qiladi va fayl yopilib keyin fayl yozish uchun ochilib, ma'lumot yozilishi kerak.

Fayllarga murojaat qilib bo'lgandan so'ng, albatta, fayllar yopilish kerak. C++ dasturlash tilida fayllarni yozishning umumiy ko'rinishi quyidagicha:

<fayl ni mantiqiy nomi>.close();

Ikkita f va g o'zgaruvchilarni fayllarning mantiqiy nomi sifatida qabul qilib, ularni mos ravishda yozish va o'qish bo'yicha e'lon qilinishi quyida namoyish qilingan:

```
#include <iostream.h>
#include <fstream>
using namespace std;
int main()
{
    ofstream f; // faylga ma'lumotni yozish uchun e'lon qilish
    ifstream g; // fayldan ma'lumot o'qish uchun e'lon qilish
    f.close(); // f fayl yopildi
    g.close(); // g fayl yopildi
    return 0;
}
```

Bunda f va g lar fayllarning mantiqiy nomlari bo'lib, ular dasturda ko'rsatilganidek, o'qish va yozish uchun alohida - alohida e'lon qilingan. Yuqoridagi dasturga f faylga ma'lumot yozish uchun, g esa fayldan ma'lumot o'qish uchun e'lon qilindi. Shuningdek, dasturda f va g fayllarni yopish jarayoni ham keltirilgan.

Fayllarga murojaat qilish

C++ dasturlash tilida fayllar bilan ishlash uchun dastlab dasturingizni va foydalaniladigan fayllarni albatta, bitta papkaga saqalab oling. Dasturlash tillarida fayllarga murojaat qilish uchun oldin uning mantiqiy va fizik nomlarini bog'lash kerak. Fayllarning mantiqiy va fizik nomlari bog'langandan so'ng faylga uning mantiqiy nomi bilan murojaat qilinadi. Fayllarning mantiqiy nomini fizik nomiga bog'lashning umumiy ko'rinishi quyidagicha bo'ladi:

<mantiqiy nomi>.open("<fizik nomi adresi bilan>");

Faylning mantiqiy nomi fizik nomiga bog'lanish vaqtida fizik nomi adresi bilan to'liq yozilish kerak aks holda ular bog'lanmaydi.

Agar yaratilgan dastur joylashgan papkada fayl yaratilgan bo'lsa, adresi ko'rsatilmasdan uning nomi ko'rsatilishi kifoya.

Misol: *output.txt* va *input.txt* fayllarini yarating va ularni mos ravishda f va g mantiqiy nomlari bilan bog'lang.

```
#include <iostream.h>
#include <fstream>
using namespace std;
int main()
{
    ofstream f;
    f.open("output.txt");
    ifstream g;
    g.open("input.txt");
    f.close(); g.close();
    return 0;
}
```

Yuqoridagi dasturga e'tibor qaratsak, *output.txt* va *input.txt* fayllar f yozish va g o'qish fayllari bilan bog'landi. Demak, f va g fayllar mos ravishda yozish va o'qish uchun tayyor holatga keltirildi. C++ dasturlash tilida fayllarga murojaatni dasturning ixtiyoriy joyida tashkil qilish mumkin. Faqat yuqorida qayd qilganimizdek, undan oldin mantiqiy va fizik nomlari bog'langan bo'lishi kerak.

Keltirilgan dasturda fayllarni bog'lash jarayoni

```
ofstream f;
f.open("output.txt");
ifstream g;
g.open("input.txt");
```

satrlarda amalga oshirildi. Bu ishni qisqacha quyidagicha ham amalga oshirish mumkin:

```
ofstream f("output.txt"); // f fayl "output.txt" fayliga ma'lumot
yozish maqsadida bog'landi.
ifstream g("input.txt"); // g fayl "input.txt" fayliga undan ma'lumot
o'qish maqsadida bog'landi.
```

Fayllar ustida amallar bajarish

Fayllar e'lon qilinib, mantiqiy va fizik nomlari bog'langandan keyin ular ustida quyidagi amallarni bajarish mumkin:

1) faylga ma'lumot yozish;

- 2) fayldan ma'lumot o'qish;
- 3) fayl tarkibidagi ma'lumotlarni o'chirish.

C++ dasturlash tilida fayllarga ma'lumot yozishning ikki usuli mavjud:

- birinchi usuldan foydalanganimizda faylda bo'lgan oldingi ma'lumotlar o'chirilib, uning joyiga yangi ma'lumot yoziladi;
- ikkinchi usuldan foydalanganimizda esa faylda bo'lgan eski ma'lumotlar ham saqlanib qoladi va yangi yozilayotgan ma'lumotlar eskisining davomidam yoziladi.

Faylga ma'lumot yozishning birinchi usulidan foydalanganimizda

```
ofstream < mantiqiy nom >;
```

```
< mantiqiy nom >.open("fizik nom");
```

buyruqlaridan yoki yuqorida qayd qilganimizdek, qisqacha

```
ofstream < mantiqiy nom >("fizik nom");
```

buyrug'idan foydalanamiz.

Shundan so'ng faylning fizik nomi tarkibiga ma'lumot yozish mumkin. C++ dasturlash tilida faylga birinchi usul bo'yicha ma'lumot yozishning umumiy ko'rinishi quyidagicha:

```
< fayl ni mantiqiy nomi > << < qiymat yoki qiymatga ega bo'lgan o'zgaruvchi >;
```

Unutmaslik kerakki, yuqorida qayd qilganimizdek, fayl tarkibiga bu usulda ma'lumot yozilganda unda oldin mavjud bo'lgan ma'lumotlari o'chirilib, yangi ma'lumotlar yoziladi.

Misol: n ta elementli massiv berilgan massiv elementlari ichidan toqlarini toq.txt fayliga juftlarini juft.txt fayliga yozing.

```
#include <iostream.h>
```

```
#include <fstream>
```

```
using namespace std;
```

```
int main()
```

```
{ int n; int a[90];
```

```
cout<<"Massiv elementlari soni n ni tanlang n="; cin>>n;
```

```
cout<<"massiv elementlarini kiriting \n";
```

```
for(int i=0;i<n;i++) cin>>a[i];
```

```
ofstream f;
```

```
f.open("toq.txt");
```

```
ofstream g;
```

```
g.open("juft.txt");
```

```
for(int i=0;i<n;i++)
if(a[i]%2==1) f<<a[i];
else g<<a[i];
f.close();
g.close();
```

```
return 0;
```

Dastur natijasini quyidagi ma'lumotlarni kiritish bilan ko'rib oling:

Massiv elementlari soni n ni tanlang n=5

massiv elementlarini kiriting

5 6 9 8 1

Shundan so'ng toq.txt faylini ochib ko'rsanizingiz

5 9 1

kabi ma'lumotni, juft.txt faylini ochib ko'rsanizingiz

6 8

kabi ma'lumotni ko'rasiz.

Ushbu dasturni birinchi usulning ikkinchi, qisqacha ko'rinishida quyidagicha ham tashkil qilish mumkin:

```
#include <iostream.h>
```

```
#include <fstream.h>
```

```
using namespace std;
```

```
int main()
```

```
{ int i,j,k,n, a[100];
```

```
cout<<"Massiv elementlari sonini kiriting"<<"\n"; cin>>n;
```

```
for(i=0;i<=n;i++) cin>>a[i];
```

```
ofstream f("juft.txt");
```

```
ofstream s("toq.txt");
```

```
for(i=0;i<=n;i++) if(a[i]%2==1) f<<a[i]<<" ";
```

```
else s<<a[i]<<" ";
```

```
f.close();s.close();
```

```
system("pause");
```

```
return 0;
```

```
}
```

Yuqoridagi dasturda massiv elementlarining toqlari f fayl orqali toq.txt fayliga, juft elementlari esa g fayl orqali juft.txt fayliga joylashtirildi.

C++ dasturlash tilida ikkinchi usul bo'yicha faylga ma'lumot yozish uchun `ofstream < mantiqiy nom>; < mantiqiy nom>.open("fizik nom", ios::app);` buyruqlari oldin yoziladi. Shundan so'ng uning tarkibiga ma'lumot yozish mumkin. Ikkinchi usul bo'yicha ma'lumot yozish deganda tashkilot kengaytmali faylga bo'lgan eski ma'lumotlar o'chirilmaganidan uning oxiridan ma'lumot yozish tushuniladi. Faylning oxiriga ma'lumot yozish uchun e'lon qilish birinchi usulda bo'lganidek e'lon qilinadi. Faqatgina `ios::app` xossasi qo'shimcha sifatida qo'shiladi. C++ dasturlash tilida faylga ikkinchi usul bo'yicha ma'lumot o'qishning umumiy ko'rinishi quyidagicha bo'ladi:

```
<faylning mantiqiy nomi> << <o'zgaruvchi>;
Fayl tarkibiga ikkinchi usul bo'yicha ma'lumot yozish uchun
<mantiqiy nom>.open("fizik nom", ios::app);
yoki <mantiqiy nom>("fizik nom", ios::app);
buyruqlarining biridan foydalaniladi. Demak, bu usul bo'yicha yozish
tashkilot qilinganda kursor fayl oxiriga boradi va yangi yozilgan
ma'lumot fayldagi ma'lumotlarning davomidan yoziladi.
```

Misol: To'rtta 1 2 3 4 elementlari mavjud bo'lgan tt.txt fayl oxiriga 10 dan 20 gacha bo'lgan sonlarni yozing.

```
#include <iostream.h>
#include <fstream>
using namespace std;
int main()
{ ofstream f;
  f.open("tt.txt", ios::app);
  for(int i=10; i<=20; i++)
    f<<" "<<i;
  f.close();
return 0;
}
```

Shundan so'ng tt.txt faylini ochib ko'rsangiz quyidagi ma'lumotlarni ko'rasiz.

Файл	Программа	Формат	Вид	Справка
1	2	3	4	10 11 12 13 14 15 16 17 18 19 20

Yuqoridagi dasturda tt.txt faylida 1 2 3 4 sonlar mavjud edi, 10 dan 20 gacha bo'lgan sonlar 1 2 3 4 sonlaridan keyin tt.txt fayliga joylashtirildi.

Fayldan ma'lumot o'qish

Fayldan ma'lumot o'qish jarayoni ham unga ma'lumot yozishdan deyarli farq qilmaydi. Faqat `ofstream` kalit so'zi o'rniga `ifstream` kalit so'zidan foydalaniladi. Bu holda ham faylning mantiqiy nomi fizik nomi bilan bog'langandan so'ng undagi ma'lumotlar o'qiladi. Fayldan ma'lumotlarni o'qishda qandaydir o'zgaruvchi .txt kengaytmali fayldagi ma'lumotlarni qiymat sifatida oladi.

