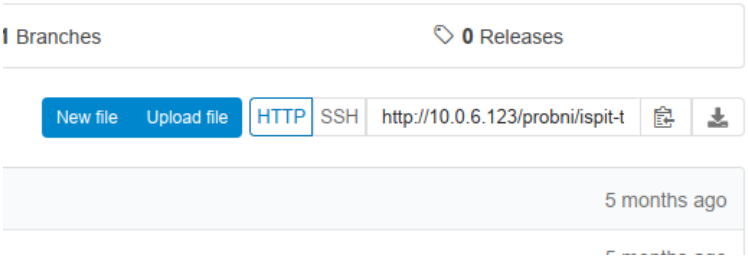
Elektrotehnički fakultet Sarajevo
Predmet: Razvoj programskih rješenja
Akademska 2018/2019. godina

Parcijalni ispit - Upute za rad

1. Upalite okruženje IntelliJ IDEA.
2. Pristupite web baziranom okruženju GOGS na adresi:
   http://10.0.6.123
3. Kliknite na opciju **Register** (gore desno) kako biste kreirali novi korisnički račun.
4. Unesite vašu *pravu ETF email adresu*, za korisničko ime uzmite *prvi dio email* adrese prije znaka @ (npr. mbajraktar1), izaberite neki password koji nije lako pogoditi. **Jako je važno da baš ovakve podatke unesete**! U suprotnom nećete dobiti zadatak.
5. Refreshujte GitLab stranicu: ubrzo ćete u sekciji **Collaborative repositories** ugledati repozitorij koji se zove **ispit-test**. Otvorite taj repozitorij i kliknite na opciju **Fork**.
6. Sada se vratite na naslovnu stranicu, ugledaćete vašu kopiju/fork repozitorija u sekciji **My Repositories**. Uđite u taj novi repozitorij.
7. Kliknite na opciju **HTTP**. Ugledaćete URL koji možete iskoristiti u IDEA da preuzmete projekat sa Git servera kako smo radili na tutorijalima.



8. Pokrenite projekat kako biste se uvjerili da sve radi. Trebali biste ugledati prozor “Sve je spremno za ispit”.
9. Vratite se na početnu GitLab stranicu i refreshajte je. Tačno 15 minuta nakon početka ispita trebali biste ugledati novi repozitorij pod imenom **razvoj-softvera-2-parcijalni**.
10. ***Ponovite sve korake*** kao i za repositorij ispit-test: Fork, vratite se na početnu, otvorite vaš primjerak repozitorija, Clone, otvorite iz IDEA!
11. Radite samostalno projekat i nemojte zaboraviti raditi commit i push! Biće pregledana posljednja verzija koju push-ate.
12. Nemojte gledati ekran od kolega/kolegica pored vas i raditi isto! Zabilježili smo ko je gdje sjedio, poslije ispita biće korišten poznati sistem za provjeru prepisivanja i bićete bodovani sa 0 bodova ako se vaš kod bude neznatno razlikovao od osoba pored vas.

Sarajevo, 4. 9. 2019

Vedran Zubrić

Elektrotehnički fakultet Sarajevo
Predmet: Razvoj programskih rješenja
Akademska 2018/2019. godina

Potpis studenta: _____

I parcijalni ispit

Ukupno bodova: 20 (Bodovi će se dodijeliti proporcionalno broju uspješnih testova.)

Na repozitoriju se nalazi gotov projekat koji sadrži samo praznu Main klasu i testove. Vaš zadatak je da napravite kompletan Java program koji zadovoljava postavku zadatka i prolazi testove.

Svojim potpisom student izjavljuje da je saglasan sa ovim sistemom bodovanja i da se rješenje postavljenog zadatka nalazi na serveru kako je objašnjeno u uputama za izradu ispita.

Zadatak:
- testovi trebaju padati ako je izuzetak izveden iz Throwable

Potrebno je implementirati Java aplikaciju za potrebe neke avio-kompanije. Projekat treba da sadrži sljedeće klase, sa navedenim atributima i metodama. Pored tih vi možete dodati i druge klase, metode i atribute po želji kako biste ispunili zadatak. Pri tome se pridržavati pravila vezanih za pisanje kvalitetnog koda obrađenih na predmetu, između ostalog svi atributi obavezno moraju biti privatni.

