Objektno programiranje, Ispit JUN2, Grupa 1
Matematički fakultet
Školska godina 2017/2018

Napomena: Na Desktop-u napraviti direktorijum pod imenom `oop_Ime_Prezime_Indeks_Asistent` (npr. `oop_Pera_Peric_mi12082_NM`). Pokrenuti *Intellij Idea* i u napravljenom direktorijumu napraviti projekat sa istim nazivom. U napravljenom projektu, paket takođe nazvati isto tako.
Kod **ne sme** imati sintaksnih grešaka niti izbacivanje `NullPointerException`-a.
Vreme za rad: **2.5 sata**
Inicijalini asistenata: `Biljana - BS`, `Anja - AB`, `Božidar - BA`, `Nemanja - NM`

U tekstu je dat opis klasa, njihovih atributa i metoda. **Dozvoljeno** je (i ohrabrujemo Vas) dodati nove atribute, klase, metode, enume, interfejse u slučaju da Vam olakšavaju implementaciju, i/ili smatrate da Vam poboljšavaju kvalitet koda i slično. Nekada će zahtevi u zadatku i zahtevati od Vas da dodate novi atribut ili slično.

Da bi se uspešno položio ispit potrebno je osvojiti barem 50% poena.

1. Napraviti apstraktnu klasu `Racunar` koju karakterišu atributi ime (`String`) i cenovniKoef (`int`). Implementirati konstruktor koji prima vrednosti za oba atributa, kao i potrebne `get` metode. Klasa poseduje apstraktan metod `public abstract double izracunajCenu(List<Integer> duzineKoriscenja)` - lista duzineKoriscenja sadrži dužine vremenskih intervala korišćenja datog računara bez prestanka u minutima.

2. Napraviti klasu `Korisnik` koja se karakteriše atributima korisnickoIme (`String`), vipKrediti (`int`) i dugovi (`Map <Racunar, Double>` - mapa pomoću koje korisnik može da prati svoja zaduženja na svakom računaru). Implementirati konstruktor koji prihvata vrednost za atribute `korisnickoIme` i `vipKrediti` i pravi praznu mapu dugova, neophodne `get` metode, metod `public double dodajRacunar(Racunar r)` kojim se r dodaje u mapu sa dugom 0.0, metod `public void azurirajDug(Racunar r, double cena)` koji postavlja dug računara `r` na `cena`, metod `public void umanjiVipKredite()` koji smanjuje `vipKrediti` za 1 i metod `public double ukupanDug()` koji računa ukupan dug korisnika.

3. Napraviti klasu `RacunarZaObradu` koja nasleđuje klasu `Racunar` i ima dodatno polje brojProcesora (`int`). Implementirati potrebne `get` metode i metod `public double izracunajCenu(List<Integer> duzineKoriscenja)` koji na osnovu liste `duzineKoriscenja` računa cenu korišćenja tog računara. Cena se računa kao suma cena Ci pojedinačnih dužina korišćenja, gde se `Ci` dobija kao proizvod broja procesora, korena i-te dužine korišćenja i cenovnog koeficijenta. Ako se računar bez prestanka koristio duže od 1800 minuta, pali se dodatno hlađenje procesora čija cena za svaki prekoračeni minut košta 0.2 dinara pomnoženo brojem procesora koje računar ima. Implementirati metod `toString` tako da vraća nisku oblika:

```
[Obrada] ime sa brojProcesora procesora, koef. cene: cenovniKoef
[Obrada] Azdaja sa 16 procesora, koef. cene: 4.5
```

4. Napisati klasu `RacunarZaCuvanje` koja nasleđuje klasu `Racunar` i dodatno sadrži atribute velicinaDiska (`int`) izraženo u MB, i polje backup (`boolean`) koje označava da li su podaci skladišteni na datom računaru *bekapovani*. Implementirati metod `public double izracunajCenu(List<Integer> duzineKoriscenja)` koji na osnovu liste `duzineKoriscenja` računa cenu korišćenja tog računara. Cena se računa kao suma cena Ci pojedinačnih korišćenja računara, gde je `Ci` i-ta cena korišćenja računara. Cena po minutu korišćenja iznosi 0.000001 dinara pomnožena koeficijentom cene, i-tom dužinom korišćenja i veličinom diska. Ukoliko računar ima mogućnost *bekapovanja* ukupna cena je veća za 50%. Implementirati metod `toString` tako da vraća nisku oblika:

```
[Cuvanje] ime sa velicinaDiskaMB diska(backup/nobackup), koef. cene: cenovniKoef
[Cuvanje] Mordor sa 1000000MB diska(backup), koef. cene: 1.5
```

5. Napraviti klasu `FarmaRacunara` koja nasleđuje `Application` klasu biblioteke `javafx` i izgleda kao na slici 1.
Obezbediti da je unapred selektovano prvo radio dugme (gledano sleva) i da u svakom trenutku može biti selektovano tačno jedno radio dugme.

   Klasa sadrži polje korisnik (`Korisnik`), koga treba instancirati sa proizvoljnim korisničkim imenom i brojem vip kredita, statičko polje *random* (`Random`), polje izabrani (`Racunar`) i polje racunari (`Map<Racunar, List<Integer>>`).

   Na klik dugmeta `Ucitaj` iz datoteke `racunari.txt` učitavaju se računari i smeštaju u mapu `racunari` i dodaju se prethodno napravljenom korisniku (koristeći metodu `dodajRacunar`). Vrednosti u mapi `racunari` su liste dužina vremenskih intervala za koje je korisnik koristio računare.

