ZADAĆA 6: PREDLOŠCI

Svaki zadatak nosi određeni broj bodova. Svi studenti koji predaju zadaće trebaju pristupiti na ScriptRunner5 sustav (https://mathos.scriptrunner.carnet.hr/) i tamo objavljivati svoje zadaće u mapi ZADACE->ZADACA_5.

Svi studenti koji predaju zadaće moraju biti nazočni na vježbama.

ZADATAK 1: (20)

Napišite C++ program koji će implementirati predložak (*template*) `sort` za sortiranje polja podataka veličine N. Tip podataka je parametar predloška.

Dan je prototip predloška `sort` i klasa `Pair`:

```
template<class T>
void sort(T *x,int N);


class Pair
{
public:
        Pair();
        Pair(float x,float y);
        Pair(const Pair &T);

        bool operator<(const Pair &) const;
        bool operator>(const Pair &) const;
};
```

klasa `Pair` sadrži:

- Pretpostavljeni konstruktor, konstruktor koji uzima 2 parametra i konstruktor kopiranja.
- `bool operator<(const Pair &P) const` predstavlja implementaciju relacije *'biti manji po x-komponenti'*
- Testirajte sortiranje na tipu podataka `int`, `float` i na tipu podataka klase `Pair`.

ZADATAK 2 (30):

Napravite implementaciju predloška za n-terac `Tuple` klasu koja je dana sljedećim prototipom:

```
template<class T, int n>
class Tuple
{
public:
        T *podaci;
        Tuple();
```

```
        Tuple(T *input);
        ~Tuple();

        bool operator<(const Tuple &) const;
        bool operator>(const Tuple &) const;

};
```

Gdje klasa sadrži:

- Dinamički alocirano polje `podaci` od n elemenata (preuzeto od predloška)
- Pretpostavljeni konstruktor i konstruktor koji uzima pokazivač na polje `input` i uzima prvih n elemenata polja.
- Sadrži relaciju *'biti leksikografski manji'* odnosno *'biti leksikografski veći'* kao bool operator:

  `bool operator<(const Tuple &) const; bool operator>(const Tuple&) const;`

➔ Testirajte program tako da definirate 10 objekta klase Tuple veličine 5 sa slučajno generiranim elemenatima te primijenite sortiranje na njima koje je implementirano u prvom zadatku.

NAPOMENA:

Za x kažemo da je veći ili jednak od y u leksikografskom smislu ukoliko postoji $k \in 1,2,..,n$ tako da je $x_1 = y_1, x_2 = y_2, \dots, x_k \leq y_k$ i pišemo $(x_1, x_2, \dots, x_n) \leq (y_1, y_2, \dots, y_n)$. Uređaj je potpun, tj. $x \leq y$ ili $y \leq x$ za bilo koji n-terac x, y.

ZADATAK 3 (20+30):

U *zadaći 2* kao zadatak je bila implementacija vektora kao dinamičko polje s pripadnim prototipom:

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


template<class T>
class vector
{
    public:
      vector(){}
      vector(size_t size);
      size_t size() const; //size_t = unsigned int
      size_t capacity() const;

      // dodavanje i brisanje elemenata s kraja
      void push_back(T x);
      T pop_back();
      void resize_to_fit();
      void clear();

    // u ovom dijelu potrebno je napraviti izmjene u potpisu metodu kako bi radilo kao
metode predloska

     // operatori pridruzivanja
     vector& operator   = (const vector<T>&); // pridruzivanje
     vector& operator  += (const vector<T>&); // zbrajanje + pridruzivanje
     vector& operator  -= (const vector<T>&); // oduzimanje + pridruzivanje
     vector& operator  *= (T lambda);         // mnozenje sa skalarom + pridruzivanje

    // aritemticki operatori
     vector operator  + (const vector<T>&);   // zbrajanje vektora
     vector operator  - (const vector<T>&);   // oduzimanje vektora
     vector operator  * (T lamba);   // oduzimanje vektora

    // usporedba
    vector operator == (const vector<T>&); // == u smislu relacije  ekvivalencije
    vector operator != (const vector<T>&);

    // kraj izmjena
};
```

Funkcionalnost klase vektor kao dinamičko polje objašnjeno je u *zadaći 2*. Učinite sljedeće:

1. Predefinirajte prototip klase na odgovarajućim mjestima kako bi predložak bio funkcionalan.
2. Implementirajte tako predefinirane metode i testirajte sve operatore u glavnom programu na vektoru s tipom `double` i tipom `Complex` (programski kod dostupan s ranijih vježbi).

Glavni program:

```
int main()
{
    vector<double> L1;
    vector<Complex> L2; // Pair je tip definiran u prvom zadatku


    // ...


    return 1;


}
```