17-AMALIY MASHG'ULOT. FUNKSIYALAR VA ULARNI E'LON QILISHGA OID DASTURLAR TUZISH

3. Funksiyani qayta yuklash

Ayrim algoritmlar berilganlarning har xil turdagi qiymatlari uchun qo'llanishi mumkin. Masalan, ikkita sonning maksimumini topish algoritmida bu sonlar butun yoki haqiqiy turda bo'lishi mumkin. Bunday hollarda bu algoritmlar amalga oshirilgan funksiyalar nomlari bir xil bo'lgani ma'qul. Bir nechta funksiyani bir xil nomlash, lekin har xil turdagi parametrlar bilan ishlatish *funksiyani qayta yuklash* deyiladi.

Kompilyator parametrlar turiga va soniga qarab mos funksiyani chaqiradi. Bunday amalni "*hal qilish amali*" deyiladi va uning maqsadi parametrlarga ko'ra aynan (nisbatan) to'g'ri keladigan funksiyani chaqirishdir. Agar bunday funksiya topilmasa kompilyator xatolik haqida xabar beradi. Funksiyani aniqlashda funksiya qaytaruvchi qiymat turining ahamiyati yo'q. Misol:

```
#include <iostream>
using namespace std;
int Max(int,int);
char Max(char,char);
float Max(float,float);
int Max(int,int,int);
int main()
{
   int a,b;
   char c,d;
   int k;
   float x,y;
   cin>>a>>b>>k>>c>>d>>x>>y;
   cout<<Max(a,b)<<endl;
   cout<<Max(c,d)<<endl;
   cout<<Max(a,b,k)<<endl;
   cout<<Max(x,y);
   return 0;
}

int Max(int i,int j)
{
   return (i>j)?i:j;
}
```

```
char Max(char s1,char s2)
{
   return (s1>s2)?s1:s2;
}
float Max(float x,float y)
{
   return (x>y)?x:y;
}
int Max(int i,int j,int k)
{
   if(i>j)
      if(i>k)
      return i;
      else
         return k;
      if(j>k)
         return j;
      else
         return k;
}
```

Agar funksiya chaqirilishida argument turi uning prototipidagi xuddi shu o‘rindagi parametr turiga mos kelmasa, kompilyator uni parametr turiga keltirilishga harakat qiladi - bool va char turlarini int turiga, float turini double turiga va int turini double turiga o‘tkazishga.

Qayta yuklanuvchi funksiyalardan foydalanishda quyidagi qoidalarga rioya qilish kerak:

1) qayta yuklanuvchi funksiyalar bitta ko‘rinish sohasida bo‘lishi kerak;

2) qayta yuklanuvchi funksiyalarda kelishuv bo‘yicha parametrlar ishlatilsa, bunday parametrlar barcha qayta yuklanuvchi funksiyalarda ham ishlatilishi va ular bir xil qiymatga ega bo‘lish kerak;

3) agar funksiyalar parametrlarining turi faqat «const» va ‘&’ belgilari bilan farq qiladigan bo‘lsa, bu funksiyalar qayta yuklanmaydi.

4. Kelishuv bo‘yicha argumentlar

C++ tilida funksiya chaqirilganda ayrim argumentlarni tushirib qoldirish mumkin. Bunga funksiya prototipida ushbu parametrlarni kelishuv bo‘yicha qiymatini ko‘rsatish orqali erishish mumkin. Masalan, quyida prototipi keltirilgan funksiya turli chaqirishga ega bo‘lishi mumkin:

```
//funksiya prototipi
void Butun_Son(int I,bool Bayroq=true,char Blg='\n');
 //funksiyani chaqirish variantlari
Butun_Son(1,false,'a');
Butun_Son(2,false);
Butun_Son(3);
```

Birinchi chaqiruvda barcha parametrlar mos argumentlar orqali qiymatlarini qabul qiladi, ikkinchi holda I parametri 2 qiymatini, bayroq parametri false qiymatini va Blg o‘zgaruvchisi kelishuv bo‘yicha ‘\n’ qiymatini qabul qiladi.

