Mundarija

C++ algoritmik tilining asosiy tushunchalari

C++ algoritmik tilining alifbosi quyidagilardan iborat:

- katta va kichik lotin harflari;
- 0 dan 9 gacha raqamlari;
- maxsus belgilar (+,-,*,/,=,>,<,{,},[,],') ni o'z ichiga oladi.

C++ tilida **so'z** deb bir nechta belgilar ketma – ketligi tushuniladi. Xizmatchi so'z deb C++ tilidagi standart nom tushuniladi. Bu nom maxsus ma'noni anglatadi va uni ma'lumotlarga berib bo'lmaydi. Masalan: **int, float, for, while** va hokazo.

C++ tilida ma'lumotlarning elementlari bo'lib o'zgaruvchilar, o'zgarmaslar, izohlar xizmat qiladi.

O'zgaruvchi. Xotiraning nomlangan qismi bolib, o'zida ma'lum bir toifadagi qiymatlarni saqlaydi. O'zgaruvchining nomi va qiymati bo'ladi. O'zgaruvchining nomi orqali qiymat saqlanayotgan xotira qismiga murojaat qilinadi. Programma ishlashi jarayonida o'zgaruvchining qiymatini o'zgartirish mumkin. Har qanday o'zgaruvchini ishlatishdan oldin, uni e'lon qilish lozim.
Quyida butun sonlardan foydalanish uchun b, haqiqiy sonlardan foydalanish uchun h o'zgaruvchisi e'lon qilingan:

```
int b;
float h;
```

O'zgarmaslar (const)
Hisoblash jarayonida qiymatini o'zgartirmaydigan kattaliklarga aytiladi.
float const pi = 3.14;
Izohlar. Programmaning ma'lum qismini tavsiflash uchun ishlatiladi va bu qatorda hech qanday amal bajarilmaydi, ya'ni programmaning biror qismini yaxshiroq tushuntirish uchun xizmat qiladi. Izoh "/*" va "*/" simvollari orasida beriladi.
/ Bu yerga izoh yoziladi. */*
Bundan tashqari bir satrli izohlardan ham foydalanish mumkin. Buning uchun izoh boshiga "//" belgisi qo'yiladi.

Operator. Tilning yakunlangan jumlasi hisoblanadi va ma'lumolar taxlilining tugallangan bosqichini ifodalaydi. Operatorlar nuqtali vergul “;” bilan ajratiladi. Ya'ni “;” operatorning tugallanganligini bildiradi. C++ da operatorlar progammada keltirilgan ketma - ketlikda bajariladi.

Identifikator. Programmist tomonidan programma elementlari (funksiya, o'zgaruvchilar, o'zgarmaslar …) uchun ixtiyoriy tanlangan nom.

Identifikator tanlaganda quyidagilarga ahamiyat berish kerak:

- Identifikator lotin harflaridan boshlanishi shart;
- Ikkinchi simvoldan boshlab raqamlardan foydalanish mumkin;
- C++ da katta kichik harflar farq qiladi. Ya'ni quyidagilarning har biri alohida identifikator hisoblanadi: KATTA, katta, KaTTa, kAttA, Katta, KattA, ...
- Probel C++ da so'zlarni ajratish uchun ishlatiladi. Shuning uchun identifikatorda probeldan foydalanib bo'lmaydi;
- Xizmatchi (**int, float, for, while** kabi) so'zlardan identifikator sifatida foydalanib bo'lmaydi;

C++ tilining kalit so'zlariga quyidagilar kiradi:

asm, auto, break, case, catch, char, class, const, continue, default, delete, do, double, else, enum, explicit, extern, float, for, friend, goto, if, inline, int, long, mutable, new, operator, private, protected, public, register, return, short, signed, sizeof, static, struct, swith, template, this, throw, try, typedef, typename, union, unsigned, virtual, void, volatile, while.

Protsessor registrlarini belgilash uchun quyidagi so'zlar ishlariladi:

_AH, _AL, _AX, _EAX, _BH, _BL, _BX, _EBX, _CL, _CH, _CX, _ECX, _DH, _DL, _DX, _EDX, _CS, _ESP, _EBP, _FS, _GS, _DI, _EDI, _SI, _ESI, _BP, _SP, _DS, _ES, _SS, _FLAGS.

Eslatma. Identifikator tanlashda birinchi belgi sifatida "_" belgisidan foydalanmaslik tavsiya etiladi.

C++ da programma funksiya yoki funksiyalardan tashkil topadi. Agar programma bir nechta funksiyadan iborat bo'lsa, bir funksiyaning nomi **main** bo'lishi shart. Programma aynan main funksiyasining birinchi operatoridan boshlab bajariladi.

Funksiyaning aniqlashishi quyidagicha bo'ladi:

```
qaytariluvchi_qiymat_toifasi funksiya_nomi ( [parametrlar] )
{
   funksiya tanasini tashkil qiluvchi operatorlar
}
```

Qoida bo'yicha funksiya qandaydir bir qiymatni hisoblash uchun ishlatiladi. Shuning uchun funksiya nomi oldidan, funksiya qaytaradigan qiymat toifasi yoziladi. Agar funksiya hech qanday qiymat qaytarmaydigan bo'lsa, **void** toifasi yoziladi. Agar funksiya qaytaradigan qiymat toifasi yozilmagan bo'lsa, **int** (butun) toifali qiymat qaytariladi deb qabul qilinadi.

Funksiyalar bilan keyingi mavzularda batafsil tanishamiz.

C++da oddiy matnni ekranga chiqaruvchi programmani ko'rib chiqamiz

```
1 // Muallif: Qudrat Abdurahimov
2 // Sana:    23 fevral 2011 yil
3 // Maqsad:  Matnni ekranga chiqaruvchi programma
4
5 #include <iostream> // ekranga ma'lumot chiqarish uchun
6
7 int main()
8 {
9   std::cout << "Assalomu alaykum bo'lajak programmist!\n";
10
11  return 0;
12 }
```

Har bir satrni o'rganib chiqamiz:

1, 2, 3 - satrlar izoh hisoblanadi. Malakali programmistlar har qanday programma muallif, programmaning tuzilish sanasi va maqsadini ifodalovchi izoh bilan boshlanishini maslahat berishadi.

4, 6, 10 - satrlar bo'sh satrlar hisoblanadi. Bosh satrlar programma qismlarini bir - biridan ajratib qo'yish uchun ishlatiladi. Programma qismlarining bir - biridan ajralib turishi, programma o'qilishini osonlashtiradi.

5 - satrda, klaviaturadan ma'lumotlarni kiritish va ekranga chiqarish uchun **<iostream>** sarlavha fayli programmaga qo'shilyapti. Bu satr klaviatura orqali ma'lumot kirituvchi va ekranga nimadir chiqaruvchi har qanday programmada bo'lishi shart. Aks xolda xato sodir bo'ladi.

Agar sizning kompilyatoringiz eski bo'lsa, unda **<iostream.h>** yozishingiz lozim bo'ladi.

"// ekranga ma'lumot chiqarish uchun" yozuvi bir satrli izoh hisoblanadi.

7 - satrda butun toifadagi qiymat qaytaruvchi **main** funksiyasi berilgan. **int** xizmatchi so'zi butun toifadagi ma'lumotlarni e'lon qilishi uchun ishlatiladi.

8 - satrdagi ochuvchi figirali { funksiya tanasining boshlanganini bildiradi.

12 - satrdagi yopuvchi figirali } funksiya tanasining tugaganini bildiradi.

9 - satrda **std::cout** << orqali ma'lumotlar ekranga chiqariladi. Qo'shtirnoq ("_") orasida yozilgan ma'lumotlar satr deyiladi. Qo'shtirnoq orasida nima yozilsa, hech qanday o'zgarishsiz ekranga chiqariladi.

9 - satr oxiridagi nuqtali vergul (;) **std::cout** operatori tugallanganligini bildiradi. ; operatorlarni bir - biridan ajratish uchun xizmat qiladi. Ya'ni operator tugallanganligini bildiradi. 5 - satrdagi kabi preprotsessor amalidan keyin ; qo'yilmaydi.

11 - satrdagi **return** xizmatchi so'zi orqali funksiya 0 qiymat qaytaradi va programma muvoffaqiyatli yakunlanadi.

O'zgaruvchilarni e'lon qilish. Programmada ishlatilgan barcha o'zgaruvchilarni qaysi toifaga tegishli ekanligini e'lon qilish kerak. Ma'lulotlarni e'lon qilishning umumiy ko'rinishi quyidagicha:

toifa_nomi o'zgaruvchi;

Agar bir nechta o'zgaruvchi bir toifaga mansub bo'lsa, ularni vergul bilan ajratib berish mumkin.

Butun sonlarni ifodalash uchun **int** va haqiqiy sonlarni ifodalash uchun **float** xizmatchi so'zlaridan foydalaniladi. Bu ma'ruzada shu 2 tasini bilish bizga kifoya qiladi. Keyingi mavzuda butun va haqiqiy sonlar haqida batafsil gaplashamiz.

```
int x,y; // butun toifadagi o'zgaruvchilarni e'lon qilish
float a,b,c;// haqiaiy toifadagi o'zgaruvchilar e'lon qilish
```

Kiritish va chiqarish operatorlari. Programmada klaviatura orqali ma'lumot kiritish va ekranga chiqarish uchun preprotsessor direktivasini, ya'ni **#include <iostream>** ni programmaga qo'shish shart. Ma'lumotlarni kiritish **std::cin** >>, ma'lumotlarni chiqarish **std::cout** << operatori orqali amalga oshiriladi.

std::cin >> a;

Bu operator bajarilganda ekranda kursor paydo bo'ladi. Kerakli ma'lumot klaviatura orqali kiritilgandan so'ng **Enter** tugmasi bosiladi. cout orqali ekranga ixtiyoriy ma'lumotni chiqarish mumkin. Satrli ma'lumotlarni ekranga chiqarish uchun, ularni qo'shtirnoq orasida yozish kerak.

Quyida a va b sonlarining yig'indisini chiqaruvchi programma berilgan:

```
#include <iostream>
// standart nomlar fazosidan foydalanishni e'lon qilish
using namespace std;
int main()
{
    int a, b, c;
```

```
    cout << "a="; cin >> a;
    cout << "b="; cin >> b;

    c = a + b;

    cout << c << endl;

    return 0;
}
```

Ba'zi matematik funksiyalar:

Matemetik funksiyalardan programmada foydalanish uchun **math.h** faylini progarmmaga qo'shish kerak.

#include <math.h>

Funksiyaning **C++** da ifodalanishi	Funksiyaning matematik ifodalanishi
abs(x) - butun sonlar uchun fabs(x) - haqiqiy sonlar uchun	/x/
pow(x, y)	x^y
sqrt(X)	$\sqrt{X}$

Matematik funksiyalardan foydalanish

```
// Muallif: Qudrat Abdurahimov
// Sana:    23 fevral 2011 yil
// Maqsad:  kiritilgan son ildizini chiqaruvchi programma
#include <iostream>
#include <math.h>
using namespace std;
int main()
{
    float a;
    cout << "a="; cin >> a;
    a = sqrt(a);
    cout << a << endl;
    return 0;
}
```

Quyidagi savollarga og'izaki yoki yozma javob bering

1. C++ algoritmik tilining alifbosi nimalardan iborat?
2. O'zgaruvchi, o'zgarmaslarga ta’rif bering.
3. Operator nima?
4. Identifikator nima?
5. Ma'lumotlar qanday e'lon qilinadi?
6. Butun va haqiqiy sonlar qanday e'lon qilinadi?

Ma’ruzalar matnining elektron versiyasidan mavzuga oid 49 ta masalani ishlab chiqing.

Butun va haqiqiy sonlar

Programmistlar doim programma ishlashi jarayonida xotiradan kamroq joy talab qilishligi haqida bosh qotirishadi. Bu muammolar programmadagi o'zgaruvchilar sonini kamaytirish, yoki o'zgaruvchilar saqlanadigan yacheyka hajmini kamaytirish orqali erishiladi.

Biz butun va haqiqiy sonlarni e'lon qilishni bilamiz. Bulardan tashqari C++ da butun va haqiqiy sonlarni e'lon qilish uchun bir nechta toifalar mavjud. Ular bir - biridan kompyuter xotirasida qancha hajm egallashi va qabul qiluvchi qiymatlar oralig'i bilan farq qiladi.

Butun sonlar

Toifa ko'rinishi	Qabul qiladigan qiymatlar oralig'i	Kompyuter xotirasida egallagan hajmi
unsigned short int	0..65535	2 bayt
short int	-32768..32767	2 bayt
unsigned long int	0..42949667295	4 bayt
long int	-2147483648..2147483647	4 bayt
int (16 razryadli)	-32768..32767	2 bayt
int (32 razryadli)	-2147483648..2147483647	4 bayt
unsigned int (16 razryadli)	0..65535	2 bayt
unsigned int (32 razryadli)	0..42949667295	4 bayt

Haqiqiy sonlar

Toifa ko'rinishi	Qabul qiladigan qiymatlar oralig'i	Kompyuter xotirasida egallagan hajmi
float	1.2E-38..3.4E38	4 bayt
double	2.2E-308..1.8E308	8 bayt
long double (32 razryadli)	3.4e-4932..-3.4e4932	10 bayt

Boshqa toifalar

Toifa ko'rinishi	Qabul qiladigan qiymatlar oralig'i	Kompyuter xotirasida egallagan hajmi
bool	**true** yoki **false**	1 bayt
char	0..255	1 bayt
void	2 yoki 4	

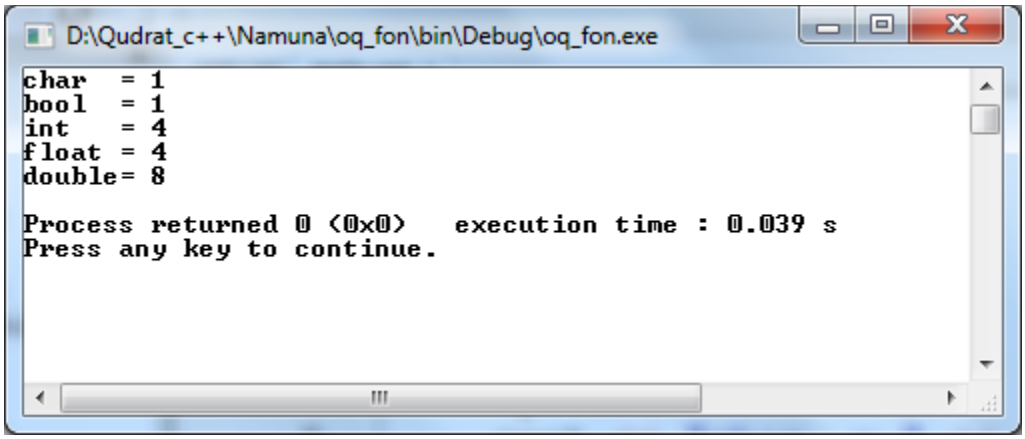
Har xil toifadagi o'zgaruvchilar kompyuter xotirasida turli xajmdagi baytlarni egallaydi. Xattoki bir toifadagi o'zgaruvchilar ham qaysi kompyuterda va qaysi operatsion sistemada ishlashiga qarab turli o'lchamdagi xotirani egallashi mumkin.

C++ da ixtiyoriy toifadagi o'zgaruvchilarning o'lchamini sizeof funksiyasi orqali aniqlash mumkin. Bu funksiyani o'zgarmasga, biror toifaga va o'zaruvchiga qo'llash mumkin.

Toifalarni kompyuter xotirasida egallagam xajmini aniqlash

```
// Muallif : Qudrat Abdurahimov
// Sana : 11 sentyabr 2011 yil
// Maqsad : Toifalarni kompyuter xotirasida egallagam xajmini
aniqlash

#include <iostream>
using namespace std;
int main()
{
    cout << "char  = " << sizeof(char)  << endl;
    cout << "bool  = " << sizeof(bool)  << endl;
    cout << "int   = " << sizeof(int)   << endl;
    cout << "float = " << sizeof(float) << endl;
    cout << "double= " << sizeof(double)<< endl;

    return 0;
}
```

Ekranga quyidagicha natija chiqariladi:



Matemetik funksiyalardan programmada foydalanish uchun **math.h** sarlavha faylini progarmmaga qo'shish kerak. **#include <math.h>**

Funksiyaning **C++** da ifodalanishi	Funksiyaning matematik ifodalanishi
1. abs(x) - butun sonlar uchun 2. fabs(x) - haqiqiy sonlar uchun 3. labs(x) - uzun butun son uchun	/x/
pow(x, y)	x^y
pow10(x)	10^x
sqrt(X)	$\sqrt{X}$
ceil(x)	haqiqiy toifadagi x o'zgaruvchisi qiymatini unga eng yaqin katta butun songa aylantiradi.
floor(x)	haqiqiy toifadagi x o'zgaruvchisi qiymatini unga eng yaqin kichik butun songa aylantiradi
cos(x)	x burchak kosinusini aniqlash. x radian o'lchovida.
sin(x)	x burchak sinusini aniqlash. x radian o'lchovida.
exp(x)	e^x
log(x)	x sonining natural logarifmini qaytaradi.
log10(x)	x sonining 10 asosli logarifmini qaytaradi.

Eslatma: Barcha trigonometrik funksiyalar radian o`lchovida beriladi.

1 - Misol: n va m natural sonlari berilgan. n sonini m soniga bo'lib, qoldiqni aniqlovchi programma tuzilsin

```
// Muallif: Qudrat Abdurahimov
// Sana:    24 fevral 2011 yil
// Maqsad:  n sonini m soniga bo'lib, qoldiqni aniqlash
#include <iostream>
int main()
{
    int n, m, qoldiq;
    cout << "n="; cin >> n;
    cout << "m="; cin >> m;

    // % qoldiqni olishni bildiradi
    qoldiq = n % m;
    cout << "Qoldiq=" << qoldiq << endl;
    return 0;
}
```

Programmalash san'ati

1. Har bir programma, muallif, sana, programma maqsadini anglatuvchi izoh bilan boshlanishi kerak.
2. Programma yozayotganda joy tashlashlarni kelishilgan, aniq bir qoida asosida olib borgan maqul.

Masalan, tabulyatsiyani 4 ta probel deb qabul qilish mumkin. Ammo bu har kimning tasavvuriga bog'liq, maqsad shuki, programma sodda oq'ishli va ko'rinishli bo'lsin.
3. Har bir verguldan keyin probel tashlang, programma oson o'qilsin.
4. O'zgaruvchilarni e'lon qilishni boshqa operatorlardan bo'sh satr bilan ajratib qo'ying.
5. (+, -, *, /) kabi amallarni har ikkala tomonidan probel qo'ying. Bu programma o'qilishini qulaylashtiradi.

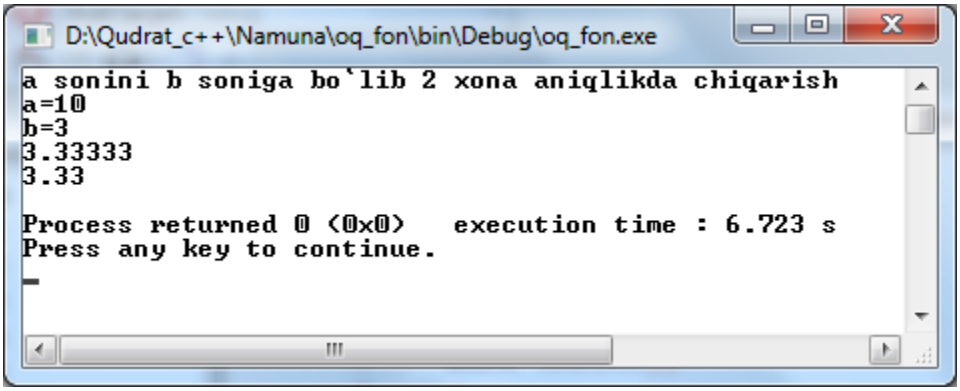
2 - Misol: n va m natural sonlari berilgan. n sonini m soniga bo'lib, butun qismini aniqlovchi programma tuzilsin

```
// Muallif: Qudrat Abdurahimov
// Sana:    24 fevral 2011 yil
// Maqsad:  n sonini m soniga bo'lib, butun qismini aniqlash

#include <iostream>

int main()
{
    int n, m, b;

    cout << "n="; cin >> n;
    cout << "m="; cin >> m;

    b = n / m;

    cout << "Butun qismi=" << b << endl;

    return 0;
}
```

3 - misol. a sonini b soniga bo`lib 2 xona aniqlikda chiqarish.

```
// Muallif: Qudrat Abdurahimov
// Sana : 8 senyabr 2011 yil
// Maqsad: Haqiqiy sonni 2 xona aniqlikda chiqarish

#include <iostream>
#include <iomanip>
// <iomanip> sarlavha faylini qo'shamiz

using namespace std;

int main()
{
    float a, b;
   cout << "a sonini b soniga bo`lib 2 xona aniqlikda chiqarish"<<endl;
    cout << "a="; cin >> a;
    cout << "b="; cin >> b;
    a = a / b;
    cout <<  a << endl;
    cout << setprecision(2) << fixed << a << endl;

   return 0;
}
```

Programma ishga tushganda ekranda quyidagicha natija chiqariladi:



4 - misol. Bir toifadan boshqasiga o'tish

c++ da bir toifadan boshqasiga o'tishning oshkor va oshkormas usullari mavjud. Oshkor ravishda toifaga keltirish uchun qavs ichida boshqa toifa nomi yoziladi.

