NASLEĐIVANJE I POLIMORFIZAM

1. Tačka u dvodimenzionalnom prostoru opisana je realnim koordinatama x i y, koje mogu i da se pročitaju. Stvara se zadavanjem vrednosti koordinata (podrazumevano 0). Tačka se može ispisati na standardni izlaz.

Apstraktna figura u ravni ima zadatu tačku težišta. Stvara se zadavanjem tačke težišta (podrazumevano koordinatni početak), koje može naknadno da se promeni. Moguće je figuru pomeriti sa zadatim vrednostima pomeraja, izračunati joj obim i površinu i ispisati na standardni izlaz.

Krug je figura koja ima dodatno definisan poluprečnik. Stvara se zadavanjem dužine poluprečnika (podrazumevano 1) i položaja centra (težišta, podrazumevano koordinatni početak).

Kvadrat je figura sa definisanom veličinom stranice. Stvara se zadavanjem dužine stranice (podrazumevano 1) i položajem težišta (podrazumevano koordinatni početak).

Napisati program na programskom jeziku C++ koji prvo učita nekoliko figura, ispiše ih na standarni izlaz, zatim svaku figuru pomeri za zadati pomeraj i ispiše novo stanje figura u ravni. Na kraju, program treba da oslobodi sve zauzete resurse.

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

class Tacka
{
   double x, y;
public:
   Tacka(double a = 0, double b = 0) : x(a), y(b) {}
   ~Tacka() {}
   double aps() { return x; }
   double ord() { return y; }
   void Tcitaj() { cout << "(" << x << "," << y << ")"; }
};
```

Figure/Tacka.h

```
#pragma once
#include "Tacka.h"

const Tacka KP;

class Figura
{
   Tacka teziste;
public:
   static const double PI;
   Figura(Tacka t = KP) : teziste(t) {}
```

```
12    virtual ~Figura() {}
13    void pomeri(double, double);
14    virtual double Obim() const = 0;
15    virtual double Povrsina() const = 0;
16    virtual void citaj() { cout << "T="; teziste.Tcitaj(); }
17
18 };
```

Figure/Figura.h

```
1  #include "Figura.h"
2
3  const double Figura::PI = 3.141592;
4
5  void Figura::pomeri(double a, double b)
6  {
7     teziste = Tacka(teziste.aps() + a, teziste.ord() + b);
8  }
```

Figure/Figura.cpp

```
1  #pragma once
2  #include "Figura.h"
3  class Krug : public Figura
4  {
5     double poluprecnik;
6  public:
7     Krug(double rr = 1, Tacka k = KP) : Figura(k), poluprecnik(rr) {};
8     ~Krug() {}
9     double Obim() const { return 2 * poluprecnik * PI; }
10    double Povrsina() const { return pow(poluprecnik, 2) * PI; }
11    void citaj();
12 };
```

Figure/Krug.h

```
1  #include "Krug.h"
2
3  void Krug::citaj()
4  {
5     cout << "Figura krug: [ r=" << poluprecnik << ",";
6     Figura::citaj();
7     cout << ", O=" << Obim() << ", P=" << Povrsina() << " ]" << endl;
8  }
```

Figure/Krug.cpp

```
1  #pragma once
2  #include "Figura.h"
3
4  class Kvadrat : public Figura
5  {
6     double osnovica;
```

```
public:
   Kvadrat(double a = 1, Tacka t = KP) : Figura(t), osnovica(a) {}
   ~Kvadrat() {}
   double Obim() const { return 4 * osnovica; }
   double Povrsina() const { return pow(osnovica, 2); }
   void citaj();
};
```

Figure/Kvadrat.h

```
#include "Kvadrat.h"

void Kvadrat::citaj() {
   cout << "Figura kvadrat: [ a=" << osnovica << ",";
   Figura::citaj();
   cout << ", O=" << Obim() << ", P=" << Povrsina() << " ]" << endl;
}
```

Figure/Kvadrat.cpp

```
#include "Krug.h"
#include "Kvadrat.h"

int main()
{
   Figura *pf[4];
   pf[0] = new Krug;
   pf[1] = new Kvadrat;
   pf[2] = new Krug(2, Tacka(3, 3));
   pf[3] = new Kvadrat(2.5, Tacka(1.3, 2));
   for (int j = 0; j<4; j++) pf[j]->citaj();
   pf[0]->pomeri(1, 0.5);
   pf[1]->pomeri(0.5, 1);
   for (int j = 0; j<2; j++) pf[j]->citaj();
   for (int j = 0; j<4; j++) { delete pf[j]; pf[j] = 0; }

