**5-Tajriba ishi**

**Mavzu:** Funksiyalarni tashkil etish.

**Ishdan maqsad.** Funksiya tushunchasi, C++ dafunksiya qanday yaratilishini, qanday qismlardan iborat bo’lishini va unga qanday murojaat qilinishini o’rganish.

**Nazariy qism.**

Dasturlashda ko’pincha bir xil amalni bir necha marta bajarishga to’g’ri keladi.

**Masala 1**: Radiyuslari r1 va r2 ga teng bo’lgan sharlar berilgan. Ularning har birining hajmini toping toping.

Shar hajmi formulasi: V=

Bu masalani avvaldan ma’lum usulda yechadigan bo’lsak:

#include <iostream>

#include <math.h>

using namespace std;

int main() {

double r1, r2;

cin>>r1>>r2;

double v1 = 4 / 3.0 \* M\_PI \* r1 \* r1 \* r1;

double v2 = 4 / 3.0 \* M\_PI \* r2 \* r2 \* r2;

cout<<"v1="<<v1<<endl<<"v2="<<v2;

}

Bunda bir xil amallar ikki marta yozilgan. Ularda faqat o’zgaruvchi farq qiladi.

Bir xil amallarni birnecha marta yozmaslik uchun funksiyadan foydalanamiz.

**Funksiya** – dastur kodining biror nomlangan qismi bo’lib, unga boshqa funksiyalardan va asosiy (main) funksiyasidan murojaat qilish mumkin. Aniq bir vazifani bajaradi, birnecha marta murojaat qilinishi mumkin.

Funksiyaning umumiy ko’rinishi:

<funksiya qaytaradigan tipi> <funksiya nomi>(<parametrlar tipladi va identifikatorlari ro’yxati>) {

<funksiya tanasi>

}

Yuqoridagi masalani har qanday shar radiyusi berilganda uning hajmini hisoblan qaytarib beradigan funksiya yaratish orqali yechamiz:

#include <iostream>

#include <math.h>

using namespace std;

double volume\_cube(double r) {

return 4 / 3.0 \* M\_PI \* r \* r \* r;

}

int main() {

double r1, r2;

cin>>r1>>r2;

double v1 = volume\_cube(r1);

double v2 = volume\_cube(r2);

cout<<"v1="<<v1<<endl<<"v2="<<v2;

}

**volume\_cube –** funksiyanomi**.** Funksiya nomi istalgan identifikator bo’lishi mumkin. Identifikator kata va kichik lotin alfaviti harflaridan, raqamlardan va \_ belgisidan iborat bo’ladi va raqam bilan boshlanmaydi.

**double** **–** funksiya qaytaradigan tip. Qaytaradigan tip har xil bo’lishi mumkin**.**

Funksiya tanasi **return** so’zi bilan tugaydi**. double** r funksiyaning parametrik o’zgaruvchisi.

Funksiyaga parametrlari soni 1 ta, ko’p yoki umuman bo’lmasligi mumkin. Agar funksiya parametrlari yo’q bo’lsa u holda qavs ichida hech narsa yozilmaydi. Masalan **int** **main**() funksiyasi parametrga ega emas.

Agar funksiya parametrlari soni 1 dan ko’p bo’lsa u holda ular bir-biridan vergul bilan ajratiladi. Har bir parametrning tipi har xil bo’lishi mumkin.

**Misol3.**

Eng oddiy misol sifatida ikkita butun sonning yig’indisini hisoblab qaytarib beradigan funksiyani yaratamiz. Funksiya nomini summa deb nomlaymiz. Unga parameter sifatida ikkita butun son keladi. Natija ham butun son bo’ladi:

#include <iostream>

using namespace std;

int summa(int a, int b) {

int c = a+b;

return c;

}

int main() {

int x, y;

cin>>x>>y;

int z = summa(x, y);

cout<<z;

}

Huddi shu masalani qyidagicha ham yozishimiz mumkin:

#include <iostream>

using namespace std;

int summa(int a, int b) {

return a+b;

}

int main() {

int x, y;

cin>>x>>y;

cout<<summa(x, y);