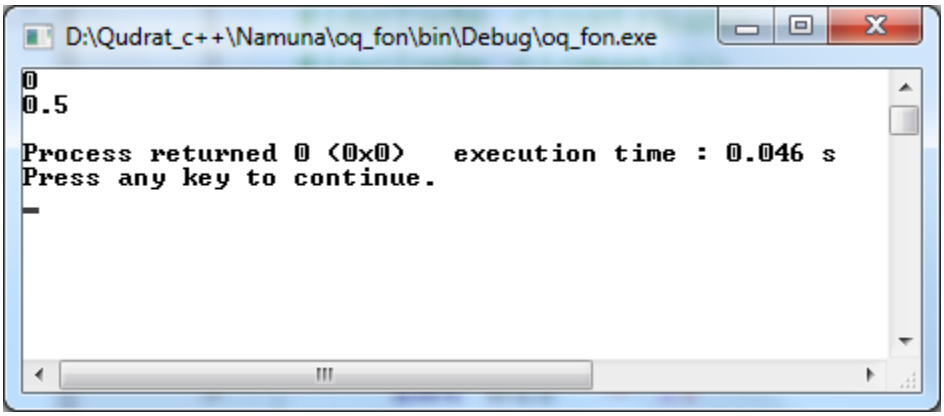
```
#include <iostream>
using namespace std;
int main()
{
    float haqiqiy = 5.57;
```

```
    int oshkor, oshkormas;
     // oshkormas ravishda butun toifaga o'tish
    oshkormas = haqiqiy;
    oshkor = (int) haqiqiy; // oshkor holda butun toifaga o'tish

    cout << "haqiqiy = " << haqiqiy << endl;
     cout << "oshkor = "  << oshkor  << endl;
     cout << "oshkormas = " << oshkormas << endl;

     return 0;
}
```

5 - misol. Butun sonni bo'lish

```
#include <iostream>
using namespace std;
int main()
{
     int bir  = 1;
     int ikki = 2;

     cout << bir / ikki << endl;
     cout << ((float)bir) / ((float)ikki) << endl;
     return 0;
}
```

Programma ishga tushganda ekranda quyidagicha natija chiqariladi:

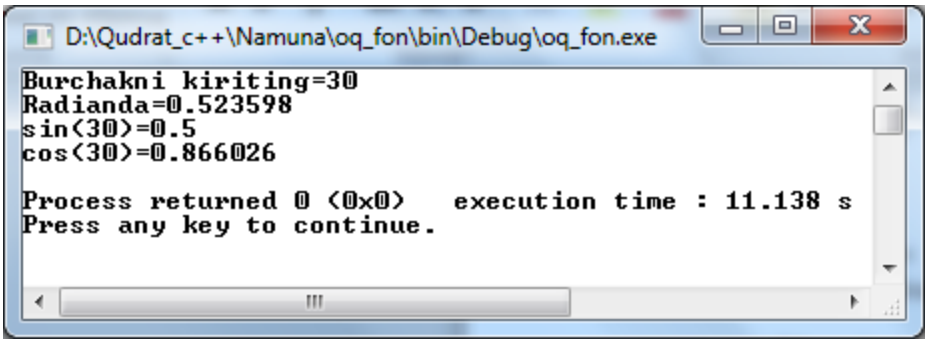


6 - misol. Trigonometrik funksiyalar bilan ishlash

```
// Muallif: Qudrat Abdurahimov
// Sana : 05.11.2011
// Maqsad: Kiritilgan burchak sin va cosinusini topish

#include <iostream>
#include <math.h>

using namespace std;
```

```
int main()
{
    float const pi = 3.14159;
    float burchak, burchak_radian;

    cout << "Burchakni kiriting="; cin >> burchak;
    burchak_radian = burchak * pi / 180;

    cout << "Radianda=" << burchak_radian << endl;
    cout << "sin(" << burchak << ")=" << sin(burchak_radian) << endl;
    cout << "cos(" << burchak << ")=" << cos(burchak_radian) << endl;
    return 0;
}
```

Programma ishga tushganda ekranda quyidagicha natija chiqariladi:



Quyidagi savollarga og'izaki yoki yozma javob bering. Bergan javobingiz qanchalik to'g'riligini tekshirish uchun savolni bir marta bosing.

Nazorat savollari:

1. Butun sonlar toifalarini sanab bering. Ular nimasi bilan farq qiladi?
2. Haqiqiy sonlar toifalarini sanab bering. Ular nimasi bilan farq qiladi?
3. Matematik funksiyalarni sanab bering.

Ma’ruzalar matnining elektron versiyasidan mavzuga oid 49 ta masalani ishlab chiqing.

C++ tilida ifodalar

C++ tilida o'zgaruvchi qiymatini birga oshirish va kamaytirishning samarali usullari mavjud. Ular inkrement (++) va dekrement (--) unar amallaridir.
Inkrement va dekrement amallarining prefiks va postfiks ko'rinishlari mavjud.
x = y++; // postfiks
x = --y; // prefiks
sanagich++; // unar amal, "++sanagich;" bilan ekvivalent
a--; // unar amal, "--a;" bilan ekvivalent

Quyida keltirilgan amallar bir xil vazifani bajaradi:

`i++;`	`i = i + 1;`
`i--;`	`i = i - 1;`
`a += b;`	`a = a + b;`
`a -= b;`	`a = a - b;`
`a *= b - c;`	`a = a * (b - c);`
`++i;`	`i++;`
`--c;`	`c--;`

C++ da ifodalar quyidagi tartibda hisoblanadi:

1. Qavs ichidagi ifodalar hisoblanadi
2. Funksiyalar qiymati hisoblanadi. (sin(x), cos(x), sqrt(x) va xakazo)
3. Inkor amali (! - not)
4. Bo'lish, ko'paytirish kabi amallar (/,*, %, ...)
5. Qo'shish kabi amallar (+, -, or, xor)
6. Munosabat amallari (=, <>, <, >, <=, >=)

Ma`lumotlarning mantiqiy toifalari

Mantiqiy toifa **bool** ikki hil qiymat qabul qilishi mumkin: true (rost, 1) va false (yolg`on, 0). Mantiqiy ma`lumotlarni e`lon qilish uchun **bool** xizmatchi so'zidan foydalaniladi.

```
bool a, b;
```

Mantiqiy toifadagi o`zgaruvchilarga qiymat berish quyidagicha amalga oshiriladi:

```
a = true; // rost
b = 0; // yolg'on, false
```

Mantiqiy amallar:

! (inkor qilish) - mantiqiy operatori mantiqiy ifodalar yoki o`zgaruvchilar oldidan qo`yiladi. Mantiqiy ifoda yoki o`zgaruvchining qiymatini teskarisiga o`zgartiradi.

&& (Mantiqiy ko`paytirish) - mantiqiy operatori ikkita mantiqiy o`zgaruvchini birlashtiradi. Agar ikkala o`zgaruvchi ham rost qiymatga ega bo`lsa natija rost, aks holda yolg`on natija beradi.

|| (mantiqiy qo`shish) - mantiqiy operatori ikkita mantiqiy o`zgaruvchini birlashtiradi. Agar o`zgaruvchilardan kamida bittasi rost qiymatga ega bo`lsa natija rost, aks holda yolg`on natija beradi.

! - mantiqiy inkor operatori jadvali

X	!X
false	true
true	false

&&, || manqiqiy operatorlai jadvali

X	Y	X && Y	X || Y
false	false	false	false
false	true	false	true
true	false	false	true
true	true	true	true

Mantiqiy amallarga misollar

a = true; b=false;
c = !a; *{ c = false }*
c = a && b; *{ c = false }*
c = a || b; *{ c = true }*

Munosabat amallari

== - teng <= - kichik yoki teng
!= - teng emas >= - katta yoki teng
< - kichik > - katta

Munosabat amallariga misollar

```
c = 5 < 0;                      // c=false
c = ( 4 % 2 == 0 );             // c=true
c = ( k > 0 ) && ( k < 7 );     // c=true, agar 0<k<7 bo`lsa
```

1. A = true, B = false, C = true, D = false bo`lsa, quyidagi mantiqiy ifoda natijasini aniqlang.

!((A && B) || (C && D)) || (A || B)

2. A = true, B = false, C = true, D = false bo`lsa, quyidagi mantiqiy ifoda natijasini aniqlang.

((A && B) || (C && D)) && (A || B)

3. A = true, B = false, C = true, D = false bo`lsa, quyidagi mantiqiy ifoda natijasini aniqlang.

!(A || B) && (C || D)

Quyidagi savollarga og`izaki yoki yozma javob bering. Bergan javobingiz qanchalik to`g`riligini tekshirish uchun savolni bir marta bosing.

Nazorat savollari:

1. Mantiqiy toifalar qanday e'lon qilinadi?
2. Mantiqiy amallarni tushuntirib bering.
3. Munosabat amallarini tushuntiring.
4. Mantiqiy amallar jadvalini tuzib bering.

Ma'ruzalar matnining elektron versiyasidan mavzuga oid 48 ta masalani ishlab chiqing.

Shart operatori

Programma tuzish mobaynida o`zgaruvchilar qiymatiga qarab u yoki bu natijani qabul qilishga to`g`ri keladi. Bu o`z navbatida programmani tarmoqlanishga olib keladi. Tarmoqlarning qaysi qismi bajarilishi ayrim shartlarga qarab aniqlanadi.

Shart operatori: Shart operatori boshqarishni qaysi tarmoqqa uzatishni ta'minlaydi. Shart operatorining ikki xil ko`rinishi mavjud. Operatorning umumiy ko`rinishi va qisqa ko`rinishi.

Shart operatorining umumiy ko`rinishi:

```
if  (<shart>)
    <operator1>;
else
     <operator2>;
```

if agar, else aks holda ma`nolarini anglatadi.
Shart operatorining qisqa ko`rinishi:

```
if  (<shart>) <operator1>;
```

<**shart**> tekshirilishi lozim bo`lgan mantiqiy ifoda
<**operator** 1> Agar shart rost (**true**) qiymatga ega bo`lsa bajarilishi lozim bo`lgan operator.
<**operator** 2> Agar shart yolg`on (**false**) qiymatga ega bo`lsa bajarilishi lozim bo`lgan operator.
Shart operatori tarkibida ixtiyoriy operatordan foydalanish mumkin. Shu o`rinda Shart operatoridan ham.
Misol: Berilgan a sonini juft yoki toqligini aniqlovchi programma tuzilsin.
Agar a sonini 2 ga bo'lganda qoldiq 0 ga teng bo'lsa, bu son juft, aks xolda toq.

```
#include <iostream>
using namespace std;
int main()
{
  int a;
  cin >> a;
  if (a % 2 == 0)
    cout << "juft";
  else
    cout << "toq";
  return 0;
}
```

C++ tili operatorlarni blok ko’rinishida bo’lishiga imkon beradi. Blok ‘{‘ va ‘}’ belgi oralig'iga olingan operatorlar ketma-ketligi bo’lib, u kompilyator tomonidan yaxlit bir operator deb qabul qilinadi. Blok ichida yangi o'zgaruvchilarni ham e'lon qilish mumkin. Bu o’zgaruvchilar faqat blok ichida ko’rinadi, undan tashqarida ko’rinmaydi, ya'ni blokdan tashqarida bu o'zgaruvchilarni ishlatib bo'lmaydi. Blokdan keyin nuqtali vergul qo’yilmaydi, lekin blok ichida har bir operator nuqtali vergul bilan yakunlanishi shart.

Shart operatorida bir nechta operatordan foydalanish uchun bu operatorlarni blok ichiga yozish lozim bo'ladi. Yuqoridagi masalani blok orqali ifodalash quyidagicha bo'ladi.

Misol: Berilgan a sonini juft yoki toqligini aniqlovchi programma tuzilsin.

```
#include <iostream>
using namespace std;
int main()
{
  int a;
  cin >> a;
  if (a % 2 == 0)
```

```
  {
       cout << "juft";
  }
  else
  {
    cout << "toq";
  }
  return 0;
}
```

Programmalashning yaxshi usuli:
Shart operatorida doimiy ravishda bloklardan foydalanish yo'l qo'yilishi mumkin bo'lgan xatoliklarni oldini oladi. Ba'zi programmistlar oldin ochuvchi va yopuvchi qavslarni {, } yozish, undan keyin blok ichidagi operatorlarni yozish lozimligini takidlashadi.

? : shart amali

Agar tekshirilayotgan shart nisbatan sodda bo'lsa, shart amalini «?: » ko'rinishini ishlatish mumkin. Bu operator quyidagi ko'rinishga ega:
<shart ifoda> ? <ifoda1> : <ifoda2>;
if shart operatoriga o'xshash holda bu shart amali quyidagicha ishlaydi: agar <shart ifoda> rost (true) bo'lsa <ifoda1> bajariladi, aks holda <ifoda2>. Odatda ifodalar qiymatlari birorta o'zgaruvchiga o'zlashtiriladi.
Misol: 2 ta sondan kattasini topuvchi programma tuzilsin.

```
#include <iostream>
using namespace std;
int main()
{
   int a, b, max;
   cout << "a="; cin >> a;
   cout << "b="; cin >> b;
   max = ( a > b ) ? a : b;
   cout << max << endl;
   return 0;
}
```

Agar a>b shart bajarilsa max o'zgaruvchisi a ni, aks xolda b ni o'zlashtiradi.

Nazorat savollari:

1. Shart operatorining qanday ko`rinishlarini bilasiz?
2. Shart operatori ichida shart operatoridan foydalanish mumkinmi?
3. Shart operatorida bir nechta operatordan foydalanish uchun nima qilinadi?

Tanlash operatori

Boshqarishni uzatish operatorlaridan yana biri tanlash operatoridir. Tanlash operatori asosan bir nechta qiymatdan, o'zgaruvchiga mos qiymatni tanlashda va qiymatlarga mos ravishda boshqarishni uzatishda ishlatiladi.

Tanlash operatorining umumiy ko'rinishi:

```
switch (<o'zgaruvchi>)
{
case <o'zgarmas ifoda1> : <operator 1>; break;
case <o'zgarmas ifoda2> : <operator 2>; break;
. . .
case <o'zgarmas ifodaN> : <operator N>; break;
[default : operator N + 1];
}
```

Tanlash operatorida boshqarilish o`zgaruvchiga mos ravishda qiymatlarga uzatiladi va mos operator ishga tushadi. **default** operatori birorta ham qiymat o`zgaruvchiga to`g`ri kelmasa ishlatiladi. **default** operatorini ishlatmasdan tashlab ketish ham mumkin.

Eslatma: Dasturlashga doir kitoblarni o`qiganingizda, biror operatorning umumiy ko'rinishining to'rtburchak qavs [ ] belgisi oralig'ida yozilgan qismini ishlatmasdan tashlab ketish mumkin. Operatorning bu qismidan foydalanish ixtiyoriy bo`ladi.

Misol: Kiritilgan songa mos ravishda hafta kunini chiqaruvchi programma tuzilsin.

```
#include <iostream>
using namespace std;
int main()
{
  int n;
  cout << "Hafta kunini kiriting" << endl;
  cin >> n;
  switch (n)
  {
     case 1: cout << "Dushanba"; break;
     case 2: cout << "Seshanba"; break;
     case 3: cout << "Chorshanba"; break;
     case 4: cout << "Payshanba"; break;
     case 5: cout << "Juma"; break;
     case 6: cout << "Shanba"; break;
     case 7: cout << "Yakshanba"; break;
```

```
      default: cout << "Bunday hafta kuni yo'q";
   }
   return 0;
}
```

Tanlash operatorida bir nechta qiymatga bir hil operator ishlatishi quyidagicha bo'ladi.

```
#include <iostream>
using namespace std;
int main()
{
   int n;
   cout << "1..10 oraliqdan son kiriting" << endl;
   cin >> n;
   switch (n)
   {
     case 1:
     case 3:
     case 5:
     case 7:
     case 9: cout << "Toq son kiritildi"; break;
     case 2:
     case 4:
     case 6:
     case 8:
     case 10: cout << "Juft son kiritildi"; break;
default: cout << "1 dan kichik yoki 10 dan katta son kiritildi";
   }
   return 0;
}
```

Nazorat savollari:

1. Tanlash operatori nima uchun ishlatiladi?
2. Tanlash operatori umumiy ko'rinishidagi [ ] ichiga yozilgan qismi nima ma'noni bildiradi?

Ma’ruzalar matnining elektron versiyasidan mavzuga oid 20 ta masalani ishlab chiqing.

for sikl operatori

Bir hil hisoblash jarayonlarini bir necha bor takrorlanishi **Sikl** deyiladi. C++ programmalashtirish tilida sikl operatorining bir necha xil turi mavjud.

- for sikl operatori
- do .. while sikl operatori
- while sikl operatori

Yechilayotgan masalaga qarab, programmist o`zi uchun qulay bo`lgan sikl operatoridan foydalanishi mumkin.

for takrorlash operatorining sintaksisi quyidagicha:

```
for (<ifoda1>; <ifoda2>; <ifoda3>)
      <operator yoki blok>;
```

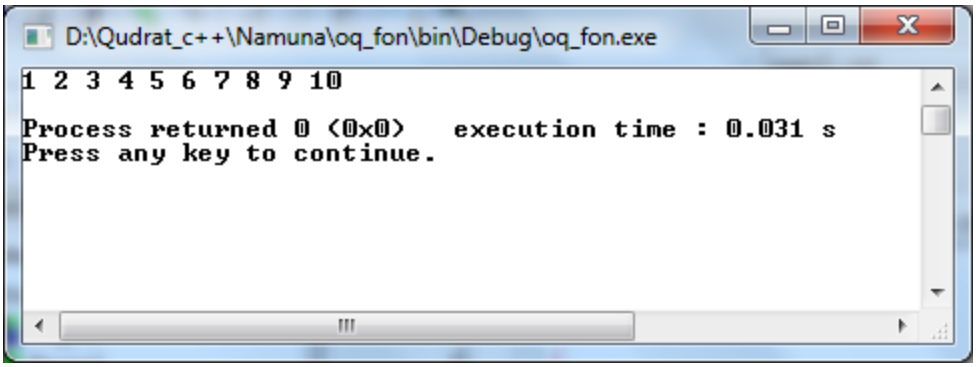
Bu operator amal qilishni <ifoda1> bajarishdan boshlaydi. Keyin takrorlash qadamlari boshlanadi. Har bir qadamda <ifoda2> bajariladi, agar natija 0 dan farqli yoki rost (true) bo'lsa, sikl tanasi - <operator yoki blok> bajariladi va oxirida <ifoda3> bajariladi, aks holda boshqaruv takrorlash operatoridan keyingi operatorga o'tiladi. Sikl tanasi – <operator yoki blok> sifatida bitta operator, shu jumladan bo'sh operator, yoki operatorlar bloki kelishi mumkin.