U klasi `Racunar` potrebno je obezbediti sortiranje instanci tako da prvo idu računari za obradu (u datoteci počinju sa `o`) i to opadajuće po broju procesora, a potom računari za čuvanje (u datoteci počinju sa `c`) opadajuće po veličini diska - na primer može se koristiti operator `instanceof`. Nakon učitavanja u mapu, u gornji `TextArea` element se ispisuje sadržaj, svaki u novom redu (koristeći `toString` nad klasom `Racunar` - pogledati sliku 1).

Na klik dugmeta `Koristi` korisnik pokušava da koristi računar čiji naziv unosi u prvi element `TextField` za vreme koje unosi u drugi element `TextField`. Ukoliko je korisnik označio standardni prioritet, on sa verovatnoćom 0.5 uspešno zauzima računar i u donji `TextArea` ispisuje poruku "`ime` je uspesno zauzet" - kao na slici 1, i dodaje vreme korišćenja za taj računar u mapu `racunari`. U suprotnom, u donji element `TextArea` ispisuje poruku "`ime` je trenutno zauzet, pokusajte malo kasnije!" - kao na slici 1. Ako je korisnik odabrao VIP prioritet, proverava se njegovo stanje vip kredita i ukoliko ih ima (više od 0), umanjuje ga za 1 i uspešno zauzima (sa verovatnoćom 1) računar za korišćenje, u donji `TextArea` ispisuje poruku "`ime` je uspesno zauzet" i dodaje izabrano vreme u mapu `racunari`. U slučaju unosa pogrešnog imena računara, u donji `TextArea` ispisati poruku "Ne postoji racunar sa trazenim imenom" - kao na slici 2.

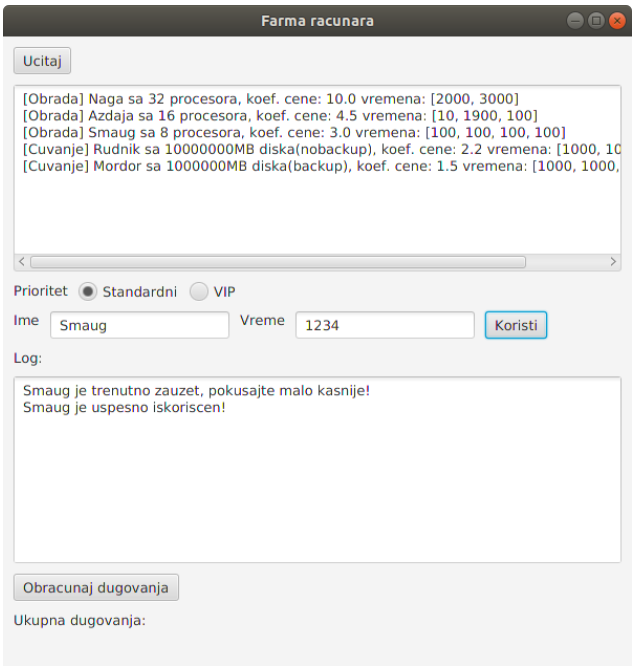
Na klik dugmeta `Obracunaj dugovanja`, potrebno je polimorfno pozivati metod `izracunajCenu` nad sadržajem koji se nalazi u mapi `racunari` i korisniku ažurirati cenu računara na dobijene vrednosti. U donji element `TextArea` ispisati imena računara i dugove na njima, a u najniži `Label` element ispisati ukupna dugovanja korisnika (slika 2).

Dozvoljeno je proširiti klase dodatnim atributima i metodama kako biste realizovali prethodno navedene zahteve.

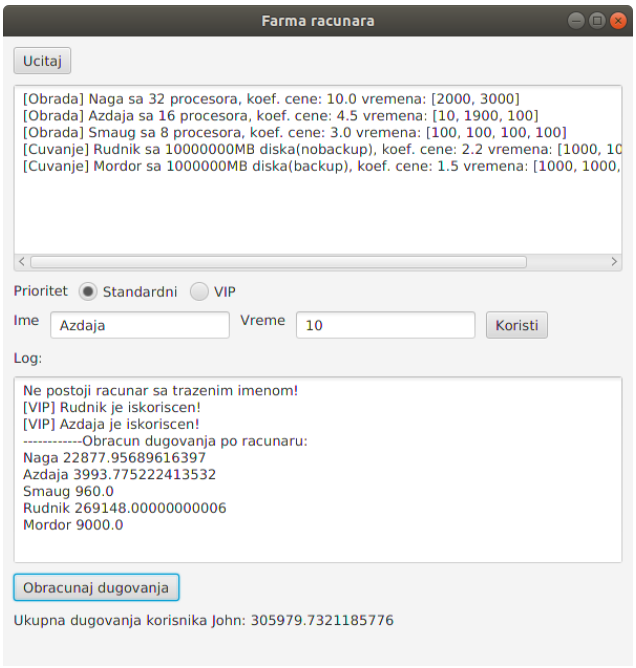
```
[racunari.txt]
c, naziv, cenovniKoef, velicinaDiska, da li ima backup, vremena
o, naziv, cenovniKoef, brojProcesora, vremena
```

Napomena: vremena može biti proizvoljan broj, odnosno za svaki računar može postojati različit broj vremena.

```
c, Mordor, 1.5, 1000000, da, 1000, 1000, 1000, 1000
o, Azdaja, 4.5, 16, 10, 1900, 100
o, Smaug, 3, 8, 100, 100, 100, 100
c, Rudnik, 2.2, 10000000, ne, 1000, 10000
o, Naga, 10, 32, 2000, 3000
```



Slika 1: Zauzimanje računara.



Slika 2: Obračun dugovanja