}

**Misol4.**

Ikkita sonning eng katta bo’luvchisini topadigan funksiyani tuzamiz. Eng katta umumiy bo’luvchi(EKUB) bu ikkala sonni qoldiqsiz bo’ladigan sonlar ichida eng kattasi. Demak funksiya parametrlari 2 ta butun son. Funksiya qaytardigan natija bitta butun son – ikki sonning EKUBi.

long long gcd(long long a, long long b) {

while (b != 0) {

long long c = a % b;

a = b;

b = c;

}

return a;

}

int main() {

long long a,b;

cin>>a>>b;

long long g = gcd(a, b);

cout<<g;

}

Funksiya qiymat qaytaradigan va qaytarmaydigan turlarga bo’linadi. Qiymat qaytarmaydigan funksiya tipi sifatida **void** so’zi yoziladi. Bunday funksiya faqat hisob kitoblarni amalga oshiradi yoki natijalarni chiqaradi.

Bir o’lchamli massivning [L..R] elementlarini chiqarishni funksiya yordamida amalga oshiruvchi dastur tuzing.

Bu masalada funksiyaga parametr sifatida massiv va uning boshlang’ich va ohirgi insekslari uzatiladi.

void print\_array(int a[], int L, int R) {

for (int i = L; i <= R; i++) {

cout<<a[i]<<" ";

}

cout<<endl;

}

int main(){

int a[] = {1, 4, 6, 14, 32, 7, 9};

print\_array(a, 0, 3);

int b[] = {4, 67, 13, 56, 61, 4};

print\_array(b, 2, 3);

}

Agar o’zgaruvchi funksiyadan oldin tavsiflangan bo’lsa u holda bu o’zgaruvchini funksiya tanasida foydalanishimiz mumkin. Yuqoridagi masalada chiqarish faqat *a* massiv ustida amalga oshiriladigan bo’lsa u holda quyidagicha yozishimiz mumkin.

#include <iostream>

using namespace std;

int a[1001];

void print\_array(int L, int R) {

for (int i = L; i <= R; i++) {

cout<<a[i]<<" ";

}

cout<<endl;

}

int main(){

int n;

cin>>n;

for (int i = 0; i < n; i++) {

cin>>a[i];

}

print\_array(0, 3);

print\_array(2, 3);

}

**Funksiya prototipi**

C++ da funksiya unga murojaat qilingan funksiyadan oldin aniqlangan bo’lishi kerak. Agar kub hajmini topishdagi **volume\_cube** funksiyasi main funksiyasidan dan keyin yozilgan bo’lsa hatolik bo’ladi.

Funksiya prototipi deb funksiyaning tanasini yozmasdan uni e’lon qilishga aytiladi. Prototip funksiya qaytaradigan tipi, nomi va parametrlar ro’yxatidan iborat bo’ladi. Funksiya tanasini undan so’ng yozishimiz mumkin.

#include <iostream>

#include <math.h>

using namespace std;

double volume\_cube(double r); //funksiya prototipi

int main() {

double r1, r2;

cin>>r1>>r2;

double v1 = volume\_cube(r1);

double v2 = volume\_cube(r2);

cout<<"v1="<<v1<<endl<<"v2="<<v2;

}

double volume\_cube(double r) {

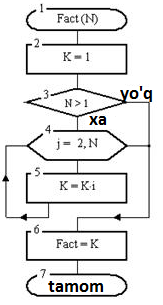
return 4 / 3.0 \* M\_PI \* r \* r \* r;

}

Funksiya prototipi nuqtali vergul bilan yakunlanadi.

**Blok sxema**

n! ni hisoblaydigan funksiya tuzamiz. *n*!=1•2•3•…•*n*;



Asosiy dasturda

**K=Fact(n)**

orqali qism dasturga(fumsiyaga) murojaat qilamiz.

**Misol 1:**

F(x, y) =

Z=F(a, b)+F(a/b, F(a2, b2))

a va b sonlari berilgan.

Z ning qiymatini toppish kerak.