Sikl takrorlanishi davomida bajarilishi lozim bo'lgan operatorlar majmuasi sikl tanasi deyiladi. Sikl tanasi sifatida bir yoki bir nechta operatordan foydalanish mumkin. Agar sikl tanasida bir nechta operatordan foydalanmoqchi bo`lsak bu operatorlarni blok **{ }** orasiga olishimiz kerak.

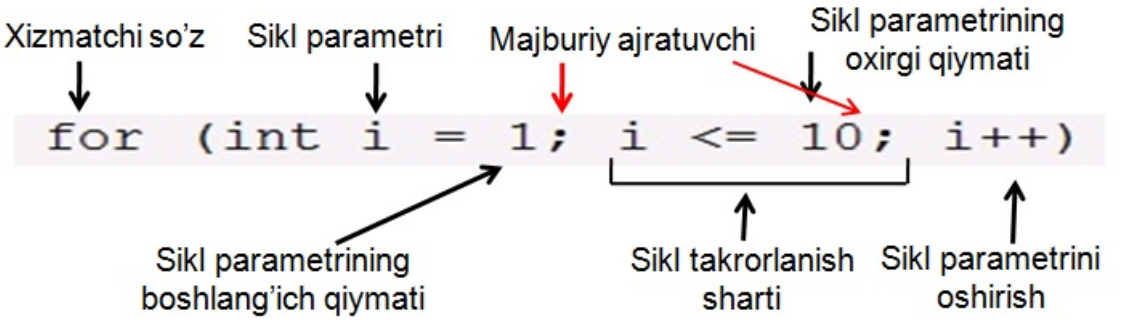
1 dan 10 gacha bo'lgan sonlarni chiqaruvchi dastur:

```
#include <iostream>
using namespace std;
int main()
{
    for (int i = 1; i <= 10; i++)
    cout << i << " ";

    cout << endl;
    return 0;
}
```

Ekranda quyidagicha natija xosil bo'ladi:



Quyidagi rasmda for sikl operatori batafsil berilgan.



for sikl operatorini umumiy ko'rinishda quyidagicha ifodalash mumkin:

for (sikl_o'zgaruvchisining_boshlang'ich_qiymati**;** takrorlanish_sharti**;**
sikl_o'zgaruvchisini_oshirish)
sikl_tanasi**;**

sikl_o'zgaruvchisini_oshirish o'rnida kamaytirish ham bo'lishi mumkin.
Agar sikl tanasida bir nechta operatordan foydalanmoqchi bo`lsak bu operatorlarni blok { } orasiga olishimiz kerak.
Programma taxlili:

Yuqoridagi 1 dan 10 gacha bo'lgan sonlarni chiqaruvchi programmani taxlil qilib chiqamiz.

```
for (int i = 1; i <= 10; i++)
cout << i << " ";
```

1. Sikl parametri (i) boshlang'ich qiymat 1 ni o'zlashtiradi. Ya'ni i = 1;
2. Sikl takrorlanish sharti tekshiriladi. (i <= 10;). 1 <= 10 shart rost bo'lgani uchun sikl tanasi (cout << i << " ";) bajariladi. Ekranga "1 " chiqariladi.
3. Sikl parametrini oshirish (i++) bajariladi. i ning qiymati 2 ga teng bo'ladi.
4. Sikl takrorlanish sharti tekshiriladi. (i <= 10;). 2 <= 10 shart rost bo'lgani uchun sikl tanasi (cout << i << " ";) bajariladi. Ekranga "2 " chiqariladi.

5. Sikl parametrini oshirish (i++) bajariladi. i ning qiymati 3 ga teng bo'ladi.
6. Sikl takrorlanish sharti tekshiriladi. (i <= 10;). 3 <= 10 shart rost bo'lgani uchun sikl tanasi (cout << i << " ";) bajariladi. Ekranga "3 " chiqariladi.
7. Sikl parametrini oshirish (i++) bajariladi. i ning qiymati 4 ga teng bo'ladi.
8. Sikl takrorlanish sharti tekshiriladi. (i <= 10;). 4 <= 10 shart rost bo'lgani uchun sikl tanasi (cout << i << " ";) bajariladi. Ekranga "4 " chiqariladi.
9. Sikl parametrini oshirish (i++) bajariladi. i ning qiymati 5 ga teng bo'ladi.
10. Sikl takrorlanish sharti tekshiriladi. (i <= 10;). 5 <= 10 shart rost bo'lgani uchun sikl tanasi (cout << i << " ";) bajariladi. Ekranga "5 " chiqariladi.
11. Sikl parametrini oshirish (i++) bajariladi. i ning qiymati 6 ga teng bo'ladi.
12. Sikl takrorlanish sharti tekshiriladi. (i <= 10;). 6 <= 10 shart rost bo'lgani uchun sikl tanasi (cout << i << " ";) bajariladi. Ekranga "6 " chiqariladi.
13. Sikl parametrini oshirish (i++) bajariladi. i ning qiymati 7 ga teng bo'ladi.
14. Sikl takrorlanish sharti tekshiriladi. (i <= 10;). 7 <= 10 shart rost bo'lgani uchun sikl tanasi (cout << i << " ";) bajariladi. Ekranga "7 " chiqariladi.
15. Sikl parametrini oshirish (i++) bajariladi. i ning qiymati 8 ga teng bo'ladi.
16. Sikl takrorlanish sharti tekshiriladi. (i <= 10;). 8 <= 10 shart rost bo'lgani uchun sikl tanasi (cout << i << " ";) bajariladi. Ekranga "8 " chiqariladi.
17. Sikl parametrini oshirish (i++) bajariladi. i ning qiymati 9 ga teng bo'ladi.
18. Sikl takrorlanish sharti tekshiriladi. (i <= 10;). 9 <= 10 shart rost bo'lgani uchun sikl tanasi (cout << i << " ";) bajariladi. Ekranga "9 " chiqariladi.
19. Sikl parametrini oshirish (i++) bajariladi. i ning qiymati 10 ga teng bo'ladi.
20. Sikl takrorlanish sharti tekshiriladi. (i <= 10;). 10 <= 10 shart rost bo'lgani uchun sikl tanasi (cout << i << " ";) bajariladi. Ekranga "10 " chiqariladi.
21. Sikl parametrini oshirish (i++) bajariladi. i ning qiymati 11 ga teng bo'ladi.
22. Sikl takrorlanish sharti tekshiriladi. (i <= 10;). 11 <= 10 shart rost bo'lmagani uchun sikl tugatiladi va boshqarilish sikl operatoridan keyingi operatorga uzatiladi.

10 dan 1 gacha bo'lgan sonlarni chiqaruvchi dastur:

```
#include <iostream>
using namespace std;
int main()
{
    for (int i = 10; i >= 1; i--)
    cout << i << " ";
    cout << endl;
    return 0;
}
```

break – funksiyasini har qanday sikl operatoriga qo`llash mumkin. Bu funksiya sikl tugatilishini ta’minlaydi. Ya`ni boshqarilishni sikl operatoridan keyingi operatorga uzatadi.

continue – funksiyasini har qanday sikl operatoriga qo`llash mumkin. Bu funksiya sikl parametrining keyingi qiymatni qabul qilishini taminlaydi. Boshqacha so`z bilan aytganda sikl tanasi tugatiladi. Bunda siklning o`zi tugatilmaydi.
Misol. n natural soni berilgan. Birdan n gacha bo`lgan sonlar yig`indisini hisoblovchi programma tuzilsin

```
#include <iostream>
using namespace std;
int main()
{
    int n, s = 0;

    cout << "n="; cin >> n;

    for (int i = 1; i <= n; i++)
    s += i;
    cout << s << endl;
    return 0;
}
```

for sikl operatorining boshqa imkoniyatlari

for sikl operatorida qavs ichidagi ifodalar bo'lmasligi mumkin, lekin ";" bo'lishi shart. Eng sodda doimiy takrorlanuvchi sikl operatori quyidagicha:

```
for ( ; ; )
cout << "doimiy takrorlanish";
```

Agar takrorlash jarayonida bir nechta o'zgaruvchi bir vaqtda sinxron o'zgarishi lozim bo'lsa, ularni <ifoda1> va <ifoda3> da zarur bo'lgan o'rinda vergul bilan ajratib yozish mumkin.

```
#include <iostream>
using namespace std;
int main()
{
    int n;
    cin >> n;

    for (int i = 1, j = 1; i <= n; i++, j += i)
    cout << i << " " << j << endl;
    return 0;
}
```

Nazorat savollari:

1. for sikl operatorining sintaksisi qanday?

2. for sikl operatorida sikl tanasi deb nimaga aytiladi?
3. break va continue funksiyalari qo`llanilishini tushuntiring?
4. for sikl operatorida sikl parametrlari qanday toifali bo'lishi mumkin?
5. Siklning takrorlanishlar soni qanday aniqlanadi?

Ma'ruzalar matnining elektron versiyasidan mavzuga oid 55 ta masalani ishlab chiqing.

do – while sikl operatori

do - while operatorining umumiy ko'rinishi :

```
do {
  operator;
} while ( shart );
```

Bu yerda **do** va **while** xizmatchi so`zlar. ( **shart )** sikl tanasi bajarilgandan so`ng, sikldan chiqish uchun tekshiriladigan shart. (mantiqiy ifoda).

do - while operatorning ishlash tartibi:

do xizmatchi so`zidan keyingi operatorlar bajariladi, keyin **while** xizmatchi so'zidan keyingi shart tekshiriladi. Agar shart rost (true) natija bersa **do** xizmatchi so`zidan keyingi operatorlar qayta bajariladi. Shart qayta tekshiriladi, bu jarayon shart yolg'on (false) natija berguncha takrorlanadi. Qachon while xizmatchi so'zidan keyingi shart yolg'on (false) qiymatga ega bo'lsa, boshqarilish **do - while** operatoridan keyingi operatorga uzatiladi.

do - while sikl operatorida sikllanib qolish

DIQQAT: do - while sikl operatoridan, qachon **while** xizmatchi so'zidan keyingi **(shart)** false (yolg'on) qiymat qabul qilsa chiqiladi. Ya'ni boshqarilish **do - while** operatoridan keyingi operatorga uzatiladi. Agar **(shart)** false qiymat qabul qilmasa, do - while sikl operatoridan chiqib ketilmaydi va bu jarayon sikllanib qolish deyiladi.

1 dan 10 gacha bo'lgan sonlarni chiqaruvchi programma tuzilsin.

```
#include <iostream>
using namespace std;
int main()
{
    int i = 1;
    do {
        cout << i << endl;
        i++;
```

```
    } while ( i <= 10);

    return 0;
}
```

Misol. Quyidagi yig`indini hisoblovchi programma tuzilsin.

$$s = 1 + \frac{1}{2} + \frac{1}{3} + \ldots + \frac{1}{50}$$

Bu programma parametrli sikl operatoridan foydalangan holda oldingi darsda tuzilgan edi. Endi do - while sikl operatori orqali programma tuzamiz va sikl operatorlarini farqini ko`rib olamiz.

```
#include <iostream>
using namespace std;
int main()
{
    float i = 1; // i - sikl uchun
    float s = 0; // s - yig'indi

    do {
        s += 1 / i;
        i++;
    } while ( i <= 50);

    cout << "yig`indi = " << s << endl;
    return 0;
}
```

Nazorat savollari:

1. do - while operatori sikl tanasida qanday operatorlar bo`lishi mumkin?
2. do - while operatorining umumiy ko'rinishi qanday?
3. do - while operatoridan qanday chiqiladi?
4. do - while operatori ishlash tartibini tushintirib bering.

while sikl operatori

do - while operatorida siklning tanasi kamida bir marta takrorlanadi. Shu bir marta hisoblash ham yechilayotgan masalani mohiyatini buzib yuborishi mumkin. Bunday hollarda **while** sikl operatoridan foydalangan maqsadga muvofiq.

while operatorining umumiy ko'rinishi:

```
while ( shart ) {
 sikl_tanasi;
}
```

while operatori sikl tanasida qanday operatorlar bo`lishi mumkin?
sikl_tanasi ixtiyoriy operator yoki operatorlar majmuidan iborat bo'lishi mumkin..

while sikl operatorning ishlash tartibi:
Agar (shart) rost (**true**) qiymatga ega bo`lsa, **sikl_tanasi** bajariladi. Qachonki shart yolg`on (**false**) qiymatga teng bo`lsa sikl tugatiladi.
Agar (shart) true qiymatga ega bo`lmasa sikl tanasi biror marta ham bajarilmaydi.

while sikl operatoridan qanday chiqiladi?
while sikl operatoridan, qachon **(shart)** false (yolg'on) qiymat qabul qilsa chiqiladi. Ya'ni boshqarilish **while** operatoridan keyingi operatorga uzatiladi. Agar **(shart)** false qiymat qabul qilmasa, while sikl operatoridan chiqib ketilmaydi va bu jarayon sikllanib qolish deyiladi.
Programmalash san'ati . **do - while** va **while** sikl operatorlarida sikl tanasi sifatida faqat bitta operator ishlatiladiga bo'lsa, bu operatorni blok orasiga { } olmasdan ham yozish mumkin. Lekin professional programmistlar har qanay xolda sikl tanasini blokka { } olib yozishni tavsiya qilishadi. Bu esa sodir bo'lishi mumkin bo'lgan mantiqiy xatoliklarni oldini oladi.

1 dan 10 gacha bo'lgan sonlarni chiqaruvchi programma tuzilsin.

```
#include <iostream>
using namespace std;
int main()
{
    int i = 1;

    while ( i <= 10 ) {
        cout << i << endl;
        i++;
    }
    return 0;
}
```

Misol. Quyidagi yig`indini hisoblovchi programma tuzilsin.

$$s = 1 + \frac{1}{2} + \frac{1}{3} + \ldots + \frac{1}{50}$$

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

```
{
    float i = 1; // sanagich
    float s = 0; // yig'indi

    while ( i <= 50 ) {
        s += 1 / i;
        i++;
    }

    cout << s << endl;

    return 0;
}
```

Kompyuter o'ylagan sonni topish dasturi

```
#include <iostream>
#include <ctime>

using namespace std;

int main()
{
    int x, y = 0, u = 0;
    srand(time(NULL));
    x = rand() % 1000 + 1;

    cout << "Kompyuter o'ylagan sonni toping" << endl;
    while (x != y)
    {
      cin >> y;
      u++;
      if (x > y) cout << "Kompyuter o'ylagan son katta" << endl;
      else if (x < y) cout << "Kompyuter o'ylagan son kichik" << endl;
    }

    cout << "Qoyil topdingiz!!!" << endl;
    cout << "Urinishlar soni=" << u << endl;

    return 0;
}
```

Nazorat savollari:

1. while operatori sikl tanasida qanday operatorlar bo`lishi mumkin?

2. while operatoridan qanday chiqiladi?
3. while operatori ishlashini tushintirib bering.

Ma’ruzalar matnining elektron versiyasidan mavzuga oid 38 ta masalani ishlab chiqing.

Ko'rsatkichlar bilan ishlash

Assalomu alaykum bo`lajak dasturchi! Yangi mavzuni boshlashdan oldin, oldingi mavzuni qisqacha takrorlab olsak. Quyidagi savollarga og`izaki yoki yozma javob bering. Javob qanchalik to`g`riligini tekshirish uchun savolni bir marta bosing.

Bu mavzuda siz bilan C++ ning eng zo'r elementlaridan biri - ko'rsatkichlar bilan tanishamiz.

Ko'rsatkich. O'zining qiymati sifatida xotira adresini ko'rsatuvchi (saqlovchi) o'zgaruvchilarga - ko'rsatkich o'zgaruvchilar deyiladi.

Masalan : Ko'rsatkichning qiymati

1) 0x22ff40
2) 0x22ff33
3) va xakazo kabi xotiraning aniq qismi bo'lishi mumkin.

Boshqa o'zgaruvchilar kabi, ko'rsatkichlardan foydalanish uchun ularni e'lon qilish, toifasini aniqlash shart.

```
int *countPtr, count;
```

bu yerda countPtr - int toifasidagi ob'ektga ko'rsatkich, count esa oddiy butun (int) toifasidagi o'zgaruvchi. Ko'rsatkichlarni e'lon qilishda har bir o'zgaruvchi oldigan * qo'yilishi shart.

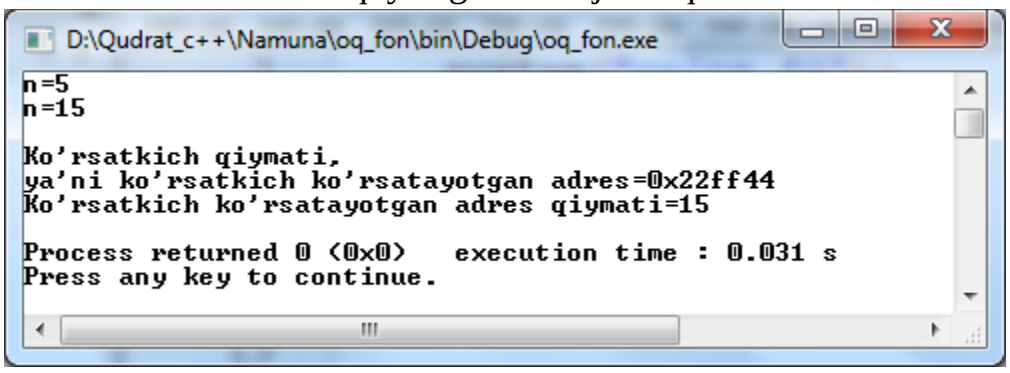
Ko'rsatkichga doir oddiy misol

```
#include <iostream>

using namespace std;

int main()
{
    int n = 5;

    int * nPtr;
```

```
    // & adresni olish amali
    nPtr = &n;
    cout << "n=" << n << endl;

    *nPtr = 15;
    cout << "n=" << n << endl;

  cout << "\nKo'rsatkich qiymati,\n";
  cout << "ya'ni ko'rsatkich ko'rsatayotgan adres=" << nPtr<<endl;
  cout << "Ko'rsatkich ko'rsatayotgan adres qiymati="
<<*nPtr<<endl;

    return 0;
}
```

Yuqoridagi masalaning to'liq izohi video ma'ruzada tushuntiriladi.

Ekranda quyidagicha natija chiqariladi:

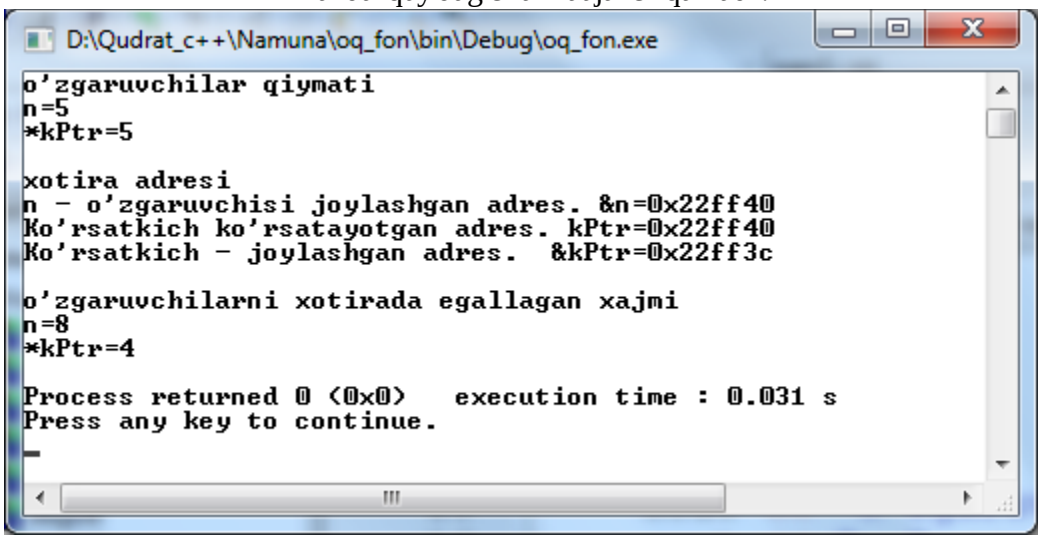


Ko'rsatkich bilan ishlash

```
#include <iostream>
using namespace std;
int main()
{
    double n = 5;
    double *kPtr;

    kPtr = &n;

    cout << "o'zgaruvchilar qiymati" << endl;
    cout << "n=" << n << endl;
    cout << "*kPtr=" << *kPtr << endl;

    cout << "\nxotira adresi" << endl;
  cout << "n - o'zgaruvchisi joylashgan adres. &n=" << &n << endl;
```

```
  cout << "Ko'rsatkich ko'rsatayotgan adres. kPtr=" << kPtr <<
endl;
  cout << "Ko'rsatkich - joylashgan adres.  &kPtr=" << &kPtr <<
endl;

cout << "\no'zgaruvchilarni xotirada egallagan xajmi" <<
endl;
cout << "n=" << sizeof(n) << endl;
cout << "*kPtr=" << sizeof(kPtr) << endl;

    return 0;
}
```

Ekranda quyidagicha natija chiqariladi:



Murojaatlar

Murojaatlar e'londa ko'rsatilgan nomning sinonimi sifatida ishlatiladi, yani bitta o'zgaruvchiga xar xil nom bilan murojaat qilish mumkin. Murojaatni doimiy qiymatga ega bo'lgan ko'rsatkich deb qarash mumkin xam bo'ladi. Murojaat quyidagicha e'lon qilinadi:

<toifa> & <nom>;

Bu yerda <toifa> – murojaat ko'rsatuvchi qiymatning toifasi, '&' belgisi, undan keyin yozilgan <nom>- murojaat toifasidagi nom ekanligini bildiruvchi operator. Boshqacha aytganda '&' belgisiga adresni olish amali deyiladi.