   return 0;
}
```

Figure/Source.cpp

2. Vozilo ima sopstvenu težinu, može da mu se pročita vrsta, da se odredi ukupna težina (podrazumenano jednaka sopstvenoj težini) i da se ispiše u izlazni tok podataka, tako što se ispiše oznaka vrste, a u zagradama navede sopstvena težina. Stvara se zadavanjem sopstvene težine.

Putničko vozilo je vozilo koje prevozi putnike i ima podatak o broju putnika i težina prosečnog putnika. Stvara se zadavanjem sopstvene težine, broja putnika i njihove prosečne težine. Vrsta mu je označena sa 'P', ukupna težina se dobija sabiranjem sopstvene težine i težine putnika, a ispisuje se isto kao i Vozilo, s tim što se u zagradama navodi još i broj putnika i srednja težina putnika.

Teretno vozilo je Vozilo koje prevozi teret zadate težine. Stvara se zadavanjem sopstvene težine i težine tereta. Ukupna težina mu se dobija sabiranjem sopstvene i težine tereta. Oznaka vrste je 'T'. Ispisuje se isto kao i Vozilo, s tim što se u zagradama, pored sopstvene težine, navodi i težina tereta.

Napisati na programskom jeziku C++ klase za opisane koncepte i glavni program koji testira njihove funkcionalnosti.

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

class Vozilo
{
   double sopTez; // Sopstvena tezina.
public:
   Vozilo(double st) { sopTez = st; } // Konstruktor.
   virtual char vrsta() const = 0; // Vrsta vozila.
   virtual double tezina() const { return sopTez; } // Ukupna tezina.
   static Vozilo* napraviVozilo(int vrsta); // Pravljenje vozila zadate vrste.
protected:
   virtual void pisi(ostream& it) const // Pisanje.
   {
      it << vrsta() << '(' << sopTez << ',';
   }
   friend ostream& operator<< (ostream& it, const Vozilo& v)
   {
      v.pisi(it); return it;
   }
};

class PVozilo : public Vozilo
{
   double srTez; // Srednja tezina putnika.
   int brPut; // Broj putnika.
public:
   PVozilo(double st, double srt, int bp) // Konstruktor.
      : Vozilo(st)
   {
      srTez = srt; brPut = bp;
   }
   char vrsta() const { return 'P'; } // Vrsta vozila.
   double tezina() const // Ukupna tezina.
   {
      return Vozilo::tezina() + srTez * brPut;
   }
private:
   void pisi(ostream& it) const // Pisanje.
   {
      Vozilo::pisi(it); it << srTez << ',' << brPut << ')';
   }
};

class TVozilo : public Vozilo
{
   double teret; // Tezina tereta.
public:
   TVozilo(double st, double t) // Konstruktor.
      : Vozilo(st)
   {
      teret = t;
   }
   char vrsta() const { return 'T'; } // Vrsta vozila.
   double tezina() const // Ukupna tezina.
   {
      return Vozilo::tezina() + teret;
   }
private:
```

```
61    void pisi(ostream& it) const // Pisanje.
62    {
63       Vozilo::pisi(it); it << teret << ')';
64    }
65 };
```

Vozila/vozila.h

```
1  #include "vozila.h"
2
3  Vozilo* Vozilo::napraviVozilo(int vrsta)
4  {
5     switch (vrsta)
6     {
7     case 't': case 'T':
8        cout << "Sopstvena tezina? "; double sTez; cin >> sTez;
9        cout << "Tezina tereta? "; double ter; cin >> ter;
10       return new TVozilo(sTez, ter);
11       break;
12    case 'p': case 'P':
13       cout << "Sopstvena tezina? "; cin >> sTez;
14       cout << "Sr. tezina putnika? "; double srTez; cin >> srTez;
15       cout << "Broj putnika? "; int brPut; cin >> brPut;
16       return new PVozilo(sTez, srTez, brPut);
17       break;
18    default:
19       cout << "*** Nepoznata vrsta vozila!\n";
20       return nullptr;
21    }
22 }
```

Vozila/vozila.cpp

```
1  #include "vozila.h"
2
3  int main()
4  {
5     Vozilo* vozila[100];
6     int n = 0;
7     while (true)
8     {
9        cout << "\nVrsta vozila (T,P,*)? ";
10       char vrsta;
11       cin >> vrsta;
12       if (vrsta == '*') break;
13       Vozilo* v = Vozilo::napraviVozilo(vrsta);
14       if (v != nullptr)
15          vozila[n++] = v;
16    }
17    cout << "\nNosivost mosta? ";
18    double nosivost;
19    cin >> nosivost;
20    cout << "\nMogu da predju most:\n";
21    for (int i = 0; i<n; i++)
22       if (vozila[i]->tezina() <= nosivost)
23          cout << *vozila[i] << " - " << vozila[i]->tezina() << endl;
24    // Dealokacija memorije.
```

```
25    for (int i = 0; i<n; i++)
26       delete vozila[i];
27    return 0;
28 }
```

