Zadaća 4.

Ova zadaća nosi ukupno 3,6 poena, pri čemu prvi i treći zadatak nose po 1,5 poen, a drugi zadatak nosi 0,6 poena. Sva tri zadatka se mogu uraditi na osnovu gradiva sa prvih 10 predavanja i pretpostavljenog predznanja iz predmeta "Osnove računarstva". Rok za predaju ove zadaće je petak, 25. V 2013. (do kraja dana) i ne može se produžiti. Zadaće se predaju putem Zamgera.

1. Potrebno je napraviti program koji vrši najavu letova na aerodromskom displeju. Program treba najavljivati sve letove u toku dana, kao i eventualna kašnjenja u polijetanjima. Za tu svrhu, u programu je potrebno razviti dvije klase nazvane "`Let`" i "`Letovi`". Klasa "`Let`" vodi evidenciju o jednom letu, dok klasa "`Letovi`" vodi evidenciju o svim letovima u toku dana. Klasa "`Let`" sadrži privatne atribute koji redom predstavljaju naziv odredišta, oznaku leta (npr. "JFK 327"), broj izlazne kapije (cijeli broj), vrijeme polijetanja (sati i minute), trajanje leta u minutama, kao i informaciju o eventualnom kašnjenju (također u minutama). Naziv odredišta i oznaka leta čuvaju se kao nizovi znakova (ne kao dinamički stringovi), sa maksimalnim dužinama od 30 i 10 znakova respektivno. Ove maksimalne dužine treba definirati kao statičke konstantne atribute. Interfejs klase "`Let`" je sljedeći:

```
Let(const char odrediste[], const char oznaka_leta[], int kapija,
  int sat_polaska, int minute_polaska, int trajanje_leta);
void PostaviKasnjenje(int kasnjenje);
bool DaLiKasni() const;
int DajTrajanjeLeta() const;
void OcekivanoVrijemePolijetanja(int &sati, int &minute) const;
void OcekivanoVrijemeSlijetanja(int &sati, int &minute) const;
void Ispisi() const;
```

Konstruktor inicijalizira sve atribute klase u skladu sa vrijednostima zadanim parametrima konstruktora, osim informacije o eventualnom kašnjenju, koja se automatski inicijalizira na 0. Konstruktor treba da baci izuzetak ukoliko bilo koji od parametara ima besmislene vrijednosti, uključujući i situaciju kada se za naziv odredišta ili za oznaku leta daju predugački nizovi znakova. Metoda "`PostaviKasnjenje`" postavlja informaciju o eventualnom kašnjenju na vrijednost zadanu parametrom, dok metoda "`DaLiKasni`" omogućava da se sazna da li odgovarajući let kasni ili ne (metoda vraća logičku vrijednost "**`true`**" u slučaju kašnjenja, a logičku vrijednost "**`false`**" u suprotnom slučaju). Metoda "`DajTrajanjeLeta`" daje kao rezultat trajanje leta u minutama, dok metode "`OcekivanoVrijemePolijetanja`" i "`OcekivanoVrijemeSlijetanja`" omogućavaju da se sazna očekivano vrijeme polaska odnosno dolaska kada se uračuna iznos kašnjenja u odnosu na predviđeno vrijeme polaska/dolaska. Obje metode treba da smjeste sate i minute očekivanog vremena polaska/dolaska u dva cjelobrojna parametra koji joj se prosljeđuju. Konačno, metoda "`Ispisi`" treba da podrži ispis objekata tipa "`Let`" na ekran. U slučaju da se radi o letu bez kašnjenja, ispis bi trebao da izgleda poput sljedećeg:

```
JFK 327   New York               12:50   19:30     5
```

Ovi podaci predstavljaju redom oznaku leta, naziv odredišta, vrijeme polijetanja, očekivano vrijeme slijetanja i broj izlazne kapije. Predvidite širinu od 10 mjesta za ispis šifre leta, 20 mjesta za naizv odredišta, po 8 mjesta za vrijeme polijetanja i slijetanja, odnosno 6 mjesta za ispis broja izlazne kapije. Oznaka leta i naziv odredišta trebaju biti poravnati ulijevo u prostoru predviđenom za ispis, dok preostali podaci trebaju biti poravnati udesno. Sati i minute se uvijek ispisuju kao dvocifreni brojevi, tako da se recimo 12 sati i 9 minuta ispisuje kao "12:09" a ne kao "12:9".

U slucaju da se radi o letu koji kasni, ispis bi trebao da izgleda poput sljedećeg:

```
IST 932   Istanbul               15:50 (Planirano 15:30, Kasni 20 min)
```

Oznaku leta, naziv odredišta i vrijeme polijetanja formatirajte kao i u prethodnom slučaju, a dopunske informacije pišite u produžetku iza vremena polijetanja.