Namuna:

```
int k;
int & p = k;
//  p murojaati -  k o'zgaruvchisining alternativ nomi
```

Murojaat asosan funktsiyalarda adres orqali uzatiluvchi parametrlar sifatida ishlatiladi.
Murojaatning ko'rsatkichdan bir nechta farqi bor:

1) Murojaatni e'lon qilishda initsializatsiya qilish kerak
2) Murojaatning qiymatini o'zgartirib bo'lmaydi, ko'rsatkichning qiymatini, ko'rsatib turgan adresini o'zgartirish mumkin.

Funksiya va massivga ko'rsatkich orqali murojaatni keyingi mavzularda o'rganamiz.

Funksiyalar bilan ishlash

Assalomu alaykum bo`lajak dasturchi! Yangi mavzu video fayli bilan tanishib chiqing.

Video faylni ko`rib bo`lgandan keyin ma`ruza matnini diqqat bilan o`qib chiqing.

Dasturlash mobaynida bir xil ifodalarni, hisoblash jarayonlarini qayta – qayta hisoblashga to`g`ri keladi.

Dasturlash tillarida, kompyuter hotirasini va dasturchining vaqtini tejash maqsadida, bunday takkorlanuvchi jarayonlarni dasturda ajratib yozib, unga asosiy daturdan, boshqa funksiyalardan murojaat qilish imkoniyatlari keltirilgan.

Dasturning istalgan qismidan murojaat qilib, bir necha bor ishlatish mumkin bo`lgan operatorlar guruhiga funksiya deyiladi.

C++ funksiyalar tili deyiladi. Chunki dasturda kamida bitta main funksiyasi bo'ladi. Asosiy dastur, asosiy funksiya deganda aynan manashu **main** funksiyasini tushunamiz.

Asosiy programmadan (yoki chaqiruvchi funksiyadan) xech qanday parametr qabul qilib olmaydigan funksiyalarga, parametrsiz funksiyalar deyiladi.

Parametrsiz funksiyaning o'zi ham 2 xil bo'lishi mumkin:

1) Asosiy programmaga (yoki chaqiruvchi funksiyaga) natijani qaytaruvchi.

2) void turidagi funksiya bo'lib, asosiy programmadan (yoki chaqiruvchi funksiyadan) xech qanday parametr qabul qilib olmaydi xam, asosiy programmaga xech qanday natija qaytarmaydi ham.

Parametrsiz funksiyaga murojaat qilishda dastur tanasida funksiya nomi yoziladi. Dasturda funksiya nomi operatorlar kabi ishlatiladi. Parametrsiz funksiyada asosiy dasturning barcha global o`zgaruvchilaridan foydalanish mumkin.

Global o`zgaruvchilar

Ham asosiy programmada, ham funksiyada ishlatish mumkin bo'lgan o`zgaruvchilar global o'zgaruvchilar deyiladi. Global o'zgaruvchilar asosiy programmada e`lon qilishi kerak.

Lokal o`zgaruvchilar

Faqat funksiyada ishlatish mumkin bo'lgan o'zgaruvchilarga local o'zgaruvchilar deyiladi. Ular funksiyada e`lon qilinadi. Funksiyada yana bir nechta ichki funksiyalardan foydalanish mumkin.

Blok ichida e'lon qilingan o'zgaruvchilar, shu blok uchun lokal o'zgaruvchilar hisoblanadi. Bu o'zgaruvchilardan faqat blok ichida foydalanish mumkin.

Parametrli funksiyalar

Asosiy dasturdan (funksiyadan) chaqiriluvchi funksiyaga uzatilgan parametrlarni qabul qilib qayta ishlovchi funksiyalar parametrli funksiyalar deyiladi.

Qiymat parametrlar

Qiymat parametrlar – asosiy dasturdan funksiyaga uzatiladigan o'zgaruvchilar qiymatlarni qabul qilib oluvchi parametrlar. Funksiyaga murojaat qilinganida qiymat parametrlari uchun xotiradan joy ajratiladi. Funksiya tugaganida qiymat parametrlari uchun ajratilgan xotira bo'shatiladi.

Ko'rsatkich parametrlar

Ko'rsatkich parametrlar - asosiy dasturdan funksiyaga uzatiladigan o'zgaruvchilarning xotiradagi adresini qabul qilib oluvchi parametrlar.

Ko'rsatkich parametrlari ustida bajarilgan har qanday o'zgarish, asosiy dasturdagi o'zgaruchilarning xotira adresida sodir bo'ladi. (Ya'ni asosiy dasturdagi o'zgaruvchi qiymati o'zgaradi)

Eslatma: Qiymat parametrlari va ko'rsatkich parametrlar toifasi, asosiy dasturdagi qiymati uzatilayotgan o`zgaruvchilar toifasi bilan bir xil bo`lishi lozim.

Funksiyadan chiqish

Ixtiyoriy funksiyadan chiqish uchun **return** xizmatchi so'zi ishlatiladi.

Misol: To'g'ri burchakli uchburchakning katetlari berilgan. (3, 4), (6, 8), (12, 5) bo'lgan xollar uchun uchburchak gipotenuzasini hisoblovchi programma tuzilsin.

1) Parametrli funksiya

```
// Muallif: Qudrat Abdurahimov
// Sana : 1 noyabr 2011 yil
// Maqsad : Parametrli funksiyani o'rganish

#include <iostream>
#include <math.h>

using namespace std;

// funksiya prototipi
float hisobla(float , float );

int main()
{
    float c;

    c = hisobla(3, 4);
    cout << c << endl;

    c = hisobla(6, 8);
    cout << c << endl;

    c = hisobla(12, 5);
    cout << c << endl;

    return 0;
}

float hisobla(float a, float b)
{
    //lokal o'zgaruvchi
    float natija;
    natija = sqrtf(a*a + b*b);
    return natija;
}
```

2) void toifasidagi parametrli funksiya

```
// Muallif: Qudrat Abdurahimov
// Sana : 1 noyabr 2011 yil
// Maqsad : void toifasidagi parametrli funksiyani o'rganish

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

```
    using namespace std;

    // funksiya prototipi
    void hisobla(float , float );

    int main()
    {
        hisobla(3, 4);
        hisobla(6, 8);
        hisobla(12, 5);

        return 0;
    }

    void hisobla(float a, float b)
    {
        float c;
        c = sqrtf(a*a + b*b);
        cout << c << endl;
    }
```

Global va lokal o'zgaruvchilarga murojaatni o'rganish

```
// Muallif : Qudrat Abdurahimov
// Sana : 05.11.2011
// Maqsad : Global va lokal o'zgaruvchilarga murojaatni
// o'rganish

#include <iostream>

int x = 5; // global o'zgaruvchi
int main()
{
    int x = 9; // lokal o'zgaruvchi

    std::cout << "lokal  x=" <<   x << std::endl;
    std::cout << "global x=" << ::x << std::endl;

    return 0;
}
```

Kiritilgan n sonini 3 - darajasini hisoblovchi funksiya tuzilsin

```
// Muallif : Qudrat Abdurahimov
```

```
// Sana : 04.12.2011
// Maqsad : Funksiyaga ko'rsatkich parametrlari
// orqali murojaatni o'rganish

#include <iostream>

using namespace std;

void kub (int *);

int main()
{
    int n;
    cout << "n="; cin >> n;

    kub (&n);

    cout << "n ning qiymati =" << n << endl;
    return 0;
}

void kub (int *nPtr)
{
    *nPtr = *nPtr * *nPtr * *nPtr;
}
```

Ikkita son yig'indisini funksiya orqali hisoblovchi programma tuzilsin

```
// Muallif : Qudrat Abdurahimov
// Sana : 04.12.2011
// Maqsad : Funksiyaga qiymat va ko'rsatkich parametrlari
// orqali murojaatni o'rganish

#include <iostream>

using namespace std;
// funksiya prototipi
int  sum(int , int);
void sum(int , int, int *);
int  sum(int *, int *);
void sum(int *, int *, int *);

int main()
{
    int a, b, c;
```

```
    cout << "a="; cin >> a;
    cout << "b="; cin >> b;

    c = sum(a, b);
    cout << "1-sul natijasi=" << c << endl;

    sum(a, b, &c);
    cout << "2-sul natijasi=" << c << endl;

    c = sum(&a, &b);
    cout << "3-usul natijasi=" << c << endl;

    sum(&a, &b, &c);
    cout << "4-usul natijasi=" << c << endl;

    return 0;
}

// 1 - usul
int sum(int son1, int son2)
{
    int natija;

    natija = son1 + son2;

    return natija;
}

// 2 - usul
void sum(int son1, int son2,int *natija)
{
    *natija = son1 + son2;
}

// 3 - usul
int sum(int *son1, int *son2)
{
    int natija;

    natija = *son1 + *son2;

    return natija;
}
```

```
// 4 - usul
void sum(int *son1, int *son2,int *natija)
{
    *natija = *son1 + *son2;
}
```

Nazorat savollari:

1. Funksiya deb nimaga aytiladi?
2. Parametrsiz funksiyaga murojaat qanday amalgam oshiriladi?
3. Qanday o`zgaruvchilar global o`zgaruvchilar deyiladi?
4. Qanday o`zgaruvchilar lokal o`zgaruvchilar deyiladi?
5. Parametrli funksiya deb qanday funksiyalarga aytiladi?
6. Qanday parametrlar qiymat parametrlar deyiladi?
7. Qanday parametrlar ko'rsatkich parametrlar deyiladi?
8. Funksiyadan chiqish uchun qaysi operatordan foydalaniladi?

Ma’ruzalar matnining elektron versiyasidan mavzuga oid 60 ta masalani ishlab chiqing.

Massivlar. Bir o`lchamli massivlar

Massiv - bu bir xil toifali, chekli qiymatlarning tartiblangan to`plamidir. Massivlarga misol qilib matematika kursidan ma`lum bo`lgan vektorlar, matritsalarni ko`rsatish mumkin.

Massiv bir o`lchamli deyiladi, agar uning elementiga bir indeks orqali murojaat qilish mumkin bo`lsa.

Bir o`lchamli massivni e`lon qilish quyidagicha bo`ladi:

<toifa> <massiv_nomi> [elementlar_soni] = { boshlang'ich qiymatlar };

Quyida massivlarni e`lon qilishga bir necha misollar keltirilgan:

```
1) float a[5];
2) int   m[6];
3) bool  b[10];
```

1) a elementlari haqiqiy sonlardan iborat bo`lgan, 5 ta elementdan tashkil topgan massiv. Indekslari esa 0 dan 4 gacha bo`lgan sonlar

float a[5];					
Massiv elementilari	a[0]	a[1]	a[2]	a[3]	a[4]
qiymati	4	-7	15	5.5	3

2) m elementlari butun sonlardan iborat bo`lgan, 6 ta elementdan tashkil topgan massiv. Indekslari esa 0 dan 5 gacha bo`lgan sonlar.

int m[6];						
Massiv elementilari	m[0]	m[1]	m[2]	m[3]	mas2[4]	mas2[5]
qiymati	2	-17	6	7	13	-3

3) b elementlari mantiqiy qiymatlardan (**true, false**) iborat bo`lgan 10 ta elementdan tashkil topgan massiv. Indekslari esa 0 dan 9 gacha bo`lgan sonlar.

Massiv elementlariga murojaat qilish oddiy o`zgaruvchilarga murojaat qilishdan biroz farq qiladi. Massiv elementiga murojaat qilish uning indeksi orqali bo`ladi.

a[1] = 10; a massivining 1 – elementi 10 qiymat o’zlashtirsin;
cin >> a[2]; a massivining 2 – elementi kirtilsin;
cout << a[3]; a massivining 3 – elementi ekranga chiqarilsin;

Massivni e'lon qilishda uning elementlariga boshlang'ich qiymat berish mumkin va buning bir nechta usuli mavjud.

1) O'lchami ko'rsatilgan massivni to'liq initsializatsiyalash.

int k[5] = { 2, 3, 7, 8, 6};

Bu yerda 5 ta elementdan iborat bo'lgan k massivi e'lon qilingan va massivning barcha elemantlariga boshlang'ich qiymat berilgan.

2) O'lchami ko'rsatilgan massivni to'liqmas initsializatsiyalash.

int k[5] = { 2, 3, 7 };

Bu yerda 5 ta elementdan iborat bo'lgan k massivi e'lon qilingan va massivning dastlabki 3 ta elemantlariga boshlang'ich qiymat berilgan.

3) O'lchami ko'rsatilmagan massivni to'liq initsializatsiyalash.

int k[] = { 2, 3, 7, 8, 6};

Shuni takidlash lozimki, agar massiv o'lchami ko'rsatilmasa, uni to'liq initsializatsiyalash shart. Bu xolda massiv o'lchami kompilyatsiya jarayonida massiv elementlari soniga qarab aniqlanadi. Bu yerda massiv o'lchami 5 ga teng.

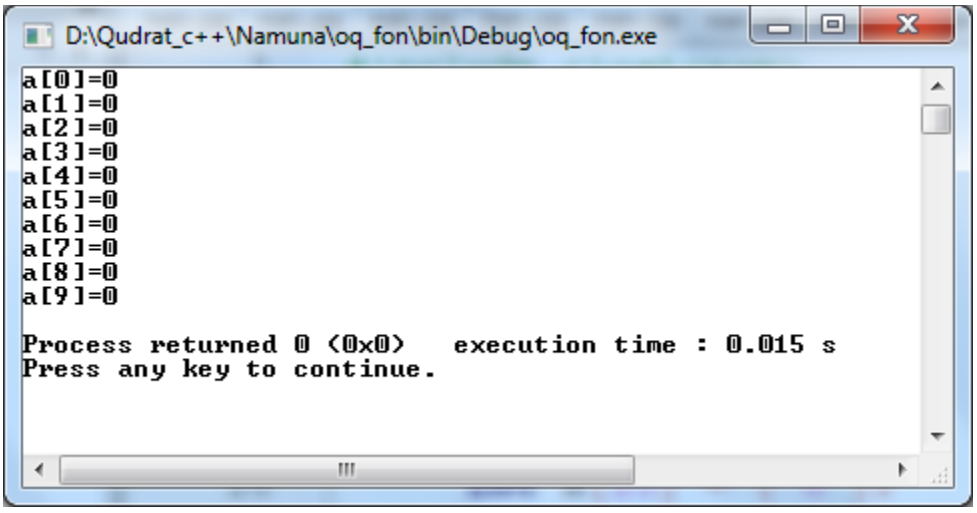
4) O'lchami ko'rsatilgan massivning barcha elementlariga boshlang'ich qiymat 0 berish:

int k[5] = { 0 };

O'lchami ko'rsatilgan massivning barcha elementlariga boshlang'ich qiymat 0 berish

```
#include <iostream>
using namespace std;
int main()
{
    int a[10] = { 0 };
    //massivning barcha elementlariga 0 qiymat berish

    for (int i = 0; i < 10; i++)
    cout << "a[" << i << "]=" << a[i] << endl;

    return 0;
}
```

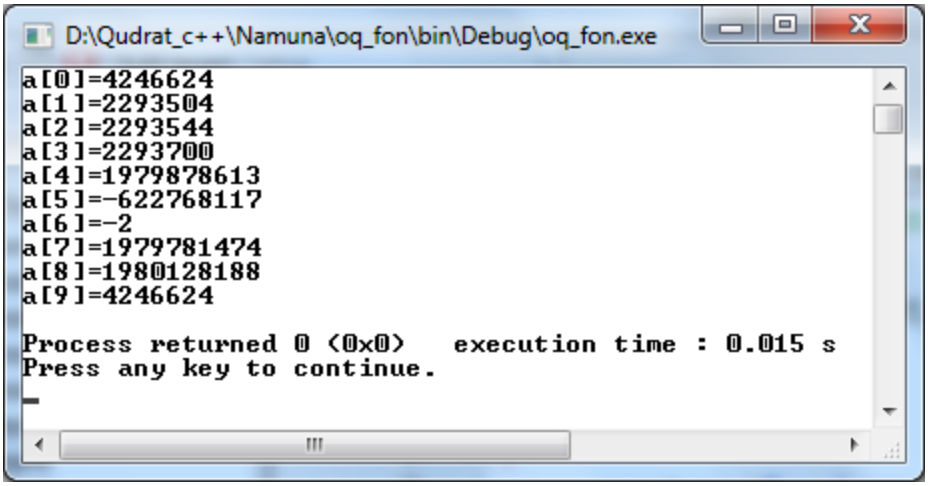
Ekranga quyidagicha natija chiqariladi:



Agar massiv elementlariga boshlang'ich qiymatlar berilmasa xatolik sodir bo'lishi mumkin.

```
#include <iostream>
using namespace std;
int main()
{
    int a[10];

    for (int i = 0; i < 10; i++)
    cout << "a[" << i << "]=" << a[i] << endl;

    return 0;
}
```

Ekranga quyidagicha natija chiqariladi:



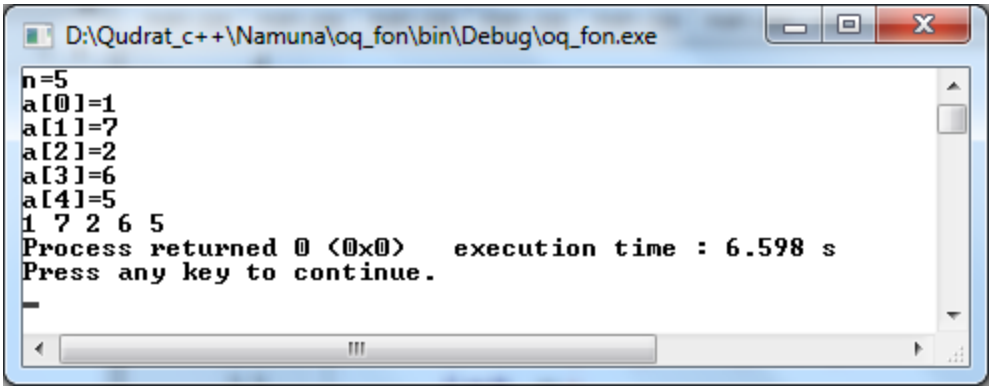
Bunday natija chiqishining sababi video ma'ruzada batafsil tushuntiriladi.

Elementlari butun sonlardan iborat bo`lgan, n elementdan tashkil topgan massiv elementlarini kirituvchi va ekranga chiqaruvchi programma tuzilsin. (n <= 10)

```
#include <iostream>
using namespace std;
int main()
{
    int a[10] = { 0 };
    int n;
    cout << "n="; cin >> n;
    for (int i = 0; i < n; i++)
    {
        cout << "a[" << i << "]=";
        cin >> a[i];
    }

    for (int i = 0; i < n; i++)
    cout << a[i] << " ";
```

```
    return 0;
}
```

Ekranga quyidagicha natija chiqariladi:



n ta elementdan tashkil topgan massiv berilgan. Shu massiv elementlari yig'indisini chiqatuvchi programma tuzilsin. (n <= 10)

```
#include <iostream>

using namespace std;

int main()
{
    int a[10] = { 0 };  // a massivini e'lon qilish
    int n;      // massiv elamentlari soni
    int s = 0;  // massiv elementlari yig'indisini hisoblash
uchun

    cout << "n="; cin >> n;

    for (int i = 0; i < n; i++)
    {
        cout << "a[" << i << "]=";
        cin >> a[i];
        s += a[i];
    }

    cout << "Massiv elementlari yig`indisi = " << s <<
endl;

    return 0;
}
```

Massiv elementlari sonini quyidagicha aniqlash mumkin

Massivning kompyuter xotirasida egallagan hajmini, bitta elementi (massiv elementi toifasi) hajmiga bo'lish orqali.

