

Napišite sučelje i implementaciju za strukture `avion` i `zracniprostor`. Struktura `avion` predstavlja avion u zračnom prostoru. Svaki avion ima naziv (`string`), donju lijevu točku te gornju desnu točku (zadani nenegativnim cjelobrojnim `x` i `y` koordinatama). Avion zamišljamo da je pravokutnog oblika. Struktura `zracniprostor` predstavlja zračni prostor koja sadrži kolekciju podataka tipa `avion`. Ukupan broj aviona na zračnom prostoru nikad neće biti veći od 100. Avioni moraju u cijelosti biti sadržani unutar zračnog prostora i smiju se preklapati. Sučelje (deklaraciju) za obje strukture spremite u datoteku `zracniprostor.h`, a implementaciju u datoteku `zracniprostor.cpp`. Strukture moraju imati sljedeće elemente (smijete dodati i elemente po želji, a neke ćete i morati dodati):

Struktura i funkcije	Opis
avion	
<code>avion(string naziv, int x1, int y1, int x2, int y2)</code>	Stvara avion (konstruktor). Možete pretpostaviti da su koordinate točaka (<code>x1</code> , <code>y1</code>) i (<code>x2</code> , <code>y2</code>) dobro zadane tako da prva čini donju lijevu točku aviona, a druga gornju desnu (tj. <code>x1 <= x2</code> , <code>y1 <= y2</code>).
<code>string getNaziv()</code>	Vraća naziv aviona.
<code>int brojTocaka()</code>	Vraća broj točaka koje avion obuhvaća.
zracniprostor	
<code>zracniprostor(int x, int y)</code>	Stvara zračni prostor koji u sebi nema aviona. Donja lijeva točka zračnog prostora ima koordinate <code>(0, 0)</code> , a gornja desna <code>(x, y)</code> .
<code>int brojAviona()</code>	Vraća broj aviona
<code>avion getAvion(int index)</code>	<code>index</code> je broj između 0 i <code>brojAviona-1</code> . Vraća određeni avion prema redoslijedu dodavanja u kolekciju.
<code>bool dodajAvion(avion a)</code>	Dodaje avion kolekciji, ukoliko u kolekciji ima manje od 100 aviona, i ako su koordinate aviona takve da se u cijelosti nalazi unutar zračnog prostora i vraća <code>true</code> , a <code>false</code> inače (pogledajte primjer dolje).
<code>bool makniAvion(string naziv)</code>	Uklanja avion s nazivom <code>naziv</code> iz kolekcije, ako postoji u kolekciji, i vraća <code>true</code> , a <code>false</code> inače. Možete pretpostaviti da ne postoje dva aviona s istim nazivom.
<code>zracniprostor sadrzi(int x, int y)</code>	Vraća zračni prostor iste veličine kao početni koji se sastoji samo od aviona koji sadrže točku s koordinatama <code>(x, y)</code> . Avioni trebaju

	biti u onom redoslijedu u kakvom su bili u početnom zračnom prostoru.
int promijeniVelicinu(int x1, int y1, int x2, int y2)	Mijenja veličinu zračnog prostora tako da mu donja lijeva točka ima koordinate (x1, y1), a gornja desna (x2, y2). Također treba ukloniti sve avione koji se ne prikazuju u cijelosti unutar novog okvira. Funkcija treba vratiti broj uklonjenih aviona.

Primjer klijentskog programa

```
#include "zracniprostor.h"
#include <iostream>

using namespace std;

void ispisiAvione(zracniprostor z) {
    for (int i = 0; i < z.brojAviona(); ++i) {
        avion a = z.getAvion(i);
        cout << a.getNaziv() << ": " << a.brojTocaka() << endl;
    }
}

int main() {
    zracniprostor z(100, 100);
    cout << z.dodajAvion(avion("Prvi", 0, 0, 0, 0)); // 1
    cout << z.dodajAvion(avion("Drugi", 10, 10, 20, 20)); // 1
    cout << z.dodajAvion(avion("Treci", 80, 15, 90, 30)); // 1
    cout << z.dodajAvion(avion("Cetvrti", 10, 10, 30, 90)); // 1
    cout << z.dodajAvion(avion("Peti", 15, 15, 85, 50)); // 1
    cout << z.dodajAvion(avion("Sesti", 1, 1, 101, 99)); // 0 - izvan
    granica zracnog prostora
    cout << endl;

    ispisiAvione(z);
    /*
    Prvi: 1
    Drugi: 121
    Treci: 176
    Cetvrti: 1701
    Peti: 2556
    */

    cout << z.brojAviona() << endl; // 5
    z.makniAvion("Prvi"); // postoji
    z.makniAvion("Sedmi"); // ne postoji
    cout << z.brojAviona() << endl; // 4

    ispisiAvione(z.sadrzi(15, 15));
    /*
    Drugi: 121
    Cetvrti: 1701
    */
}
```

```

    Peti: 2556
    */

    ispisiAvione(z.sadrzi(90, 90)); // nista

    cout << z.promijeniVelicinu(15, 15, 99, 99) << endl; // 2
    ispisiAvione(z);
    /*
    Treci: 176
    Peti: 2556
    */

    cout << z.promijeniVelicinu(10, 10, 85, 50) << endl; // 1
    ispisiAvione(z);
    /*
    Peti: 2556
    */
    cout << z.dodajAvion(avion("Osmi", 9, 9, 80, 40)); // 0 - izvan granica
    cout << z.dodajAvion(avion("Deveti", 10, 10, 85, 50)); // 1
    cout << endl;
    ispisiAvione(z);
    /*
    Peti: 2556
    Deveti: 3116
    */
    return 0;
}

```

Opće napomene

- Struktura, funkcije i datoteke koje šalžete 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 šalžete *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 implementacijâ 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 šalžete!