Zadaci za Tutorijal 14.

NAPOMENA: Ovaj tutorijal je predviđen nastavnim planom i programom, ali kako za njega nema prostora u nastavnom kalendaru da se fizički održi, on će se održati kao "virtualan". Ovaj tutorijal je stimulacione prirode: njegova (tačna) izrada donosi dodatne poene (**2 poena** maksimalno, uz kviz koji nosi još **0.5 poena**). Postoje striktna pravila vezana za ovaj tutorijal, koja su prikazana ispod, uz neke sugestije:

- Tutorijal se radi "od kuće" i šalje se kao zadaća, njegova izrada će biti moguća od 9:00 (kada će zadaci biti i objavljeni) do 17:00, na dan 18. VI 2016. godine.
- Tutorijal imaju pravo raditi svi studenti (pa i oni koji nisu poništili prisustvo tj. "prenijeli" su poene).
- Tutorijal se radi isključivo na C9 razvojnom okruženju. Nije dozvoljena izrada tutorijala u nekom drugom orkuženju (IDE-u) i kopiranje istog na C9!
- Nije dozvoljeno kopiranje teksta prilikom izrade tutorijala! Maksimalno je dozvoljeno kopirati do 80 karaktera (jedna linija koda) istovremeno! Ovo pravilo vrijedi za prva dva zadatka.
- Vršiće se provjera na prepisivanje, isto kao i za zadaće, samo će biti restriktivnija (zbog iznad navedenih pravila). Svaki pokušaj prepisivanja se kažnjava davanjem **0 poena** na prepisani zadatak. Pošto je ovo tutorijal stimulacione prirode, te kao takav skroz neobavezan, dodatno se kažnjavaju oni koji su prepisali sa **oduzimanjem 0.5 poena** po zadatku od ukupno osvojenih poena. Da, dobro ste pročitali, (poštena) izrada ovog tutorijala može donijeti dodatna 2 poena, ali Vas može koštati i 2 poena koja su osvojena negdje drugo (zadaće, ispiti, ostali tutorijali).
- Kviz će se raditi tačno u **12.00**. Studenti će imati **3 minute** vremena da se prijave za kviz (tj. do **12.03**), nakon čega prijava nije moguća. Drugim riječima, kviz će biti vidljiv u intervalu od 12.00 do 12.03 (zbog moguće nesaglasnosti u sistemskim satovima na različitim računarima, poželjno je da studenti dođu malo ranije i da vrše "refresh" dok im kviz ne postane vidljiv). Od trenutka prijave, student kao što je uobičajeno ima 3 minuta vremena za rješavanje samog kviza.
- Kviz mogu raditi **samo oni studenti koji su poništili prisustvo**!
- Tutorijal ima 4 zadatka, pri čemu svaki od zadataka nosi po **0.5 poena**.
- Zadaci će biti autotestirani, od čega će biti jedan do dva autotesta po zadatku biti dostupni studentima da sami (auto)testiraju.
- Tema tutorijala je rad sa datotekama, stoga se studentima preporučuje da pogledaju kako se kreira datoteka na C9 razvojnom okruženju i kako se smješta u trenutni direktorij (najlakše vam je ako smjestite datoteku u isti folder gdje je se nalazi i vaš "`main.cpp`" fajl). Datoteke možete kreirati na više načina: File > New, Alt + N, ili klikom na znak "+" poslije prikazanih tabova. Pazite da ispravno odaberete lokaciju datoteke. Najviše testova će testirati upravo rad sa datotekama!
- Imena datoteka na Unixoidnim sistemima (na kojima se, između ostalog, nalazi i C9 i autotester) su *case-sensitive* (dok npr. na Windows operativnim sistemima nisu), tj. pravi se razlika između velikih i malih slova! Obratite pažnju na to!
- Dobro provjerite da li ste funkcije, klase i ostale identifikatore nazvali upravo onako kako je to postavkom rečeno, da niste izostavili neku metodu i da vam je ispis upravo onakav kakav je očekivan!
- Ne podrazumijeva se da datoteke na svome kraju sadrže novi red (mnogi studentski programi se oslanjaju na tu činjenicu prilikom čitanja podataka koji se završavaju novim redom)
- Ukoliko je moguće da vaš program prilikom izvršavanja modificira datoteku, napravite njenu kopiju da ne biste svaki put morali ponovo upisivati podatke, nego radite brisanje originalne datoteke i preimenujte kopiju (ili napravite još jednu kopiju) na originalno ime.

