14-Ma’ruza. Rekursiv funksiyalar

Ma’ruza rejasi:

14.1 Rekursiv jarayon nima

14.2 Rekursiv funksilari

14.3 Rekursiv funksiya parametrlari

Kalit so’zlar:*, ro’yxat, manzil, nolinchi ko’rchsatkich, tugun, adres olish &, bo’shatish, ko’rsatkich, virtual destruktor, xotira, xotira chiqishi, destruktor, toifani o’zlashtirish, resurslar chiqishi, a’zo destruktori.*

Joylashtiriladigan (inline) funksiyalar

Kompilyator ishlashi natijasida har bir funksiya mashina kodi ko‘rinishida bo‘ladi. Agar programmada funksiyani chaqirish ko‘rsatmasi bo‘lsa, shu joyda funksiyani adresi bo‘yicha chaqirishning mashina kodi shakllanadi. Odatda funksiyani chaqirish protsessor tomonidan qo‘shimcha vaqt va xotira resurslarini talab qiladi. SHu sababli, agar chaqiriladigan funksiya hajmi unchalik katta bo‘lmagan hollarda, kompilyatorga funksiyani chaqirish kodi o‘rniga funksiya tanasini o‘zini joylashtirishga ko‘rsatma berish mumkin. Bu ish funksiya prototipini inline kalit so‘zi bilan e’lon qilish orqali amalga oshiriladi. Natijada hajmi oshgan, lekin nisbatan tez bajariladigan programma kodi yuzaga keladi.

Funksiya kodi joylashtiriladigan programmaga misol.

```
#include <iostream.h>
inline int Summa(int,int);
int main()
{
 int a=2,b=6,c=3;
 char yangi_qator='\n';
 cout<<Summa(a,b)<<yangi_qator;
 cout<<Summa(a,c)<<yangi_qator;
 cout<<Summa(b,c)<<yangi_qator;
 return 0;
}
int Summa(int x,int y)
{
 return x+y;
}
```

Keltirilgan programma kodini hosil qilishda Summa() funksiyasi chaqirilgan joylarga uning tanasidagi buyruqlar joylashtiriladi.

Rekursiv funksiyalar

YUqorida qayd qilingandek *rekursiya* deb funksiya tanasida shu funksiyaning o‘zini chaqirishiga aytiladi. Rekursiya ikki xil bo‘ladi:

1) *oddiy* - agar funksiya o‘z tanasida o‘zini chaqirsa;

vositali - agar birinchi funksiya ikkinchi funksiyani chaqirsa, ikkinchisi esa o‘z navbatida birinchi funksiyani chaqirsa.

Odatda rekursiya matematikada keng qo‘llaniladi. CHunki aksariyat matematik formulalar rekursiv aniqlanadi. Misol tariqasida faktorialni hisoblash formulasini

$$n!=\begin{cases}1, & \text{arap } n=0;\\ n*(n-1)!, & \text{arap } n>0,\end{cases}$$

va sonning butun darajasini hisoblashni ko‘rishimiz mumkin:

$$x^n=\begin{cases}1, & \text{arap } n=0;\\ x*x^{n-1}, & \text{arap } n>0.\end{cases}$$

Ko‘rinib turibdiki, navbatdagi qiymatni hisoblash uchun funksiyaning «oldingi qiymati» ma’lum bo‘lishi kerak. S++ tilida rekursiya matematikadagi rekursiyaga o‘xshash. Buni yuqoridagi misollar uchun tuzilgan funksiyalarda ko‘rish mumkin. Faktorial uchun:

```
long F(int n)
{
 if(!n) return 1;
 else return n*F(n-1);
}
```

Berilgan haqiqiy x soning n- darajasini hisoblash funksiyasi:

```
double Butun_Daraja(double x, int n)
{
 if(!n) return 1;
 else return x*Butun_Daraja(x,n-1);
}
```