```
#include <iostream>
using namespace std;
int main()
{
    int a[10];
    int n;

    cout << "n="; cin >> n;

    for (int i = 0; i < n; i++)
    {
        cout << "a[" << i << "]=";
        cin >> a[i];
    }

    cout << "1 - usul=" << sizeof(a) / sizeof(a[0]) <<
endl;
    cout << "2 - usul=" << sizeof(a) / sizeof(int) <<
endl;

    return 0;
}
```

C stilida dinamik massivlar bilan ishlash

```
#include <iostream>
#include <alloc.h>
using namespace std;
int main()
{
    int n, s = 0;
    int *a;

    cout << "n="; cin >> n;

    // xotira ajratish, xotira yetarli bo'lmasa
    // NULL (0) ko'rsatkich qaytariladi

    a =(int*) malloc(n * sizeof(int));
```

```
    if ( a == NULL)
    {
        cout << "Xotira yetarli emas";
        return 1;
    }

    for (int i = 0; i < n; i++)
    {
        cout << "a[" << i << "]=";
        cin >> a[i];

        s += a[i];
    }

    // xotirani bo'shatish
    free(a);

    cout << s << endl;

    return 0;
}
```

C++ stilida dinamik massivlar bilan ishlash

```
#include <iostream>
using namespace std;
int main()
{
    int n, s = 0;
    int *a;

    cout << "n="; cin >> n;

    // xotira ajratish, xotira yetarli bo'lmasa
    // NULL (0) ko'rsatkich qaytariladi
    a = new int[n];

    if ( a == NULL)
    {
        cout << "Xotira yetarli emas";
        return 1;
    }
```

```
    for (int i = 0; i < n; i++)
    {
        cout << "a[" << i << "]=";
        cin >> a[i];

        s += a[i];
    }

    // xotirani bo'shatish
    delete []a;

    cout << s << endl;
    return 0;
}
```

Nazorat savollari:

1. Massiv nima?
2. Bir o'lchamli massiv deb nimaga aytiladi?
3. Massiv elementi nima?
4. Massiv indeksi nima?
5. Bir o'lchamli massiv qanday e`lon qilinadi?
6. Massiv elementlari soni qanday aniqlanadi?

Ma’ruzalar matnining elektron versiyasidan mavzuga oid 140 ta masalani ishlab chiqing.

Ko’p o’lchamli massivlar

Assalomu alaykum bo`lajak dasturchi! Yangi mavzuni boshlashdan oldin, oldingi mavzuni qisqacha takrorlab olsak. Quyidagi savollarga og`izaki yoki yozma javob bering. Javob qanchalik to`g`riligini tekshirish uchun savolni bir marta bosing.

Bir o`lchamli massivlar uchun ishlatilgan o`zgaruvchilar, bir xil jinsdagi berilganlarni xotirada saqlash uchun foydalaniladi. Ikki o`lchamli massivlarda esa, satr va ustunlar orqali bir xil jinsdagi qiymatlarni ikki o`lchamli o`zgaruvchilar ichida saqlash uchun foydalaniladi.

Ikki o`lchamli statik massivlarni e`lon qilish.

toifa massiv_nomi [massiv_satrlari_soni][massiv_ustunlari_soni];

Ikki o`lchamli statik massivlarning e`lon qilinishida, bir o`lchamlidan farqi, massiv nomidan keyin qirrali qavs ichida ikkita qiymat yozilganligidadir. Bulardan birinchisi, satrlar sonini, ikkinchisi esa ustunlar sonini bildiradi. Ya'ni ikki o'lchamli

massiv elementiga ikkita indeks orqali murojaat qilinadi. Ikki o`lchamli massivlar matematika kursidan ma`lum bo`lgan matritsalarni eslatadi.

Ikki o'lchamli massiv e'loniga misol:

int a[3][3], b[2][4];

A matritsa	B matritsa
a_{00} a_{01} a_{02} a_{10} a_{11} a_{12} a_{20} a_{21} a_{22}	b_{00} b_{01} b_{02} b_{03} b_{10} b_{11} b_{12} b_{13}

A matritsa 3 ta satr, 3 ta ustunga ega;
B matritsa 2 ta satr, 4 ta ustunga ega;

Ikki o'lchamli massivlarda 1 - indeks satrni, 2 - indeks ustunni bildiradi.

Birinchi satrning dastlabki elementi a_{10} – a biru nol element deb o`qiladi. a o`n deyilmaydi.

m ta satr va n ta ustunga ega bo`lgan massivga (mxn) o`lchamli massiv deyiladi. Agar m=n (satrlar va ustunlar soni teng) bo'lsa kvadrat massiv deyiladi.

Ko'p o'lchamli massivlarni initsializatsiyalash misollar:

int a[2][2]={1,2,7,3};
int b[2][3]={ {0,1,2}, {3,4,5} };

Massivlarni qo`llanilishiga misol keltiradigan bo`lsak, satrlar talabalarni, ustunlar fanlardan olgan baholarini bildirsin. Ya`ni m ta talaba, n ta fan. n - ustunga talabalarning o`rtacha baholari hisoblanib, shu asosida stipendiya bilan ta`minlansin. Va hakazo, bunga o`xshash ko`plab misollar keltirish mumkin. Bu masalalarga to`xtalishdan oldin bir ikkita oddiy masalar bilan tanishib chiqaylik.

1 - Masala. A(mxn) matritsa berilgan. Shu matritsa elementlarini kirituvchi va ekranga jadval ko`rinishida chiqaruvchi programma tuzilsin.

```
#include <iostream>

using namespace std;

int main()
{
  int  m, n, a[10][10];

  cout << "Satrlar sonini kiriting \nm=";
```

```
  cin >> m;
  cout << "Ustunlar sonini kiriting \nn=";
  cin >> n;

  cout <<"Massiv elementlarini kiriting \n";
  for(int satr = 0;  satr < m ; satr++)
  for(int ustun = 0; ustun < n; ustun++)
  {
    cout << "a[" << satr << "][" << ustun << "]=";
    cin >> a[satr][ustun];
  }

  // matritsani jadval shaklida chiqarish
  for(int satr = 0; satr < m; satr++)
  {
    for(int ustun = 0; ustun < n; ustun++)
    cout << a[satr][ustun] << "\t";

    cout<<"\n";
  }

  return 0;
}
```

Funksiya parametri sifatida massivni jo'natish va funksiya natijasi sifatida massivni olish ham mumkin. Funksiyaga matritsani uzatishda matritsa nomi bilan uning satrlar va ustunlar sonini ham jo'natish kerak bo'ladi. Funksiyada massivdan foydalanishni bir necha xil usuli bor, shularning ba'zilari bilan tanishamiz.

Funksiyaga matritsani uzatish

```
// Muallif: Qudrat Abdurahimov
// Sana: 20.08.2012
// Maqsad: Funksiyaga matritsani uzatishni o'rganish
#include <iostream>

using namespace std;

void matrix_print(int a[10][10], int m, int n)
{
    // matritsani jadval shaklida chiqarish
    for (int i = 0; i < m; i++)
    {
        for (int j = 0; j < n; j++)
        {
            cout << a[i][j] << "\t";
        }
        cout << "\n";
    }
}
```

```
void matrix_input (int a[10][10], int m, int n)
{
    cout <<"Massiv elementlarini kiriting \n";
    for (int i = 0; i < m; i++)
    for (int j = 0; j < n; j++)
    cin >> a[i][j];
}
int main()
{
    int  m, n, a[10][10];

    cout << "Satrlar sonini kiriting \nm=";  cin >> m;
    cout << "Ustunlar sonini kiriting \nn="; cin >> n;

    //funksiyaga matritsa, satrlar va ustunlar soni jo'natiladi
    matrix_input(a, m, n);

    cout << "Kiritilgan matritsa\n";
    //funksiyaga matritsa, satrlar va ustunlar soni jo'natiladi
    matrix_print(a, m, n);

    return 0;
}
```

Matritsadagi har bir satrning eng kattasini topish

```
// Muallif: Qudrat Abdurahimov
// Sana: 20.08.2012
// Maqsad: Matritsadagi har bir satrning eng kattasini topish
#include <iostream>

using namespace std;

void matrix_print(int a[10][10], int m, int n)
{
    // matritsani jadval shaklida chiqarish
    for (int i = 0; i < m; i++)
    {
        for (int j = 0; j < n; j++)
        {
            cout << a[i][j] << "\t";
        }
        cout << "\n";
    }
}

int satr_max(int a[], int n)
{
    // massivning eng katta elementini aniqlash
    int max = a[0];
```

```
    for (int i = 1; i < n; i++)
    if (max < a[i]) max = a[i];

    return max;
}

int main()
{
    int  m, n, a[10][10];

    cout << "Satrlar sonini kiriting \nm=";  cin >> m;
    cout << "Ustunlar sonini kiriting \nn="; cin >> n;

    cout <<"Massiv elementlarini kiriting \n";
    for (int i = 0; i < m; i++)
    for (int j = 0; j < n; j++)
    cin >> a[i][j];

    cout << "Kiritilgan matritsa\n";
    //funksiyaga matritsa, satrlar va ustunlar soni jo'natiladi
    matrix_print(a, m, n);

    for (int i = 0; i < m; i++)
    {
        // funksiyaga i-satrning 0-elementi adresini
        // va elementlar sonini jo'natamiz
        cout << i << "-satrning eng kattasi=" <<
satr_max(&a[i][0], n);
        cout << endl;
    }

    return 0;
}
```

Statistik massivlar bilan ishlashning kamchiliklari:

1. Statik massivlarning uzunligi oldindan ma’lum bo’lishi kerak;
2. Statik massivning o'lchami berilganlarga ajratilgan xotiraning o’lchami bilan chegaralangan;
3. Katta o’lchamdagi massiv e’lon qilib, masalani yechilishida ajratilgan xotira to’liq ishlatimasligi mumkin.

Bu kamchiliklarni dinamik massivlardan foydalanish orqali xal qilish mumkin.

Ikki o'lchamli dinamik massivlar bilan ishlash

Ikki o’lchamli dinamik massivni tashkil qilish uchun «ko’rsatkichga ko’rsatkich» dan foydalaniladi:

int **a;

Endi satrlar soniga qarab ko’rsatkichlar massiviga dinamik xotira ajratish kerak:

a = new int *[satrlar_soni];

Har bir satrga takrorlash operatori yordamida xotira ajratish kerak va uning boshlang’ich adresini a massiv elementlariga joylashtirish zarur:

for (int i = 0; i < satrlar_soni; i++)
a[i] = new int [ustunlar_soni];

Dinamik massivda har bir satr xotiraning turli joylarida joylashishi mumkin.

Dinamik massivlarni ishlatib bo’lgandan keyin albatta delete amali bilan uni o’chirish kerak. Yuqoridagi misolda ikki o’lchamli massiv uchun avval massivning har bir elementi, oxirida massivning o’zi o'chiriladi:

for (int i = 0; i < satrlar_soni; i++)
delete [] a[i];

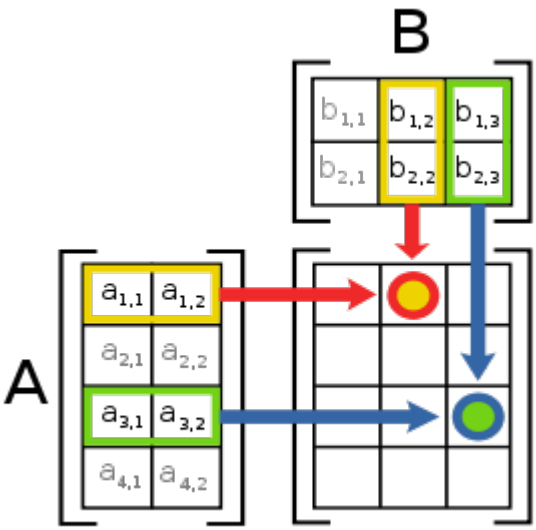
delete []a;

1 - Masalani dinamik massivdan foydalanga holda ishlashni ko'rib chiqamiz.

```
#include <iostream>

using namespace std;

int main()
{
  int  m, n;
  int **a; // ko'rsatkichga ko'rsatkich

  cout << "Satrlar sonini kiriting \nm=";
  cin >> m;
  cout << "Ustunlar sonini kiriting \nn=";
  cin >> n;

  // m ta ko'rsatkichlar massivi uchun xotira ajratish
  a = new int *[m];

  // har bir satr uchun dinamik xotira ajratish
  for (int i = 0; i < m; i++)
  a[i] = new int [n];

  cout <<"Massiv elementlarini kiriting \n";
  for(int satr = 0;  satr < m ; satr++)
  for(int ustun = 0; ustun < n; ustun++)
  {
```

```
    cout << "a[" << satr << "][" << ustun << "]=";
    cin >> a[satr][ustun];
  }

  // matritsani jadval shaklida chiqarish
  for(int satr = 0; satr < m; satr++)
  {
    for(int ustun = 0; ustun < n; ustun++)
    cout << a[satr][ustun] << "\t";

    cout<<"\n";
  }

  // Dinamik massivdan foydalanib bo'lgandan so'ng
  // uni xotiradan o'chirish lozim
  // oldin massiv satrlari o'chiriladi (xotira bo'shatiladi)
  for (int i = 0; i < m; i++)
  delete []a[i];

  // endi massivning o'zini o'chirish mumkin
  delete [] a;

  return 0;
}
```

a(m, n) matritsani b(n, p) matritsaga ko'paytirishdan hosil bo'lgan c(m, p) matritsani chiqaruvchi programma tuzilsin.




```
// Muallif: Qudrat Abdurahimov
// Sana: 20.08.2012
// Maqsad: Matritsani matritsaga ko'paytirish
#include <iostream>

using namespace std;
```

```
void matrix_print(int **a, int m, int n)
{
    // matritsani jadval shaklida chiqarish
    for (int i = 0; i < m; i++)
    {
        for (int j = 0; j < n; j++)
        {
            cout << a[i][j] << "\t";
        }
        cout << "\n";
    }
}

void matrix_input (int **a, int m, int n)
{
    for (int i = 0; i < m; i++)
    for (int j = 0; j < n; j++)
    cin >> a[i][j];
}

void matrix_delete (int **a, int m)
{
    // Dinamik massivdan foydalanib bo'lgandan so'ng
    // uni xotiradan o'chirish lozim
    // oldin massiv satrlari o'chiriladi (xotira bo'shatiladi)
    for (int i = 0; i < m; i++)
    delete []a[i];

    // endi massivning o'zini o'chirish mumkin
    delete []a;
}

// create_matrix - ko'rsatkichga ko'rsatkich qaytaradi
int ** create_matrix(int m, int n)
{
    int **ptr;
    // matritsa satrlari uchun xotira ajratish
    ptr = new int *[m];

    for (int i = 0; i < m; i++)
    // matritsa ustunlari uchun xotira ajratish
    ptr[i] = new int [n];

    return ptr;
}
int main()
{
    int  m, n, p, **a, **b, **c;

    cout << "A matritsa satrlar sonini kiriting \nm=";  cin >> m;
    cout << "A matritsa ustunlar sonini kiriting \nn="; cin >> n;
    cout << "B matritsa ustunlar sonini kiriting \np="; cin >> p;
```

```
    // matritsalarni hosil qilish
    a = create_matrix(m, n);
    b = create_matrix(n, p);
    c = create_matrix(m, p);

    cout << "A massiv elementlarini kiriting \n";
    matrix_input(a, m, n);

    cout << "B massiv elementlarini kiriting \n";
    matrix_input(b, n, p);

    cout << "Kiritilgan A matritsa\n";
    matrix_print(a, m, n);

    cout << "Kiritilgan B matritsa\n";
    matrix_print(b, n, p);

    // c matritsa elementlarini 0 qilish
    for (int i = 0; i < m; i++)
    for (int j = 0; j < p; j++)
    c[i][j] = 0;

    // A matritsani B matritsaga ko'paytirish
    for (int i = 0; i < m; i++)
    for (int j = 0; j < n; j++)
    for (int k = 0; k < p; k++)
    {
        c[i][k] += a[i][j] * b[j][k];
    }

    cout << "Natija C matritsa\n";
    matrix_print(c, m, p);

    matrix_delete(a, m); // a matritsani o'chirish
    matrix_delete(b, n); // b matritsani o'chirish
    matrix_delete(c, m); // c matritsani o'chirish

    return 0;
}
```

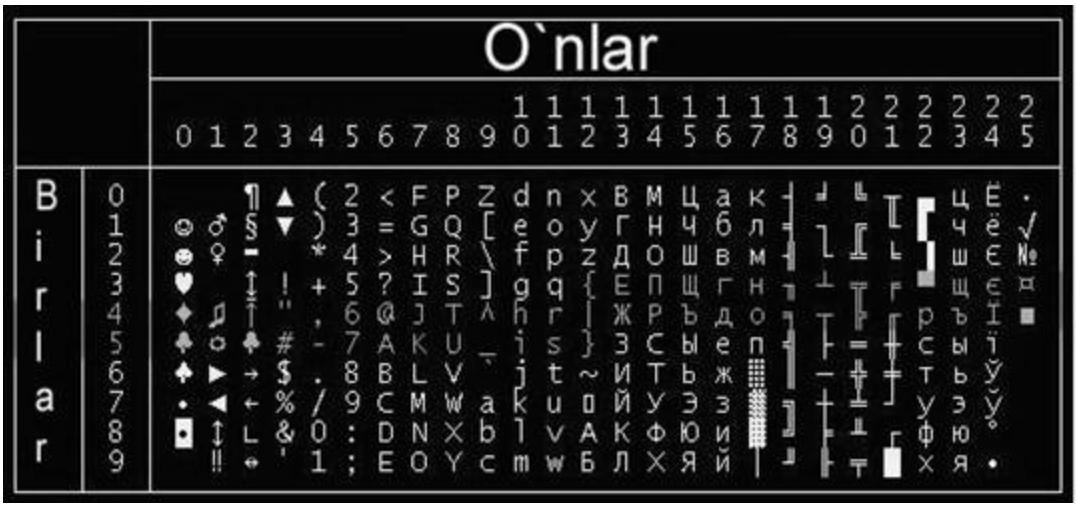
Nazorat savollari:

1. Ko`p o`lchamli massivlarni e`lon qilishning qanday usullarini bilasiz?
2. Qanday massivlarga kvadrat massivlar deyiladi?
3. Bir o'lchamli va ikki o'lchamli massivlar orasidagi farq nimada?
4. Statistik massivlar bilan ishlashning kamchiliklari nimada?
5. Ikki o'lchamli dinamik massivlar bilan ishlash qanday amalga oshiriladi?

Belgili o`zgaruvchilar. char toifasidagi satrlar

Belgini (simvolni) saqlash uchun mo`ljallangan o'zgaruvchilarga belgili o'zgaruvchilar deyiladi. C++ tilida bu o'zgaruvchilar uchun **char** toifasi keltirilgan. char toifasidagi o'zgaruvchi ASCII kodidagi 255 ta belgidan ixtiyoriy birisi bo'lishi mumkin.

ASCII kodi jadvali



Belgili o'zgaruvchilarni e'lon qilish quyidagicha bo'ladi:
char c, s;

belgili o'zgaruvchilar apostraf ichida yoziladi.
a='q'; c='*'; s='/';

char toifasini oshkor ravishda butun toifaga o'tkazish orqali, berilgan belgiga mos ASCII kodini aniqlash mumkin.

ASCII kodi jadvaliga e`tibor bering. Katta va kichik lotin harflari alifbo tartibida joylashtirilgan. Bu esa sikl orqali char toifasini tashkil qilish imkoniyatini beradi.

Sikl orqali lotin harflarini chiqarish