Klasa "Letovi" čuva podatke o letovima u dinamički alociranim objektima tipa "Let", kojima se opet pristupa preko dinamički alociranog niza pokazivača na takve objekte. Pored pokazivača koji se koristi za pristup ovom nizu, klasa još sadrži i privatne atribute koji predstavljaju broj registriranih letova u toku dana, te maksimalni broj letova koji se mogu registrirati (ovaj atribut mora biti konstantan). Interfejs ove klase je sljedeći:

```
explicit Let(int max_broj_letova);
~Letovi();
Letovi(const Letovi &letovi);
Letovi &operator =(const Letovi &letovi);
void RegistrirajLet(const char odrediste[], const char oznaka_leta[],
  int kapija, int sat_polaska, int minute_polaska, int trajanje_leta);
void RegistrirajLet(Let *let);
int DajBrojLetova() const;
int DajBrojLetovaKojiKasne() const;
int DajProsjecnoTrajanjeLetova() const;
Let &DajPrviLet() const;
Let &DajPosljednjiLet() const;
void IsprazniKolekciju();
void Ispisi(int sati, int minute) const;
```

Konstruktor vrši odgovarajuće konfigurisanje memorije za prihvatanje podataka o letovima, a njegov parametar predstavlja maksimalan broj letova koji se mogu registrirati. Destruktor oslobađa svu memoriju koja je zauzeta tokom života objekta, dok konstruktor kopije i preklopljeni operator dodjele omogućavaju bezbjedno kopiranje i međusobno dodjeljivanje objekata tipa "Letovi" korištenjem strategije dubokog kopiranja. Pri tome, dodjelu treba podržati samo ukoliko su izvorišni i odredišni objekat istih kapaciteta. U suprotnom treba baciti izuzetak. Metoda "RegistrirajLet" podržana je u dvije verzije. Prva verzija kreira novi let u skladu sa parametrima (koji su identični kao kod konstruktora klase "Let") i registrira ga u kolekciji, dok druga verzija prosto kao parametar prihvata pokazivač na objekat tipa "Let" (za koji pretpostavljamo da je već na neki način kreiran) i registira ga u kolekciji. U oba slučaja, treba baciti izuzetak u slučaju da je dostignut maksimalan broj letova koji se mogu registrirati. Metode "DajBrojLetova", "DajBrojLetovaKojiKasne" i "DajProsjecnoTrajanjeLetova" daju respektivno ukupan broj registriranih letova, broj letova koji kasne, te prosječno trajanje svih registriranih letova u minutama. Metode "DajPrviLet" i "DajPosljednjiLet" daju kao rezultat prvi i posljednji let (tj. odgovarajući objekat tipa "Let") u toku dana (uključujući i eventualna kašnjenja). Obje metode vraćaju kao rezultat referencu da se izbjegne nepotrebno kopiranje objekata. Metoda "IsprazniKolekciju" uklanja sve registrirane letove, tako da je nakon poziva ove metode kolekcija u identičnom stanju kakva je bila neposredno nakon kreiranja. Konačno, metoda "Ispisi" ispisuje informacije o svim letovima, počev od zadanog vremena do kraja dana, sortiranu po očekivanim vremenima polijetanja (za sortiranje iskoristite funkciju "sort", uz pogodno definiranu funkciju kriterija koja treba biti privatna statička funkcija članica klase). Ispis pojedinačnih letova vrši se prostim pozivom metode "Ispisi" nad objektima tipa "Let" pohranjenim u kolekciji.

Sve metode implementirajte izvan deklaracije klase, osim trivijalnih metoda koje trebate implementirati direktno unutar deklaracije klase. Obavezno napišite i testni program u kojem ćete testirati sve elemente napisanih klasa. Posebno se trebate uvjeriti da konstruktor kopije i preklopljeni operator dodjele rade ispravno, kao i da ni u kom slučaju ne dolazi do curenja memorije.

2. Izmijenite program "ucenici_obp.cpp" priložen uz Predavanje 10 u skladu sa sljedećim zahtjevima:
   a) Atributi "ime" i "prezime" u klasi "Ucenik" sada će biti tipa "string" umjesto niza znakova. Kao posljedica toga, konstruktor klase "Ucenik" treba također da se promijeni, tim prije što više nije potrebno provjeravati da li je predena maksimalna dozvoljena dužina imena odnosno prezimena.

b) Ocjene se više neće čuvati u nizu cijelih brojeva, nego u vektoru cijelih brojeva (njegov kapacitet i dalje treba da bude "`BrojPredmeta`" elemenata).