C++ dasturlash tilida fayldan ma'lumot o'qish uchun `ifstream < mantiqiy nom>; < mantiqiy nom>.open("fizik nom");` yoki

```
ifstream <mantiqiy nom>; <mantiqiy nom>("fizik nom");
buyruqlari berilgandan so'ng fayl tarkibidan ma'lumot o'qish
mumkin. C++ dasturlash tilida fayldan ma'lumot o'qishning umumiy
ko'rinishi quyidagicha bo'ladi:
```

```
<faylning mantiqiy nomi> >> <o'zgaruvchi>;
```

Fayldan ma'lumot o'qish jarayonini quyidagi dastur vositasida namoyish qilamiz:

Misol. 1 dan 10 gacha sonlar yozilgan tt.txt faylidan uning elementlarini ko'chirib olib ikkiga ko'paytirib ekranga chiqaring.

```
#include <iostream.h>
#include <fstream>
using namespace std;
int main()
{ ifstream f; int a;
  f.open("tt.txt");
  for(int i=1; i<=10; i++)
  { f>>a;
    cout<<2*a<<" ";
  }
  f.close();
return 0;
}
```

Dastur natijasi ekranda quyidagicha bo'ladi
2 4 6 8 10 12 14 16 18 20

Misol . 1 dan 20 gacha sonlar yozilgan tt.txt fayli elementlari orasidan juftlarini ikkiga ko'paytirib g faylga yozing.

Buning uchun f va g fayllarni mos ravishda o'qish va yozish maqsadida e'lon qilamiz va ularni tt.txt fayli bilan bog'laymiz.

```
#include <iostream.h>
#include <fstream>
using namespace std;
int main()
{ int a; ifstream f;
  f.open("tt.txt");
  ofstream g;
  g.open("g.txt");
  for(int i=1;i<=20;i++)
  { f>>a;
    if(a%2==0) g<<2*a<<" ";
  }
  f.close();
  g.close();
  return 0;
}
```

Kiritishga ma'lumot f(tt.txt) fayl

Файл	Правка	Формат	Вид	Справка
1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20

Dastur natijasi g(th.txt) fayl

Файл	Правка	Формат	Вид	Справка
4	8	12	16	20
24	28	32	36	40

C++ dasturlash tilida fayldan ma'lumotlar yuqoridagi holatlar bo'yicha o'qiladi va qayta ishlanadi.

Fayl tarkibidagi ma'lumotlarni o'chirish

Fayl tarkibidagi ma'lumotlarni o'chirish imkoniyati ham mavjud. Faylga ma'lumot yozilgan oldin faylni tozalash yoki uning tarkibidagi elementlarni o'chirish uchun maxsus fayllardagi ma'lumotlarni uchirish buyrug'i mavjud. C++ dasturlash tilida tuzilgan dastur tarkibida foydalaniladigan fayllarni tozalash uchun o'sha fayl tozalash maqsadida e'lon qilinishi shart. Fayl elementlarini uchirish maqsadida e'lon qilishning umumiy ko'rinishi quyidagicha bo'ladi:

<mantiqiy nom>.open("fizik nom",ios::trunc);

C++ dasturlash tilida fayl elementlarini o'chirish uchun **ofstream <mantiqiy nom>; <mantiqiy nom>.open("fizik nom", ios::trunc);** buyruqlari oldin yozilgandan so'ng fayl tarkibidagi elementlarni o'chirish mumkin. Faylni o'chirish uchun e'lon qilish xuddi yozish uchun e'lon qilishdek bo'ladi faqatgina ios::trunc xossasini qo'shib yozish kerak. C++ dasturlash tilida fayl elementlarini o'chirishning umumiy ko'rinishi quyidagicha bo'ladi:

<faylni mantiqiy nomi>.clear();

Fayl tarkibidagi ma'lumotlarni uchirish uchun faylni yuqoridagi buyruq asosida mantiqiy nomi bilan bog'lab, shu mantiqiy nom yordamida fayl elementlari o'chirish jarayoni bajariladi.

Misol: 1 dan 10 gacha sonlar yozilgan th.txt fayl elementlarini o'chirish.

```
#include <iostream.h>
#include <fstream>
using namespace std;
int main()
{ ofstream f;
  f.open("th.txt",ios::trunc);
  f.clear();
  f.close();
  return 0;
}
```

Dastur natijasi g(th.txt) fayli

Файл	Правка	Формат	Вид	Справка

Yuqoridagi dastur asosida th.txt fayli tozalandi.

C++ dasturlash tili tarkibida fayllardan foydalanish jarayonlari yuqoridagi holatlar bo'yicha amalga oshiriladi. Fayldan ma'lumot o'qish, faylga ma'lumot yozish va fayldan ma'lumot o'chirish jarayonlari asosida fayllarga tegishli masalalarni hal etish imkoniyati mavjud.

Endi amaliy misollar keltiramiz.

Misol. 1 dan n gacha bo'lgan natural sonlar yozilgan g fayl yaratib, undagi tub sonlarni f faylga, murakkab sonlarni h faylga joylashtiring.

```
#include <iostream>
#include <fstream>
using namespace std;
int main()
{ int n;
  cin>>n;

  int a; ofstream f,h;
  f.open("t1.txt"); h.open("t2.txt");
  ofstream g;
  g.open("t3.txt");
  for(int i=1;i<=n;i++)
  { g<<i<<" ";
    }g.close();

    ifstream l;
    l.open("t3.txt");
  for(int i=1;i<=n;i++)
  { l>>a;int k=0;
    for(int j=2;j<=a/2;j++)
    {
      if(a%j==0){k++;}
    }
    if(k==0 && a!=1){f<<a<<" "; cout<<" "<<a<<" "<<"\n";}
    else{h<<a<<"\n "; cout<<" "<<a<<" "<<"\n"; }
  }

  f.close();
  h.close();l.close(); system("pause");
  return 0;
}
```

Dasturni ijro etishda n ning qiymati sifatida 10 soni tanlansa, dastur natijasi quyidagicha bo'ladi:

```
10
1
2
3
4
5
6
7
8
9
10
```

Для продолжения нажмите любую клавишу...

Misol. 1 dan n gacha bo'lgan sonlarni g faylga joylashtirib, bu sonlarning buluvchilari yig'indisini f faylga joylashtiring

```
#include <iostream>
#include <fstream>
using namespace std;
int main()
{ int n;
  cin>>n;

  int s,a; ofstream f;
  f.open("t1.txt");
  ofstream g;
  g.open("t2.txt");
  for(int i=1;i<=n;i++)
  { g<<i<<" ";
    }g.close();

    ifstream l;
    l.open("t2.txt");
  for(int i=1;i<=n;i++)
  { l>>a;f<<a<<" sonining bo'luvchilar yig'indisi="; int
    k=0;cout<<"\n";
    for(int j=1;j<=a/2;j++) { if(a%j==0){ f<<j<<" ";
      cout<<j<<" "; k+=j; }} f<<a<<"="<<k+a; if(k!=0)
    cout<<"="<<k<<"\n";}
    f.close();
  }
```

```

l.close();
system("pause");
return 0;
}

```

Dasturni ijro etishda n ning qiymati sifatida 8 soni tanlanib dastur natijasi quyidagicha bo'ladi:

```

8
1 = 1
1 = 1
1 2 = 3
1 = 1
1 2 3 = 6
1 = 1
1 2 4 = 7

```

Для продолжения нажмите любую клавишу . . .

Quyidagi savollarga javob bering

1. Fayl deb nimaga aytiladi.
2. Dasturlashda qanday kengaytmali fayllardan foydalaniladi.
3. Dasturlashda fayl nomi qanday qismlardan tashkil topgan.
4. Fayllarni e'lon qilishda qaysi kutubxonaga murojaat qilish kerak.
5. Fayllarni e'lon qilishning qanday turlari mavjud.
6. Fayllarni yozish uchun e'lon qilishning qanday turlari mavjud?
7. Fayllarni yozish uchun e'lon qilishning umumiy ko'rinishini ayting?
8. Fayllarni o'qish uchun e'lon qilishning umumiy ko'rinishini ayting?
9. Fayllar elementlarini o'chirish uchun e'lon qilishning umumiy ko'rinishini ayting?
10. Fayllarni yopish komandasining umumiy ko'rinishini ayting?
11. Fayllarni fizik va mantiqiy nomlarini bog'lashning umumiy ko'rinishi?
12. Faylga ma'lumot yozishning ikki turdagi komandasining umumiy ko'rinishi?
13. Fayldan ma'lumot o'qish komandasining umumiy ko'rinishi?
14. Fayldan ma'lumot o'chirish komandasining umumiy ko'rinishi?

Mustaqil ishlash uchun topshiriqlar.

1. Elementlari alohida so'z hisoblanuvchi turlashgan fayl berilgan. "t" harfi bilan boshlanuvchi hamma so'zlarni chop qiling. Bunda ikki variantni qarang:

- a) ma'lumki, mavjud faylga 30 ta so'z yozilgan;
- b) mavjud fayl o'lchami ma'lum emas.

2. Elementlari sonlardan iborat bo'lgan mavjud turlashgan faylning talab etilgan elementlarini boshqa sonlar bilan almashtiring (yangi qiymat klaviaturadan kiritiladi):

- a) birinchi elementni;
- b) beshinchi elementni;
- v) k-nchi elementni;
- g) oxirgi elementni.

3. Elementlari alohida so'zlar bo'lgan mavjud turlashgan faylni berilgan qiymat bilan almashtiring (yangi so'z klaviaturadan kiritiladi):

- a) birinchi elementni;
- b) uchinchi elementni;
- v) s-nchi elementni;
- g) oxirgi elementni.

4. Turlashgan fayl elementlari alohida so'z hisoblanadi. Uning barcha elementlarini boshqa so'z bilan almashtiring. Bunda ikkita variantni qarang:

- a) ma'lumki, mavjud faylga 12 ta so'z yozilgan;
- b) mavjud fayl hajmi ma'lum emas.

5. Sonli turlashgan fayl mavjud. Uning tartib nomeri 3 ga karrali bo'lgan hamma elementlarini almashtiring. Yangi qiymat klaviaturadan kiritiladi. Bunda ikkita variantni qarang:

- a) ma'lumki, mavjud faylga 20 ta son yozilgan;
- b) mavjud fayl hajmi ma'lum emas.

6. Butun sonli turlashgan fayl mavjud. Uning hamma juft elementlarini nol bilan almashtiring. Bunda ikkita variantni qarang:

- a) ma'lumki, mavjud faylga 13 ta son yozilgan;
- b) mavjud fayl hajmi ma'lum emas.