Studentima se savjetuje da svakako urade zadatke predviđene za ovaj tutorijal, da bi se upoznali sa tehnikama rada sa datotekama (pogotovo ukoliko nemaju namjeru raditi posljednju zadaću).

A sada slijede i sami zadaci:

1. Uz pomoć nekog tekstualnog editora kreirajte tekstualnu datoteku "STUDENTI.TXT" koja sadrži podatke o studentima na nekom fakultetu. Datoteka je organizirana na sljedeći način. U prvom redu nalazi se puno ime i prezime studenta. U drugom redu nalazi se njegov broj indeksa, u trećem redu datum rođenja u formatu *dan*/*mjesec*/*godina* (npr. 3/10/1998), dok se u četvrtom redu nalaze ocjene studenta, međusobno razdvojene zarezima (iza posljednje ocjene *nema zareza*). Dalje se podaci ponavljaju za svakog od studenata za koji su registrirani podaci. Slijedi primjer mogućeg izgleda ove datoteke:

```
Damir Damirovic
1234
3/10/1998
6,8,5,10,7,7,9,8
Brus Li Ramadanovic
4312
23/9/1997
7,9,6,5,8,8,7
Ena Popovic-Hodzic
2143
4/12/1998
9,10,8,6,5,7,9,10,6
```

   Zatim napravite program koji iščitava sadržaj ove tekstualne datoteke i kreira drugu tekstualnu datoteku "IZVJESTAJ.TXT" koja sadrži izvještaj koji izgleda poput sljedećeg. Za ispis imena i prezimena rezervirajte 30 mjesta, za indeks 10 mjesta, za datum rođenja 20 mjesta i za prosjek također 10 mjesta. Pri tome, prosjek se ispisuje fiksno na dvije decimale:

```
Student                       Indeks      Datum rodjenja      Prosjek
-------                       ------      --------------      -------
Ena Popovic-Hodzic            2143        4/12/1998           8.12
Damir Damirovic               1234        3/10/1998           7.86
Brus Li Ramadanovic           4312        23/9/1997           7.50
```

   Spisak treba biti sortiran po prosjeku, kao što je gore prikazano. U slučaju da dva studenta imaju isti prosjek, prije treba doći student sa manjim brojem indeksa. U slučaju da čitanje datoteke protekne korektno, program ne ispisuje ništa na ekran. S druge strane, ukoliko datoteka ne postoji, na ekran treba ispisati tekst "Datoteka STUDENTI.TXT ne postoji!" (sa prelaskom u novi red nakon ispisa), a ukoliko datoteka nije u odgovarajućem formatu, na ekran treba ispisati tekst "Neispravan format datoteke STUDENTI.TXT" (također uz prelazak u novi red). To uključuje i besmislene datume, besmislene ocjene, nailazak na nenumeričke podatke prilikom čitanja numeričkih podataka. Legalne su ocjene u rasponu od 5 do 10, pri čemu se ocjena 5 prihvata, ali ne uračunava u prosjek (jedino u slučaju da ni jedna registrirana ocjena nije veća od 5, tada se uzima da student ima prosjek 5). Konačno, u slučaju nepopravljivih problema, poput fizičkog oštećenja datoteke, na ekran treba ispisati tekst "Problemi pri citanju datoteke STUDENTI.TXT".

