KLASE - KONSTRUKTORI

1. Napisati na programskom jeziku C++ klasu kompleksnih brojeva. Kompleksni broj se stvara zadavanjem realnog i imaginarnog dela.

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

class Complex
{
   double real, imag;
public:
   Complex();
   Complex(double, double);
   ~Complex();

   Complex Saberi(Complex);
   Complex Oduzmi(Complex);
   Complex Mnozi(Complex);
   Complex Deli(Complex);

   double Real() const { return real; } // inline metoda
   double Imag() const { return imag; } // inline metoda
};
```

Complex/Complex.h

```
#include "Complex.h"

Complex::Complex() : real(0), imag(0)
{
   cout << "Pozvan je default konstruktor" << endl;
}

Complex::Complex(double a, double b) : real(a), imag(b)
{
   //this->real = a;
   //this->imag = b;
   cout << "Pozvan je konstruktor sa dva parametra" << endl;
}

Complex::~Complex()
{
}

Complex Complex::Saberi(Complex a)
{
```

```
21    //Complex res;
22    //res.real = this->real + a.real;
23    //res.imag = this->imag + a.imag;
24    Complex res(this->real + a.real, this->imag + a.imag);
25    return res;
26 }
27
28 Complex Complex::Oduzmi(Complex a)
29 {
30    Complex res;
31    res.real = this->real - a.real;
32    res.imag = this->imag - a.imag;
33    return res;
34 }
35
36 Complex Complex::Mnozi(Complex a)
37 {
38    Complex res;
39    res.real = this->real * a.real - this->imag * a.imag;
40    res.imag = this->real * a.imag + this->imag * a.real;
41    return res;
42 }
43
44 Complex Complex::Deli(Complex a)
45 {
46    Complex res;
47    res.real = (this->real * a.real + this->imag * a.imag)
48       / (this->imag * this->imag - a.imag * a.imag);
49    res.imag = (this->real * a.imag + this->imag * a.real)
50       / (this->imag * this->imag - a.imag * a.imag);
51    return res;
52 }
```

Complex/Complex.cpp

```
1  #include "Complex.h"
2
3  int main()
4  {
5     double a, b;
6     cout << "Unesite vrednost za realan i imaginarni deo c1 " << endl;
7     cin >> a >> b;
8     Complex c1(a, b);
9
10    cout << "Unesite vrednost za realan i imaginarni deo c2 " << endl;
11    cin >> a >> b;
12    Complex c2(a, b);
13
14    Complex c3;
15    c3 = c1.Saberi(c2);
16
17    cout << "(" << c3.Real() << "," << c3.Imag() << ")" << endl;
18    return 0;
19 }
```

Complex/Source.cpp

2. Napisati na programskom jeziku C++ klasu za rad sa krugovima. Krug se zadaje svojim centrom

u ravni (instanca klase Tacka), kao i poluprecnikom (realan broj).

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

class Tacka
{
   double x, y;
public:
   Tacka();
   Tacka(double, double);
   Tacka(const Tacka&);
   ~Tacka();

   double rastojanje(Tacka) const;
   double apscisa() const;
   double ordinata() const;
};
```

Krug/Tacka.h

```
#include "Tacka.h"

Tacka::Tacka()
{
}

Tacka::Tacka(double a, double b) : x(a), y(b)
{

}

Tacka::Tacka(const Tacka& t) : x(t.x), y(t.y)
{
   cout << "[Tacka] Konstruktor kopije" << endl;
}

double Tacka::rastojanje(Tacka t) const
{
   return sqrt(pow(x - t.x, 2) + pow(y - t.y, 2));
}

double Tacka::apscisa() const
{
   return x;
}

double Tacka::ordinata() const
{
   return y;
}

Tacka::~Tacka()
{
}
```

Krug/Tacka.cpp

```
#pragma once
#include "Tacka.h"
class Krug
{
   Tacka c;
   double r;
public:
   static const double PI;
   Krug();
   Krug(Tacka, double);
   Krug(double, double, double);
   Krug(const Krug&);
   ~Krug();

   double Povrsina() const;
   double Obim() const;
   bool PostojiPresek(Krug) const;
   bool SuKoncentricni(Krug) const;
   bool Sadrzi(Krug) const;
};
```

Krug/Krug.h

```
#include "Krug.h"

double const Krug::PI = 3.14159;

Krug::Krug()
{
}

Krug::Krug(Tacka t, double pp) : c(t), r(pp)
{

}

Krug::Krug(double x, double y, double pp) : c(x, y), r(pp)
{

}

Krug::Krug(const Krug &k) : c(k.c), r(k.r)
{
   cout << "[Krug] Konstruktor kopije" << endl;
}

double Krug::Obim() const
{
   return 2 * r * PI; // moze, ali je jasnije Krug::PI
}

double Krug::Povrsina() const
{
```

```
31    return r * r * Krug::PI;
32 }
33
34 bool Krug::PostojiPresek(Krug k) const
35 {
36    return this->c.rastojanje(k.c) <= this->r + k.r;
37 }
38
39 bool Krug::SuKoncentricni(Krug k) const
40 {
41    return (this->c.apscisa() == k.c.apscisa())
42       && (this->c.ordinata() == k.c.ordinata());
43 }
44
45 bool Krug::Sadrzi(Krug k) const
46 {
47    return this->r >= this->c.rastojanje(k.c) + k.r;
48 }
49
50 Krug::~Krug()
51 {
52 }
```

Krug/Krug.cpp

```
1  #include "Krug.h"
2
3  int main()
4  {
5     cout << "Unesite centar kruga i pp" << endl;
6     double x, y, r;
7     cin >> x >> y >> r;
8     Krug k1(x, y, r);
9     cout << "Povrsina kruga je: " << k1.Povrsina() << endl;
10    cout << "Obim kruga je: " << k1.Obim() << endl;
11
12    cout << "Unesite c i pp drugog kruga" << endl;
13    cin >> x >> y >> r;
14    Tacka t(x, y);
15    Krug k2(t, r);
16
17    cout << (k1.PostojiPresek(k2) ? "Seku se" : "Ne seku se") << endl;
18    cout << (k1.SuKoncentricni(k2) ? "Koncentricni su" :
19       "Nisu koncentricni") << endl;
20    cout << (k1.Sadrzi(k2) ? "Prvi sadrzi drugi" :
21       "Prvi ne sadrzi drugi") << endl;
22    return 0;
23 }
```

Krug/Source.cpp