7. Elementlari alohida so'zlardan iborat turlashgan fayl mavjud. Agar so'z "k" harfi bilan boshlansa, uni "K" harfi bilan almashtiring.

8. Sonli turlashgan fayl mavjud. Quyidagilarni aniqlang (hamma holatlarda fayl hajmi ma'lum emas):

- a) faylning birinchi va ikkinchi sonlar yig'indisini;
 - b) faylning k-nchi va q-nchi sonlari yig'indisi;
 - v) faylning birinchi va oxirgi sonlari ko'paytmasini;
 - g) faylning hamma sonlarining yig'indisini;
 - d) fayldagi sonlar miqdorini;
 - e) a sonidan oshib ketmaydigan fayldagi sonlar miqdorini;
 - j) faylning musbat sonlarining o'rta arifmetigini;
 - i) mavjud fayldagi eng katta sonni;
 - k) fayldagi eng kichik sonning tartib nomeri (agar bunday sonlar bir nechta bo'lsa, ularning birinchisini nomerini toping).
9. Elementlari alohida so'zlardan iborat turlashgan fayl mavjud. Quyidagilarni aniqlang (hamma holatlarda fayl hajmi ma'lum emas):
- a) "m" harfi bilan boshlanuvchi so'zlar sonini;
 - b) "l" harfi bilan boshlanuvchi barcha so'zlar va ularning birinchisini tartib nomerini chop qiling;
 - v) tartib nomeri juft bo'lgan so'zlar va ulardan gap tuzing;
 - g) eng uzun so'zni;
 - d) eng qisqa so'zni.
10. Turlashgan faylda shaxarlar nomlari va uning aholisi soni yozilgan. Har bir shaxarning aholisini 5% ga oshiring (aholi miqdori har doim butun son).
11. Butun sonli turlashgan fayl mavjud. Undan birinchi noldan keyin yozilgan sonni o'chiring (faylda nol mavjud deb qabul qilinng). Natijani boshqa faylga yozing.
12. Elementlari alohida belgilardan iborat turlashgan fayl mavjud. Undan birinchi "o" harfini o'chiring ("o" harfi faylda mavjud deb qabul qilinsin). Natijani boshqa faylga yozing.
13. Butun sonli turlashgan fayl mavjud. Birinchi 100 sonidan keyin 100 sonini qo'ying. Natija boshqa faylga yozing.
14. Elementlari 20 ta alohida belgidan iborat turlashgan fayl mavjud. Boshqa faylga mavjud fayl belgilarini teskari tartibda yozing.
15. Elementlari sondan iborat, o'lchamlari bir xil turlashgan ikkita fayl mavjud. Dastlab birinchi faylning, keyin ikkinchisining sonlarini yozib uchinchi faylni hosil qiling. Sonlar ketma-ketlik tartibi saqlansin.
16. Elementlari alohida belgilardan iborat turlashgan fayl mavjud. Bu faylning barcha raqamlarini ikkinchi faylga, qolgan

belgilarni esa uchinchi faylga yozing. Sonlar ketma-ketlik tartibi saqlansin.

17. Elementlari alohida belgilardan iborat bo'lgan, o'lchamlari bir xil turlashgan ikkita fayl mavjud. Elementlari tartibi saqlangan holda birinchi fayl elementlarini ikkinchisiga, ikkinchisini birinchisiga yozing. Yordamchi fayldan foydalaning.

18. Elementlari alohida belgilardan iborat bo'lgan, o'lchamlari bir xil turlashgan ikkita fayl mavjud. Ularning elementlarini mos kelishini aniqlang. Agar mos kelmasa, u holda bu fayllarning bir-biridan farq qiluvchi birinchi elementi nomerini aniqlang.

19. Elementlari butun sonlardan iborat bo'lgan, turlashgan ikkita tartiblashgan fayl mavjud. Berilgan ikkita faylni tartiblashgan holda birlashtirib, Yangi turlashgan faylni hosil qiling. Birlashtirish algoritmidan berilgan fayllar juda katta yozuvlardan iborat bo'lish mumkinligi ham qaralishi kerak.

20. Matnli fayl mavjud. Undan elementlarni o'chiring:

7.2 C++ dasturlash tilida struktura va ulardan foydalanish

Amalda biron bir tashkilot yoki muassasalarning ma'lumotlar bazasini yaratish va ma'lumotlar bazasi tarkibidagi ma'lumotlarni qayta ishlashga mo'ljallangan dasturlar tuzishga to'g'ri keladi. Ma'lumotlar bazasini yaratishda bitta ob'ekt uchun bir nechta parametrlardan foydalaniladi va bu parametrlarni birlashtirish kerak bo'ladi. Ma'lumotlar bazasini yaratish uchun qulay bo'lgan dasturiy ta'minotlar ham mavjud. Lekin mavjud dasturiy ta'minotlar hamisha ham foydalanuvchi ehtiyojlarini qondira olmaydi. Dasturlash tillarida ma'lumotlar bazasini yaratish jarayoniga mo'ljallangan keng imkoniyatlari mavjud.

C++ dasturlash tilida bir nechta turlarni birlashtirib, yangi tur hosil qilish jarayoni strukturalar yaratish deb yuritiladi.

Ma'lumotlar bazasini hosil qilish uchun ham strukturalardan foydalaniladi. Masalan, talaba haqida ma'lumotlar bazasi hosil qilishda talaba deb nomlangan struktura tarkibida talaba familiyasi, ismi, ota-onasi haqidagi ma'lumotlar, adresi, reytingi, tug'ilgan yili va joyi kabi parametrlari saqlanishi kerak. Demak, bunda talaba uchun yuqoridagi barcha parametrlar inobatga olinishi kerak. Struktura tarkibida bitta o'zgaruvchining bir nechta qiymatini saqlash mumkin. Bir nechta talabalar haqida ma'lumotlar bazasi yaratish

jarayonida har bir talaba ushun bir nechta parametrlardan yoritishga to'g'ri kelganligi uchun strukturalardan foydalanish maqsadga muvofiq.

Struktura va ularni e'lon qilish

Demak, C++ dasturlash tili tarkibida ham bir nechta turlarni birlashtirish yo'li bilan strukturalar hosil qilib, tashkilotlarga tegishli bo'lgan ko'p masalalarni hal etish imkoniyati mavjud.

Strukturalarni ba'zi hollarda yozuvlar ham deb ataladi, strukturalar tarkibidagi turlarning har biri maydon deb nomlanadi.

Ta'rif: Obyektning bitta xususiyatini ichida saqlaydigan parametr maydon deb nomlanadi.

Talaba haqida ma'lumotlar strukturasini hosil qilinishda quyidagicha maydonlar bo'lishi mumkin:

Talabani familiyasi, ismi, otasining ismi, tug'ilgan vaqti, telefoni, manzili va hakoza. Strukturalar maydonlardan tashkil topgan ekan, struktura tarkibidagi har bir maydon strukturani bitta parametrini o'zida saqlaydi. Struktura maydoni uchun xotira hajmi, maydon uzunligiga teng bo'ladi. Struktura maydoni qiymatlari ifodalarda ishlatilishi mumkin. C++ dasturlash tilida strukturalarni e'lon qilish uchun *struct* xizmatchi so'zidan foydalaniladi. Struktura e'lon qilingandan so'ng dastur tarkibida o'zgaruvchilarni struktura nomi bilan e'lon qilinadi.

C++ dasturlash tilida strukturalarni tasvirlashning umumiy ko'rinishi quyidagicha:

```
struct <struktura nomi>
{ <tur1> <maydon 1>;
  <tur2> <maydon 2>;
  ...
  <tur n> <maydon n>;
};
```

Strukturalarni tasvirlashda *struct* xizmatchi so'zidan keyin struktura nomi va figurali qavsda uning maydonlari kiritilishi shart. Struktura nomi masala mohiyatiga qarab, unga mos holda tanlanishi maqsadga muvofiq. Masalan, talabalar haqida ma'lumotlar bazasida struktura nomini talaba deb yoki supermarket mahsulotlari narxlari bazasida esa narx deb nomlash mumkin. Strukturalar tarkibidagi maydonlarni e'lon qilish xuddi oddiy o'zgaruvchilarni e'lon qilishdek

bajariladi va oxirgi blok yopilgandan so'ng, albatta, nuqtali vergul qo'yilishi shart.

Strukturalar yaratish jarayoni tushunarli bo'lishim uchun C++ dasturlash tilida talabalar haqida ma'lumotlar bazasini struktura yordamida yaratish jarayoniga misol keltiramiz.

Misol: Talabalar(familyasi, ismi, telefoni, reytingi, tug'ilgan yili) haqida ma'lumotlarga asosan C++ dasturlash tilida struktura hosil qiling.

```
#include <iostream.h>
using namespace std;
struct talaba
{
    char[20] fam; //familyani saqlovchi maydon
    char[20] ism; //ismni saqlovchi maydon
    char[20] tel; //telefonni saqlovchi maydon
    int reyting; //reyting balini saqlovchi maydon
    int ty; //tug'ilgan vaqtini saqlovchi maydon
};
int main()
{
    buyruqlar;
    return 0;
}
```

Keltirilgan misolda talaba haqida ma'lumotlar strukturasiga talaba deb nom berildi va uning maydonlari familiyasi(fam), ismi(ism), telefoni(tel), reytingi(reyting), tug'ilgan yili(ty) ko'rinishlarida nomlandi. Maydonlarni e'lon qilishda dastur tarkibida qanday maqsadlarda foydalanilishi e'tiborga olinishi kerak. Agar dastur tarkibida maydonlar ustida hisob ishlari olib borilmasa, ularning turini char yoki string turi ko'rinishida e'lon qilinsa, dastur bir muncha yengil ishlaydi.

Strukturalarga murojaat qilish va ulardan foydalanish

Talaba haqida ma'lumotlarni qayta ishlash davrida bitta talabaning bir nechta parametrlariga murojaat qilish mumkin.

Dasturlashda struktura o'zgaruvchisini e'lon qilish uchun unga tur sifati struktura nomi beriladi. Talaba haqida ma'lumotlar bazasini hosil qilishda talaba deb nomlangan struktura yaratiladi. Talabalar ma'lumotlar bazasida ma'lumotlardan foydalanish uchun o'zgaruvchilar e'lon qilinishi kerak va o'sha o'zgaruvchining turi talaba bo'lishi kerak. Quyida struktura o'zgaruvchisini e'lon qilish jarayoni namoyish qilingan.