1. Klasa **Aerodrom** treba da sadrži:
- Privatne atribute **nazivAerodroma**, **grad** i **sifra** znakovnog tipa, te **sirina** i **duzina** tipa double koje predstavljaju geografske koordinate na kojima se nalazi aerodrom. Klasa treba slijediti JavaBean specifikaciju.
- Potrebno je osigurati da se šifra aerodroma može sastojati isključivo od 3 velika slova engleskog alfabeta. Ukoliko se pokuša postaviti neka drugačija šifra, potrebno je baciti izuzetak tipa **IlegalnaSifraAerodroma**. Tekst izuzetka treba glasiti “Ilegalna sifra XYXY, probajte ABC”, gdje umjesto XYXY treba staviti šifru koju je neko pokušao postaviti. U slučaju da šifra sadrži mala slova (a inače zadovoljava sve uslove zadatka), predložena šifra ABC treba biti ista ta šifra velikim slovima. U suprotnom, predložena šifra se treba sastojati od prva tri slova imena grada, također velikim slovima, pri čemu ta prva tri slova sadrže razmak ili bilo šta što nije slovo engleskog alfabeta, takve ilegalne karaktere treba izbaciti i koristiti sljedeće po redu slovo.

2. Klasa **Let** treba da sadrži:
- Privatne atribute: **polazniAerodrom** i **dolazniAerodrom** tipa Aerodrom, te **vrijemePolaska** i **vrijemeDolaska**.
- Klasa također treba slijediti JavaBean specifikaciju.
- Pored standardnih metoda, klasa Let treba sadržavati i metodu **trajanje** koja vraća trajanje leta u minutama kao vrijednost tipa **int** (pretpostaviti da su u vremenima polaska i dolaska sekunde postavljene na nulu).

- Također treba sadržavati i metodu **duzinaLeta** koja vraća udaljenost između polaznog i dolaznog aerodroma u “stepenima”. Radi jednostavnosti zadatka, pretpostavićemo da je Zemlja ravna ploča, tako da se udaljenost između dvije tačke može izračunati koristeći uobičajenu euklidsku formulu za udaljenost između dvije tačke, na osnovu atributa **duzina** i **sirina** klase Aerodrom. Ipak, za svaki slučaj, obavezno je u ovoj metodi napisati komentar sljedeće sadržine: “Znam da Zemlja nije ravna ploča, ali radi jednostavnosti ćemo koristiti euklidsku udaljenost”.

3. Klasa **BrziLet** predstavlja let obavljen najmodernijim supersoničnim avionima koji su tek u razvoju. Kako bismo dočarali činjenicu da ovi avioni lete dvostruko brže, metoda **duzinaLeta** u ovoj klasi treba vraćati broj dvostruko manji od onog koji bi vratila ista metoda klase **Let**.

4. Klasa **Aviokompanija** posjeduje sljedeće metode:
- Konstruktor koji prima jedan cijeli broj koji predstavlja maksimalan broj letova koji se mogu registrirati.
- Metoda **brojLetova** vraća broj trenutno registrovanih letova.
- Metodu **registrujLet** koja kreira novu instancu klase **Let** i pohranjuje u nekoj internoj kolekciji. Metoda ima parametre: **polazniAerodrom**, **dolazniAerodrom**, **vrijemePolaska**, **vrijemeDolaska** i **brzi**. Prva 4 parametra imaju isti tip kao istoimeni atributi klase Let, a parametar **brzi** je tipa boolean i označava da je u pitanju BrziLet. Ako je već registrovan maksimalan broj letova, metoda treba baciti izuzetak tipa **SizeLimitExceededException**.
- Metoda **dolazniLetovi** vraća mapu (Map) u kojoj je ključ naziv grada, a vrijednost je lista (List) svih letova koji dolaze u taj grad danas.
- Metoda **uZraku** vraća skup (Set) svih letova koji su bili u zraku u datom trenutku. Funkcija prima LocalTime. Vraćeni skup treba biti sortiran po vremenu polaska.
- Metoda **nadjiNajkraci** prima dva stringa koji predstavljaju imena polaznog i dolaznog grada. Funkcija treba vratiti vrijednost tipa ArrayList<Let> koja sadrži niz uzastopnih letova takvih da se iz polaznog grada stigne do dolaznog grada. Npr. ako su primljeni parametri Sarajevo i New York, može se vratiti niz letova: Sarajevo-Beč, Beč-London i London-New York ali pri tome voditi računa da vrijeme dolaska prvog leta treba biti prije vremena polaska drugog leta. Ako postoji više kombinacija letova koje ispunjavaju uslov, treba vratiti onu čija je ukupna dužina najmanja (pozivom metode **duzinaLeta**). Metoda će biti bodovana sa dodatna dva boda ako se sastoji od samo jedne linije koda.
- Metoda **nadjiNajbolji** radi slično kao prethodna metoda, ali prima lambda funkciju kriterija po kojoj će se tražiti najbolji let. Lambda funkcija treba da prima Let a vraća realan broj, tako da treba vratiti onu listu letova za koje je ukupna suma vrijednosti koje je vratila funkcija najveća. Ponovo, dostupni su bonu bodovi ako se funkcija sastoji od samo jedne linije.

Sarajevo, 4. 9. 2019