Vozila/Source.cpp

3. Pojektovati klase Pravougaonik i Trougao koje su izvedene iz klase Figura (sadrži težište kao zajedničku karakteristiku za sve figure, funkciju koja omogućava pomeraj težišta za zadatu vrednost i virtuelne funkcije dijagonala, poluprečnik opisanog kruga i čitaj). Klase treba da imaju specifične funkcije za računanje dijagonale i poluprečnika opisanog kruga kao i očitavanje odgovarajućih podataka članova. Za klasu Trougao treba realizovati konstruktor tako da može stvoriti jednakostranični trougao kada se unosi podatak za dužinu jedne stranice kao i opšti slučaj. Funkcija za računanje dijagonale u slučaju trougla treba da vraća -1. Poluprečnik opisanog kruga, za trougao (stranica a, b i c) može se računati kao:

$$R = \frac{abc}{\sqrt{(a^2+b^2+c^2)^2 + 2(a^4+b^4+c^4)}}.$$

```
1  #pragma once
2  #include <iostream>
3  using namespace std;
4  class Tacka
5  {
6     double x, y;
7  public:
8     Tacka(double a = 0, double b = 0)
9        : x(a), y(b) {}
10    ~Tacka() {}
11    void tcitaj() const
12    {
13       cout << "(" << x << "," << y << ")"
14          << endl;
15    }
16    double aps() const { return x; }
17    double ord() const { return y; }
18 };
```

Figure2/Tacka.h

```
1  #pragma once
2  #include "Tacka.h"
3  const Tacka KP;
4
5  class Figura
6  {
7  protected:
8     Tacka teziste;
9  public:
10    Figura(Tacka t = KP) : teziste(t) {}
11    ~Figura() {}
12    void pomeri(double, double);
13    virtual double dijagonala() const = 0;
14    virtual double poluprecnik() const = 0;
```

```
15    virtual void citaj() const = 0;
16 };
```

Figure2/Figura.h

```
1  #include "Figura.h"
2
3  void Figura::pomeri(double dx, double dy)
4  {
5     teziste = Tacka(dx + teziste.aps(),
6        dy + teziste.ord());
7  }
```

Figure2/Figura.cpp

```
1  #pragma once
2  #include "Figura.h"
3  class Pravougaonik :
4     public Figura
5  {
6     double a, b;
7  public:
8     Pravougaonik(double a1 = 1, double b1 = 1,
9        Tacka t = KP) : a(a1), b(b1), Figura(t)
10    { }
11    ~Pravougaonik() {}
12    double dijagonala() const
13    {
14       return sqrt(a * a + b * b);
15    }
16    double poluprecnik() const
17    {
18       return dijagonala() / 2;
19    }
20    void citaj() const;
21 };
```

Figure2/Pravougaonik.h

```
1  #include "Pravougaonik.h"
2
3  void Pravougaonik::citaj() const
4  {
5     cout << "Pravougaonik: [a = " << a
6        << ", b = " << b;
7     cout << ", Teziste je: ";
8     teziste.tcitaj();
9     cout << ", d = " << dijagonala()
10       << ", r = " << poluprecnik()
11       << "]" << endl;
12 }
```

Figure2/Pravougaonik.cpp

```
#pragma once
#include "Figura.h"
class Trougao :
   public Figura
{
   double a, b, c;
public:
   Trougao(double a1 = 1, Tacka t = KP)
      : a(a1), b(a1), c(a1), Figura(t)
   {}
   Trougao(double a1, double b1, double c1,
      Tacka t = KP)
      : a(a1), b(b1), c(c1), Figura(t)
   {}
   ~Trougao() {}
   double dijagonala() const { return -1; }
   double poluprecnik() const;
   void citaj() const;
};
```

Figure2/Trougao.h

```
#include "Trougao.h"

double Trougao::poluprecnik() const
{
   return a * b * c /
      sqrt(pow(a * a + b * b + c * c, 2)
         + 2 * (pow(a, 4) + pow(b, 4)
            + pow(c, 4)));
}

void Trougao::citaj() const
{
   cout << "Trougao: [a = " << a
      << ", b = " << b << ", c = " << c;
   cout << ". Teziste je: ";
   teziste.tcitaj();
   cout << ", d = " << dijagonala()
      << ", r = " << poluprecnik()
      << "]" << endl;
}
```