c) U klasi "`Razred`", za evidenciju dinamički alociranih objekata tipa "`Ucenik`" umjesto dinamički alociranog niza pokazivača treba koristiti vektor čiji su elementi pokazivači na objekte tipa "`Ucenik`". Pri tome, atributi koji čuvaju broj upisanih učenika i kapacitet razreda više neće biti potrebni. Zaista, broj upisanih učenika se može saznati testiranjem trenutne veličine vektora, dok kapacitet razreda više nije potrebno zadavati, s obzirom da vektor može u toku rada po volji povećavati svoju veličinu. Samim tim, konstruktoru više neće biti potreban parametar (zahvaljujući fleksibilnosti vektora, moći će se upisati onoliko učenika koliko želimo, bez potrebe da unaprijed specificiramo njihov maksimalan broj). Isto tako, metoda za evidentiranje novog učenika više ne treba provjeravati da li je dostignut maksimalan broj studenata, s obzirom da ograničenje na maksimalan broj učenika više ne postoji. Destruktor će i dalje biti potreban (da oslobodi sve dinamički alocirane objekte tipa "`Ucenik`" nakon što objekat tipa "`Razred`" prestane postojati).

d) Potrebno je u klasu "`Razred`" dodati konstruktor kopije i preklopljeni operator dodjele, s ciljem da se omogući bezbjedno kopiranje i međusobno dodjeljivanje objekata tipa "`Razred`", zasnovano na strategiji dubokog kopiranja. Naime, treba primijetiti da su dinamički alocirani objekti tipa "`Ucenik`" u vlasništvu klase "`Razred`", iako nisu njen sastavni dio (razmislite dobro šta konstruktor kopije i preklopljeni operator dodjele tačno trebaju kopirati).

Uvjerite se da uz ovakve modifikacije program i dalje radi ispravno. Posebno se trebate uvjeriti da konstruktor kopije i preklopljeni operator dodjele rade ispravno (za tu svrhu ćete morati ponešto dodati i u glavni program), i da nigdje ni pod kakvim okolnostima ne dolazi do curenja memorije.

3. Implementirajte jednostavan program koji olakšava vođenje administrativnih poslova u nekoj videoteci. Program se zasniva na tri klase "`Korisnik`", "`Film`" i "`Videoteka`". Primjerci ovih klasa modeliraju respektivno korisnike videoteke, filmove u videoteci te samu videoteku (ova posljednja klasa je tzv. singlton klasa, što znači da će u čitavom programu biti samo jedan primjerak te klase).

   Klasa "`Korisnik`" sadrži privatne atribute koji čuvaju informacije o članskom broju korisnika, njegovom imenu i prezimenu (oboje u istom atributu), adresi, te broju telefona (svi ovi atributi su tipa "`string`", osim članskog broja koji je cijeli broj). Interfejs klase sadrži konstruktor sa 4 parametara koji inicijalizira sve atribute na vrijednosti zadane parametrima (redoslijed parametara je isti kao i redoslijed gore navedenih atributa), zatim trivijalne pristupne metode "`DajClanskiBroj`", "`DajImeIPrezime`", "`DajAdresu`" i "`DajTelefon`" koje prosto vraćaju vrijednosti odgovarajućih atributa, te metodu "`Ispisi`" koja ispisuje podatke o korisniku na ekran. Ispis treba da izgleda ovako:

   `Clanski broj:` ***članski_broj***
   `Ime i prezime:` ***ime_i_prezime***
   `Adresa:` ***adresa***
   `Telefon:` ***broj_telefona***

   Klasa "`Film`" sadrži privatne atribute koji čuvaju informacije o evidencijskom broju video trake ili CD-a na kojem je film, zatim da li je film na video traci ili DVD-u, nazivu filma, žanru i godini produkcije, kao i informaciju o eventualnom zaduženju filma. Evidencijski broj i godina izdavanja su cijeli brojevi, informacija da li je film na video traci ili DVD-u je logičkog tipa, informacija o zaduženju čuva se kao pokazivač na korisnika koji je zadužio film odnosno 0 ukoliko film nije zadužena, dok su ostali atributi stringovnog tipa. Interfejs klase sadrži konstruktor sa 5 parametara koji inicijalizira sve atribute klase na vrijednosti zadane parametrima (redoslijed parametara je isti kao i redoslijed gore navedenih atributa), osim informacije o zaduženju koja se postavlja tako da signalizira da film nije zadužen. Pored konstruktora, interfejs klase sadrži trivijalne pristupne metode "`DajEvidencijskiBroj`", "`DajNaziv`", "`DajZanr`", "`DajGodinuProdukcije`" i "`DajKodKogaJe`" koje vraćaju vrijednosti odgovarajućih atributa, metodu "`DaLiJeDVD`" koja vraća informaciju da li je film na DVD-u ili ne, te metode