```
#include <iostream>
using namespace std;
int main()
{
    cout << "KATTA LOTIN HARFLARI" << endl;
    for (int i = 65; i <= 90; i++)
    cout << i << "->" << (char) i << endl;

    cout << "kichik lotin harflari" << endl;
    for (int i = 97; i <= 122; i++)
```

```
    cout << i << "->" << (char) i << endl;

    return 0;
}
```

Yoki siklni quyidagicha ham tashkil qilish mumkin:

```
    for (char c = 'a'; c <= 'z'; c++)
    cout << (int) c << " " << c << endl;
```

C++ da satr. C++ da satr deb - satr oxiri ('\0') belgisi bilan tugaydigan belgilar massiviga aytiladi. Demak C++ da satr - birinchi belgiga o'rnatilgan ko'rsatkich ekan. Chunki massiv ham, birinchi elementiga o'rnatilgan ko'rsatkichdir.

Satrni belgilar massivi ko'rinichida yoki char toifasidagi ko'rsatkich sifatida e'lon qilish mumkin.

char satr1[ ] = "dastur.uz";
char * satr2[ ] = "dastur.uz";

satr1 massivi 10 ta elementdan, 'd', 'a', 's', 't', 'u', 'r', '.', 'u', 'z', '\0' belgilaridan iborat.

Satrni kiritishda cin.getline funksiyasidan foydalanish mumkin.
cin.getline(satr, satr_uzunligi);

char satr[15];
cin.getline(satr, 15);

getline funksiyasining 2 - parametri sifatida sizeof funksiyasidan foydalanish tavsiya etiladi.

cin.getline(satr, sizeof(satr));

sizeof va strlen funksiyalarining farqi

```
#include <iostream>
using namespace std;
int main()
{
    char s[20];

    cout << "Satr kiriting" << endl;
    cin.getline(s, sizeof(s));

    cout << "sizeof(s)=" << sizeof(s) << endl;
    cout << "strlen(s)=" << strlen(s) << endl;
```

```
    return 0;
}
```

Belgiga mos ASCII kodini aniqlash

```
#include <iostream>
using namespace std;
int main()
{
    char * cPtr;
    char s[] = "Assalomu alaykum";

    cPtr = s;

    cout << s << endl;
    while ( *cPtr != '\0')
    {
        cout << *cPtr << "="<< (int) *cPtr << endl;
        cPtr++;
    }

    return 0;
}
```

Belgilarni qayta ishlovchi funksiyalar

Quyidagi funksiyalardan foydalanish uchun ctype.h sarlavha faylini progarmmaga qo'shish kerak.

Funksiya prototipi	Funksiya tavsifi
int isdigit(int c)	Agar c raqam bo'lsa true, aks xolda false qiymat qaytaradi.
int isalpha(int c)	Agar c harf bo'lsa true, aks xolda false qiymat qaytaradi.
int isalnum(int c)	Agar c raqam yoki harf bo'lsa true, aks xolda false qiymat qaytaradi.
int islower(int c)	Agar c kichik harf bo'lsa true, aks xolda false qiymat qaytaradi.
int isupper(int c)	Agar c katta harf bo'lsa true, aks xolda false qiymat qaytaradi.
int tolower(int c)	Agar c katta harf bo'lsa kichik harf qaytariladi, aks xolda tolower argumentni o'zgarish qaytaradi.
int toupper(int c)	Agar c kichik harf bo'lsa katta harf qaytariladi, aks xolda toupper argumentni o'zgarish qaytaradi.

Kiritilgan satrni katta harflar bilan chiqaruvchi programma tuzilsin

```
#include <iostream>
#include <ctype.h>
using namespace std;
int main()
{
    char c[20];

    cout << "satr kiriting\n";
    cin.getline(c, sizeof(c));

    for (int i = 0; i < strlen(c); i++)
    c[i] = toupper(c[i]);

    cout << c << endl;
    return 0;
}
```

Toifalarni o'zgartirish funksiyalari

Quyidagi funksiyalardan foydalanish uchun stdlib.h sarlavha faylini progarmmaga qo'shish kerak.

Funksiya prototipi	Funksiya tavsifi
`double atof(const char *c)`	c satrini **double** toifasiga o'zgartiradi.
`int atoi(const char *c)`	c satrini **int** toifasiga o'zgartiradi.
`int atol(const char *c)`	c satrini **long int** toifasiga o'zgartiradi.
`double strtod(const char *c, char **endPtr)`	c satrini **double** toifasiga o'zgartiradi.
`char * itoa(int n, char *satr, int radix)`	n sonini radix sanoq sistemasida satr o'zgaruvchisiga o'zlashtiradi.

Satrni butun va haqiqiy songa aylantirish

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

using namespace stds;
```

```
int main ()
{
  char c[] = "3.1415";
  double f;
  int n;

  f = atof(c);
  n = atoi(c);
  cout << f << endl;
  cout << n << endl;

  return 0;
}
```

Nazorat savollari:
1. Qanday o'zgaruvchilar simvolli o'zgaruvchilar deyiladi?
2. ASCII kodida "$" belgisining kodi necha?
3. Satr deb nimaga aytiladi?
4. Belgilarni qayta ishlovchi funksiyalardan qaysilarini bilasiz?
5. Toifalarni o'zgartirish funksiyalardan qaysilarini bilasiz?

Standart kutubxonadagi string sinfi

C++ da satrlar bilan ishlashni qulaylashtirish uchun string sinfi kiritilgan. string sinfi satrlarida satr oxirini '\0' belgisi belgilamaydi.

string sinfidan foydalanish uchun qaysi sarlavha faylini dasturga qo'shish kerak?

Standart kutubxonadagi string sinfidan foydalanish uchun <string> sarlavha faylini dasturga qo'shish kerak.

Lekin ba'zi eski kompilyatorlarda <cstring.h> yoki <bstring.h> sarlavha faylini qo'shish kerak bo'ladi. Oddiy eski usuldagi satrlar bilan ishlash uchun esa, <string.h> sarlavha fayli qo'shiladi.

Eng afzali, o'zingiz ishlatayotgan kompilyator bilan yaxshilab tanishib chiqing.

Satrlar bilan ishlovchi asosiy funksiyalar bilan tanishib chiqamiz.

```
Satr xususiyatlarini aniqlash uchun quyidagi funksiyalardan
foydalanish mumkin:
  unsigned int size() const;        // satr o'lchami
  unsigned int length() const;      // satr elementlar soni
  unsigned int max_size() const;    // satrning maksimal uzunligi
  unsigned int capacity() const;    // satr egallagan xotira hajmi
```

```
  bool empty() const;                // satrning bo’shligini
aniqlash
```

Satrning uzunligini aniqlash uchun length() yoki size() funksiyalaridan foydalanish mumkin.

```
// Muallif: Qudrat Abdurahimov
// Sana : 04.02.2012
// Maqsad: Satr uzunligini aniqlash

#include <iostream>
#include <string>

using namespace std;

int main()
{
    string s;

    cout << "Satr kiriting" << endl;
    getline(cin, s);

    cout << "Siz kiritgan satr " << s.length() << " ta belgidan iborat";
    cout << "Siz kiritgan satr " << s.size() << " ta belgidan iborat";

    return 0;
}
```

Satr uzunligini o'zgartirish uchun resize funksiyasidan foydalaniladi

```
1) void resize ( size_t n, char c );
2) void resize ( size_t n );

#include <iostream>
#include <string>

using namespace std;

int main()
{
    size_t n;

    string str ("I like to code in C");
    cout << str << endl;

    // satr uzunligini aniqlash
    n = str.size();

    // satr uzunligini 2 ta belgiga uzaytirish
    str.resize (n + 2, '+');
    cout << str << endl;
```

```
    //satr uzunligini o'zgartirish
    str.resize(14);
    cout << str << endl;

    return 0;
}
```

void clear(); - funksiyasi satrni tozalash (to'liq o'chirish) uchun ishlatiladi.

bool empty() const; - funksiyasi satrni bo'shligini tekshirish uchun ishlatiladi. Agar satr bo'sh bo'lsa, true qiymat qaytaradi.

Satrning biror qismidan nusxa olish

string& assign (const string &str);

Satrga str o'zgaruvchisidagi satrning to'liq nusxasini olish.

string& assign (const string& str, size_t pos, size_t n);

Satrga str o'zgaruvchisidagi satrning pos o'rindagi belgisidan boshlab n ta belgi nusxasini olish.

string& assign (const char* s, size_t n);

string toifasidagi satrga char toifasidagi satrning n ta belgisi nusxasini olish.

```
string s1, s2, s3;
s1 = "dastur.uz";
s2.assign(s1); // s2 = "dastur.uz"
s3.assign(s1, 0, 6); // s3 = "dastur"
```

append funksiyasining assigndan farqi satrning davomiga satr qismining qo'shishidadir.

string& append (const string& str);
string& append (const string& str, size_t pos, size_t n);
string& append (const char* s, size_t n);

Satrdan nusxa olish

```
// Muallif: Qudrat Abdurahimov
// Sana : 04.02.2012
// Maqsad: Satrdan nusxa olish

#include <iostream>
#include <string>
```

```
using namespace std;

int main()
{
    string s1, s2, s3;

    s1 = "GULBAHOR";

    s2.assign(s1, 0, 3); // s2 = "GUL"

    s3.assign(s1, 3, 5); // s3 = "BAHOR"

    cout << s1 << endl;
    cout << s2 << endl;
    cout << s3 << endl;

    s1 = s3 + s2;       // s1 = "BAHORGUL"
    cout << s1 << endl;

    s2.append(s3);      // s2 = "GULBAHOR"
    cout << s2 << endl;

    return 0;
}
```

char toifasidagi satrni string toifasiga o'tkazish. clear funksiyasiga misol.

```
// Muallif: Qudrat Abdurahimov
// Sana : 30.07.2012
// Maqsad: char toifasidagi satrni string toifasiga o'girish

#include <iostream>
#include <string>

using namespace std;

int main()
{
    char c[100];
    string s = "Boshlang'ich qiymat";

    s.clear(); // s ning qiymati tozalandi

    cout << "Satr kiriting" << endl;
    cin.getline(c, sizeof(c));

    // s satriga char toifasidagi c satrini nusxasini olish
    s.assign(c, strlen(c));

    cout << s << endl;

    return 0;
}
```

string toifasidagi satrni char toifasiga o'tkazish

string toifasidagi satrni char toifasiga o'tkazish uchun c_str yoki data funksiyalaridan foydalanish mumkin.

const char * c_str() const;
const char * data() const;

Bularning bir - biridan farqi, data funksiyasida satr oxiriga '\0' satr oxiri belgisi qo'shilmaydi.

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

int main()
{
    char c[100];
    string s;

    cout << "Satr kiriting" << endl;
    getline(cin, s);

    strcpy(c, s.c_str());

    cout << c << endl;

    return 0;
}
```

Satrning biror qismini o’chirish

erase(unsigned int pos=0, unsigned int n=npos);

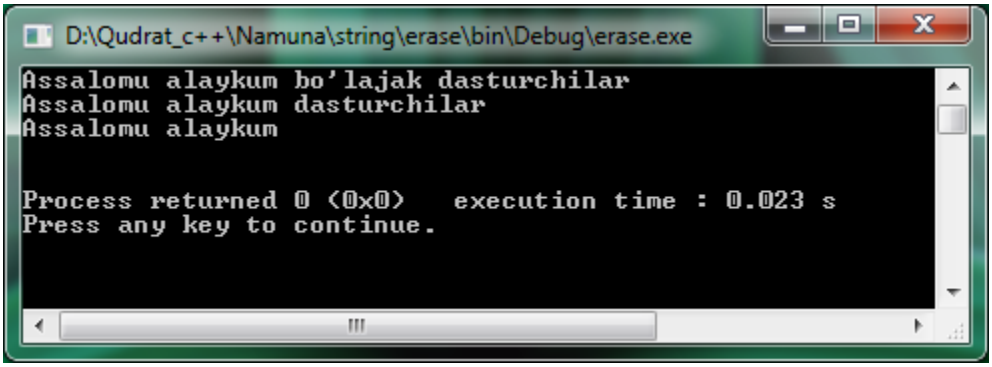
erase funksiyasi satrni pos o'zgaruvchisida ko'rsatilgan o'rindan boshlab n ta belgini o'chiradi.

Agar nechta belgi o'chirilishi n ko'rsatilmagan bo'lsa, pos o'zgaruvchisida ko'rsatilgan o'rindan boshlab satr oxirigacha o'chiriladi.

Agar pos va n ko'rsatilmagan bo'lsa, satr to'liq o'chiriladi.

```
// Muallif: Qudrat Abdurahimov
// Sana : 30.07.2012
// Maqsad: Satr qismini o'chirish
```

```
#include <iostream>
#include <string>
using namespace std;

int main()
{
    string s = "Assalomu alaykum bo'lajak dasturchilar";

    cout << s << endl;

    // 16 - belgidan boshlab 9 ta belgini o'chirish
    s.erase(16, 9);
    cout << s << endl; // Assalomu alaykum dasturchilar - chiqadi

    // s = "Assalomu alaykum dasturchilar";
    // 16 - belgidan boshlab satr oxirigacha o'chirish
    s.erase(16);
    cout << s << endl; // Assalomu alaykum - chiqadi

    s.erase();
    cout << s << endl; // bo'sh satr chiqariladi

    return 0;
}
```



Satrni satr orasiga qo'shish

Biror satrga boshqa satrning istalgan qismini qo'shish uchun insert funksiyasidan foydalaniladi.

insert(unsigned int pos1, const string &str);

Satrga pos1 o'rindan boshlab, str satrini qo'shish.

---

insert(unsigned int pos1, const string &str, unsigned int pos2,unsigned int n);

Satrga pos1 o'rindan boshlab, str satrining pos2 o'rnidan boshlab n ta belgini qo'shish.

---

insert(unsigned int pos1, const char *str, int n);

Satrga pos1 o'rindan boshlab, char toifasidagi satrning n ta belgisini qo'shish.

```
// Muallif: Qudrat Abdurahimov
// Sana : 30.07.2012
// Maqsad: Satrga boshqa satr qismini qo'shish

#include <iostream>
#include <string>
using namespace std;

int main()
{
    string s = "Assalomu alaykum do'stlar";
    string c = "Merhibon_va_muhtarama_ayollar";
    char p[] = "Salomlashish odobi";

    cout << s << endl;

    // 17 - belgidan boshlab yangi satrni qo'shish
    s.insert(17, "aziz_");
    cout << s << endl; // Assalomu alaykum aziz do'stlar

    // c satridan 8 - belgidan boshlab 10 ta belgi qo'shish
    s.insert(21, c, 8, 10);
    cout << s << endl; // Assalomu alaykum aziz_va_muhtar_do'stlar

    // char toifasidagi satrdan 13 ta belgini qo'shish
    s.insert(0, p, 13);
    cout << s << endl;

    return 0;
}
```

Satr qismini almashtirish

Satrning biror qismini almashtirish kerak bo'lsa, replace funksiyasidan foydalanish mumkin.

replace (unsigned int pos1, unsigned int n1, const string &str);

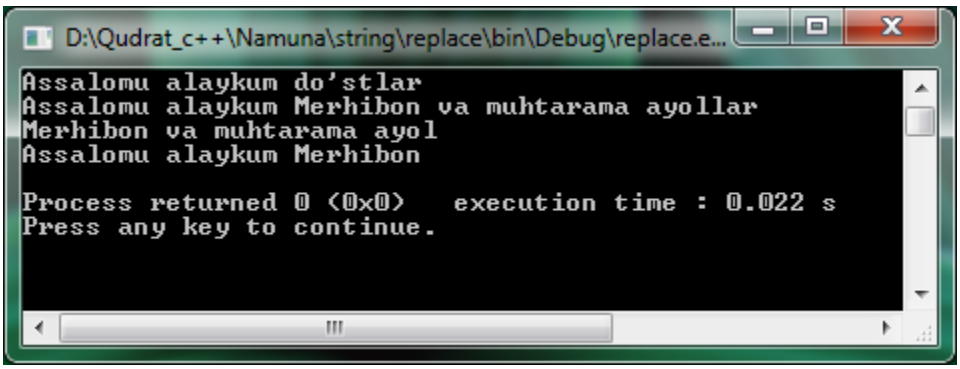
replace (unsigned int pos1, unsigned int n1, const string & str, unsigned int pos2, unsigned int n2);

replace (unsigned int pos1, unsigned int n1, const char *str, int n);

replace funksiyasi insert kabi ishlaydi, faqat qo'shilishi kerak bo'lan satrni pos1-o'rindan boshlab n1 ta belgi o'rniga qo'shadi.

2 ta satrni to'la almashtirish uchun swap funksiyasi ishlatiladi.

```
// Muallif: Qudrat Abdurahimov
// Sana : 30.07.2012
// Maqsad: Satr qismini almashtirish funsiyasiga misol

#include <iostream>
#include <string>
using namespace std;

int main()
{
    string s = "Assalomu alaykum do'stlar";
    string c = "Merhibon va muhtarama ayol";

    cout << s << endl;

    // 17 - belgidan boshlab 5 ta belgi o'rniga c satrni qo'shish
    s.replace(17, 5, c);
    cout << s << endl;

    s.swap(c); // 2 ta satrni to'la almashtirish
    cout << s << endl;

    s.replace(0, 0, c, 0, 17);
    s.erase(25);
    cout << s << endl;

    return 0;
}
```



Satrlarni solishtirish

Satrlarni solishtirish uchun compare funktsiyasi ishlatiladi:

int compare (const string &str) const;
int compare (unsigned int pos1, unsigned int n1, const string & str) const;
int compare (unsigned int pos1, unsigned int n1, const string & str, unsigned int pos2, unsigned int n2) const;

compare funksiyasini chaqiruvchi satr, str o'zgaruvchisidagi satrdan kichik bo'lsa, manfiy qiymat qaytariladi. Katta bo'lsa musbat va teng bo'lsa 0 qiymat qaytariladi.

```
#include <iostream>
#include <string>
using namespace std;

int main()
{
    string s = "Assalomu alaykum";
    string c = "vaalaykum dasturchi";

    cout << "s=" << s << endl;
    cout << "c=" << c << endl;
    cout << "s.compare(c)=" << s.compare(c) << endl; // -1
    cout << "c.compare(s)=" << c.compare(s) << endl; // 1
    cout << "c.compare(2,7,s,9,7)=" << c.compare(2,7,s,9 7) << endl; //0

    return 0;
}
```

Satr qismini ajratib olish funksiyasi
string substr(unsigned int pos=0, unsigned int n=npos) const;

Bu funktsiya chaqiruvchi satrdan pos o’rnidan boshlab n belgini natija sifatida qaytaradi, agarda pos ko’rsatilmasa, satr boshidan boshlab n ta belgi ajratib olinadi.

```
string s,c;
s = "GULOY";
c = s.substr(3, 2) + s.substr(0, 3);
cout << c << endl; // OYGUL - chiqadi
```

Satrdan qidirish funksiyalari

unsigned int find(const string &str, unsigned int pos=0) const;

Bu funksiyani chaqirgan satrning pos o'zgaruvchisida ko’rsatilgan joyidan boshlab str satrni qidiradi.
Agar qidirilayotgan satr (str) topilsa, mos keluvchi satr qismining boshlanish indeksini javob sifatida qaytaradi, aks holda (satrning maksimal uzunligi qiymati) npos sonini qaytaradi. (npos=4294967295)

Agar pos ko'rsatilmasa, satr boshidan boshlab izlanadi.

unsigned int rfind(const string &str, unsigned int pos=npos) const;

Bu funksiyani chaqirgan satrdan pos o'ringacha str satri qidiriladi. Agar str topilsa, oxirgi uchragan indeks qaytariladi.

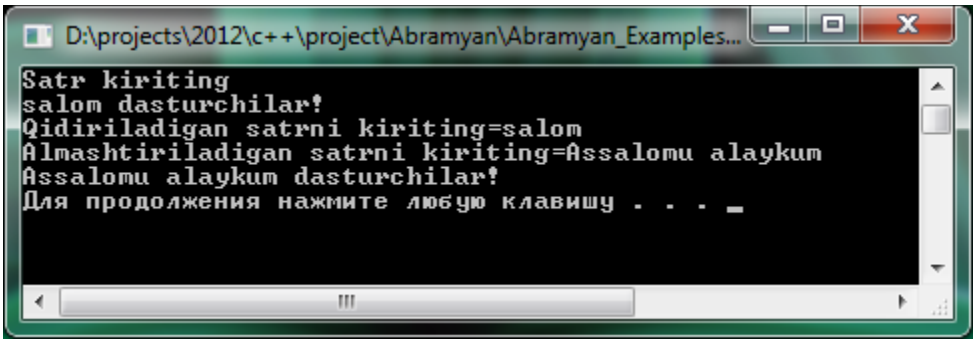
Agar pos ko'rsatilmasa, satr oxirigacha izlanadi. Ya'ni oxirgi uchragan indeks qaytariladi. Agar topilmasa, npos qaytariladi.

