Programabilni uređaji i objektno orjentisano programiranje

Računske vježbe 8

1. Realizovati klasu **razlomak** koja predstavlja racionalne brojeve. Izvršiti preklapanje operatora +, += kao i operatora za prefiksno i postfiksno inkrementiranje. Prilikom realizacije operatora + uzeti u obzir i mogućnost sabiranja racionalnih brojeva sa cijelim brojem, pri čemu se cijeli broj može očekivati i kao lijevi i kao desni operand.

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

class razlomak
{
    private:
        int br;
        int im;
    public:
        razlomak(int=0,int=1);
        ~razlomak(){};

        // Prijateljskoj funkciji ne smijemo proslijediti objekte po referenci jer nam je neophodan
       //  poziv konstruktora zbog potencijalnog sabiranja sa cijelim brojevima
        friend razlomak operator+(razlomak, razlomak);
        razlomak& operator+=(razlomak);
        razlomak& operator++();            //prefiksno inkrementiranje
        razlomak operator++(int);          //postfisksno inkrementiranje
        void stampaj(){cout<<br<<"/"<<im<<endl;}
};

razlomak::razlomak(int a,int b):br(a),im(b){}

razlomak operator+(razlomak r1, razlomak r2)
{
    razlomak rez;
    rez.br = r1.br*r2.im + r2.br*r1.im;
    rez.im = r1.im*r2.im;
    return rez;
}

razlomak& razlomak::operator+=(razlomak r)
{
    br = br*r.im + im*r.br;
    im = im*r.im;
    return *this;
}

//prefiksno inkrementiranje
razlomak& razlomak::operator++()
{
    //možemo iskoristiti operator += jer je već realizovan
    return (*this)+=1;
}

//postfiksno inkrementiranje
razlomak razlomak::operator++(int i)
{
    razlomak pom(*this);      //kreiramo kopiju objekta kojeg inkrementiramo
    (*this)+=1;               //inkrementiramo vrijednost originalnog objekta
    return pom;               //vraćamo vrijednost prije inkrementiranja (vrijednost kopije)
}

int main()
{
    int b,i;

    cout<<"Unesi vrijednosti za brojioce i imenioce razlomaka"<<endl;
    cin>>b>>i;
    razlomak r1(b,i);
```

```
    cin>>b>>i;
    razlomak r2(b,i);

    razlomak pom;
    pom=r1+r2;
    pom.stampaj();

    r1++;
    r1.stampaj();

    pom=r1++;
    pom.stampaj();

    ++r1;
    r1.stampaj();

    r1+=r2;
    r1.stampaj();

    r1+=(r2++);
    r1.stampaj();

    pom=r1+r2+5;
    pom.stampaj();
    (5+r2).stampaj();
}
```

2. Realizovati klasu **student** koja sadrži podatke o imenu i prezimenu studenta (jedan niz karaktera), godini studija (cijeli broj) i prosječnoj ocjeni (realni broj), kao i statičku promjenljivu koja sadrži informaciju o ukupnom broju studenata. Klasa sadrži odgovarajuće konstruktore i destruktor, preklopljene operatore dodjele, prefiksnog i postfiksnog inkrementiranja (inkrementiranje povećava godinu studija za jedan).

   Za veću ocjenu
   Potrebno je realizovati prijateljsku funkciju, koja za proslijeđeni skup studenata treba da odredi niz studenata sa najvećom prosječnom ocjenom na svakoj godini studija i da odštampa ime, godinu studija i prosječnu ocjenu studenata rezultujućeg niza.

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

class Student
{
   private:
          char *ime;
          int gds;
          float pros;
   public:
          Student(){ime=0; gds=0; pros=0; ukupno++;}
          Student(char *, int, float);
          Student(const Student &);
          ~Student(){delete []ime; ime=0; ukupno--;}
          Student & operator=(const Student &);
          Student & operator++();
          Student operator++(int);
          friend void pretrazi(Student *, int);
          void stampaj(){cout<<ime<<" "<<gds<<" "<<pros<<endl;}
          static int ukupno;
};

int Student::ukupno=0;
```

```
Student::Student(char *a, int b, float c):gds(b), pros(c)
{
    ime = new char[strlen(a)+1];
    strcpy(ime,a);
    ukupno++;
}

Student::Student(const Student & a):gds(a.gds), pros(a.pros)
{
    ime = new char[strlen(a.ime)+1];
    strcpy(ime, a.ime);
    ukupno++;
}

Student & Student::operator=(const Student &a)
{
    // ako se razlikuju memorijske adrese objekta koji je pozvao funkciju i objekta koji je argument
    // to znači da se dodjela vrijednosti (operator =) izvršava nad dva različita objekta u memoriji
    if(this != &a)
    {
           gds  = a.gds;   pros = a.pros;
           delete []ime;   ime = new char[strlen(a.ime)+1];   strcpy(ime, a.ime);
    }
    return *this;
}

//prefiksno inkrementiranje
Student & Student::operator++()
{
    gds++;
    return *this;
}

//postfiksno inkrementiranje
Student Student::operator++(int)
{
    Student pom(*this);        //kreiramo kopiju objekta kojeg inkrementiramo
    gds++;                     //inkrementiramo vrijednost originalnog objekta
    return pom;                //vraćamo vrijednost prije inkrementiranja (vrijednost kopije)
}

void pretrazi(Student *grupa, int duz)
{
    Student pom[5];
    for(int i=0; i<5; i++)
           pom[i]=Student(" ", 0, 0);

    for(int i=0; i<duz; i++)
    {
           if(grupa[i].gds==1 && grupa[i].pros > pom[0].pros)
                  pom[0]=grupa[i];
           else if(grupa[i].gds==2 && grupa[i].pros > pom[1].pros)
                  pom[1]=grupa[i];
           else if(grupa[i].gds==3 && grupa[i].pros > pom[2].pros)
                  pom[2]=grupa[i];
           else if(grupa[i].gds==4 && grupa[i].pros > pom[3].pros)
                  pom[3]=grupa[i];
           else if(grupa[i].gds==5 && grupa[i].pros > pom[4].pros)
                  pom[4]=grupa[i];
    }

    for(int i=0; i<5; i++)
           pom[i].stampaj();
}
```

```
int main()
{
    Student niz[5];
    char ime[20];
    float prosjek;
    int godina, n;

    Student a("Marko Markovic", 2, 8.55);
    Student b;
    //primjer poziva operatora dodjele (operator=)
    b=a;
    b.stampaj();

    cout<<"Unesite duzinu niza:"<<endl; cin>>n;
    cout<<"Unesite podatke za studente:"<<endl;
    for(int i=0; i<n; i++)
    {
           cin>>ime>>godina>>prosjek;
           niz[i]=Student(ime, godina, prosjek);
    }

    pretrazi(niz, n);
}
```