“ZaduziFilm”, “RazduziFilm”, “DaLiJeZaduzen” i “Ispisi”. Metoda “ZaduziFilm” vrši zaduživanje filma, a parametar joj je referenca na korisnkia koji zadužuje film. Metoda “RazduziFilm” nema parametara, a vrši razduživanje filma. Metoda “DaLiJeZaduzen” također nema parametara i prosto vraća informaciju da li je film zadužen ili ne, dok metoda “Ispisi” vrši ispis podataka o filmu na ekran. Ispis treba da izgleda ovako:

Evidencijski broj: ***evidencijski_broj***
Medij: Video traka (*ili* DVD *ako je film na DVD-u*)
Naziv filma: ***naziv_ filma***
Zanr: ***žanr***
Godina produkcije: ***godina_produkcije***

Klasa “Videoteka” objedinjuje u sebi podatke o svim korisnicima videoteke, kao i o svim filmovima. Podaci o svakom korisniku odnosno svakom filmu čuvaju se u dinamički alociranim varijablama, kojima se pristupa pomoću dva vektora čiji su elementi pokazivači na korisnike (tj. na objekte tipa “Korisnik”) odnosno pokazivači na fimove (tj. na objekte tipa “Film”). Ova dva vektora su ujedno i jedini atributi klase “Videoteka”. Broj elemenata ovih vektora nije unaprijed određen, nego raste po potrebi sa dodavanjem novih korisnika odnosno filmova u evidenciju. Klasi nije potreban ručno pisani konstruktor (s obzirom da će atributi koji su vektori svakako biti automatski inicijalizirani na prazne vektore), ali treba imati destruktor koji će po završetku postojanja objekta tipa “Videoteka” obrisati sve korisnike i filmove koji su evidentirani u njoj (tj. koji su u njenom vlasništvu). Kopiranje i međusobno dodjeljivanje objekata tipa “Videoteka” treba zabraniti. Pored ovih elemenata, klasa “Videoteka” sadrži i nekoliko metoda. Metoda “RegistrirajNovogKorisnika” prima kao parametre podatke o korisniku (ovi parametri su isti kao parametri konstruktora klase “Korisnik”), nakon čega kreira odgovarajući objekat tipa “Korisnik” i upisuje ga u evidenciju. U slučaju da već postoji korisnik sa istim članskim brojem, metoda baca izuzetak. Metoda “RegistrirajNoviFilm” radi analognu stvar, ali za filmove (tj. objekte tipa “Film”). Metoda “NadjiKorisnika” prima kao parametar članski broj korisnika i vraća kao rezultat referencu na korisnika sa zadanim članskim brojem, ili baca izuzetak ako takvog korisnika nema. Metoda “NadjiFilm” radi analognu stvar, ali za filmove (parametar je evidencijski broj). Metoda “IzlistajKorisnike” ispisuje podatke o svim registriranim korisnicima, jedan za drugim, sa po jednim praznim redom između svaka dva korisnika, dok metoda “IzlistajFilmove” tu istu stvar radi za filmove. Pri tome, ukoliko je film zadužen, iza standardnih podataka koji se ispisuju za film treba ispisati i tekst “Zadužen kod korisnika: ” iza čega slijedi korisnički broj korisnika koji je zadužio film. Metoda “ZaduziFilm” prima kao parametre evidencijski broj filma, kao i članski broj korisnika koji zadužuje film. Ona vrši registraciju da je navedeni film zadužen kod navedenog korisnika. U slučaju da je evidencijski broj filma ili članski broj korisnika neispravan, ili ukoliko je film već zadužen, metoda baca izuzetak. Metoda “RazduziFilm” prima kao parametar evidencijski broj filma. Ona registrira da film više nije zadužen. U slučaju da je evidencijski broj neispravan, ili ukoliko film uopće nije zadužen, metoda baca izuzetak. Konačno, metoda “PrikaziZaduzenja” prima kao parametar članski broj korisnika, a vrši ispis podataka o svim filmovima koje je zadužio navedeni korisnik (na isti način kao u metodi “IzlistajKorisnike”). U slučaju da korisnik nije zadužio niti jedan film, metoda ispisuje tekst “Korisnik nema zaduženja!”, dok u slučaju da je članski broj neispravan, metoda baca izuzetak.

Napisane klase demonstrirajte u testnom programu u kojem se korisniku prikazuje meni koji mu nudi da odabere neku od mogućnosti koje su podržane u klasi “Videoteka”. Nakon izbora opcije, sa tastature treba unijeti eventualne podatke neophodne za izvršavanje te opcije, te prikazati rezultate njenog izvršenja. Ovo se sve izvodi u petlji dok korisnik programa ne izabere da želi završiti sa radom.