Agar faktorial funksiyasiga n>0 qiymat berilsa, quyidagi holat ro‘y beradi: shart operatorining else shoxidagi qiymati (n qiymati) stekda eslab qolinadi. Hozircha qiymati noma’lum n-1 faktorialni hisoblash uchun shu funksiyaning o‘zi n-1 qiymati bilan bilan chaqiriladi. O‘z navbatida, bu qiymat ham eslab qolinadi (stekka joylanadi) va yana funksiya chaqiriladi va hakoza. Funksiya n=0 qiymat bilan chaqirilganda if operatorining sharti (!n) rost bo‘ladi va «return 1;» amali bajarilib, ayni shu chaqirish bo‘yicha 1 qiymati qaytariladi. SHundan keyin «teskari» jarayon boshlanadi - stekda saqlangan qiymatlar ketma-ket olinadi va ko‘paytiriladi: oxirgi qiymat - aniqlangandan keyin (1), u undan oldingi saqlangan qiymatga 1 qiymatiga ko‘paytirib F(1) qiymati hisoblanadi, bu qiymat 2 qiymatiga ko‘paytirish bilan F(2)

topiladi va hakoza. Jarayon F(n) qiymatini hisoblashgacha «ko‘tarilib» boradi. Bu jarayonni, n=4 uchun faktorial hisoblash sxemasini 5.2-rasmda ko‘rish mumkin:

↓	F(4)=4*F(3)	↓	F(4)=4*F(3)	↓	F(4)=4*F(3)	↓	F(4)=4*F(3)	↑	F(4)=4*6
↓	F(3)=3*F(2)	↓	F(3)=3*F(2)	↓	F(3)=3*F(2)	↑	F(3)=3*2		
↓	F(2)=2*F(1)	↓	F(2)=2*F(1)	↑	F(2)=2*1				
↓	F(1)=1*F(0)	↑	F(1)=1*1						
↑	F(0)=1								

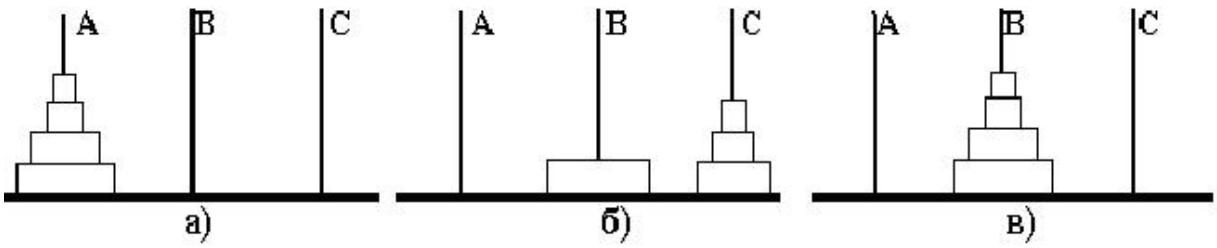
14.1-rasm. 4! hisoblash sxemasi

Rekursiv funksiyalarni to‘g‘ri amal qilishi uchun rekursiv chaqirishlarning to‘xtash sharti bo‘lishi kerak. Aks holda rekursiya to‘xtamasligi va o‘z navbatida funksiya ishi tugamasligi mumkin. Faktorial hisoblashida rekursiv tushishlarning to‘xtash sharti funksiya parametri n=0 bo‘lishidir (shart operatorining rost shoxi).

Har bir rekursiv murojaat qo‘shimcha xotira talab qiladi - funksiyalarning lokal ob’ektlari (o‘zgaruvchilari) uchun har bir murojaatda stekdan yangidan joy ajratiladi. Masalan, rekursiv funksiyaga 100 marta murojaat bo‘lsa, jami 100 lokal ob’ektlarning majmuasi uchun joy ajratiladi. Ayrim hollarda, ya’ni rekursiyalar soni etarlicha katta bo‘lganda, stek o‘lchami cheklanganligi sababli (real rejimda 64Kb o‘lchamgacha) u to‘lib ketishi mumkin. Bu holatda programma o‘z ishini «Stek to‘lib ketdi» xabari bilan to‘xtadi.

Quyida, rekursiya bilan samarali echiladigan «Xanoy minorasi» masalasini ko‘raylik.

Masala. Uchta A, B, C qoziq va n-ta har xil o‘lchamli xalqalar mavjud. Xalqalarni o‘lchamlari o‘sish tartibida 1 dan n gacha tartib-langan. Boshda barcha xalqalar A qoziqqa 5.3a - rasmdagidek joylash-tirilgan. A qoziqdagi barcha xalqalarni B qoziqqa, yordamchi S qoziqdan foydalangan holda, quyidagi qoidalarga amal qilgan holda o‘tkazish talab etiladi: xalqalarni bittadan ko‘chirish kerak va katta o‘lchamli xalqani kichik o‘lchamli xalqa ustiga qo‘yish mumkin emas.