F(x, y)

Blok sxemasi:

x+y<0

R=

R=

Tamom

F=R

Tamom

Z

Z=F(a,b)+F(a/b,F(a2+b2))

Boshlash

a,b

#include <iostream>

#include <math.h>

#include <stdio.h>

using namespace std;

double F(double x, double y) {

double R;

if (x+y < 0)

R = x\*x + y\*y;

else

R = sqrt(x+y);

return R;

}

int main() {

double a, b;

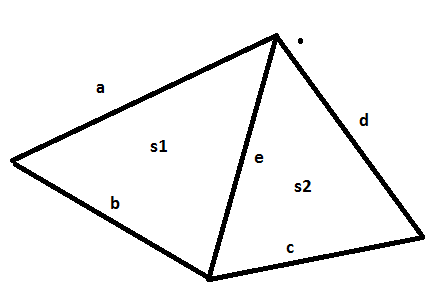
cin>>a>>b;

double Z = F(a, b)+F(a/b, F(a\*a, b\*b));

printf("%.20f", Z);

}

**Misol2**

****

To’rtburchakning tamonlari va bitta diagonali berilgan. Uning yuzasini topuvchi dastur tuzing.

Yechimi:

To’rtburchak yuzasi diagonal ajratgan tamonlari (a, b, e) va (c, d, e) bo’lgan ikkita uchburchaklar yuzalari yi’gindisiga teng.

Uchburchak yuzasini ikki marta hisoblashga lozim. Ikki marta bir xil amalni yozmaslik uchun uchburchak yuzasini topadigan funksiya tuzamiz va unga ikki marta murojaaat qilamiz.

#include <iostream>

#include <math.h>

#include <stdio.h>

using namespace std;

double surface\_traingle(int a, int b, int c) {

double p = (a+b+c) / 2.0;

double s = sqrt(p\*(p-a)\*(p-b)\*(p-c));

return s;

}

int main() {

int a, b, c, d, e;

cin>>a>>b>>c>>d>>e;

double s = surface\_traingle(a, b, e) + surface\_traingle(c, d, e);

printf("%.10f", s);

}

**Rekursiv funksiyalar**

Funksiya tanasida o‘zini o‘zi chaqirsa **rekursiya** deyiladi. Rekursiya ikki xil bo‘ladi:

* **Oddiy** – agar funksiya o‘z tanasida o‘zini chaqirsa;
* **Vositali** – agar birinchi funksiya ikkinchi funksiyani chaqirsa, ikkinchisi esa o‘z navbatida birinchi funksiyani chaqirsa.

**Misol 1**: Masala sifatida faktorialni rekursiv funksiya yordamida hisoblashni olamiz. Bunda n faktarialni hisoblashda oldingi n-1 faktarialni, n-1 faktarialni hisoblashda esa undan oldinikini hisoblashimiz zarur. Faktarialni hisoblaydigan funksiya yaratamiz. U o‘zining ichida oldingilarini chaqiradi.

Masalaning matematik ifodasi:

|  |
| --- |
|  |

Masala yechimining dasturi

#include <bits/stdc++.h>

using namespace std;

long Faktorial(int n)

{

if (!n) return 1;

else return n \* Faktorial (n – 1);

}

int main()

{

int n;

cin>>n;

cout<<Faktorial(n);

}

**Misol 2.** Xuddi faktorialni hisoblashdagi kabi darajani xisoblash funksiyasini ham misol keltirishimiz mumkin.

Masalaning matematik ifodasi:

|  |
| --- |
|  |

Masala yechimining dasturi:

#include <bits/stdc++.h>

using namespace std;

double Daraja(double x, int n)

{

if (!n) return 1;

else return x \* Daraja(x, n - 1);

}

int main()

{

int n;

double x;

cin>>x>>n;

cout<<Daraja(x, n);

}

Darajani xisoblashning yana bir usuli bu ikkilik daraja. Bunda ham rekursiv funksiyadan foydalanib darajani hisoblaymiz.

