3-Amaliy mashg‘ulot.

Mavzu: C++ tilida takrorlanish operatorlari (while, do while, for).

For strukturasi sanovchi (counter) bilan bajariladigan takrorlashni bajaradi.Boshqa takrorlash bloklarida (while, do/while) takrorlash sonini control qilish uchun ham sanovchini qo'llasa bo'lardi, bu holda takrorlanish sonini o'ldindan bilsa bo'lardi, ham boshqa bir holatning vujudga kelish-kelmasligi orqali boshqarish mumkin edi. Ikkinchi holda ehtimol miqdori katta bo'ladi. Masalan qo'llanuvchi belgilangan sonni kiritmaguncha takrorlashni bajarish kerak bo'lsa biz while li ifodalar-ni ishlatamiz. for da esa sanovchi ifodaning qiymati oshirilib (kamaytirilib) borilvuradi, va chegaraviy qiymatni olganda takrorlanish tugatiladi. for ifodasidan keyingi bitta ifoda qaytariladi. Agar bir necha ifoda takrorlanishi kerak bo'lsa, ifodalar bloki {} qavs ichiga olinadi.

```
//Ekranda o'zgaruvching qiymatini yozuvchi dastur, for ni ishlatadi.
# include <iostream.h>
int main()
{
 for (int i = 0; i < 5; i++){
  cout << i << endl;
 }
 return (0);
}
```

Ekranda:

```
0
1
2
3
4
```

for strukturasi uch qismdan iboratdir. Ular nuqtavergul bilan bir-biridan ajratiladi. for ning ko'rinishi:

```
for( 1. qism ; 2. qism ; 3. qism ){
takror etiladigan blok
}
```

1. qism - e'lon va initsalizatsiya.
2. qism - shartni tekshirish (oz'garuvchini chegaraviy qiymat bilan solishtirish).
3.qism - o'zgaruvchining qiymatini o'zgartirish.

Qismlarning bajarilish ketma-ketligi quyidagichadir:

Boshida 1. qism bajariladi (faqat bir marta), keyin

2. qismdagi shart tekshiriladi va agar u true bo'lsa takrorlanish bloki ijro ko'radi, va eng ohirda 3. qismda o'zgaruvchilar o'zgartiriladi, keyin yana ikkinchi qismga

o'tiladi. for strukturamizni while struktura bilan almashtirib ko'raylik:

```
for (int i = 0; i < 10 ; i++)
  cout << "Hello!"<< endl;
```

Ekranga 10 marta Hello! so'zi bosib chiqariladi. I o'zgaruvchisi 0 dan 9 gacha o'zgaradi. i 10 ga teng bo'lganda esa i < 10 sharti noto'g'ri (false) bo'lib chiqadi va for strukturasi nihoyasiga yetadi. Buni while bilan yozsak:

```
int i = 0;

while ( i<10 ){
  cout << "Hello!" << endl;
  i++;
}
```

Endi for ni tashkil etuvchi uchta qismninig har birini alohida ko'rib chiqsak.Birinchi qismda asosan takrorlashni boshqaradigan sanovchi (counter)o'zgaruvchi-

lar e'lon qilinadi va ularga boshlangich qiymatlar beriladi (initsalizatsiya). Yuqoridagi dastur misolida buni int i = 0; deb berganmiz. Ushbu qismda bir necha o'zgaruvchilarni e'lon qilishimiz mumkin, ular vergul bilan ajratilinadi. Ayni shu kabi uchinchi qismda ham bir nechta o'zgaruvchilarning qiyma-tini o'zgartirishimiz mumkin. Undan tashqari birinchi qismda for dan oldin e'lon qilingan o'zgaruvchilarni qo'llasak bo'ladi.

Masalan:

```
int k = 10;
int l;
for (int m = 2, l = 0 ; k <= 30 ; k++, l++, ++m) {
cout << k + m + l;
}
```

Albatta bu ancha sun'iy misol, lekin u bizga for ifodasining naqadar moslashuvchanligi ko'rsatadi. for ning qismlari tushurib qoldirilishi mumkin.

Masalan:

```
for(;;) {}
```

ifodasi cheksiz marta qaytariladi. Bu for dan chiqish uchun break operatorini beramiz. Yoki agar sanovchi sonni takrorlanish bloki ichida o'zgartirsak,for ning 3. qismi kerak emas. Misol:

```
for(int g = 0; g < 10; ){
  cout << g;
  g++;
}
```

Yana qo'shimcha misollar beraylik.

```
 for (int y = 100; y >= 0; y-=5){
...
```

```
ifoda(lar);
...
}
```

Bu yerda 100 dan 0 gacha 5 lik qadam bilan tushiladi.

```
for(int d = -30; d<=30; d++){
...
ifoda(lar);
...
}
```

60 marta qaytariladi.

for strukrurasi bilan dasturlarimizda yanada yaqinroq tanishamiz. Endi

1. qismda e'lon qilinadigan o'zgaruvchilarning hususiyati haqida bir og'iz aytib o'taylik. Standartga ko'ra bu qismda e'lon qilingan o'zgaruvchilarning qo'l-

lanilish sohasi faqat o'sha for strukturasi bilan chegaralanadi. Yani bitta blokda joylashgan for struk-turalari mavjud bo'lsa, ular ayni ismli o'zgaruvchilarni qo'llana ololmaydilar. Masalan quyidagi hatodir:

```
for(int j = 0; j<20 ; j++){...}
...
for(int j = 1; j<10 ; j++){...} //hato!
```

j o'zgaruvchisi birinchi for da e'lon qilinib bo'lindi. Ikkinchi for da ishlatish mumkin emas. Bu masalani yechish uchun ikki hil yo'l tutish mumkin.