<struktura nomi> <struktura o'zgaruvchisi>

Talaba ma'lumotlar bazasini qayta ishlashga mo'ljallangan talaba nomli struktura yaratib, undagi ma'lumotlardan foydalanish uchun A nomli o'zgaruvchini e'lon qilish jarayonini quyidagi dastur yordamida ko'rib chiqamiz:

```
#include <iostream.h>
using namespace std;
struct talaba
{
    char[20] fam; //familyani saqlovchi maydon
    char[20] ism; //ismni saqlovchi maydon
    char[20] tel; //telefonni saqlovchi maydon
    int reyting; //reyting balini saqlovchi maydon
    int ty; //tug'ilgan vaqtini saqlovchi maydon
};
int main()
{
    talaba A; // A o'zgaruvchini talaba turi bilan e'lon qilish
    Buyruqlar;
    return 0;
}
```

C++ dasturlash tilida struktura maydoniga murojaat qilish uchun struktura nomi va maydon nomi nuqta bilan ajratilgan holda amalga oshiriladi. Struktura ma'lumotlarini dastur tarkibiga kiritish vaqtida har bir maydon elementlarini kiritish kerak bo'ladi. C++ dasturlash tilida strukturalarga murojaat qilishning umumiy ko'rinishi quyidagicha bo'ladi:

<operator funksiya> <struktura o'zgaruvchisi>.<maydon nomi>

Strukturalarga murojaatni amalga oshirish uchun quyidagi masalaga e'tibor qarating:

Misol: Supermarketdagi ichimlik narxi, saqlash yili haqida ma'lumot berilgan 2015 yildan hozirgacha chiqarilgan ichimliklarning narxini chiqaring aks holda muddati o'tganligini aniqlang.

Bu masalani hal etish uchun ichimlik nomli struktura yaratiladi, strukturaning ikkita: ichimlik narxi va saqlash muddati haqidagi maydonlari mavjud bo'ladi. Ichimlik haqidagi ma'lumotni A o'zgaruvchiga saqlaymiz. A o'zgaruvchini saqlash muddati maydonini 2015 bilan solishtiriladi, agar kichik bo'lsa, muddati o'tgan aks holda narxi chiqariladi.

```
#include <iostream.h>
using namespace std;
struct ichimlik
{
    string nomi; //ichimlik nomini saqlovchi maydon
    int narx; //ichimlik narxini saqlovchi maydon
    int muddat; //ichimlik saqlash muddatini saqlovchi maydon
};
int main()
{
    ichimlik A;
    cin>>A.nomi;
    cin>>A.narx;
    cin>>A.muddat;
    if(A.muddat<2015) cout<<"muddat utgan";
    else cout<<A.narx<<"sum";
    return 0;
}
```

C++ dasturlash tilida strukturalar yaratib ularga murojaat qilish oddiy o'zgaruvchilar kabi bajariladi faqat struktura o'zgaruvchisi nuqta bilan maydon nomi yoziladi. Biror bir tashkilotning ma'lumotlar bazasini yaratish uchun struktura o'zgaruvchisining ko'p qiymatidan foydalanishga to'g'ri keladi. Struktura o'zgaruvchisining ikki yoki undan ortiq qiymatlariga murojaat qilish uchun struktura o'zgaruvchisini massiv sifatida e'lon qilish kerak.

Struktura massivi

Dastur tarkibida strukturalardan yakka tartibda foydalanish maqsadga muvofiq bo'lmaydi. Strukturada biror bir tashkilotning ma'lumotlar bazasi berilganda ularni qayta ishlash uchun qulay vosita hisoblanadi. Agar strukturalar bir nechta qiymatlar uchun ishlatilsa, struktura o'zgaruvchisini struktura massivi sifatida e'lon qilish maqsadga muvofiq.

Demak, struktura o'zgaruvchisi bir nechta qiymatlar uchun ishlatilsa, struktura o'zgaruvchisi struktura massivi deb yuritiladi.

Struktura massivini e'lon qilish, xuddi oddiy massiv kabi e'lon qilinadi, faqatgina massiv turiga struktura nomi ko'rsatiladi. C++ dasturlash tilida struktura massivlarini e'lon qilishning umimiy ko'rinishi quyidagicha:

<struktura nomi> <struktura massivi> [<qiymatlar soni>]

Struktura massiviga murojaat qilish uchun massiv elementlariga murojaat qilishdek bajariladi faqatgina massiv elementlariga maydon nomi bilan birga murojaat qilinadi. Talabalar haqida ma'lumotlar yaratish va ularni qayta ishlash uchun struktura massividan foydalanish kerak, chunki bitta talaba uchun bir nechta parametr ishlatiladi va ma'lumotlar bazasida n ta talaba haqida ma'lumot saqlanadi.

Misol: N ta talaba haqida ismi, familiyasi, tug'ilgan yili haqida ma'lumot berilgan. 20 yoshdan katta bo'lgan talabalarining ismi va familiyasini ekranga chiqaruvchi dastur tuzing.

```
#include <iostream.h>
using namespace std;
struct talaba
{
    string fam;
    string ism;
    int tyili;
};
int main(int argc, char* argv[])
{ talaba A[50]; int n=3;
  for(int i=0;i<n;i++)
  {
    cin>>A[i].fam;
    cin>>A[i].ism;
```

```
cin>>A[i].tyili;
}
for(int i=0;i<n;i++)
if((2017-A[i].tyili)>20)
{cout<<A[i].fam<<" "; cout<<A[i].ism<<"\n"; }
return 0;
}
```

Kiritishga ma'lumot
Allayorov Jasur 1995
Abduvaliv Umar 1998
Saydiyev Sardor 1996

Dastur natijasi
Allayorov Jasur
Saydiyev Sardor

Misol: O'qituvchilarning ismi, familiyasi, ilmiy darajasi haqida ma'lumot berilgan. Professorlar, dotsentlar, assistentlar sonini aniqlovchi dastur tuzing.

```
#include <iostream.h>
using namespace std;
struct uqituvchi
{
    string fam;
    string ism;
    string idaraja;
};
int main(int argc, char* argv[])
{ uqituvchi A[50]; int n=3;
  for(int i=0;i<n;i++)
  {
    cin>>A[i].fam;
    cin>>A[i].ism;
    cin>>A[i].idaraja;
  }
  int p=0,d=0,s=0;
  for(int i=0;i<n;i++){
    if(A[i].idaraja=="professor") p++;
    if(A[i].idaraja=="dotsent") d++;
    if(A[i].idaraja=="assistent") s++;
  }
  cout<<"professorlar soni "<<p<<"\n";
```

```
cout<<"dotsentlar soni "<<d<<"\n";
cout<<"assistentlar soni "<<s;
return 0;
}
```

Kiritishga ma'lumot

Tolipov Temur assistant
Soliev Aktam dotsent
Karimiov Obid professor

Dastur natijasi

professorlar soni 1
dotsentlar soni 1
assistentlar soni 1

Endi amaliy masalalardan namunalar keltiramiz:

Misol. Talabalarning uchta, yani matematika, informatika va fizika fanlaridan to'plagan ballari mavjud. Har bir talabaning o'rtacha bahosini hisoblab, natijalarni ekranga chop etish dasturini tuzing.

```
#include <iostream.h>
using namespace std;
struct talaba
{
    string fam;
    string ism;
    int matematika;
    int informatika;
    int fizika;
};
int main(int argc, char* argv[])
{ talaba A[50]; int n; cin>>n;
  for(int i=0;i<n;i++)
  {
    cin>>A[i].fam;
    cin>>A[i].ism;
    cin>>A[i].matematika;cin>>A[i].informatika;cin>>A[i].fizika;
  }
  float s=0;
  for(int i=0;i<n;i++){
    s=(A[i].matematika+A[i].informatika+A[i].fizika)/3;
    if(s>85) cout<<A[i].fam<<" "<<A[i].ism<<"ning
    bahosi="<<"5"<<"\n";
```

```
if(s>=71 && s<86) cout<<A[i].fam<<" "<<A[i].ism<<"ning
    bahosi="<<"4"<<"\n";
if(s>=55 && s<70) cout<<A[i].fam<<" "<<A[i].ism<<"ning
    bahosi="<<"3"<<"\n";
if(s<55) cout<<A[i].fam<<" "<<A[i].ism<<"ning
    bahosi="<<"2"<<"\n"; }
system("pause");
return 0;
}
```

Dasturdan foydalanish uchun talabalar sonini 3 ta qilib tanlab, ularning to'plagan ballarini 100 ballik tizimda quyidagicha tartibda kiriting. Albatta siz har bir fandan har bir talaba uchun 100 ballik tizimda xohlagan balni kiritishingiz mumkin.

3
Sobirov Tolib 73 81 69
Davidov Asqar 69 75 54
Nishonov Karim 57 55 56
Shundan so'ng ekranda quyidagicha natija chop etiladi:
Sobirov Tolibning bahosi=4
Davidov Asqarning bahosi=3
Nishonov Karimning bahosi=3
Для продолжения нажмите любую клавишу ...

Misol. Avtomobillar uchun ma'lumotlar bazasini yarating. Bunda avtomobil egasining familiyasi, ismi, mashinasining rusumi va rangini aniqlash mumkin bo'lsin.

```
#include <iostream>
using namespace std;
struct mashina
{ string fam;
  string ism;
  string rusum;
  string rang;
};
int main(int argc, char* argv[])
{ mashina A[50]; int n; string s; cin>>n; //cin>>n;
  for(int i=0;i<n;i++)
```



```

{
    cin>>A[i].fam;
    cin>>A[i].ism;
    cin>>A[i].rusum;cin>>A[i].rang;
}
for(int i=0;i<n;i++){
    cout<<A[i].fam<<"    "<<A[i].ism<<"ning mashinasi"
    "<<A[i].rusum<<" rusumli, rangi "<<A[i].rang<<"\n";
}
system("pause");
return 0;
}

```

Dasturdan foydalanishda haydovchilar sonini 3 ta qilib tanlab, ularga tegishli ma'lumotlarni quyidagicha tartibda kiritib, dastur ishlashini sinab ko'ring.

3
Karimov Olim Nexya Oq
Toxirov Ochilxo'ja Damas ko'k
Ochilov Yopildi Cobalt Qora

Dastur natijasi ekranda quyidagicha e'lon qilinadi:

Karimov Olimning mashinasi Nexya rusumli, rangi Oq
Toxirov Ochilxo'janing mashinasi Damas rusumli, rangi ko'k
Ochilov Yopildining mashinasi Cobalt rusumli, rangi Qora
Для продолжения нажмите любую клавишу...

