Posljednji zadatak sa petih laboratorijskih vježbi

```
#include <iostream>

/*Realizovati klasu radnik koja predstavlja trenutno zaposlene radnike u firmi. Klasa posjeduje privatne podatke članove koji

predstavljaju identifikacioni broj radnika (cijeli broj) i godinu zaposlenja radnika (cijeli broj).

U okviru klase potrebno je ralizovati sljedeće funkcije članice:

•	odgovarajuće konstruktore, destruktor, mutatore i inspektore;

•	funkciju za računanje koliko je dugo radnik zaposlen u firmi (broj godina).

U glavnom programu potrebno je inicijalizovati niz radnika (niz objekata klase radnik) proizvoljne dužine

 i u unijetom nizu pronaći radnika sa najdužim radnim stažom, čije je podatke članove potrebno štampati na izlazu.

*/


using namespace std;

class Radnik{

private:

   int ib;

   int gz;

public:

   Radnik(){}

   Radnik(int a, int b):ib(a),gz(b){}

   ~Radnik(){}

   void PromijeniIb(int a){ib = a;}

   //u main-u Radnik prvi(123456,2008); ne smije prvi.ib = 111111; jer je private
```

```
    // mora prvi.PromijeniIb(111111); Pristupa se podacima pomocu pokazivaca this
    // on ukayuje na objekat za koji je poyvana funkcija, ovdje je to prvi
    // pa kada imamo ib=a; isto je kao da pise *this.ib = a; ili this->ib = a
    // this ne mora da se pise
    void PromijeniGz(int b){gz = b;}
    int vratiIb(){return ib;}
    //u glavnom programu ne smijem da pisem cout<<prvi.ib; jer je private
    // pristupam mu preko fukcije clanice inspektor cout<<prvi.vratiIb();
    // u this se kopira pokayivac na prvi i vraca se ib od prvi
    int vratiGz(){return gz;}
    int Staz(){return 2020 - gz;}
    void citaj()const;
};

void Radnik::citaj()const{
    cout<<"Identifikacioni broj: "<<ib<<", godina zaposlenja "<<gz<<endl;
}



int main()
{
    cout << "Hello world!" << endl;
    Radnik prvi(123456,2008); // jedan radnik
    prvi.citaj();
        //da nemam f-ju citaj() koja je f-ja clanica morala bih da kucam
```

```
//cout<<"Identifikacioni broj: "<<prvi.vratiIb()<<", godina zaposlenja "<<prvi.vratiGz()<<endl;
prvi.PromijeniIb(111111);
cout<<"Novi identifikacioni broj je"<<prvi.vratiIb()<<endl;
Radnik niz[10];
int n;
cout<<"Koliko imas radnika?"<<endl;
cin>>n;
int indb,god;
cout<<"Unesi ident brojeve i godine zaposlenja "<<endl;
for(int i=0; i < n; i++){
   cin>>indb>>god;
   niz[i] = Radnik(indb,god);
   //niz[i].PromijeniIb(indb);
   //niz[i].PromijeniGz(god);
}
int mx = 0;
int indeks = 0;
for(int i=0; i<n; i++){
   if(niz[i].Staz()> mx){
      mx = niz[i].Staz();
      indeks = i;
   }
}
cout<<"Najduzi staz je: "<<niz[indeks].Staz();
cout<<"Godina zaposlenja mu je "<<niz[indeks].vratiGz()<<endl;
```

```
    return 0;
}
```

Zadatak sa šestih računskihvježbi

```
#include <iostream>
#include<string.h>

using namespace std;

/*
```

1. Realizovati klasu radnik koja ima četiri podatka člana i to:

- koeficijent za platu (cijeli broj);
- identifikacioni broj radnika (pokazivač na cijeli broj);
- ime radnika (pokazivač na niz karaktera)
- javni statički podatak koji će služiti za brojanje ukupnog broja radnika (objekata date klase).

staticku promjenljivu koja pamt najveci koeficijent ikada

Klasa posjeduje konstruktor, destruktor i konstruktor kopije, kao i funkcije članice za pristup podacima članovima radi očitavanja i izmjene.

Potrebno je realizovati i funkciju koja od dva radnika vraća ime radnika sa većim koeficijentom za platu.

```
*/

class Radnik{
private:
    int koef;
    int *ib;
    char *ime;
```

```
public:
   static int broj;
   static int najkoef;
   Radnik(){ib = 0; ime = 0; broj++;}
   Radnik(int,int,char *);
   Radnik(const Radnik &);
   ~Radnik();
   int vratiKoef(){return koef;}
   int vratiIb(){return *ib;}
   //ib je adresa na kojoj je upusan vrijednost ident. broja
   //*ib je vrijednost ident. broja
   char *vratiIme(){return ime;}
   void PromKoef(int a){
      koef = a;
      if(koef > najkoef)
         najkoef = koef;
         }
   void PromIb(int b){*ib = b;}
   void PromIme(char *);
   char *VeciKoef(Radnik);
   static void PromMaxKoef(int a){najkoef = a;}
};

int Radnik::broj = 0;
int Radnik::najkoef = 0;
```

```
Radnik::Radnik(int a,int b,char * ime1):koef(a),ib(new int(b)){
   //ib =  new int; moze i ovako
   //*ib = b;
   if(koef > najkoef)
      najkoef = koef;
   ime = new char[strlen(ime1)];
   strcpy(ime,ime1);
   broj++;
}

Radnik::Radnik(const Radnik &a):koef(a.koef){
   ib = new int;
   *ib = *a.ib;
   ime = new char[strlen(a.ime)+1];
   strcpy(ime,a.ime);
   broj++;
}

Radnik::~Radnik(){
   delete ib;
   ib = 0;
   delete []ime;
   ime = 0;
   broj--;
```

```
}



// !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
void Radnik::PromIme(char *novo){
   delete []ime; //oslobadjamo memoriju
   //jer nam novo ime moze imati vise slova
   ime = new char[strlen(novo)];
   //zauzimamo koliko nam treba za novo ime
   strcpy(ime,novo);
}

char *Radnik::VeciKoef(Radnik a){
   if(koef > a.koef)
      return ime;
   else
      return a.ime;
}

int main()
{
   Radnik prvi(9,123456,"Vesna"),drugi(8,123444,"Janko");
   cout<<drugi.VeciKoef(prvi)<<endl;
   Radnik treci(drugi);
   treci.PromIme("Vladimirijan");
```

```
    cout<<"Ime treceg je"<<treci.vratiIme()<<endl;
    cout<<"Ime drugog je"<<drugi.vratiIme()<<endl;
    cout<<"U programu imam "<<Radnik::broj<<" objekta."<<endl;
    cout<<"Maksimalni koef. ikada je"<<Radnik::najkoef<<endl;
    //Radnik::PromMaxKoef(15); preko f-je
    Radnik::najkoef = 16; // da je private ovo ne bi moglo
    //moralo bi preko funkcije
    cout<<"Postavljam novi maksimalni koeficijent"<<Radnik::najkoef<<endl;

    //poziv konstruktora kopije
    cout << "Hello world!" << endl;
    return 0;
}
```