2. Neka je *Predmet* objekat koji, između ostalog, posjeduje neku gustinu. *Lopta* je vrsta predmeta koja posjeduje i poluprečnik, dok je *Cigla* vrsta predmeta (oblika kvadra) koja posjeduje i dužine stranica. Razvijte hijerarhiju klasa koje opisuju ove objekte. Apstraktna bazna klasa "`Predmet`" posjedovaće atribut koji predstavlja gustinu predmeta, konstruktor koji inicijalizira ovaj atribut, apstraktnu metodu "`DajZapreminu`" koja daje zapreminu predmeta, te metodu "`DajTezinu`" koja daje njegovu težinu (gustina pomnožena sa zapreminom). Klasa "`Lopta`" nasljeđuje se iz klase "`Predmet`", a posjeduje dodatni atribut koji predstavlja poluprečnik lopte. Njen konstruktor ima dodatni parametar (poluprečnik r) i izmijenjenu metodu za računanje zapremine, u skladu sa načinom računanja zapremine za loptu ($V = 4\pi r^3/3$). Klasa "`Cigla`" se također nasljeđuje iz klase "`Predmet`", a posjeduje dodatne atribute koji predstavljaju dužine stranica cigle. Ona također ima dodatne parametre u konstruktoru (stranice a, b i c) i implementiranu metodu za računanje zapremine ($V = abc$). Napisane klase demonstrirajte u testnom programu koji prvo traži od korisnika da unese sa tastature ime datoteke (na ekran se ispisuje tekst "Unesite ime datoteke: "

nakon čega korisnik treba da unese ime), a koji zatim čita podatke o predmetima iz tekstualne datoteke, dinamički kreira odgovarajuće objekte i čuva njihove adrese u dinamički alociranom nizu pokazivača na predmete, sortira predmete po ukupnoj težini u opadajući poredak i na kraju ispisuje težine predmeta nakon sortiranja (svaka težina se ispisuje u posebnom redu). Prvi red datoteke sadrži ukupan broj predmeta. Nakon toga, svaki red datoteke sadrži podatke o jednom predmetu, gdje početno slovo "`L`" označava loptu, a "`C`" ciglu. Recimo, "`L0.75 3.5`" predstavlja loptu gustine 0.75 i poluprečnika 3.5, dok "`C1.3 3.4 7 2.15`" predstavlja ciglu gustine 1.3 sa dužinom stranica 3.4, 7 i 2.15 respektivno. U slučaju da zadana datoteka ne postoji, na ekran treba ispisati tekst "Trazena datoteka ne postoji", a u slučaju da format datoteke nije u skladu sa specifikacijama, treba ispisati tekst "Neispravan format datoteke". Isti tekst treba da se prijavi i ukoliko je neki od podataka negativan ili nula (s obzirom da niti gustina niti dimenzije predmeta ne mogu biti takvi).