```
// Muallif: Qudrat Abdurahimov
// Sana : 30.07.2012
// Maqsad: Satrdan qidirish

#include <iostream>
#include <string>
using namespace std;

int main()
{
    string s = "Assalomu alaykum";
    string c = "alaykum";

    cout << "s=" << s << endl;
    cout << "c=" << c << endl;
    cout << "s.find(c)=" << s.find(c) << endl; // 9
    cout << "c.find(s)=" << c.find(s) << endl; // 4294967295

    // birinchi uchragan "a" harfining o'rnini aniqlash
    cout << "s.find('a')=" << s.find("a") << endl; // 3

    // oxirgi uchragan "a" harfining o'rnini aniqlash
    cout << "s.rfind('a')=" << s.rfind("a") << endl; // 11

    return 0;
}
```

Satr qismini almashtirishga misol

```
// Muallif: Qudrat Abdurahimov
// Sana : 29.07.2012
// Maqsad: Satr qismini almashtirish
#include <iostream>
#include <stdlib.h>
#include <string>
using namespace std;

int main()
{
```

```
    string c;
    string s, s1;

    size_t index;

    cout << "Satr kiriting" << endl;
    getline(cin, s);

    cout << "Qidiriladigan satrni kiriting=";     getline(cin, c);
    cout << "Almashtiriladigan satrni kiriting="; getline(cin,s1);

    index = s.find(c);
    while (index < s.length())
    {
        s.replace(index, c.length(), s1);
        index = s.find(c, index + s1.length());
    }

    cout << s << endl;
    system("pause");
    return 0;
}
```



Nazorat savollari:

1. Satr uzunligi qanday aniqlanadi?.
2. Satrdan nusxa olish qanday amalga oshiriladi?
3. Satrning biror qismini o'chirish qanday amalga oshiriladi?.
4. Satrni satr orasiga qo'shish qanday bo'ladi?
5. Satr qismini almashtirish qanday bo`ladi?
6. Satrdan qidirish funksiyalarini tushuntirib bering.

Rekursiya. O`z - o`zini chaqiruvchi funksiyalar

Assalomu alaykum bo`lajak dasturchi!

Yangi mavzu video fayli bilan tanishib chiqing.
Video faylni ko`rib bo`lgandan keyin ma`ruza matnini diqqat bilan o`qib chiqing.

C++ dasturlash tilida funksiyalar o`z – o`zini chaqirish imkoniyatiga ega. Bunday funksiyalar rekursiyali (o`z – o`zini chaqiruvchi) funksiya deyiladi.

Rekursiyali funksiyalarga qo`yiladigan asosiy talab, qandaydir qiymatda rekursiya yolg`on yoki rost qiymat qabul qilishi kerak. Shundagina chaqirilgan funksiyalar qaytadi. Aks holda funksiya o`z – o`zini davomli ravishda chaqiradi va xatolik sodir bo'ladi.

n! faktorialni rekursiyali funksiya orqali hisoblochi dastur tuzilsin. n!=1*2*…*(n-1)*n;

```
#include <iostream>
using namespace std;
int fact(int);
int main()
{
    int n;
    cout << "n="; cin >> n;

    cout << n << "!="<< fact(n) << endl;

    return 0;
}

int fact(int k)
{
    if (k == 0) return 1;

    return k * fact(k - 1);
}
```

Dasturni n=5 uchun tahlil qilamiz.

	N=5 kiritiladi fact(5)	Rekursiya`ning qaytishi 5!=120;	
v	i=5; fact(5) := 5*fact(4);	fact(5) := 5*24; (120)	^
v	i=4; fact(4) := 4*fact(3);	fact(4) := 4*6; (24)	^
v	i=3; fact(3) := 3*fact(2);	fact(3) := 3*2; (6)	^
v	i=2; fact(2) := 2*fact(1);	fact(2) := 2*1; (2)	^
v	i=1; fact(1) := 1*fact(0);	fact(1) := 1*1; (1)	^
v	i=0; fact(0) :=1;		^

Fibonachchi ketma ketligining n – hadini rekursiya qism dastur orqali hisoblovchi dastur

```
#include <iostream>
using namespace std;
int fib(int);
int main()
{
    int n;
    cout << "n="; cin >> n;

    cout << fib(n) << endl;

    return 0;
```

```
}

int fib(int k)
{
    if (k == 0 || k == 1) return 1;
    else return fib(k - 1) + fib(k - 2);
}
```

Nazorat savollari:

1. Rekursiya nima?
2. Rekursiyali funksiyaga qo`yiladigan talablar?

Fayllar bilan ishlash. Binar fayllar

Assalomu alaykum bo`lajak dasturchi! Yangi mavzu video fayli bilan tanishib chiqing.
Video faylni ko`rib bo`lgandan keyin ma`ruza matnini diqqat bilan o`qib chiqing.

Programma ishlashi natijasida olingan ma`lumotlarni, saqlab qo`yish uchun, CD, DVD disklar, qattiq disklar va boshqa har xil tashqi qurilmalardan foydalaniladi. Ma`lumotlarni saqlab qo`yish uchun tashqi qurilmalardan foydalanish qulay va ishonchlidir.

Ma`lumotlarni saqlab qo`yish uchun, tashqi xotiraning nomlangan qismiga fayl deyiladi. Bunday fayllar fizik fayllar deyiladi.

Mantiqiy fayllar. Fizik fayllar bilan ishlash uchun, programmalashtirish tillarida maxsus strukturalashgan, toifalangan fayllar kiritilgan. Bunday fayllar mantiqiy (logicheskiy) fayllar deyiladi. Mantiqiy fayllar, hech qanday fizik xotirani band qilmasdan ma`lumotlarning mantiqiy modelini o`zida saqlaydi.

Fizik va mantiqiy fayllar bir - biri bilan fopen funksiyasi orqali bog'lanadi.

Fayl bir nechta elementdan tashkil topgan bo`lganligi uchun, faqat fayl ko`rsatkichi ko`rsatayotgan elementga murojaat qilish mumkin.

Fayldan o'qish yoki yozish mumkin bo'lgan o'rinni ko'rsatuvhi elementga fayl ko'rsatkichi deyiladi. Fayldan ma'lumot o'qiganda yoki yozganda fayl ko'rsatkichi avtomat ravishda o'qilgan yoki yozilgan bayt miqdoricha siljiydi. Fayl ko'rsatkichini magnitafon galovkasiga o'xshatish mumkin.

Binar fayl - har xil ob'ektlarni ifodalovchi baytlar ketma - ketligidir. Ob'ektlar faylda qanday ketma - ketlikda joylashganini programmaning o'zi aniqlashi lozim.

Fayllar bilan ishlovchi funksiyalardan foydalanish uchun <stdio.h> sarlavha faylini programmaga qo'shish kerak bo'ladi.

Fayldan ma'lumotlarni o'qish yoki yozish uchun ochish fopen funksiyasi orqali amalga oshiriladi.

FILE * fopen (const char * filename, const char * mode);

filename - o'zgaruvchisi char toifasidagi satr bo'lib, faylning to'liq nomini ko'rsatishi lozim
(filename = "D:\Qudrat_c++\Namuna\file\file.txt"). Agar faylning faqat nomi ko'rsatilgan bo'lsa, fayl joriy katalogdan qidiriladi (filename = "file.txt").

mode - o'zgaruvchisi ham char toifasidagi satr bo'lib, faylni qaysi xolatda ochish lozimligini bildiradi.

mode qiymati	**Faylning ochilish xolati**
"w"	Faylni yozish uchun ochish. falename o'zgaruvchisida ko'rsatilgan fayl hosil qilinadi va unga ma'lumot yozish mumkin bo'ladi. Agar fayl oldindan bor bo'lsa (ya'ni oldin hosil qilingan bo'lsa), faylning ma'lumotlari o'chiriladi va yangi bo'sh fayl faqat yozish uchun ochiq holda bo'ladi.
"r"	Fayl o'qish uchun ochiladi. Agar fayl oldindan mavjud bo'lmasa, xatolik sodir bo'ladi. Ya'ni ochilishi lozim bo'lgan fayl oldindan hosil qilingan bo'lishi shart.
"a"	Faylga yangi ma'lumotlar qo'shish - kiritish uchun ochiladi. Yangi kiritilgan ma'lumotlar fayl oxiriga qo'shiladi. Agar fayl oldindan mavjud bo'lmasa, yangi fayl hosil qilinadi.
"w+"	Yozish va o'qish uchun faylni ochish. Agar fayl oldindan bor bo'lsa (ya'ni oldin hosil qilingan bo'lsa), faylning ma'lumotlari o'chiriladi va yangi bo'sh fayl yozish va o'qish uchun ochiqholda bo'ladi.
"r+"	Oldindan mavjud bo'lgan faylni o'qish va yozish uchun ochish.
"a+"	Fayl ma'lumotlarni o'qish va yangi ma'lumot qo'shish uchun ochiladi. fseek, rewind

Faylni ochishda xatolik sodir bo'lsa, fopen funksiyasi NULL qiymat qaytaradi.

Ochilgan faylni yopish uchun fclose funksiyasi ishlatiladi.

int fclose (FILE * stream);

Faylni yopishda xato sodir bo'lmasa, fclose funksiyasi nol qiymat qaytaradi. Xato sodir bo'lsa, EOF - fayl oxiri qaytariladi.

Faylga ma'lumot yozish va o'qish

size_t fread (void * ptr, size_t size, size_t n, FILE * stream);

fread funksiyasi, fayldan ptr ko'rsatkichi adresiga size xajmdagi ma'lumotdan n tani o'qishni amalga oshiradi. Agar o'qish muvoffaqiyatli amalga oshsa fread funksiyasi o'qilgan bloklar soni n ni qaytaradi. Aksholda nol qaytariladi

size_t fwrite (const void * ptr, size_t size, size_t n, FILE * stream);

fwrite funksiyasi, faylga ptr ko'rsatkichi adresidan boshlab size xajmdagi ma'lumotdan n tani yozishni amalga oshiradi.

```
1 - Misol. fread va fwrite funksiyalarining qo'llanilishi
#include <iostream>
#include <stdio.h>
using namespace std;

int main()
{
    int    n = 5;
    double d = 10.77;
    char s[20] = "dastur.uz";
    FILE *f;

    // binar faylni yozish uchun ochamiz
    f = fopen("my_file.dat", "wb");

    fwrite(&n, sizeof(int), 1, f); // n sonini faylga yozish
    fwrite(&d, sizeof(double), 1, f); // d sonini faylga yozish
    // satrni faylga yozish
    fwrite(s,  sizeof(char), strlen(s) + 1, f);
    fclose(f); // faylni yopish

    n = 0; d = 0;

    // binar faylni o'qish uchun ochamiz
    f = fopen("my_file.dat", "rb");
    fread(&n, sizeof(int), 1, f); // n sonini fayldan o'qish
    fread(&d, sizeof(double), 1, f); // d sonini fayldan o'qish
    // satrni fayldan o'qish
    fread(s,  sizeof(char), strlen(s) + 1, f);
    fclose(f); // faylni yopish

    cout << n << endl;
    cout << d << endl;
    cout << s << endl;
    return 0;
}
```

yuqoridagi misolda satrni yozish va o'qish uchun quyidagicha kod ishlatildi:

```
fwrite(s, sizeof(char), strlen(s) + 1, f);
fread (s, sizeof(char), strlen(s) + 1, f);
```

Buning kamchiligi s satridagi har bir belgi alohida - alohida faylga yozildi va o'qildi. Bu masalani quyidagicha hal qilish mumkin edi:

```
fwrite(s, sizeof(s), 1, f);
fread (s, sizeof(s), 1, f);
```

Lekin bu usulning ham kamchiligi bor. Ya'ni s satri belgilari soni massiv o'lchamidan kam bo'lgan holda, keraksiz ma'lumotlarni saqlash va o'qish sodir bo'ladi.

Fayl ko'rsatkichi bilan ishlovchi funksiyalar

Fayldan ma'lumot o'qiganda yoki yozganda fayl ko'rsatkichi avtomat ravishda o'qilgan yoki yozilgan bayt miqdoricha siljiydi. Fayl ko'rsatkichining kelgan joyini aniqlash uchun ftell funksiyasi ishlariladi.

long int ftell (FILE * stream);

Fayl ko'rsatkichini siljitish uchun fseek funksiyasi ishlatiladi.

int fseek (FILE * stream, long int offset, int whence);

Bu funksiya offset da ko'ratilgan bayt miqdoricha siljishni amalga oshiradi. whence o'zgaruvchisi quyidagi qiymatlarni qabul qilishi mumkin:

O'zgarmas	whence	Izoh
SEEK_SET	0	Fayl boshiga nisbatan siljitish
SEEK_CUR	1	Fayl ko'rsatkichining joriy xolatiga nisbatan siljitish
SEEK_END	2	Fayl oxiriga nisbatan siljitish

Agar whence = 1 bo'lsa (SEEK_CUR), offset musbat (o'ngga siljish) yoki manfiy (chapga siljish) bo'lishi mumkin.

Fayl ko'rsatkichini faylning boshiga o'rnatish uchun rewind funksiyasi ishlatiladi.

void rewind (FILE * stream);

Bu amalni fayl ko'rsatkichini siljitish orqali ham amalga oshirish mumkin.

fseek (f, 0, SEEK_SET);

Agar faylda faqat butun sonlar yozilgan bo'lsa, uning k - elementiga murojaat quyidagicha bo'ladi.

```
fseek (f, sizeof(int) * (k - 1), SEEK_SET);
fread (&n, sizeof(int), 1, f);
```

Fayl oxirini aniqlash uchun feof funksiyasi ishlatiladi.

int feof (FILE * stream);

feof funksiyasi fayl ko'rsatkichi fayl oxirida bo'lsa, noldan farqli qiymat qaytaradi. Boshqa hollarda nol qaytaradi.

2 - misol. n natural soni berilgan. Elementlari n ta butun sondan iborat bo`lgan faylni hosil qiluvchi va ekranga chiqaruvchi programma tuzilsin.

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

int main()
{
    int n, k;
    FILE *f;

    f = fopen("binar", "wb+");
    // binar faylni yozish va o'qish uchun ochish
    if (f == NULL)
    {
        cout << "Faylni hosil qilishda xato bo'ldi";
        return 1;
    }

    cout << "n="; cin >> n;

    for (int i = 0; i < n; i++)
    {
        cin >> k;
        fwrite(&k, sizeof(k), 1, f);
    }

    // fayl ko'rsatkichini fayl boshiga qo'yish
    rewind(f);

    while (fread(&k, sizeof(k), 1, f))
    {
  //fayl boshidan fayl ko'rsatkichi turgan o'ringacha bo'lgan baytlar
        int bayt = ftell(f);
        cout << k <<" ftell(f)=" << bayt << endl;
```

```
    }

    fclose(f);

    return 0;
}
```

3 - misol. n natural soni berilgan. Elementlari n ta butun sondan iborat bo`lgan faylni hosil qiluvchi va juft elementlarini 2 marta orttiruvchi programma tuzilsin.

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

int main()
{
    int n, k;
    FILE *f;
    // binar faylni yozish va o'qish uchun ochish
    f = fopen("binar", "wb+");

    if (f == NULL)
    {
        cout << "Faylni hosil qilishda xato bo'ldi";
        return 1;
    }

    cout << "n="; cin >> n;

    for (int i = 0; i < n; i++)
    {
        cin >> k;
        fwrite(&k, sizeof(k), 1, f);
    }

    // fayl ko'rsatkichini fayl boshiga qo'yish
    rewind(f);
    while (!feof(f)) // fayl oxiri uchramasa bajar
    {
        fread(&k, sizeof(k), 1, f);
        if (k % 2 == 0 )
        {
            k *= 2;
            // fayl ko'rsatkichini sizeof(int) bayt chapga surish
            fseek(f,  -sizeof(int), SEEK_CUR);
            fwrite(&k, sizeof(int), 1, f);
            // fayl ko'rsatkichini o'rnatish
            fseek(f, ftell(f), SEEK_SET);
        }
    }
    cout << "fayl elementlari\n";
    rewind(f);
```

```
    while (fread(&k, sizeof(k), 1, f))
    cout << k << endl;

    fclose(f);
    return 0;
}

3 - misolni quyidagicha yechish ham mumkin.
#include <iostream>
#include <stdio.h>
using namespace std;

int main()
{
    int n, k;
    FILE *f;

    f = fopen("binar", "wb+");
    cout << "n="; cin >> n;
    for (int i = 0; i < n; i++)
    {
        cin >> k;
        fwrite(&k, sizeof(k), 1, f);
    }
    // fayl ko'rsatkichini fayl boshiga qo'yish
    rewind(f);
    while (!feof(f))
    {
        // fayl ko'rsatkichi o'rnini eslab qolish
        int pos = ftell(f);
        fread(&k, sizeof(k), 1, f);
        if (k % 2 == 0 )
        {
            k *= 2;
            // fayl ko'rsatkichini oldingi xolatiga o'rnatish
            fseek(f, pos, SEEK_SET);
            fwrite(&k, sizeof(int), 1, f);
            // fayl ko'rsatkichi o'rnini sizeof(int) ga surish
            pos += sizeof(int);
            fseek(f, pos, SEEK_SET);
        }
    }
    cout << "fayl elementlari\n";
    rewind(f);
    // fayl elementlarini chiqarish
    while (fread(&k, sizeof(k), 1, f))
    cout << k << endl;

