Programabilni uređaji i objektno orjentisano programiranje

Računske vježbe 5

1. Implementirati funkcije koje određuju:
   - maksimalni elemenat niza,
   - veći od dva cijela broja,
   - maksimalni od dva stringa (leksikografski),
   - "maksimalni" od dva karaktera,
   - za jedan string, karakter sa najvećom ASCII vrijednošću.

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

int maks(int *,int);
inline int maks(int,int);
char * maks(char *, char *);
inline char maks(char,char);
char maks(char*);

int main()
{
    int a[]={1,5,4,3,2,6}, n=6;
    cout<<maks(a,n)<<endl;
    cout<<maks(2,5)<<endl;
    cout<<maks("string","program")<<endl;
    cout<<maks('a','A')<<endl;
    cout<<maks("string");
}

int maks(int *a,int n)
{
    int m;
    m=a[0];
    for(int i=1; i<n; i++)
        if(a[i]>m)
            m=a[i];
    return m;
}
```

```
inline int maks(int a,int b)
{
    return (a>=b) ? a:b;
}

char * maks(char *a,char *b)
{
    if(strcmp(a,b)>=0) return a;
    else return b;
}

inline char maks(char a,char b)
{
    return (a>=b) ? a:b;
}

char maks(char *a)
{
    char m;
    m=a[0];
    for(int i=1; i<strlen(a); i++)
        if(a[i]>m)
            m=a[i];
    return m;
}
```

2. Date su deklaracije funkcija:

```
int f(int, char='0');
int f(char, char);
int f(double);
```

Šta će se desiti nakon sljedećih poziva funkcije:

```
R = f('0');
R = f(2.34);
R = f('c', 3);
```

Prvi poziv će izvršiti prvu funkciju. Njen prvi argument je cijeli broj koji se dobija konverzijom iz karaktera (ASCII) dok je drugi argument podrazumijevani karakter tako da će taj poziv biti isto što i f('0','0');

Kod drugog poziva će biti izvršena treća funkcija. Jedino ova funkcija ima realan broj kao argument.

Kod trećeg poziva i prva i druga funkcija mogu da odgovaraju prvom argumentu (bolje odgovara druga funkcija) ali drugi argument ne odgovara ni jednoj funkciji tako da će doći do greške u programu.

3. Projektovati klasu sa elementarnim operacijama nad kompleksnim brojevima (sabiranje, oduzimanje) pri čemu je potrebno koristiti konstruktore.

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

// definicija klase
class complex
{
    // privatni podaci članovi klase (dva realna broja)
    private:
        float imag, real;

    // javni podaci članovi (i funkcije) klase
    public:
        complex(){}; // konstruktor bez argumenata (kada pravimo prazan kompl. broj)
        complex(float, float); // konstruktor sa argumentima
        complex cSab(complex); // funkcija članica za sabiranje kompleksnih brojeva
        complex cOduz(complex); // funkcija članica za oduzimanje kompleksnih brojeva
        float cRe()const {return real;};
        float cIm()const {return imag;};
};

// realizacija konstruktora sa arguemntima
// funkcija nema tip rezultata, pripada klasi complex (complex::) i naziv joj je complex
// inicijalizaciona lista argumenata ima isti smisao kao da smo napisali real=a; i imag=b;
// u realizaciji funkcije
complex::complex(float a, float b):real(a),imag(b){}

// realizacija funkcije članice za sabiranje dva kompleksna broja
// funkcija ima tip rezultata complex jer vraća objekat kompleksni broj pom
// funkcija pripada klasi kompleks jer je funkcija članica i zbog toga navodimo complex::
// jedan argument funkcije je implicitan i to je objekat koji poziva funkciju (*this)
// drugi argument funkcije je eksplicitan i to je objekat a
// podacima članovima objekta (*this) se može pristupiti navođenjem real i imag
// podacima članovima objekta a se može pristupiti navođenjem a.real i a.imag
complex complex::cSab(complex a)
{
    complex pom;
    pom.real=real+a.real;
    pom.imag=imag+a.imag;
    return pom;
}

// realizacija funkcije članice za oduzimanje dva kompleksna broja
complex complex::cOduz(complex a)
{
    complex pom;
    pom.real=real-a.real;
    pom.imag=imag-a.imag;
    return pom;
}
```

```
// ovdje počinje izvršavanje programa i zauzimanje memorije za promjenljive standardnog ili
// klasnog tipa
int main()
{
    float a,b;

    cout<<"Unesi vrijednost za realni i imaginarni dio c1"<<'\n';
    cin>>a>>b;
   // poziv konstruktora sa dva argumenta
   // u memoriji je kreiran objekat (promjenljiva) c1 koji ima istu strukturu kao definicija
   // klase complex a njegovi podaci članovi su inicijalizovani na vrijednosti a i b
    complex c1(a,b);

    cout<<"Unesi vrijednost za realni i imaginarni dio c2"<<'\n';
    cin>>a>>b;
    complex c2(a,b);

   // poziv konstruktora bez argumenata (njegovi podaci članovi nijesu inicijalizovani)
    complex c3;

   // poziv funkcije za sabiranje kompleksnih brojeva c1 i c2
   // c1 je implicitni argument a c2 eksplicitni argument
   // rezultat funkcije cSab će biti smješten u prazan objekat c3
    c3=c1.cSab(c2);

   // objekte neke klase možemo odštampati isključivo štampajući njihove podatke članove!
   // da bismo pristupili privatnim podacima članovima objekta neke klase neophodno je
   // realizovati funkcije članice (inspektore - getere) čiji je zadatak da vrate pojedinačne
   // podatke članove klase kao rezultat (funkcije cRe() i cIm())
    cout<<"Zbir dva unijeta kompleksna broja je ("<<c3.cRe()<<" "<<c3.cIm()<<")"<<endl;

    c3=c1.cOduz(c2);
    cout<<"Razlika dva unijeta kompleksna broja je ("<<c3.cRe()<<" "<<c3.cIm()<<")"<<endl;
}

// Napomena: Sve funkcije u ovoj klasi su realizovane kao funkcije članice (pripadaju klasi
// complex tj. objektima klase complex. Pošto funkcije članice pripadaju objektima klase, to
// znači da ih samo objekti klase mogu pozvati. Primjer poziva funkcija članica su c1.cSab(c2)
// ili c3.cIm() ili c3.cRe(). Funkcije članice klase nikako nije moguće pozvati bez prethodnog
// navođenja naziva objekta kojem pripadaju (ne mogu se pozvati cSab() ili cIm() ili cRe() bez
// navođenja naziva objekta)!

// Svaka funkcija članica klase podrazumijevano pripada nekom kreiranom objektu klase! Tom
// objektu se, unutar realizacije funkcije, pristupa kao implicitnom argumentu (*this). To je
// razlog zašto funkcije cSab() i cOduz() imaju samo po jedan argument. Jedan se podrazumijeva
// (*this) a drugi je eksplicitno naveden u listi argumenata.

// Klasa, u ovom slučaju complex, predstavlja tip podatka. Kao što smo do sada imali tipove
// podataka int, float, char itd. tako sada možemo definisati nove, apstraktne, tipove podataka.
```