Quyidagi savollarga javob bering

- 1.Struktura deb nimaga aytiladi?
- 2.Maydon deb nimaga aytiladi?
- 3.Strukturalardan qanday maqsadlarda foydalaniladi?
- 4.Strukturalarning umumiy ko'rinishini yozing va tushuntiring?
- 5.Struktura o'zgaruvchisini e'lon qilish jarayonini bayon qiling?
- 6.Strukturalarga murojaat qilish jarayonini misollar yordamida tushuntiring?
- 7.Struktura massivi deb nimaga aytiladi?
- 8.Strukturalar massividan qanday holatlarda foydalaniladi?
- 9.Struktura massivini e'lon qilish jarayoni qanday amalga oshiriladi?

10.Strukturalar massiviga qay tartibda murojaat qilinadi?

Mustaqil ishlash uchun topshiriqlar

Quyidagi tushunchalar uchun aralash turlarni e'lon qiling.

- a) Mahsulotlarning bahosi uchun so'm va tiyinlar;
- b) Vaqt uchun soat, minut va soniyalar;
- c) Manzil uchun davlat, shaxar, ko'cha va uy nomeri;
- d) Telefon uchun kod, telefon raqamlari, abonent familiyasi va ismi;
- e) Mashg'ulotlar uchun kun, fan, o'qituvchi va dars soati;
- f) Imtihon qaydnomasi uchun fan, guruh raqami, talabani familiyasi va ismi, talaba daftarchasining raqami va bahosi;
- g) Shaxmat taxtasi uchun maydonlar;
- h) Kompleks son uchun uning haqiqiy va mavhum qismi.

2. Mashina egasining familiyasi, ismi, mashina rangi, rusumi va davlat nomeriga ega bo'lgan struktura yarating va faqat davlat nomeri A harfi bilan boshlangan mashinalar sonini ekranga chiqaring.

3. Mashina egasining familiyasi, ismi, mashina rangi, rusumi va davlat nomeriga ega bo'lgan struktura yarating va faqat oq, qora, ko'k rangli mashinalar egalarining familiyasini ekranga chiqaring.

4. Mashina egasining familiyasi, ismi, mashina rangi, rusumi va davlat nomeriga ega bo'lgan struktura yarating va faqat davlat raqami tarkibida 1 va 5 raqamlari bo'lgan mashinalar egalarining familiyasini ekranga chiqaring.

5.Tog' cho'qqilarining nomi, balandligi, joylashgan davlat nomi haqida struktura yarating. Eng baland cho'qqi joylashgan davlat nomi, cho'qqi nomi va balandligini aniqlang.

6.Tog' cho'qqilarining nomi, balandligi, ushbu cho'qqi joylashgan davlat nomi haqida struktura yarating. Eng past cho'qqi joylashgan davlat nomi, cho'qqi nomi va balandligini aniqlang.

7. Mehmonxonalar nomi, narxi, joylashgan ko'chasining nomi haqida struktura yarating. Eng arzon mehmonxona nomi va ko'cha nomini aniqlang.

8.Mehmonxonalar nomi, narxi, joylashgan ko'chasining nomi haqida struktura yarating. Eng qimmat mehmonxona nomi va ko'cha nomini aniqlang.

9.Talabalar familiyasi, ismi, kursi, reyting bali haqida struktura yarating. n - kursdagi eng a'lochi talabani familiyasi va ismini aniqlang.

10. Talabalar familiyasi, ismi, kursi, reyting bali haqida strukturani yaratish. n - kursdagi eng past ko'rsatkichli talabaning familiyasi va ismini aniqlang.
11. Talabalar familiyasi, ismi, kursi, reyting bali haqida strukturani yaratish. n - kursdagi 5 bahoga o'qiydigan talabalar familiyasi va ismini aniqlang.
12. Talabalar familiyasi, ismi, kursi, reyting bali haqida strukturani yaratish. n - kursdagi 4 bahoga o'qiydigan talabalar familiyasi va ismini aniqlang.
13. Talabalar familiyasi, ismi, kursi, reyting bali haqida strukturani yaratish. n - kursdagi 3 bahoga o'qiydigan talabalar familiyasi va ismini aniqlang.
14. Shaxar poliklinikasiga qarashli aholining har birining familiyasi, ismi, kasalligi haqida struktura yaratish. Shaxar aholisi ichidan X kasallik bilan kasallangan odamlarning familiyasi, ismi va sonini aniqlang.
15. Shaxar poliklinikasiga qarashli aholining har birining familiyasi, ismi, kasallik haqida struktura yaratish. Shaxar aholisi ichida kasalligi mavjud bo'lmagan odamlarning familiyasi, ismi va sonini aniqlang.
16. Telefon kitobida ishchilarning familiyasi, ismi, lavozimi, tel raqami haqida struktura yaratish. X familiyali insonning tel raqami va lavozimini aniqlang.
17. Telefon kitobida ishchilarning familiyasi, ismi, lavozimi, tel raqami haqida struktura yaratish. X tel raqamli insonning familiyasi va lavozimini aniqlang.
18. O'qituvchilarning familiyasi, ismi, lavozimi haqida struktura yaratish. X lavozimli o'qituvchilarning familiyasi va ismini aniqlang.
19. O'qituvchilarning familiyasi, ismi, lavozimi, tug'ilgan yili haqida struktura yaratish. X yoshga kirgan o'qituvchilarning familiyasi va ismini aniqlang.
20. Maktab o'quvchilarining familiyasi, ismi, sinfi, yashaydigan ko'chasining nomi haqida struktura yaratish. X ko'chada yashaydigan o'quvchilar familiyasi, ismi va sinfini aniqlang.

VIII BOB. C++ DASTURLASH TILINING GRAFIK IMKONIYATLARI

Barcha dasturlash tillarida bo'lgani singari C++ dasturlash tili ham juda boy grafik imkoniyatlarga ega. Odatda qo'shimcha buyruqlar berilmasa, kompyuter ekrani matn muhitida bo'ladi. Grafik muhitni joriy qilish uchun dasturga `#include<graphics.h>` kutubxonasi ham qo'shilishi lozim. Bu kutubxona yuklangandan so'ng `main()` funksiyasi blokida

```
int gd=DETECT, gm; initgraph(&gd,&gm, "");
```

funksiya yozilgandan boshlab grafik muhit ishlay boshlaydi. Grafik muhiti ishlashini yakunlash uchun `closegraph()`; funksiyasidan foydalaniladi. Demak grafik muhitni yuklash uchun `initgraph()`, yopish uchun esa `closegraph()` funksiyalaridan foydalaniladi. Grafik muhitida mashina ekranini dekart koordinatalari tekisligining birinchi choragi deb qarash kerak. Bunda koordinata boshini ekraning chap yuqori burchagida, OX o'qi gorizontal yo'nalishda o'nga, OY o'qi esa vertikal yo'nalishda pastga yo'nalgan deb qaraladi. Natijada, ekranni o'zaro tig'iz joylashgan nuqtalardan tashkil topgan deb tushunish mumkin va har bir nuqta o'z koordinatalariga ega bo'ladi. Shuningdek, nuqtalar ketma-ketligiga ekran rangidan boshqa rang berish yo'li bilan tasvir hosil qilamiz. Masalan, ekran rangida biror shakl chizsak, u ko'rinmaydi. C++ dasturlash tilining grafik muhitida har xil shakllarni chizishga mo'ljallangan ko'plab funksiyalar mavjud. Quyida asosiy funksiyalar bilan tanishamiz.

8.1. Tekislikda shakllar chizish

putpixel(x, y, rang) – nuqta chizish funksiyasi

Funksiya ko'rsatilgan rangda (x,y) koordinatali nuqta hosil qiladi.

Masalan: `putpixel(10,20,RED)` buyrug'i qizil rangli (10;20) koordinatali nuqta chizadi. Bu yerda rang nomi RED o'rniga sonda ifodalangan kodini qo'ysa ham bo'ladi. 1- jadvalda ranglar nomi va ularga mos kodlari keltirilgan.

Eslatma: Rang nomlari doimo katta harflarda yozilishi lozim.

getpixel(x,y) – (x,y) koordinatali nuqta rangini aniqlash funksiyasi.

funksiya (x,y) koordinatali nuqta rangi kodiga teng qiymatni oladi.

setcolor(color) – rang tanlash funksiyasi

Bu funksiya tomonidan tayinlangan rang keyingi shunday buyruqgacha amal qiladi.

line(x1,y1,x2,y2) - kesma chizish funksiyasi.
Bu funksiyasi bizga to'g'ri chiziq kesmasini chizishga mo'ljallangan bo'lib, bunda (x1;y1) kesma boshining (x2;y2) kesma oxirining koordinatalari.

A(x ₁ , y ₁)		B(x ₂ , y ₂)		
Rang nomi	Rang kodi	Rang nomi	Rang kodi	Rang nomi
BLACK(qora)	0	BROWN(malla)	6	LIGHTRED(och ko'k)
BLUE(ko'k)	1	LIGHTGRAY(och ko'kish)	7	LIGHTMAGENTA(och binafsha)
GREEN(yashil)	2	DARKGRAY(och qora)	8	YELLOW(sariq)
CYAN(och ko'k)	3	LIGHTBLUE(och ko'k)	9	WHITE(oq)
RED(qizil)	4	LIGHTGREEN(och yashil)	10	
MAGENTA(binafsha)	5	LIGHTCYAN(to'q ko'k)	11	

1-jadval

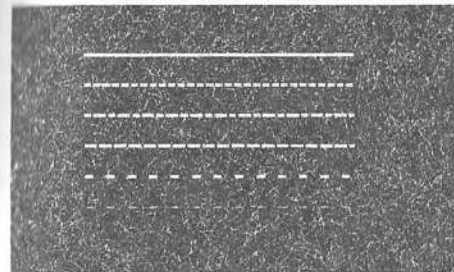
Setlinestyle(x,y,z) chiziq chizish usulini tayinlash funksiyasi

Funksiya yordamida chizilayotgan shakl chegaralarini chizish uchun usul tanlanadi. Bunda tanlangan x usul nomi, z chiziq qalinligi, y esa butun son bo'lishi lozim. Chizish usullari 0 dan 4 gacha kodlangan.

Quyidagi dasturni tahlil qiling

```
#include <graphics.h>
int main()
{
    int gd = DETECT, gm, c, x = 100, y = 50;
    initgraph(&gd, &gm, "C:\\TC\\BGI");
    for ( c = 0 ; c <= 5 ; c++ ) { setlinestyle(c, 100, 3); line(x, y,
    100, y);
        y = y + 25; }
    getch(); closegraph();
    return 0;
}
```

Dastur natijasi quydagicha ko'rinishda bo'ladi:



arc(x,y,α,β,r) – aylana yoyini chizish funksiyasi

Bu funksiya markazi (x;y) koordinatali nuqtada bo'lgan aylana yoyini chizadi. α yoy chizish boshlanadigan burchak, β yoy chizish tugatiladigan burchagi, r radius. α va β lar o'rnida sonlar kiritiladi, C++ bu qiymatlarni gradus deb qabul qiladi.