#include <bits/stdc++.h>

using namespace std;

double Daraja(double x, int n)

{

if (!n) return 1;

double a = Daraja(x, n/2);

return n%2==0 ? a\*a:a\*a\*x;

}

int main()

{

int n;

double x;

cin>>x>>n;

cout<<Daraja(x, n);

}

**Namuna**. Rekursiv funksiyadan foydalangan holda ikkita sondan raqamlari yig‘indisi katta bo‘lgan sonni topuvchi dastur tuzing.

#include <bits/stdc++.h>

using namespace std;

int sum, sum\_1, sum\_2 ;

int raqam(int son)

{

sum += son % 10;

son = son / 10;

if (son == 0) return sum;

raqam (son);

}

int main()

{

int sum\_1 = 0, sum\_2 = 0;

int son\_1, son\_2;

cin>>son\_1>>son\_2;

sum\_1 = raqam(son\_1);

sum\_2 = raqam(son\_2);

if (sum\_1 > sum\_2) cout << son\_1;

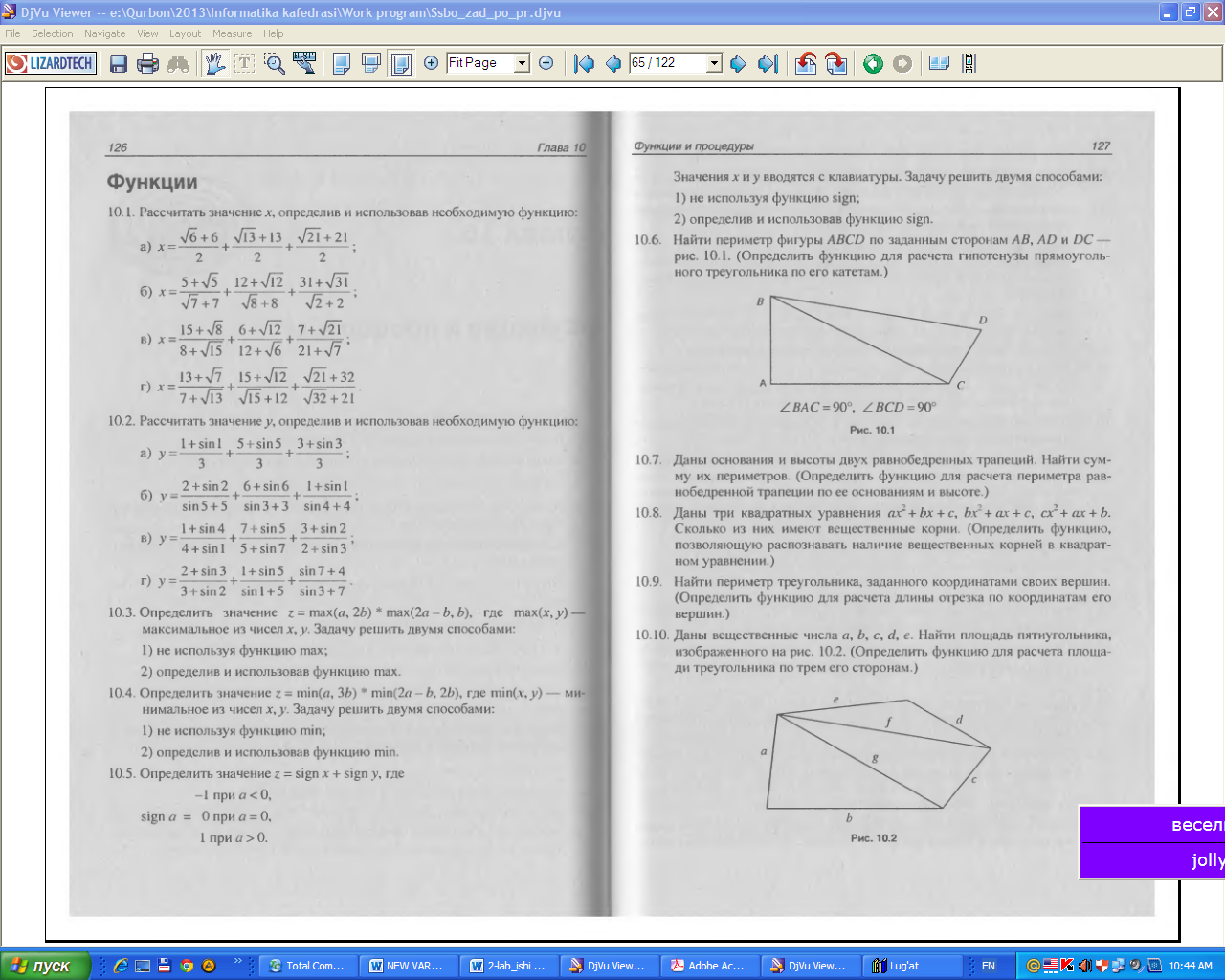
else cout<<son\_2;