4. Napraviti klasu **tačka** koja sadrži:
   - koordinate tačke (dva realna broja);
   - odgovarajuće konstruktore;
   - funkciju članicu za računanje rastojanja između dvije tačke.

   Nakon toga napraviti klasu **krug** koja sadrži:
   - tačku koja predstavlja centar kruga i tačku sa ivice kruga (objekti tipa klase tačka);
   - odgovarajuće konstruktore;
   - funkcije članice za izračunavanje obima i površine kruga.

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

class tacka
{
    private:
        float x,y;
    public:
        tacka(){}
        tacka(float, float);
        float rastojanje(tacka) const;
};

tacka::tacka(float a, float b):x(a){y=b;}

float tacka::rastojanje(tacka a)const
{
    return sqrt(pow(x-a.x,2)+pow(y-a.y,2));
}

// u programu može postojati proizvoljan broj klasa
// da bi klasa krug mogla da koristi objekte klase tacka neophodno je da klasa tacka bude
// prethodno realizovana
class krug
{
    private:
        tacka centar;
        tacka saKruga;
    public:
        krug(){} //drugi način: krug():centar(tacka(0,0)),saKruga(tacka(0,0)){};
        krug(float, float, float, float);
        //drugi način: krug(tacka, tacka);
        float kPovrsina();
        float kObim();
};


krug::krug(float x1, float x2, float x3, float x4):centar(tacka(x1,x2))
{
    saKruga=tacka(x3,x4);
}

/*
//Drugi način:
krug::krug(tacka t1, tacka t2):centar(t1)
{
    saKruga=t2;
}
*/
```

```
float krug::kPovrsina()
{
    float r;
    r=centar.rastojanje(saKruga);
    return pow(r,2)*3.14;
}

float krug::kObim()
{
    float r;
    const float pi=3.14;
    r=centar.rastojanje(saKruga);
    return 2*r*pi;
}

int main()
{
    float x1, y1, x2, y2;

    cout<<"Unesi koordinate centra kruga i tacke sa kruga"<<endl;
    cin>>x1>>y1>>x2>>y2;

    krug k1(x1,y1,x2,y2);
    // drugi način: krug k1(tacka(x1,y1), tacka(x2,y2));

    cout<<"Povrsina kruga je: "<<k1.kPovrsina()<<endl;
    cout<<"Obim kruga je: "<<k1.kObim();
}
```