Eslatma : Bu yerda (x,y,α,β,r) parametrlar butun turga (int turiga) mansub bo'lishi kerak.

Agar bu funksiyada α=0, β=360 deb olinsa, markazi (x;y) koordinatali nuqtada bo'lgan r radiusli aylana chizadi. Bunda α va β manfiy qiymatlar ham qabul qiladi. Bu holda chizish soat strelkasi yo'nalishida bo'ladi aks holda soat strelkasiga qarama-qarshi yo'nalishda bo'ladi.

circle(x, y, r) – aylana chizish funksiyasi

Bu funksiya yordamida markazi (x;y) koordinatali nuqtada bo'lgan r radiusli aylana chiziladi. Bu funksiya parametrlari ham butun sonlar bo'lishi lozim.

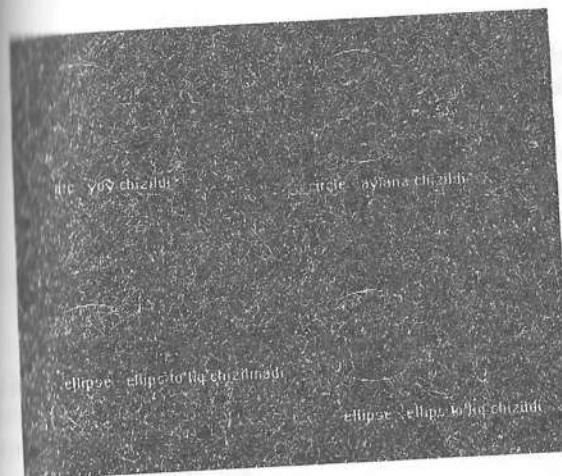
ellipse(x, y, α , β , a, b) – ellips va uning yoyini chizish funksiyasi

Funksiya markazi (x;y) koordinatali nuqtada bolgan ellips yoyini chizadi. α yoy chizish boshlanadigan burchak, β yoy chizish tugatiladigan burchak, α va β lar o'rnida sonlar kiritiladi, C++ taqiymatlarni gradus deb qabul qiladi. a – ellipsning OX o'qi bo'ylab, b – OY o'qi bo'ylab yo'nalgan yarim o'qlari.

Eslatma: bu funksiya yordamida ellipsni to'la chizish uchun α va $\beta=360$ deb olish kerak. Chunki chizish 0° dan boshlanib, 360° ga tugaydi va natijada ellips yoyi chizish boshlanadigan burchakka qaytib kelib ellipsni to'liq hosil qiladi. Quyidagi dasturda yuqoridagi funksiyalardan foydalanish namoyish qilingan. Siz ranglar jilosiga alohida e'tibor qiling.

```
#include <graphics.h>
#include <conio.h>
main()
{ int gd = DETECT, gm;
  initgraph(&gd, &gm, "C:\\TC\\BGI");
  setcolor(1); arc(100, 100, 0, 135, 50); // arc yoy chizish funksiyasi
  setcolor(4); outtextxy(70, 150, "arc - yoy chizildi"); // tushuntirish
  matni
  setcolor(14); circle(350, 100, 50); // circle - aylana chizish
  funksiyasi
  outtextxy(300, 160, "circle - aylana chizildi");
  setcolor(15); ellipse(100, 300, 0, 180, 50, 40);
  outtextxy(70, 320, "ellipse - ellips to'liq chizilmadi ");
  ellipse(350, 300, 0, 360, 50, 40);
  outtextxy(320, 360, "ellipse - ellips to'liq chizildi ");
  getch(); closegraph(); return 0;
}
```

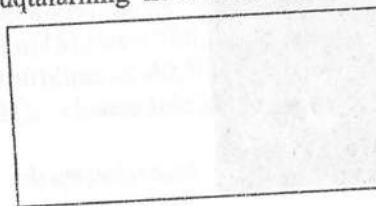
Dasturdagi `outtextxy(x, y, "matn")` funksiyasidan tushuntirish matnlarini ekranning (x;y) koordinatali nuqtasidan chiqarish uchun foydalanildi. Dastur natijasi quyidagicha



rectangle(x₁, y₁, x₂, y₂) – to'g'ri to'rtburchak chizish funksiyasi

Bu funksiya yordamida chap yuqori burchagi (x₁;y₁) nuqtada va o'ng quyi burchagi (x₂;y₂) nuqtada bo'lgan to'g'ri to'rtburchak chiziladi. Nuqtalarning koordinatalari butun sonlar bo'lishi lozim.

A(x₁, y₁)



B(x₂, y₂)

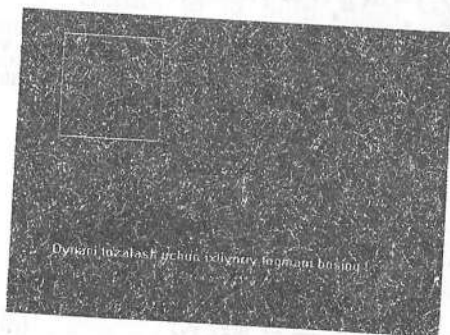
cleardevice() - javoblar darchasini tozalash funksiyasi

Funksiya yordamida javoblar darchasi oldin chizilgan grafiklardan tozalanadi. Masalan, dasturda bir nechta shakllarni ketma - ket tozalanadi. Natijani ko'rish uchun dasturni chizish buyruqlari berilgan bo'lsin. Natijani ko'rish uchun dasturni ishga tushirgan vaqtimizda oldingi berilgan buyruqlar natijasi paydo bo'ladi. Ular uctma-ust yoki yonma-yon tig'iz joylashib qolishi va natijalarni tahlil qilishni qiyinlashtirishi mumkin. Funksiya o'zidan oldingi buyruqlar natijalarini o'chirib yuborish imkonini beradi. Dasturga bu buyruq kiritilgandan so'ng, undan oldin joylashgan buyruqlar natijalari ko'rsatilmaydi. Ya'ni C++ dasturda bu buyruqni uchratgach, javoblar darchasini barcha ma'lumotlardan tozalaydi.

delay(n) - kutish funksiyasi

Bu funksiya ishga tushgandan so'ng keyingi burchaklar chizilish ketligiga o'tish uchun n millisekund kutib turadi va keyingi burchakni o'tiladi.

```
#include <graphics.h>
#include <conio.h>
int main()
{ int gd = DETECT, gm;
  initgraph(&gd, &gm, "C:\\TC\\BGI");
  rectangle(100,100,200,200); delay(1000);
  outtextxy(100,300,"Oynani tozalash uchun ixtiyoriy tugmani bosish!");
  getch(); cleardevice(); getch(); closegraph();
  return 0;
}
```



Ushbu dastur ijro etilishi jarayonida to'rtburchak chizilgandan so'ng 1000 millisekund kutib, so'ng tushuntirish matni chiqadi hamda klaviaturadan biror belgi kiritilishi bilan cleardevice(); operatori ijro etiladi va oyna tozalanadi.

getmaxx va getmaxy protseduralari

getmaxx - joriy rejim va drayverlar uchun gorizontalar nuqtalar soni, getmaxy - joriy rejim va drayverlar uchun vertikal nuqtalar soni. Bu protseduralar yordamida kompyuterning uzi ekrandagi maksimal nuqtalar sonini aniqlaydi.

Bu funksiyalar mos ravishda OX va OY o'qlaridagi eng katta nuqtalarni oladi. Masalan, putpixel(getmaxx/2, getmaxy/2, 4) bo'yicha ekranning o'rtasida qizil rangli nuqta o'rnatiladi.

setbkcolor(color) funksiyasi

Funksiyadan javob darchasi uchun fon rangini tanlash maqsadida qo'llaniladi.

linere(x,y) va lineto(x,y) funksiyasi

linere(x,y) funksiyasi (x,y) koordinatali nuqtadan joriy nuqttagacha kesma chizadi, lineto(x,y) funksiyasi esa joriy nuqtadan (x,y) koordinatali nuqttagacha kesma chizadi. Masalan, quyidagi dastur natijalarini tahlil qiling.

```
#include <iostream.h>
#include <graphics.h>
#include <conio.h>
main()
{ int gd = DETECT, gm, a, b;
  initgraph(&gd, &gm, "C:\\TC\\BGI");
  setcolor(15); linere(60,50); lineto(300,20);
  linere(20,150); linere(-40,50);
  getch(); closegraph(); return 0;
}
```

drawpoly(n,p) - ko'burchak chizish funksiyasi

drawpoly(n,p) funksiyasi yordamida ko'pburchaklar chizish mumkin. Bunda n - ko'pburchak uchlari soni, p - ko'pburchak uchlarning koordinatalari. n ko'pburchak uchlardan bitta ko'p qilib tanlanadi, chunki birinchi uchlarning koordinatalari ikki marta ko'rsatiladi, aks holda yopiq ko'pburchak hosil bo'lmaydi. Uchlarning koordinatalari esa o'lchamli yoki o'lchamsiz massivga initsializatsiyalanadi. Funksiya o'sha n ta nuqtalarni ketma - ket tutashtirib ko'pburchak chizadi. Masalan, quyidagi dastur uchburchak va beshburchaklar chizadi.

```
#include <graphics.h>
#include <conio.h>
main()
```

```

{
    int gd=DETECT, gm, k[]={320,150,440,340,230,340,320,150};
    d[]={100,40,200,20,300,80,210,200,150,150,100,40};
    initgraph(&gd, &gm, "C:\\TC\\BGI");
    drawpoly(4,k); drawpoly(6,d);
    getch(); closegraph(); return 0;
}

```

Quyidagi savollarga jabob bering

1. putpixel funksiyasi qanday shaklni chizish uchun ishlatiladi?
2. Grafik muhitda tayin nuqtaning rangi qanday aniqlanadi?
3. Tekislikda chiziladigan shakllar uchun rang qaysi funksiya yordamida tayinlanadi?
4. circle() funksiyasidan foydalanmasdan ham aylana chizishning necha xil usulini bilasiz?
5. Grafik muhitni yuklovchi funksiya qaysi?
6. rectangle() funksiyasi qanday shakllarni chizish uchun ishlatiladi?
7. line() funksiyasi qanday shaklni chizish uchun ishlatiladi?
8. drawpoly(n,m) funksiyasida n va m lar qanday parametrlar va ularning qiymatlari qanday kiritiladi?
9. getch() funksiyasi qanday vazifani bajaradi?
10. Qachon chizilgan shakl ko'rinmaydi?
11. Aylana chizish funksiyasining umumiy ko'rinishini yozing va parametrlarining vazifasini tushuntiring.
12. Dasturning javob oynasini tozalovchi funksiya qaysi?
13. Grafik muhitda javob darchasidan dastur darchasiga qanday o'tiladi?
14. Ellips chizish funksiyasi yordamida qanday qilib aylana va ellips yoylari chiziladi?
15. Uchlari (2;1), (3;2) va (4;-5) nuqtalarda bo'lgan uchburchakni qanday usullarda chiza olasiz?
16. ellipse(a,b,c,d,e,f) funksiyasida a, b, c, d, e, f parametrlari vazifasini tushuntiring.
17. Ellips chizish funksiyasining umumiy ko'rinishini yozing va parametrlarining vazifasini tushuntiring.
18. delay(500); buyrug'i vazifasini tushuntiring.
19. Grafik muhit qanday qilib joriy qilinadi?
20. Ekranda tasvir qanday hosil qilinadi?

