Zadatak:

Napišite sučelje i implementaciju za strukturu **bankomati**. Struktura **bankomati** sadrži listu bakomata. Svaki bankomat je jedinstveno određen svojim imenom. Također, svaki bankomat na račun stavlja ili krade fiksni broj kuna. Bankomati su poredani onim redoslijedom kojim ih dodajemo. Sučelje za strukturu spremite u datoteku **bankomati.h**, a implementaciju u datoteku **bankomati.cpp**.

Struktura bankomati

```
struct bankomati
{
    list<pair<string, int> > listaBank;

    list<pair<string, int> > vratiListu(void) {return listaBank;}
    void dodaj(string bankomatID, int novac);
    void izbrisi(string bakomatID);
    string najbolji(void);
    int zarada(string pocetak, string kraj);
    list<pair<string, int> > nadjiBankomat(string substr);
    set<string> viseOd(int novac);
    pair<string, string> maxZarada(void);
};
```

Funkcije članice

- **void dodaj(string bankomatID, int novac)**
  Dodaje novi bankomat (imena **bankomatID** *koji stavlja/krade* **novac** *kuna) na kraj liste* **listaBank**. *Ako je* **novac** *pozitivan onda bankomat stavlja novac na račun, a ako je negativan onda krade s računa.*

- **void izbrisi(string bakomatID)**
  Izbacuje traženi bankomat iz liste. Ostali bankomati ostaju poredani prema redoslijedu koji su imali i prije izvršavanja funkcije.

- **string najbolji(void)**
  Vraća ime bankomata koji na račun stavlja najviše kuna. Ako ima više istih onda vraća onog koji se nalazi prvi po redoslijedu.
  .
- **int zarada(string pocetak, string kraj)**
  Vraća zaradu (ili gubitak) na računu koju dobijemo ako krenemo redom od bankomata s imenom **pocetak** *i završimo s bankomatom imena* **kraj** *(bankomat* **pocetak** *se uvijek nalazi prije (ili je jednak) bankomata* **kraj**).

- **list<pair<string, int> > nadjiBankomat(string substr)**
  Vraća listu bankomata čije ime sadrži podriječ **substr**. *Bankomati u novoj listi zadržavaju isti poredak.*

- **set<string> viseOd(int novac)**
  Vraća skup bankomata (samo njihova imena) koji na račun stavljaju strogo više od **novac** *(uvije pozitivno) kuna.*

- **pair<string, string> maxZarada(void)**
  Vraća par imena bankomata takav da je suma novca koju dobijemo na računu kad krenemo redom od prvog bankomata (first u vraćenom paru) do drugog bankomata (second u vraćenom paru) (ne smije se preskakati bankomate između) maksimalna moguća. Ako postoji više takvih parova tada uzimamo onog kod kojeg se prvi bankomat nalazi bliže početku liste, a ako ima i takvih više tada uzimamo onaj par koji sadrži najmanje bankomata između. Možete pretpostaviti da uvijek postoji makar jedan bankomat koji stavlja novac na račun. Također, prvi bankomat se mora nalaziti prije (ili biti jednak) drugog bankomata.

Primjer klijentskog programa

```
#include <iostream>
#include <map>
#include <set>
#include <list>
#include <string>
#include <utility>
#include "bankomati.h"

using namespace std;

void ispisiListu(list<pair<string, int> > L)
{
    list<pair<string, int> >::iterator it;

    for(it = L.begin(); it != L.end(); it++)
        cout << it->first << " " << it->second << " ";
}

void ispisiSkup(set<string> S)
{
    set<string>::iterator it;

    for(it = S.begin(); it != S.end(); it++)
        cout << *it << " " ;
}

int main(void)
{
    bankomati B;
    B.dodaj("otp4", 3); B.dodaj("zaba3", -20); B.dodaj("pbz1", 10);
    B.dodaj("zaba", -2); B.dodaj("credo", 5); B.dodaj("splitska", 5);
    B.dodaj("bks", 4); B.dodaj("credo1", -2); B.dodaj("erste2", 2);
    B.dodaj("pbz3", -30); B.dodaj("12pbz",10); B.dodaj("erste", -3);
    B.dodaj("otp", 10); B.izbrisi("credo");

    ispisiListu(B.vratiListu());
    //    otp4 3 zaba3 -20 pbz1 10 zaba -2 splitska 5 bks 4 credo1 -2 erste2 2 pbz3 -30 12 pbz 10 erste -3
otp 10

    cout << endl;
    cout << B.najbolji() << endl;
    //    pbz1

    cout << B.zarada("bks", "erste") << endl;
    //    -19

    ispisiListu(B.nadjiBankomat("pbz"));
    //    pbz1 10 pbz3 -30 12pbz 10

    cout << endl;
    ispisiSkup(B.viseOd(4));
    //    12pbz otp pbz1 splitska

    cout << endl;
    pair<string, string> p;
    p = B.maxZarada();

    cout << p.first << " " << p.second << " " << endl;
    //    pbz1 bks

    return 0;
}
```

Opće napomene

- Struktura, funkcije i datoteke koje šaljete moraju se zvati točno onako kako je zadano u zadatku. Pazite na mala i velika slova!
- Trebate poslati samo sučelje i implementaciju. U datotekama koje šaljete ne smije se nalaziti funkcija main()!
- nijedna funkcija ne smije ništa učitavati s tipkovnice ili neke datoteke, niti išta ispisivati na ekran ili u neku datoteku.
- Svaki od main-ova pomoću kojih testiramo ispravnost vašeg programa neće pozivati sve gore navedene funkcije. Stoga, ako neku od funkcija ne znate napisati ipak možete dobiti koji bod (u tom slučaju tu funkciju nemojte navesti niti u .h niti u .cpp datoteci ili napravite neku trivijalnu implementaciju).

Ispravnost implementacija koje napišete bit će provjerena tako da ćemo mi napisati razne klijentske programe koji će deklarirati nekoliko varijabli zadane strukture, i na njima pozivati funkcije koje ste trebali napisati. Ako se poslani programi ne budu uspješno povezivali (linkali) s našim klijentskim programima, smatrat će se neispravnima.

Neki klijentski programi provjeravat će samo neke jednostavnije funkcije, dok će neki provjeravati sve funkcije koje trebate napisati. Provjera je potpuno automatska, tako da je od presudne važnosti da se pridržavate specifikacije. Nepridržavanje lako može uzrokovati osvojenih 0 bodova iz zadaće!

Naravno, za provjeru radi li implementacija prije nego što je pošaljete, preporučuje se da je testirate pomoću nekog klijentskog programa. No taj klijentski program ne šaljete!

Za sva pitanja vezana uz ovu zadaću javite se asistentu Vujčiću na jvujcic+rp1@gmail.com.