}

**Topshiriqlar**

**1-topshiriq**

1. Q sonini P darajasini topuvchi funksiya tuzing.
2. P sonini oxiridan L sonini qo‘shuvchi funksiya tuzing.
3. Berilgan to‘g‘ri burchakli uchburchakning katetlari yordamida gipotenuzasini topuvchi funksiya tuzing
4. Berilgan ikki nuqtaning koordinatasi asosida ular orasidagi masofani topuvchi funksiya tuzing.
5. Berilgan sondagi qo‘shni raqamlarining raqamlarining yig‘indisiga teng bo‘lgan raqamni o‘chirivchi funksiya tuzing.
6. Berilgan N soni ikkita tub sonni yig‘indisi bo‘lishini tekshiruvchi funksiya tuzing.
7. Hech bir uchtasi bir to’g’ri chiziqda yotmaydigan funksiya tuzing.
8. Berilgan son 11 ga bo‘linsa 1 aks holda o chiqaruvchi funksiya tuzing.
9. Berilgan sonni K o‘rindagi raqamini N o‘rindagi raqami bilan almashtiruvchi funksiya tuzing.
10. Sonni raqamlarini o‘sish (kamayish) tartibida saralovchi funksiya tuzing.
11. Uchburchakni uchta uchining koordinatalari berilgan. Uning yuzasini topuvchi funksiya tuzing.
12. Tomonlari a, b, c, d va e haqiqiy sonlardan iborat bo‘lgan beshburchak berilgan (2.1-rasm). Beshburchakni va undagi uchta uchburchaklarni yuzasini topuvchi alohida funksiyalar tuzing.

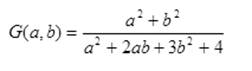


2.1-rasm.

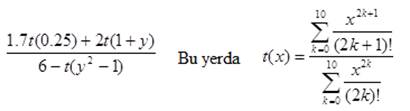
1. Besh burchakning uchlarining koordinatalari x1, y1, x2, y2, …, x5, y5 berilgan (2.1-rasm). Beshburchakdagi uchburchakni uchta uchining koordinatalari berilgan. Uning yuzasini topuvchi funksiya tuzing.
2. Sonni tublikka tekshiruvchi funksiya tuzing va barcha uch xonali tub sonlarni chiqaring.
3. Barcha tub sonlar ichidan “egizak”larini topuvchi funksiya tuzing. Tub sonlar “egizak” deyiladi, agarda ular 3 ga farq qilsa. Masalan, 41 va 43. Uch xonali sonlar ichidan barcha “egizak”larni chiqaring.
4. Ikkita son berilgan. Ularni raqamlari yig‘indisi kattasini toping. Sonni raqamlari yig‘indisini topuvchi funksiya tuzing.
5. Ikkita son berilgan. Ularning raqamlaridan soni ko‘pini toping. Sonni raqamlari sonini topuvchi funksiya tuzing.
6. 6 xonali barcha baxtli sonni toping. Agarda 6 xonali sonni dastlabki uchta raqamini yig‘indisi oxirgi uchta raqamini yig‘indisiga teng bo‘lsa baxtli son deyiladi. 6 xonali sonni raqamlarini uchtalab yig‘indisini topuvchi funksiya tuzing.
7. Berilgan sonni palindromlikka tekshiruvchi funksiya tuzing. Masalan, 1221. Son palindrom bo‘lsa “Palindrom”, aks holda “Palindrom emas” so‘zini chiqaring.
8. Berilgan a va b sonlarini EKUBini topuvchi funksiya tuzing.
9. Berilgan a va b sonlarini EKUKini topuvchi funksiya tuzing.
10. Berilgan a, b va c sonlarini EKUBini topuvchi funksiya tuzing.