Figure2/Trougao.cpp

```
#include "Trougao.h"
#include "Pravougaonik.h"

int main()
{
   Figura *pf[5];
   pf[0] = new Pravougaonik;
   pf[1] = new Trougao;
   pf[2] = new Trougao(2.5, 3);
   pf[3] = new Pravougaonik(2, 3, Tacka(1, 2));
   pf[4] = new Trougao(2, 4, 5, Tacka(1, 2));
```

```
12    for (int i = 0; i < 5; i++)
13    {
14       pf[i]->citaj(); // polimorfizam!
15    }
16    pf[0]->pomeri(2.3, 4); // nije polimorfizam!
17    pf[1]->pomeri(3.1, 2); // nije polimorfizam
18    for (int i = 0; i < 5; i++)
19    {
20       pf[i]->citaj();
21    }
22    for (int i = 0; i < 5; i++)
23    {
24       delete pf[i];
25       pf[i] = 0;
26    }
27    return 0;
28 }
```

Figure2/Source.cpp

4. Realizovati klasu Časovnik koja će predstavljati vreme u satima, minutima i sekundama. Klasa ima funkciju za povećavanje broja sekundi za po jednu. Iz date klase izvesti klasu Let koja sadrži naziv i broj leta. Neka podaci nasleđeni iz osnovne klase Časovnik predstavljaju vreme polaska i neka postoji funkcija koja će omogućiti promenu ovog vremena za neko zadato kašnjenje. Napisati glavni program u kojem se zadaje vreme polaska nekog leta i kašnjenje a čita novo vreme polaska, sa uračunatim kašnjenjem.

```
1  #pragma once
2  #include <string>
3  #include <iostream>
4  using namespace std;
5  class Casovnik
6  {
7  protected:
8     int sec, min, sati;
9  public:
10    Casovnik() {}
11    Casovnik(int h, int m, int s)
12       : sati(h), min(m), sec(s)
13    {}
14    ~Casovnik() {}
15    int daj_s() const { return sec; }
16    int daj_m() const { return min; }
17    int daj_sat() const { return sati; }
18    void Otkucaj();
19    void stampaj();
20 };
```

Let/Casovnik.h

```
1  #include "Casovnik.h"
2
3  void Casovnik::Otkucaj()
4  {
5     sec++;
6     if (sec >= 60)
```

```
7     {
8        sec = 0; min++;
9        if (min >= 60)
10       {
11          min = 0; sati++;
12          if (sati >= 24) sati = 0;
13       }
14    }
15 }
16
17 void Casovnik::stampaj()
18 {
19    cout << sati << ":" << min << ":"
20       << sec << endl;
21 }
```

Let/Casovnik.cpp

```
1  #pragma once
2  #include "Casovnik.h"
3  class Let :
4     public Casovnik
5  {
6     char *naziv;
7  public:
8     Let() { naziv = 0; }
9     Let(char*, int, int, int);
10    Let(const Let&);
11    ~Let();
12    void Kasnjenje(Casovnik);
13    void stampaj();
14 };
```

Let/Let.h

```
1  #include "Let.h"
2
3  Let::Let(char *ime, int h, int m, int s)
4     : Casovnik(h, m, s)
5  {
6     naziv = new char[strlen(ime) + 1];
7     strcpy_s(naziv, strlen(ime) + 1, ime);
8  }
9
10 Let::Let(const Let& stara)
11    : Casovnik(stara.sati, stara.min, stara.sec)
12 {
13    naziv = new char[strlen(stara.naziv) + 1];
14    strcpy_s(naziv, strlen(stara.naziv) + 1,
15       stara.naziv);
16 }
17
18 Let::~Let()
19 {
20    delete[] naziv;
21    naziv = 0;
22 }
```

```
23
24 void Let::Kasnjenje(Casovnik vreme)
25 {
26    int s = vreme.daj_s()
27       + vreme.daj_m() * 60
28       + vreme.daj_sat() * 60 * 60;
29    for (int i = 0; i < s; i++)
30    {
31       Otkucaj();
32    }
33 }
34
35 void Let::stampaj()
36 {
37    cout << "Naziv leta je: " << naziv
38       << ". Vreme odlaska je: ";
39    Casovnik::stampaj();
40 }
```

Let/Let.cpp

```
1 #include "Let.h"
2
3 int main()
4 {
5    Let prvi("Beograd-Podgorica", 20, 30, 0);
6    Let drugi(prvi);
7    prvi.stampaj();
8    cout << "Unesite kasnjenje: ";
9    int h, m, s;
10   cin >> h >> m >> s;
11   prvi.Kasnjenje(Casovnik(h, m, s));
12   prvi.stampaj();
13   drugi.stampaj();
14   return 0;
15 }
```

Let/Source.cpp