11. C++ grafik muhitini joriy qilish uchun qaysi funksiyalar kutubxonasidan foydalaniladi?
12. Ekranning tayin joyida tasvir hosil qilish uchun nima qilish kerak?
13. arc() funksiyasi yordamida aylana chizish mumkinmi? Mumkin bo'lsa qanday qilib?
14. Chiziqlarni har xil usullarda chizish uchun qaysi funksiyadan foydalaniladi?
15. linerel() va lineto() funksiyalarining vazifalarini tushuntiring?
16. linerel() va lineto() funksiyalarining qanday farqlari bor?
17. C++ grafik muhitida necha xil rangdan foydalaniladi?
18. C++ grafik muhitidan matn muhitiga qanday o'tiladi?
19. closegraph() funksiyasining vazifasi nima?
20. closegraph() funksiyasi bo'lmasa ham dastur grafik muhitda ishlaydimi?
21. putpixel() funksiyasi yordamida uzunligi 2 sm ga teng kesma chizish mumkinmi, agar mumkin bo'lsa qanday chiziladi?

Quyidagi masalalarning dasturlarini tuzing

1. Ekranda o'tkir, o'tmas, to'g'ri burchaklar chizing.
2. Ekranda parallelogramm, trapetsiya, muntazam uchburchak chizing.
3. Line funksiyasi yordamida to'rtburchak chizish dasturini tuzing.
4. To'rtburchak chizing va diagonalalarini o'tkazish dasturini tuzing.
5. Uchburchak chizib medianasini o'tkazish dasturini tuzing.
6. Kvadratga ichki aylana chizish dasturini tuzing.
7. Kvadratga tashqi aylana chizish dasturini tuzing.
8. Ikkita urinivchi aylanalar chizish dasturini tuzing.
9. Ikkita konsentrik aylanalar chizish dasturini tuzing.
10. To'g'ri burchakli uchburchak chizish dasturini tuzing.
11. Kvadrat va unga tashqi uchburchak chizish dasturini tuzing.
12. Aylanaga ichki chizilgan kvadrat dasturini tuzing.
13. Kvadrat va uning uchlarini tutashtiruvchi yoylar chizish dasturini tuzing.
14. Uch xil rangli aylana chizish dasturini tuzing.
15. arc funksiyasi yordamida aylana chizing.

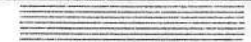
16. Vertikal va gorizontal holda yuguruvchi kesmalar hosil qilish dasturini tuzing.
17. Uy rasmini hosil qilish dasturini tuzing.
18. Mashina rasmini hosil qilish dasturini tuzing.
19. Kamalak rasmini hosil qilish dasturini tuzing.
20. Ekranda aylana hosil qiling va u har xil rangli to'viansin(jilolansin).
21. Ekranda harakatlanuvchi aylana hosil qilish dasturini tuzing.
22. Ekranda uchayotgan o'q rasmini hosil qiling.
23. Ekranda maysazor hosil qilib, unga bitta daraxt o'tqazing.
24. Ekranni tungi osmon ko'rinishiga keltiring. Bunda bir sekundda tasodifiy bitta yulduz paydo bo'lib borsin.
25. Ekranda ctul rasmini hosil qilish dasturini tuzing.
26. Eranda samolyot rasmini hosil qilish dasturini tuzing.
27. Hashamli darvoza chizish dasturini tuzing.
28. Silindr, konus, shar chizish dasturlarini tuzing.
29. Ekranda mashina faralari(chiroqlari) tunda yonib turgan holatini hosil qiling.
30. Ekranda elektron soat hosil qilish dasturini tuzing.
31. Parashut rasmini hosil qiling.

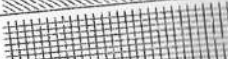
8.2 Tekislikda yassi figuralar chizish

Yassi figuralarning ichki qismi bo'yalgan bo'ladi. C++ dasturlash tilida yopiq sohani bo'yash uchun maxsus funksiya mavjud bo'lib, uning umumiy ko'rinishi quyidagicha

Setfillstyle(usul, rang) - bo'yash usullari funksiyasi

Bo'yash usulining nomi va bu usullarning sonlardagi kodi hamda ko'rinishi quyidagi jadvalda keltirilgan.

Bo'yash usuli nomi	Bo'yash usuli kodi	Natija
EMPTY_FILL	0	Soha bo'yalmasdan qoladi
SOLID_FILL	1	
LINE_FILL	2	
LTSLASH_FILL	3	

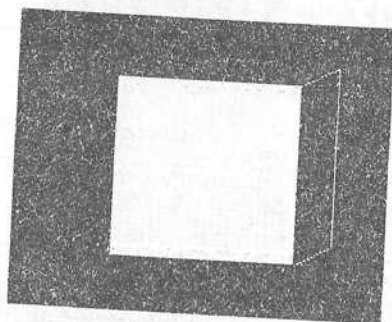
ASH_FILL	4	
BLASH_FILL	5	
BTSLASH_FILL	6	
HATCH_FILL	7	
WHATCH_FILL	8	
INTERLEAVE_FILL	9	
WIDE_DOT_FILL	10	
CLOSE_DOT_FILL	11	
USER_FILL	12	Orqa fon bilan bir xil holda bo'ladi

bar(x1, y1, x2, y2) - yassi to'g'ri to'rtburchak chizish funksiyasi

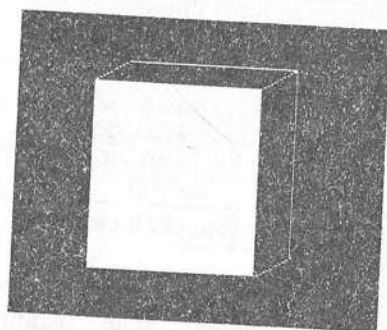
Bar funksiyasi tekislikda to'g'ri to'rtburchak chizadi. Bu yerda (x1;y1) nuqta to'g'ri to'rtburchakning yuqori chap uchining koordinatalari, (x2;y2) esa chap quyi uchining koordinatalari. Bu funksiyaga bo'yash usuli va rang berish uchun setfillstyle() funksiyasidan foydalaniladi.

bar3d(x1,y1,x2,y2,r,a) - parallelopiped chizish funksiyasi
bar3d() funksiyasi to'g'ri burchakli parallelopiped chizadi. Bu yerda (x1;y1) parallelopiped oldingi qismidagi yog'ining chap yuqori uchining koordinatalari, (x2;y2) esa o'ng quyi uchining koordinatalari, r - uning qalinligi, ya'ni yon yog'ining kengligi. a - parallelopiped ustki yog'ining bor yoki yo'qligini ta'minlovchi parametr. Agar a=0 bo'lsa parallelopiped ustki yog'i chizilmasdan qoladi, agar noldan farqli butun son bo'lsa parallelopiped ustki yog'i chiziladi. Bunda parallelopipedning faqat oldiyog'i bo'yaladi. Bo'yash usuli va rangini tanlash uchun setfillstyle() funksiyasidan foydalaniladi. Quyida dastlab a=0 bo'lgan hol, so'ngra a=1 bo'lgan hollar tegishli buyruqlar orqali namoyish qilingan.

`bar3d(100,100,200,200,20,0);` buyrug'i natijasi quydagicha



`bar3d(100,100,200,200,20,1);` dastur natijasi quydagicha



fillellipse(x, y, a, b) – yassi ellips chizish funksiyasi

Bu funksiya yordamida yassi ellipslar chiziladi. Bu yerda (x,y) ellips markazining koordinatalari, *a* va *b* lar esa ellipsning yarim o'qlari. Ellipsni bo'yash usuli va rangini tayinlash uchun `setfillstyle()` funksiyasidan foydalaniladi. Agar bu funksiyada *a*=*b* bo'lsa doira chiziladi.

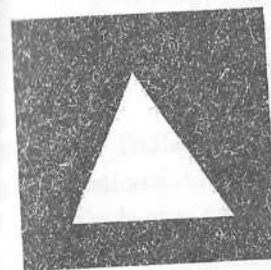
fillpoly funksiyasi;

Bu funksiya `drawpoly()` funksiyasi kabi ko'pburchak chizadi, faqat undan farqi yassi ko'pburchak, ya'ni ichi bo'yalgan ko'pburchak chizadi. `Setfillstyle()` funksiyasidan foydalanib bo'yash usuli va rangi tayinlanadi. Funksiyaning umumiy ko'rinishi quyidagicha.

fillpoly(x, p);
bu yerda *x* va *p* parametrlarning vazifasi `drawpoly()` funksiyasidagi *x* va *p* parametrlarning vazifasi bilan bir xil. Misol sifatida quyidagi dasturni va uning natijasini tahlil qiling.

```
#include <graphics.h>
#include <conio.h>
main()
{
    int gd=DETECT, gm, p[]={320,150,440,340,230,340,320,150};
    initgraph(&gd, &gm, "C:\\TC\\BGI");
    fillpoly(4, p); getch(); closegraph();
    return 0;
}
```

Dastur natijasi quyidagi shaklda aks etgan.



floodfill maydonni bo'yash funksiyasi

Funksiyaning umumiy ko'rinishi quyidagicha
floodfill(x, y, border)

Funksiya yopiq maydonning ichini tayinlangan rangga bo'yash uchun mo'ljallangan. Bunda (x,y) maydon ichidagi nuqtaning koordinatalari. Bunda (x,y) maydon chegarasining rangi. U bo'yalishi kerak bo'lgan maydon(soha) chegarasi rangi bilan bir xil bo'lishi lozim. Aks holda xatolik yuz berishi mumkin. Bu funksiyada (x,y) nuqtaning maydon ichida yoki tashqarisiga tegishiligi tekshirib ko'riladi. Agar maydon ichida tegishli bo'lsa maydon ichkarisi, aks holda maydon tashqarisi bo'yladi. Bo'yalish usuli `setfillstyle()` funksiyasi orqali ko'rib chiqiladi. Agarda `setfillstyle()` funksiyasi kiritilmagan bo'lsa, avtomatik tarzda oq rangga bo'yaladi, ya'ni bu holda avtomatik tarzda oq rang tanlanadi.


```
#include <graphics.h>
#include <conio.h>
main()
{int gd = DETECT, gm;
  initgraph(&gd,&gm,"C:\\TC\\BGI");
  setcolor(RED);
  circle(100,100,50);setfillstyle(4,RED);
  floodfill(100,100,RED);
  getch(); closegraph();return 0;
}
```

Yuqoridagi misolda qizil rang bilan chizilgan aylana ichki bo'yalishi keltirilgan. Aylana qizil rangda chizilgani uchun ham `floodfill(x,y,border)` funksiyasida `border` o'rnida `qizil(RED)` rangi tanlangan.