3. Proširite klasu "`Liga`" iz Zadatka 4 sa Tutorijaala 11. sljedećim elementima:

   a) Metodom "`ObrisiSve`" koja briše kompletnu ligu, odnosno dovodi je u stanje kakvo je bilo nakon kreiranja objekta tipa "`Liga`".
   b) Metodom "`SacuvajStanje`" koja sprema cijelo stanje lige u binarnu datoteku, pri čemu se kao parametar zadaje ime datoteke (tipa "`string`"). U slučaju bilo kakvih problema pri upisu, metoda baca izuzetak tipa "`logic_error`" uz prateći tekst "Problemi pri upisu u datoteku".
   c) Metodom "`AzurirajIzDatoteke`" koja vrši ažuriranje stanja lige iz tekstualne datoteke čije se ime zadane kao parametar (tipa "`string`"). Datoteka je organizirana tako da prva dva reda sadrže imena timova koji su odigrali utakmicu, nakon čega u trećem redu slijedi rezultat utakmice u obliku poput "`3:2`" sa značenjem "3 postignuta gola za prvi tim, a 2 postignuta gola za drugi tim". Postojeći podaci u ligi se ne brišu, nego se samo stanje ažurira dodatnim informacijama. Ukoliko datoteka ne postoji, baca se izuzetak tipa "`logic_error`" uz prateći tekst "Datoteka ne postoji", a u slučaju bilo kakvih problema pri čitanju, metoda baca izuzetak istog tipa uz prateći tekst "Problemi pri citanju datoteke" (uključujući i slučaj kada datoteka sadrži nekonzistentne podatke, poput nenumeričkih podataka gdje se očekuje broj, ili negativan broj golova, itd.).
   d) Konstruktorom koji obnavlja stanje lige iz binarne datoteke čije je ime zadano kao parametar konstruktora (tipa "`string`"). Za datoteku se očekuje da je onakva kakva je kreirana pod b). Ukoliko datoteka ne postoji, konstruktor baca izuzetak tipa "`logic_error`" uz prateći tekst "Datoteka ne postoji", a u slučaju bilo kakvih problema pri čitanju, metoda baca izuzetak istog tipa uz prateći tekst "Problemi pri citanju datoteke". Također, prije nego što se izvrši dinamička alokacija za podatke koji će biti obnovljeni, obavezno treba provjeriti da li je pročitana informacija o broju timova manja ili jednaka od kapaciteta lige. Ukoliko to nije slučaj (što se može desiti jedino ukoliko neko "podvali" neispravnu datoteku), treba baciti izuzetak tipa "`logic_error`" uz prateći tekst "Datoteka sadrzi fatalne greske". Ovo je neophodno uraditi, jer bi u suprotnom pokušaj obnove lige iz takve datoteke rezultirao krahom programa.

   Ukoliko to smatrate potrebnim, dozvoljeno je izvršiti i neke dopune u klasi "`Tim`" (ali bez mijenjanja njenog interfejsa). Nakon izvršene modifikacije, izmijenite i glavni program tako da po završetku rada obavezno snima stanje lige u binarnu datoteku "`LIGA.DAT`". Na početku rada programa, ukoliko datoteka "`LIGA.DAT`" postoji, program treba da obnovi sadržaj lige iz ove datoteke, tako da program prosto nastavlja raditi sa istom ligom i istim rezultatima sa kojima je radio prilikom prethodnog pokretanja. U slučaju da datoteka "`LIGA.DAT`" ne postoji, program treba da kreira novu ligu, onakvu kakva je bila i u testnom programu Zadatka 4 sa Tutorijala 11.

   VAŽNA NAPOMENA: U ovom zadatku morate krenuti od svoje vlastite klase "`Liga`" kakvu ste imali na Tutorijalu 11, a ne od nečije tuđe (ovo će se *provjeravati*). Ukoliko niste ranije dovršili ovu klasu, morate je sada dovršiti, a ukoliko je niste ranije ni napisali, morate je sada napisati (a ne nipošto kopirati odnekud ili od nekoga). Ukoliko ne postupite ovako, zadatak će se tretirati kao prepisan.

4. Pretpostavimo da neka binarna datoteka sadrži snimljen sadržaj nekog niza realnih brojeva (tipa "**`double`**"). Napišite funkciju "`IzvrniDatoteku`" koja kao parametar prima ime datoteke (tipa "`string`"), a koja izvrće sadržaj datoteke "u ogledalu" tako da njen prvi element postane posljednji, drugi pretposljednji, itd. Pri tome, sve se mora izvoditi direktno nad sadržajem datoteke, odnosno nije dozvoljeno prethodno učitavanje sadržaja datoteke u memoriju. U slučaju da datoteka ne postoji, funkcija treba baciti izuzetak tipa "`logic_error`" uz prateći tekst "Datoteka ne postoji", a u

slučaju bilo kakvih problema u čitanju, treba baciti isti izuzetak, uz prateći tekst "Problemi pri citanju datoteke". Obavezno napišite i testni programu kojem ćete testirati napisanu funkciju.