Kelishuv bo‘yicha qiymat berishning bitta sharti bor - parametrlar ro‘yxatida kelishuv bo‘yicha qiymat berilgan parametrlardan keyingi parametrlar ham kelishuv bo‘yicha qiymatga ega bo‘lishlari shart. Yuqoridagi misolda I parametri kelishuv bo‘yicha qiymat qabul qilingan holda, Bayroq yoki Blg parametrlari qiymatsiz bo‘lishi mumkin emas. Misol tariqasida berilgan sonni ko‘rsatilgan aniqlikda chop etuvchi programmani ko‘raylik. Qo‘yilgan masalani yechishda sonni darajaga oshirish funksiyasi - pow() va suzuvchi nuqtali uzun sondan modul olish fabsl() funksiyasidan foydal-niladi. Bu funksiyalar prototipi «math.h» sarlavha faylida joylashgan.

```
#include <iostream>
#include <math.h>
using namespace std;
void Chop_qilish(double Numb, double Aniqlik=1, bool Bayroq = true)
{
   if(!Bayroq)
      Numb=fabsl(Numb);
   Numb=(int)(Numb*pow(10,Aniqlik));
   Numb=Numb/pow(10,Aniqlik);
   cout<<Numb<<'\n';
}

int main()
{
```

```
    double Mpi=-3.141592654;
    Chop_qilish(Mpi,4,false);
    Chop_qilish(Mpi,2);
    Chop_qilish(Mpi);
    return 0;
}
```

Programmada sonni turli aniqlikda (Aniqlik parametri qiymati orqali) chop etish uchun har xil variantlarda Chop_qilish() funksiyasi chaqirilgan. Programma ishlashi natijasida ekranda quyidagi sonlar chop etiladi:

-3.1415
-3.14
-3.1

Parametrning kelishuv bo‘yicha beriladigan qiymati o‘zgarmas, global o‘zgaruvchi yoki qandaydir funksiya tomonidan qaytaradigan qiymat bo‘lishi mumkin.

5. Rekursiv funksiyalar

Rekursiya - bu nafaqat ilm-fan sohasida, balki kundalik hayotda ham uchraydigan juda keng tarqalgan hodisa.

Dasturlashda rekursiya funksiyalar bilan chambarchas bog‘liq, aniqrog‘i dasturlashdagi funksiyalar tufayli rekursiya yoki rekursiv funksiya kabi tushunchalar mavjud. Oddiy so‘zlar bilan aytganda, rekursiya - bu funksiya qismini o‘zi orqali belgilash, ya’ni o‘zini to‘g‘ridan-to‘g‘ri (tanasida) yoki bilvosita (boshqa funksiya orqali) chaqiradigan funksiya. Odatda rekursiv muammolarga sonning faktorialini topish, Fibonachchi raqamini topish va hokazolarni keltirish mumkin. Bu kabi masalalarni sikllar yordamida ham hal qilish mumkin. Umuman aytganda, iterativ ravishda yechilgan hamma narsani rekursiv, ya’ni rekursiv funksiya yordamida hal qilish mumkin.

C++ da **rekursiv** funksiya (yoki shunchaki "rekursiya") o‘zini o‘zi chaqiradigan funksiya.

Masalan:

```
#include <iostream>
using namespace std;
void countOut(int count1)
{
    cout << "push " << count1 << '\n';
    countOut(count1-1); //countOut () funksiyasi o‘zini rekursiv chaqiradi
```

```
}
int main()
{
   countOut(4);
   return 0;
}
```

countOut(4) ga murojaat qilish “push 4” yozuvini bosib chiqaradi va keyin countOut (3) ni chaqiradi. countOut (3) “push 3” ni bosib chiqaradi va countOut (2) ga murojaat qiladi.

Rekursiyani tugatish sharti. Rekursiv funksiya chaqiruvlari odatdagi funksiya murojaatlari singari ishlaydi. Shu bilan birga, yuqoridagi dastur oddiy funksiyalar va rekursivlar o‘rtasidagi eng muhim farqni aks ettiradi: siz rekursiyani tugatish shartini belgilashingiz kerak, aks holda funksiya cheksiz marta bajariladi.

Rekursiyani tugatish sharti - bu bajarilgandan so‘ng rekursiv funksiyani o‘zi chaqirishni to‘xtatadigan shart. Ushbu holat odatda if ifodasini ishlatadi.

Bu yerda yuqoridagi funksiyaga misol keltirilgan, ammo bu yerda rekursiya tugashi sharti ham mavjud:

```
#include <iostream>
using namespace std;
void countOut(int count1)
{
   cout << count1<< "-chiqish " << '\n';
   if (count1 > 1) // chiqish sharti
      countOut(count1-1);
   cout <<  count1<< "-kirish " << '\n';
}

int main()
{
   countOut(4);
   return 0;
}
```