Sektor chizish funksiyasi

Bu funksiya quydagicha e'lon qilinadi va u sektor chizadi.

sector(x,y,α,β,a,b);

bunda (x,y) sektor markazining koordinatalari α chizish boshlanadigan, β esa chizish tugallanadigan burchak kattaliklari bo'lib, ular yuqorida qayd qilinganidek butun son ko'rinishida kiritiladi, ammo C++ ularni graduslarda berilgan deb tushunadi. Shuningdek, mos ravishda a va b lar yarim o'qlaridir(radiuslari). Masalan, quyidagi dastur sariq rangli sektor chizadi.

```
#include <graphics.h>
#include <conio.h>
main()
{ int gd = DETECT, gm;
  initgraph(&gd, &gm, "C:\\TC\\BGI");
  setcolor(4);
  setfillstyle(4,14);sector(200,200,45,135,150,100);
  getch(); closegraph(); return 0;
}
```

Grafik muhitda matnlar bilan ishlash

outtextxy(x,y,"matn") – grafik muhitda matn chop etish funksiyasi

Bu funksiya yordamida izohlar, tushuntirish matnlari chop etiladi. Buning uchun chop etilayotgan matn qo'shtirnoq ichiga olib yoziladi. **outtextxy(x,y,"matn")** funksiyasi (x,y) koordinatali nuqtadan qo'shtirnoq ichiga olib yozilgan matnni ekranda chop etadi. Matn rangi `setcolor()` funksiyasi orqali kiritiladi.

settextstyle(font,direction,charsize); - matn yozish usuli funksiyasi

Bunda **font** matnning yozilish usuli, **direction** esa yozilish holati ya'ni bu funksiya ikkita qiymatni qabul qila oladi: 1) **HORIZ_DIR(0)** bunda matn gorizantal holatda chapdan o'ngga qarab yoziladi. 2) **VERT_DIR(1)** bunda matn vertikal holatda pastdan yuqoriga qarab yoziladi. **charsize** esa matn shriftining o'lchami.

Font (nomi)	Son qiymati
DEFAULT_FONT	0
TRIPLEX_FONT	1
SMALL_FONT	2
SANS_SERIF_FONT	3
GOTHIC_FONT	4
SCRIPT_FONT	5
SIMPLEX_FONT	6
TRIPLEX_SCR_FONT	7
COMPLEX_FONT	8
EUROPEAN_FONT	9
BOLD_FONT	10

```
#include <graphics.h>
#include <conio.h>
main()
```



```

{
    int gd = DETECT, gm;
    initgraph(&gd, &gm, "C:\\TC\\BGI");
    setfillstyle(9,4);fillellipse(200,200,100,100);setcolor(14);
    settextrstyle(5,0,4);outtextxy(120,310,"bu doira");
    settextrstyle(5,1,4);outtextxy(310,320,"uning radiusi r=100");
    getch(); closegraph(); return 0;
}

```

Quyidagi savollarga javob bering

1. Bar3d() funksiyasidan foydalanib qanday to'rtburchakli soha chizish mumkin? Uning parametrlarini tushuntiring.
2. Maydon ichi qanday bo'yaladi?
3. Parallelopiped chizish funksiyasini yozib, uning parametrlari vazifasini tushuntiring.
4. Doira chizishning qanday usullarini bilasiz?
5. Doiraviy sektor chizuvchi funksiyaning umumiy ko'rinishini yozib, uning parametrlarini tushuntiring.
6. Yassi figuralarni chizishda chizilayotgan soha ichi qanday bo'yaladi?
7. fillpoly() bilan floodfill() funksiyalari farqi nimada?
8. Sektor doiraviy sektor bo'lishi uchun sector() funksiyasining qaysi parametrlari qanday tanlanadi?
9. Ko'pburchak chizish funksiyasida ko'pburchakning uchlarining koordinatalari qanday kiritiladi?
10. bar() va bar3d() funksiyalari farqini tushuntiring.
11. Yassi figura deyilganda qanday figurani tushunasiz?
12. Sektor yassi ellips sektori bo'lishi uchun sector() funksiyasining qaysi parametrlari qanday tanlanadi?
13. Grafik muhitda matn qaysi funksiya yordamida chop etiladi? Uning parametrlari vazifasini tushuntiring.
14. setfillstyle() funksiyasi qaysi buyruqgacha o'z ta'sir doirasini saqlaydi?

15. fillellipse() funksiyasi yordamida ikki ekranning o'rtasida + belgisini hosil qilsin.
16. Grafik muhitda matnning ekrandagi joyini tushuntiring.
17. Yozilayotgan matn uchun usul qanday belgilanadi?
18. fillellipse funksiyasi yordamida doira qanday chiziladi?
19. fillellipse funksiyasi yordamida aylana qanday chiziladi?

Quyidagi masalalarning dasturlarini tuzing

1. Yassi kvadrat chizish dasturini tuzing.
2. Doira yuqori yarmini chizish dasturini tuzing.
3. Kvadrat va unga ichki doira chizish dasturini tuzing.
4. Doira va unga ichki kvadrat chizish dasturini tuzing.
5. 15 xil rangdagi ketma ket doirachalar chizish dasturini tuzing.
6. Diagramma chizish dasturini tuzing.
7. Doiraviy diagramma chizish dasturini tuzing.
8. Harakatlanuvchi parallelopiped chizish dasturini tuzing.
9. Shunday dastur tuzingki, har xil o'lchamli yassi doiralar ekranning tasodifiy joylarida paydo bo'lsin.
10. Kvadrat va unga ichki aylana chizish dasturini tuzing.
11. Aylananing tashqi qismini bo'yash dasturini tuzing.
12. Xalqa rasmini chizish dasturini tuzing.
13. Ichma – ich 15 xil rangdagi xalqalarni chizish dasturini tuzing.
14. Yarim oy shaklini chizish dasturini tuzing.
15. O'zbekiston bayrog'ini chizish dasturini tuzing.
16. Rangi ketma – ket almashinuvchi doira chizish dasturini tuzing, ya'ni u jilolansin.
17. sector() funksiyasi yordamida doira chizish dasturini tuzing.
18. Kvadratga tashqi doiralar chizish dasturini tuzing.
19. Parashut rasmini chizing.

20. Uy rasmini chizing.
21. Doira uning elementlarini chizib, elementlari uchun tushuntirish matnlarini bering.
22. Shunday dastur tuzingki, kodini kiritish yo'li bilan bir nechta figuralarni tanlab chizish mumkin bo'lsin.
23. Yassi to'g'ri to'rtburchakka ichki ellips chizing.
24. Muntazam oltiburchak chizing

Foydalanilgan adabiyotlar

1. Страуструп Б. Язык программирования C++. Специальное издание. – Москва. 1997. -1055 с.
2. Фролов А.Б., Фролов Г.В. Язык C#. Самоучитель. – Москва. 2003.-559с.
3. Подбельский В.В. Язык СИ++ : Учебное пособие. 5-е изд. М.: Финансы и статистика. 2003.-560 с.
4. Стефан.Р.Дэвис.С++ для “Чайников”. 4-изд.–М. 2003.-335 с.
5. Герберт Шилд. С++. Базовый курс. Москва, Издательский дом “Вильямс”. 2010.-621 с.
6. Джессе Либерти. С++ за 21 день. 3-изд. –М. 2003.-815 с.
7. Мадрахимов Ш.Ф., Гайназаров С.М. С++ тилида программалаш асослари. Услубий қўлланма. Тошкент. 2009.-200 с.
8. Madрахimov SH.F., Ikramov A.M., Babajanov M.R. С++ тилида dasturlash bo'yicha masalalar to'plami. Услубий қўлланма. Тошкент. 2013.-160 с.
9. Турский В. Методология программирования. – М.: Мир. 1981. –547 с.
10. Буч Г. Объектно-ориентированное проектирование примерами применения: пер. с англ. – М.Конкорд, 1992. – 214 с.
11. Жоголев Е.А. Система программирования использованием библиотеки подпрограмм // Система автоматизация программирования.– М.: Физматгиз, 1961. – С. 1-52.
12. Информатика. Базовый курс / С.В.Симонович и др. – СПб. Питер, 1999. –640 с.
13. Кнут Д.Э. Искусство программирования: учеб. пособие М.: Вильямс, 2000. – 832 с.
14. Программирование на С++ и C#. [Электронный ресурс] Режим доступа: <http://trubetskoy1.narod.ru/> (дата обращения 30.09.2011)
15. Eshtemirov S. Nomozov F. С++ dasturlash tili. Uslubiy qo'llanma. Samarqand 2016. -146 b.
16. Nazarov F. С++ tilida dasturlash asoslari. Uslubiy qo'llanma. Samarqand 2017. -160 b.

S.Eshtemirov, F.M.Nazarov

**ALGORITMLASH VA DASTURLASH
ASOSLARI**

*(5130200-Amaliy matematika va informatika, 5140200-Matematika
yonalishlari uchun)*

O'quv qo'llanma

Muharrir
Musahhih
Texnik muharrir

J. Bozorova
L. Xoshimov
N.Isroilov

2019 yil 14 dekabrda tahririy-nashriyot bo'limiga qabul qilindi.

2019 yil 15 dekabrda original-maketdan bosishga ruxsat etildi.

Qog'oz bichimi 60x84_{1/8}. "Times new roman" garnituras. Offset qog'ozi.

Shartli bosma tabog'i -13,0

Adadi 100 nusxa. Buyurtma № 12/7.

ISBN-978-9943-6178-9-6

SamDU tahririy-nashriyot bo'limida chop etildi.

140104, Samarqand sh., Universitet xiyoboni, 15.



978-9943-6178-9-6



9789943617896