|  |
| --- |
| S, t хaqiqiy sоnlar bеrilgan.    f(t, -2s, 1.17)+f(2.2, t, s-t) ni хisоblang,  bu еrda |
| Безымянный |

1. S, t хakikiy sоnlar bеrilgan. G(1.2,s)+G(t,s)+G(2s-1,st) ni хisоblang, bu еrda

****

1. Xakikiy sоn bеrilgan. Хisоblang



1. a , b va c хakikiy sоn bеrilgan. Хisоblang

http://algo.ubtuit.uz/168.files/image002.gif

1. a va b хakikiy sоn bеrilgan. Хisоblang

               u=min(a,b)  v=min(ab, max(a,b))  s=min(u+v, 3.14).

1. . s, t хaqiqiy sоnlar bеrilgan.  Хisоblang:

h(s,t)+max(h(s-t,st), h(s-t,s+t))+h(1 , 1), bu еrda

http://algo.ubtuit.uz/170.files/image002.jpg

1. f(n) funksiyasi quyidagicha xisoblanadi:

f(0)=0, f(1)=1, f(2n)=f(n), f(2n+1)=f(n)+f(n+1).

Sizga natural n soni beriladi. Sizdan topish talab qilinadi f(n)

**Rekursiv funksiyaga oid topshiriqlar**

**2-topshiriq**

1. Sonni raqamlar yig‘indisini topuvchi rekursiv funksiya tuzing.
2. Sonni raqamlar sonini topuvchi rekursiv funksiya tuzing.
3. Berilgan sonning raqamli ildizini topuvchi rekursiv funksiya tuzing. Sonning raqamlar yig‘indisini topamiz va bu yig‘indini ham raqamlar yig‘indisi ustma-ust tushsa sonni raqamli ildizi deb ataladi.
4. Arifmetik progressiyani birinchi hadi va ayirmasi berilgan. N ta hadini aniqlovchi rekursiv funksiya tuzing.
5. Arifmetik progressiyani birinchi hadi va ayirmasi berilgan. N ta hadini yig‘indisini hisoblovchi rekursiv funksiya tuzing.
6. Fibonachi sonini k-hadini topuvchi rekursiv funksiya tuzing.
7. Fibonachi sonini k ta hadini yig‘indisini hisoblovchi rekursiv funksiya tuzing.
8. Manfiy bo‘lmagan n va m sonlari uchun Akkerman funksiyasini hisoblovchi rekursiv funksiya tuzing. Akkerman funksiyasi quyidagicha aniqlanadi:

m+1, agar n = 0;

A(n, m) = A(n – 1, 1), agar n ≠ 0, m = 0;

A(n-1, A(n, m-1)), agar n>0, m>0.

1. Berilgan a va b sonlarining EKUBini topuvchi rekursiv funksiya tuzing.
2. Berilgan a va b sonlarining EKUKini topuvchi rekursiv funksiya tuzing.
3. Berilgan a, b va c sonlarining EKUBini topuvchi rekursiv funksiya tuzing.
4. Berilgan sonning raqamlarini teskarisiga yozuvchi rekursiv funksiya tuzing.
5. Soni noma’lum bo‘lgan sonlar ketma-ketligini massiv ishlatmagan holda teskarisiga yozuvchi rekursiv funksiya tuzing.
6. N-Fibonachi sonini oxirgi 17 xonasini aniqlovchi rekursiv dastur tuzing.
7. Maxraji n bo‘lgan [0;1] orasidagi barcha qisqarmas kasrlarni topuvchi rekursiv dastur tuzing.
8. Berilgan sonni necha N faktorialga tengligini aniqlovchi rekursiv funksiya tuzing. Masalan, 6 = 3!.
9. 22^n ning 109 ga bo‘lgandagi qoldiqni hisoblovchi rekursiv funksiya tuzing. n soni [1; 1000000] oraliqda bo‘lishi mumkin.