Birinchisi bitta blokda berilgan for larning har birida farqli o'zgaruvchilarni qo'llashdir. Ikkinchi yo'l for lar guruhidan oldin sanovchi vazifasini bajaruvchi bir o'zgaruvchini e'lon qilishdir. Va for larda bu o'zgaruv-

chiga faqat kerakli boshlangich qiymat beriladi halos.

for ning ko'rinishlaridan biri, bo'sh tanali for dir.

```
for(int i = 0 ; i < 1000 ; i++);
```

Buning yordamida biz dastur ishlashini sekinlashtirishimiz mumkin.

while operatori orqali murakkab konstruktsiyalarni tuzish. while operatori shartida murakkab mantiqiy ifodalarni ham qo‘llash mumkin. Bunday ifodalarni qo‘llashda && (mantiqiy ko‘paytirish), || (mantiqiy qo‘shish) , hamda !(mantiqiy INKOR) kabi operatsiyalardan foydalaniladi. Quyida while operatori konstruktsiyasida murakkabroq shartlarni quyilishiga misol keltirilgan .

while konstruktsiyasidagi murakkab shartlar.

```
#include <iostream>
using namespace std;
int main()
{
unsigned short kichik;
```

```
unsigned long katta;
const unsigned short  MaxKichik=65535;
cout << "Kichik sonni kiriting:";
 cin >> kichik;
 cout << "Katta sonni kiriting:";
 cin >> katta;
 cout << "kichik son:" << kichik << "…";
 //Xar bir iteratsiyada uchta shart tekshiriladi.
 while (kichik<katta && katta>0 &&
    kichik< MaxKichik )
 {
 if(kichik%5000==0) //Xar 5000 satrdan
  //keyin nuqta chikariladi
 cout<<"." ;
kichik++;
katta-=2 ;
}
cout<<"\n kichik son:"<<kichik<<" katta son:"
<<katta << endl ;
return 0 ;
}
```

Natija:

```
Kichik sonni kirit : 2
Katta sonni kirit : 100000
Kichik son : 2 ……
Kichik son :33335  katta son : 33334
```

TAHLIL

Dastur quyidagi mantiqiy o‘yinni ifodalaydi. Oldin ikkita son – kichik va katta kiritiladi. Undan so‘ng toki ular bir biriga teng bo‘lmaguncha, ya’ni «uchrashmaguncha» kichik son birga oshiriladi, kattasi esa ikkiga kamaytiriladi. O‘yinni maqsadi qiymatlar «uchrashadigan» sonni topishdir. Qiymatlar kiritilgandan so‘ng siklni davom ettirishning quyidagi uchta sharti tekshiriladi:

- kichik o‘zgaruvchisi qiymati katta o‘zgaruvchisi qiymatidan oshmasligi.
- katta o‘zgaruvchisi qiymati manfiy va nolga teng emasligi
- kichik o‘zgaruvchisi qiymati MaxKichik qiymatidan oshib ketmasligi

So‘ngra kichik soni 5000 ga bo‘lingandagi qoldiq hisoblanadi. Agarda kichik 5000 ga qoldiqsiz bo‘linsa bu operatsiyaning bajarilishi natijasi 0 ga teng bo‘ladi. Bu holatda hisoblash jarayonini vizual ifodasi sifatida ekranga nuqta chiqariladi. Keyin esa kichik qiymati bittaga oshiriladi, katta qiymati esa 2 taga kamaytiriladi. Sikl agarda tekshirish sharti tarkibidagi birorta shart bajarilmasa to‘xtatiladi.

do…while konstruktsiyasi yordamida sikl tashkil etish. Ayrim hollarda while operatori yordamida sikllarni tashkil etishda uning tanasidagi amallar umuman bajarilmasligi mumkin. Chunki siklni davom etish sharti har bir iteratsiyadan oldin tekshiriladi. Agarda boshlang‘ich berilgan shart to‘g‘ri bo‘lmasa sikl tanasining birorta operatori ham bajarilmaydi.

do...while konstruktsiyasining qo‘llanilishi

```
#include <iostream>
using namespace std;
int main()
{
int counter;
cout<<"How manu hellos ?";
cin >>counter;
do
{
cout << "hello \n";
counter--;
}
while(counter>0);
cout << "Counter is :" << counter <<endl;
return 0 ;
}
```

Natija:

```
how manu hellos ? 2
hello
hello
Sounter is : 0
How manu hellos ? 0
Hello
Counter is: - 1
```