14.2-rasm. Xanoy minorasi masalasini echish jarayoni

Amallar ketma-ketligini chop etadigan («Xalqa q dan r ga o‘tkazilsin» ko‘rinishida, bunda q va r - 5.3-rasmdagi A,V yoki S xalqalar). Berilgan n ta xalqa uchun masala echilsin.

Ko‘rsatma: xalqalarni A dan B ga to‘g‘ri o‘tkazishda 5.3b –rasmlar-dagi holat yuzaga keladi, ya’ni n xalqani A dan B o‘tkazish masalasi n-1 xalqani A dan S ga o‘tkazish, hamda bitta xalqani A dan B o‘tkazish masalasiga keladi. Undan keyin S qoziqdagi n-1 xalqali A qoziq yordamida B qoziqqa o‘tkazish masalasi yuzaga keladi va hakoza.

```
#include <iostream.h>
void Hanoy(int n,char a='A',char b='B',char c='C')
{
 if(n)
 {
  Hanoy(n-1,a,s,b);
  cout<<”Xalqa”<< a<<” dan ”<<b<<” ga o’tkazilsin\n”;
  Hanoy(n-1,c,b,a);
 }
}
int main()
{unsigned int Xalqalar_Soni;
 cout<<”Hanoy minorasi masalasi”<<endl;
 cout<<”Xalqalar sonini kiriting: ”;
 cin>>Xalqalar_Soni;
 Hanoy(Xalqalar_Soni);
 return 0;
}
```

Xalqalar soni 3 bo‘lganda (Xalqalar_Soni=3) programma ekranga xalqalarni ko‘chirish bo‘yicha amallar ketma-ketligini chop etadi:

```
Xalqa A dan B ga o’tkazilsin
Xalqa A dan C ga o’tkazilsin
Xalqa B dan C ga o’tkazilsin
Xalqa A dan B ga o’tkazilsin
Xalqa C dan A ga o’tkazilsin
Xalqa C dan B ga o’tkazilsin
Xalqa A dan B ga o’tkazilsin
```

Rekursiya chiroyli, ixcham ko‘ringani bilan xotirani tejash va hisoblash vaqtini qisqartirish nuqtai-nazaridan uni imkon qadar iterativ hisoblash bilan almashtirilgani ma’qul. Masalan, x haqi-qiy sonining n-darajasini hisoblashning quyidagi echim varianti nisbatan kam resurs talab qiladi (n- butun ishorasiz son):

```
double Butun_Daraja(double x, int n)
{
 double p=1;
 for(int i=1; i<=n; i++)p*=x;
 return p;
}
```

Ikkinchi tomondan, shunday masalalar borki, ularni echishda rekursiya juda samarali, hattoki yagona usuldir. Xususan, grammatik tahlil masalalarida rekursiya juda ham o‘ng‘ay hisoblandi.

ch05/account.cpp

```
1 #include <iostream>
2
3 using namespace std;
4 5 /**
6  Withdraws the amount from the
given balance, or withdraws
7  a penalty if the balance is
insufficient.
8 @param balance the balance from
which to make the withdrawal 9 @param
amount the amount to withdraw
10 */
11 void withdraw(double& balance,
double amount)
12 {
13 const double PENALTY = 10;
14 if (balance >= amount)
15 {
16  balance = balance - amount;
17 }
18 else
19 {
20  balance = balance - PENALTY;
21 }
22 }
23
24 int main()
25 {
26 double harrys_account = 1000;
27 double sallys_account = 500;
28 withdraw(harrys_account, 100);
29  // Now harrys_account is 900
30 withdraw(harrys_account, 1000);
// Insufficient funds
31  // Now harrys_account is 890
32 withdraw(sallys_account, 150);
33 cout << "Harry's account: " <<
harrys_account << endl; 34 cout << "Sally's
account: " << sallys_account << endl; 35
36 return 0;
37 }
```

program run

```
Harry's account: 890 Sally's account: 350
```

Nazorat savollari

1. C++da funksiya qanday ishlaydi?
2. funksiyaga kutubxona kerakmi?
3. Rekursiv funksiya nima?
4. Rekursiv funksiya parametrlari.
5. For operatori funksiyada qanday ishlatiladi?
6. Matematik funksiyalar qanday ishlaydi?
7. Funksiya parametrlar nima?
8. Funksiya qanday chaqiriladi?
9. Funksiya parametrlari orqali nima uzatiladi?
10. If operatorining nechta turi bor?