    fclose(f);
    return 0;
}
```

Nazorat savollari:

1. Fizik fayl deb qanday fayllarga aytiladi?
2. Mantiqiy fayl deb qanday fayllarga aytiladi?
3. Fizik va matiqiy fayllar qanday bog`laniladi?
4. Fayllar nima uchun va qanday ochiladi?
5. Fayl qanday yopiladi?
6. Fayl ko`rsatkichi deb nimaga aytiladi?
7. Fayllar necha xil xolatda bo'lishi mumkin?
8. Fayllar bilan ishlovchi funksiyalardan foydalanish uchun qaysi sarlavha faylini programmaga qo'shish kerak bo'ladi?
9. Qanday fayllar binar fayllar deyiladi?

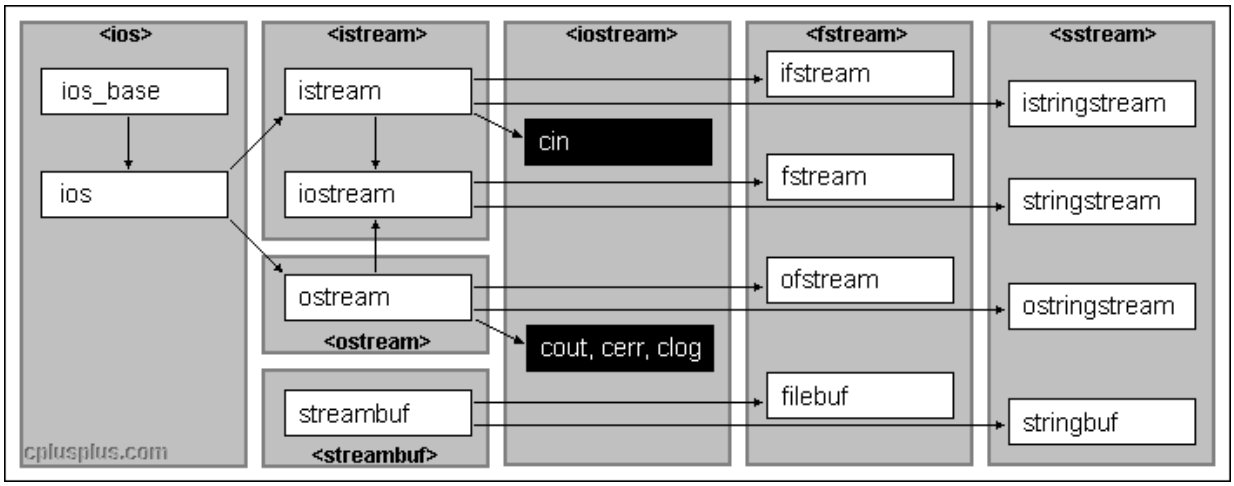
Fayllar bilan ishlash. Matnli fayllar

Matnli fayllar bilan ishlash binar fayllar bilan ishlashdan bir oz farq qiladi. Matnli fayllarda ma`lumotlar satrlarda saqlanadi. Matnli fayl elementilari har xil uzunlikdagi satrlardir. Bu satrlar bir biridan satr oxiri belgisi bilan ajratiladi. Matnli fayl elementlari indekslanmagan bo`lganligi uchun, faylning istalgan elementiga bevosita murojaat qilib bo`lmaydi.

C++ da matnli yoki binar fayllar bilan ishlash uchun keng imkoniyatlar berilgan. Matnli fayllar bilan ishlashda oddiy C ning funksiyalaridan ham foydalanish mumkin. Masalan, formatli o'qish va yozish funksiyalari yoki oldingi mavzudagi funksiyalardan foydalanishimiz mumkin. Matnli fayllar bilan ishlashning bunday usuli kitoblarda keng yoritilgan. Ularni mustaqil o'qib - o'rganishingiz mumkin.

Bu mavzu fayllar bilan ishlovchi oqimlarni qisqacha o'rganamiz va buni matnli fayl misolida ko'ramiz.

Standart kiritish / chiqarish kutubxonasi sinflari quyidagicha shajaraga ega:



Fayllar bilan ishlash uchun quyidagi sifnlar ob'ektlari hosil qilinadi:

- ofstream - faylga ma'lumot yozish uchun
- ifstream - fayldan ma'lumot o'qish uchun
- fstream - fayldan ma'lumot o'qish uchun va yozish uchun

Bu sinflarni dasturda ishlatish uchun <fstream> sarlavha faylini qo'shish kerak bo'ladi. Bundan keyin programmada aniq fayllar oqimini aniqlash mumkin. Masala:

ofstream yozish; *// faylga yozish oqimini e'lon qilish*
ifstream oqish; *// fayldan o'qish oqimini e'lon qilish*
fstream yoz_oqi; *// faylga yozish va o'qish oqimini e'lon qilish*

Keyin faylni ochish kerak bo'ladi. Faylni ochish deganda, uning ustida nima amal qilinishi haqida amaliyot tizimiga xabar berish tushuniladi.

void open (const char * filename, ios_base::openmode mode = ios_base::out);

mode parametri quyidagicha qiymatlarni qabul qilishi mumkin:

ios::in	faqat ma'lumot o'qish uchun
ios::out	faqat ma'lumot yozish uchun
ios::ate	faylni ochishda fayl ko'rsatkichini fayl oxiriga qo'yish
ios::app	fayl oxiriga ma'lumotlarni yozish uchun
ios::trunc	bor bo'lgan faylning ustidan yangi faylni yozish
ios::binary	binar holda ma'lumotlarni almashish uchun

Har bir sinf uchun mode parametrining odatiy qiymatlari mavjud:

class	**default mode parameter**
ofstream	ios::out
ifstream	ios::in
fstream	ios::in | ios::out

Fayl ustida o'qish yoki yozish amalini bajarib bo'lgandan song, faylni yopish kerak bo'ladi. Faylni yopish uchun close funksiyadi ishlatiladi. Masalan:

```
yozish.close();
oqish.close();
```

```
Matnli faylga ma'lumot yozish
#include <iostream>
```

```
#include <fstream>

using namespace std;

int main ()
{
    ofstream yozish; // faylga yozish oqimini hosil qilish

    yozish.open("namuna.txt");
    // yangi namuna.txt nomli fayl hosil qilinadi.
    // agar namuna.txt fayli oldindan bo'lsa,
    // uning eski qiymatlari o'chiriladi
    // va yangi fayl hosil qilinadi

    yozish << "Matnli faylga ma'lumot yozish" << endl;
    yozish << "Juda oson!" << endl;

    yozish.close(); // faylni yopish

    return 0;
}
```

Matnli fayldan o'qish

```
#include <iostream>
#include <fstream>
#include <string>

using namespace std;

int main ()
{
    ifstream oqish; // fayldan o'qish oqimini hosil qilish
    string satr;

    oqish.open("namuna.txt");

    // faylni ochishda xatolik sodir bo'lsa
    if (!oqish.is_open())
    {
        cout << "Faylni ochishda xatolik sodir bo'ldi." << endl;
        exit(1); // dasturni tugatish
    }

    while (!oqish.eof())
    {
        // fayldan o'qish
        getline(oqish, satr);
        // ekranga chiqarish
        cout << satr << endl;
    }

    // namuna.txt fayli bilan oqish oqimi aloqasini uzish
```

```
    oqish.close();

    return 0;
}
```

istream sinfi funksiyalari

istream& seekg (streampos pos);
istream& seekg (streamoff off, ios_base::seekdir dir);;

oqish oqimi ko'rsatkichini o'rnatish (siljitish).

pos - oqim buferining yangi pozitsiyasi.
dir parametri quyidagilardan birini qabul qilishi mumkin:

Qiymat	**Izoh**
ios::beg	oqimning boshlanishi
ios::cur	oqimning joriy xolari
ios::end	oqim oxiri

long tellg();

o'qish oqimining joriy xolatini (pozitsiyasi) aniqlash.

ostream sinfi funksiyalari

ostream& seekp (streampos pos);
ostream& seekp (streamoff off, ios_base::seekdir dir);

yozish oqimi o'rnini (pozitsiyasini) o'rnatish.
pos - oqim buferining yangi pozitsiyasi
dir parametri beg, cur, end qiymatlaridan birini qabul qilishi mumkin.

long tellp() - yozish oqimining kelgan joyini aniqlash.

Fayldan nusxa olish

```
//ushubu dastur orqali ixtiyoriy fayldan nusxa olish mumkin
#include <iostream>
#include <fstream>
using namespace std;

int main ()
{
    int length;
    char * buffer, fayl[] = "matn.txt", yangi[]="yangi_fayl.txt";
```

```
    // fayl - nusxalanadigan fayl nomi
    // yangi - yangi nusxalangan fayl nomi

    // o'qish oqimi
    ifstream fromfile(fayl, ios::binary );
    if (!fromfile.is_open())
    {
        cout << "faylni o'qishda xatolik sodir bo'ldi\n";
        exit(1);
    }

    // yozish oqimi
    ofstream tofile(yangi, ios::binary );

    // fayl xajmini aniqlash:
    fromfile.seekg (0, ios::end); // fayl oxiriga o'tish
    length = fromfile.tellg();
    fromfile.seekg (0, ios::beg); // fayl boshiga o'tish

    // xotira ajratish:
    buffer = new char [length];

    // blokka ma'lumotlarni o'qish:
    fromfile.read (buffer, length);

    fromfile.close();

    // nusxalanishi kerak bo'lgan faylga yozish
    tofile.write (buffer, length);

    // xotirani bo'shatish
    delete[] buffer;

    cout << "fayl nusxalandi\n";

    return 0;
}
```

dic.txt nomli fayl berilgan . Faylning har bir satrida inglizcha va o`zbekcha so`zlar "-" belgisi bilan ajratilgan. Inglizcha so`zlarni english.txt fayliga, o`zbekcha so`zlarni uzbek.txt fayliga o`tkazuvchi programma tuzilsin.

```
dic.txt fayli quyidagicha bo'ladi:
hello - salom
bread - non
car - mashina

#include <iostream>
#include <fstream>
#include <string>
```

```
using namespace std;

int main ()
{
    ifstream dic("dic.txt");
    ofstream uzbek("uzbek.txt");
    ofstream english("english.txt");

    if (!dic.is_open())
    {
        cout << "dic.txt - fayli topilmadi\n";
        exit(1);
    }

    string s, uzb, eng;
    int p;

    cout << "dic.txt fayli ma'lumotlari\n";
    while (!dic.eof())
    {
        getline(dic, s);
        p = s.find("-");
        eng.assign(s, 0, p - 1);
        uzb.assign(s, p + 1, s.length() - (p + 1));

        uzbek << uzb << endl;
        english << eng << endl;

        cout << s << endl;
    }

    dic.close();
    uzbek.close();
    english.close();

    return 0;
}
```

Nazorat savollari:

1. Fayllar bilan ishlovchi qaysi sinflarni bilasiz?
2. Matnli fayllarni toifali fayllardan qanday farqi bor?
3. Fayllarga oqimli yozish va o'qishda, oqimni ochish qanday bo'ladi?
4. istream sinfi funksiyalarni tushuntirib bering?
5. ostream sinfi funksiyalarni tushuntirib bering?

Ma`lumotlarning murakkab toifalari. Strukturalar

Ma`lumotlarning barcha toifalari oddiy bo`lsin, murakkab bo`lsin faqat bir toifadagi ma`lumotlarni saqlash uchun ishlatiladi. Masalan **Integer** toifasi faqat butun sonlarni saqlash uchun ishlatiladi. Massivlar va to`plamlarda esa elementlari qaysi toifada e`lon qilingan bo`lsa, faqat shu toifadagi ma`lumotlarni saqlaydi.

Amaliyotda esa axborotlarni saqlash, qayta ishlash uchun ma`lumotlarning har hil toifalarini aralashtirib ishlashga to`g`ri keladi. Ma`lumotlarning aralash toifasi bilan ishlash uchun C++ da **struct** (struktura) dan foydalanish mumkin.

Struktura deb, har hil toifadagi ma`lumotlarning cheklangan to`plamiga aytiladi. Strukturalar – maydon deb ataluvchi chekli sondagi hadlardan tashkil topadi. Struktura nimaligini tushinish uchun quyidagi misolni ko`rib chiqamiz.

№	Familiya Ism	Baholar
1	Abdullaev Dilshod	5 5 4 5
2	Abdurahimov Ne`mat	3 3 4 5
3	Rejepova Dilbar	5 5 5 5
4	Karimova Hafiza	4 4 5 5

Bu jadvalning har bir satri har hil toifadagi alohida elementlardan tuzilgan:

1. Tartib raqami – butun sonlardan
2. Familiya Ism – belgili satrlardan
3. Baholar – butun sonlar massividan tashkil topgan.

Bu ma`lumotlarni bir guruhga, strukturaga birlashtirish mumkin.
Quyidagicha belgilashlar kiritamiz: Imtixon – strukturaning nomi; N – tartib raqami; FI – Familiya ism; B – baholar;

Strukturalarni e`lon qilish:

Strukturalarni e`lon qilish uchun struct xizmatchi so'zidan foydalaniladi.

```
struct struktura_nomi
{
    toifa_1 nom1;
    toifa_2 nom2;
    . . .
    toifa_n nomn;
};
```

Struktura elementiga murojaat

Struktura orqali yangi ma'lumotlar toifasi hosil qilinadi. Programmada strukturalardan foydalanish shu toifadagi o’zgaruvchilar e’lon qilinish orqali bo'ladi.

Yuqoridagi misol uchun strukturani e`lon qilish quyidagicha bo`ladi:

```
struct imtixon
{
    int n;          // tartib raqami
    char FI[30];    // Familiya ismi
    int b[4];       // Baholar
};
```

Struktura elementlari (maydonlari) programmada oddiy o`zgaruvchilar kabi ishlatiladi. Struktura maydoni elementlari ustida, uning toifasida nima amal bajarish mumkin bo`lsa shu amallarni bajarish mumkin. Strukturaning mayddoniga murojaat qilish uchun, o'zgaruvchidan keyin nuqta (.) qo'yiladi.

```
imtixon t; // talaba

t.n = 1;
t.b[0] = 5;
t.b[1] = 5;
t.b[2] = 4;
t.b[3] = 5;
strcpy(t.FI, "Abdullaev Dilshod");
```

Butun va haqiqiy toifadagi o'zgaruvchilarga qiymat berish oddiy amalga oshiriladi. Satrlarga qiymat berish esa, strcpy - funksiyasi orqali amalga oshiriladi.

Keling oddiy ishchi nomli struktura (sinf) hosil qilamiz va undan programmada foydalanishni o'rganamiz.

Ishchi strukturasi quyidagi maydonlarni o'z ichiga oladi:

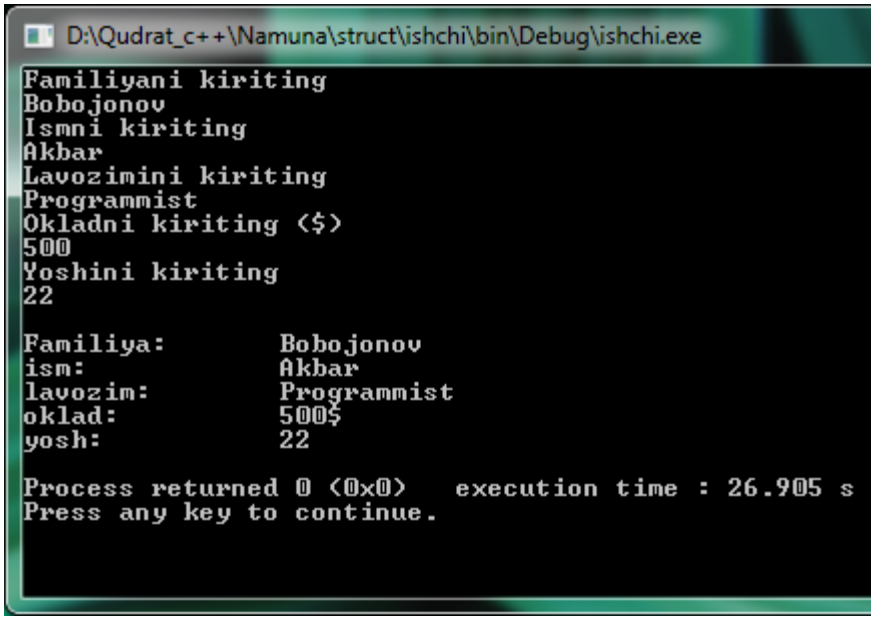
- Familiy
- Ism
- Lavozim
- Oklad
- Yosh

Strukturaga ma'lumotlarni kiritish va chiqarishda amallarni qayta yuklashdan foydalanamiz. Amallarni qayta yuklashni, C++ da ob'ektga yo'naltirilgan dasturlash qismida batafsil to'xtalamiz. Xozircha asosiy e'tiborni strukturaga qaratamiz.

Ishchi strukturasin hosil qilish va dasturda foydalanish

```
#include <iostream>
```

```
using namespace std;
// ishchi sinfini e'lon qilish
struct ishchi
{
    char familiya[30];
    char ism[30];
    char lavozim[30];
    float oklad;
    int yosh;
};
// kiritish amalini qayta yuklash
istream& operator >> (istream& input, ishchi& k)
{
    cout << "Familiyani kiriting\n";
    input >> k.familiya;

    cout << "Ismni kiriting\n";
    input >> k.ism;

    cout << "Lavozimini kiriting\n";
    input >> k.lavozim;

    cout << "Okladni kiriting ($)\n";
    input >> k.oklad;

    cout << "Yoshini kiriting\n";
    input >> k.yosh;

    return input;
}
// chiqarish amalini qayta yuklash
ostream& operator << (ostream& out, ishchi k)
{
    out << "\nFamiliya:\t" << k.familiya;
    out << "\nism:\t\t" << k.ism;
    out << "\nlavozim:\t" << k.lavozim;
    out << "\noklad:\t\t" << k.oklad << "$";
    out << "\nyosh:\t\t" << k.yosh << endl;

    return out;
}
int main()
{
    // ishchi sinfidagi ob'ektni e'lon qilish
    ishchi p;

    cin >> p;
    cout << p;

    return 0;
}
```



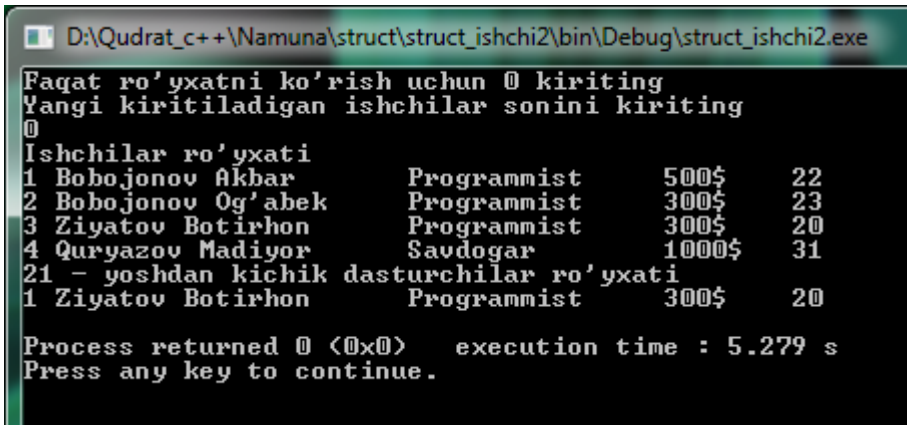
---

n ta ishchi haqidagi ma'lumotlarni o'zida saqlovchi fayl hosil qiling. 21 yoshdan kichik bo'lgan ishchilarni chiqaruvchi programma tuzilsin.

```
#include <iostream>

using namespace std;
// ishchi sinfini e'lon qilish
struct ishchi
{
    char familiya[30];
    char ism[30];
    char lavozim[30];
    float oklad;
    int yosh;
};
// kiritish amalini qayta yuklash
istream& operator >> (istream& input, ishchi& k)
{
    cout << "Familiyani kiriting\n";
    input >> k.familiya;

    cout << "Ismni kiriting\n";
    input >> k.ism;

    cout << "Lavozimini kiriting\n";
    input >> k.lavozim;

    cout << "Okladni kiriting ($)\n";
    input >> k.oklad;

    cout << "Yoshini kiriting\n";
```

```
    input >> k.yosh;

    return input;
}
// chiqarish amalini qayta yuklash
ostream& operator << (ostream& out, ishchi k)
{
    out << k.familiya;
    out << " " << k.ism;
    out << "\t" << k.lavozim;
    out << "\t" << k.oklad << "$";
    out << "\t" << k.yosh << endl;

    return out;
}
int main()
{
    // ishchi sinfidagi ob'ektni e'lon qilish
    ishchi p;
    FILE * f;
    int n = 0;

    f = fopen("ishchi.dat", "a+");

    cout << "Faqat ro'yxatni ko'rish uchun 0 kiriting\n";
    cout << "Yangi kiritiladigan ishchilar sonini kiriting\n"; cin >> n;

    if (n >= 1)
    for (int i = 1; i <= n; i++)
    {
        cout << i << " - ishchi ma'lumotlarini kiriting\n";
        cin >> p;

        // faylga ma'lumotlarni yozish
        fwrite(&p, sizeof(ishchi), 1, f);
    }
    // fayl boshiga o'tish
    rewind(f);

    cout << "Ishchilar ro'yxati\n";
    n = 0;
    while (fread(&p, sizeof(ishchi), 1, f))
    {
        n++;
        cout << n << " ";
        cout << p;
    }

    // fayl boshiga o'tish
    rewind(f);
    cout << "21 - yoshdan kichik dasturchilar ro'yxati\n";
    n = 0;
```

```
    while (fread(&p, sizeof(ishchi), 1, f))
    {
        if (p.yosh <= 21)
        {
            n++;
            cout << n << " ";
            cout << p;
        }
    }

    fclose(f);
    return 0;
}
```



Nazorat savollari:

1. Struktura nima?
2. Strukturalar qanday e`lon qilinadi?
3. Struktura maydonlari ustida qanday amallarni bajarish mumkin?
4. Strukturalar massivlardan qanday farq qiladi?