Objektno-orijentisano programiranje, Ispit JUN2, Grupa 2
Matematički fakultet
Školska godina 2018/2019

Napomena: Na Desktop-u napraviti direktorijum pod imenom `oop_Asistent_Grupa_Prezime_Ime_Indeks` (npr. `oop_NM_2_Peric_Pera_mi12082`). Pokrenuti *Intellij Idea* i u napravljenom direktorijumu napraviti projekat sa istim nazivom. U napravljenom projektu, napraviti paket sa istim nazivom.
Kod ne sme imati sintaksnih grešaka niti izbacivanje `NullPointerException`-a.
Vreme za rad: **2.5 sata**
Inicijalni asistenata: `Biljana - BS`, `Nemanja - NM`, `Anja - AB`, `Ivan - IR`, `Rastko - RD`

1. Napraviti apstraktnu klasu `BusPlus` koja predstavlja karticu za gradski prevoz i koju karakterišu atributi: `id` - identifikator kartice (`int`) i `zona` (`int`). Zona može uzeti vrednosti 1 ili 2, što ne treba proveravati. Implementirati konstruktor koji prima vrednosti za oba atributa, `get` metode za oba atributa i metod `toString()` koji vraća nisku sa brojem zone kao u test primeru: `zona: 1`

2. Napraviti nabrojivi tip `Kategorija` čije su vrednosti oznake kategorija personalizovanih BusPlus kartica: `A1`, `P2`, `P3` ili `P13`. Implementirati metod `toString()` koji za oznaku kategorije vraća nisku sa njenim značenjem, pri čemu za `A1` vraća `zaposleni`, za `P2` `srednjoskolci`, za `P3` `studenti` a za `P13` `penzioneri`. Implementirati statički metod `Kategorija izBroja(int i)` koji na osnovu prosleđenog broja vraća element nabrojivog tipa, pri čemu za `i=0` vraća `A1`, za `i=1` vraća `P2`, za `i=2` vraća `P3` a za `i=3` vraća `P13`.

3. Napraviti klasu `Personalizovana` koja nasleđuje klasu `BusPlus` i predstavlja personalizovanu BusPlus karticu. Dodatno se karakteriše poljem `kategorija` (tipa `Kategorija`) i poljem `imaDopunu` (`boolean`) koje označava da li je kartica dopunjena za tekući mesec. Implementirati konstruktor koji prima sve potrebne vrednosti, `get` metode za oba atributa i metod `toString()` koji vraća nisku kao u test primeru.

   Primer za metod `toString` (kategorija, zona, imaDopunu)

   ```
   [srednjoskolci] zona: 2 | dopunjena
   [studenti] zona: 1 | bez dopune
   ```

4. Napraviti klasu `Nepersonalizovana` koja nasleđuje klasu `BusPlus` i predstavlja nepersonalizovanu BusPlus karticu. Dodatno se karakteriše poljem `kredit` (`int`) i poljem `ocitana` (`boolean`) koje označava da li je kartica očitana u vozilu. Klasa sadrži i statičko polje `cenaVoznje` sa vrednošću 90 koje označava cenu karte za jednu vožnju u datoj zoni. Implementirati konstruktor koji prima sve potrebne vrednosti, `get` metode za oba atributa, metod `boolean nedovoljnoKredita()` koji proverava ima li dovoljno kredita za jednu vožnju i metod `toString()` koji vraća nisku kao u test primeru (ako kartica nije očitana, metodom `nedovoljnoKredita()` utvrditi razlog).

   Primer za metod `toString` (oznaka `[NP]`, zona, kredit, ocitana)

   ```
   [NP] zona: 1, kredit: 70 | nedovoljno kredita
   [NP] zona: 2, kredit: 270 | ocitana
   [NP] zona: 2, kredit: 180 | nije ocitana
   ```

5. Napraviti klasu `GSPVozilo` koja predstavlja vozilo na liniji gradskog prevoza. Klasa sadrži polje `kartice` (`Map<Integer, BusPlus>`) koje predstavlja mapu koja preslikava id karticu putnika u vozilu u odgovarajući objekat klase `BusPlus` sa tim id-em i polje `nevalidne` (`List<Integer>`) koje predstavlja listu id-eva nevalidnih kartica. Klasa sadrži i statička polja `random` (`Random`) i `noviId` (`int`, sa vrednošću 10000). Implementirati konstruktor bez argumenata koji instancira polja `kartice` i `nevalidne`, kao i `get` metod `List<BusPlus> getKartice()`.

   Implementirati metod `boolean putniciUVozilu(String putanja)` koji iz datoteke koja se nalazi na prosleđenoj putanji učitava podatke o karticama putnika u vozilu i smešta ih u mapu `kartice`. Ukoliko je učitavanje uspešno, vraća `true`, inače vraća `false`.

   Datoteka sadrži prvo indikator da li je kartica personalizovana (P) ili nepersonalizovana (N), zatim id i zonu. Personalizovane kartice imaju u nastavku oznaku kategorije i indikator da li kartica ima dopunu (tekst `da` ili `ne` na kraju linije) dok nepersonalizovane kartice imaju iznos kredita i indikator da li je kartica očitana ili ne (oznaka `+` ili `-` na kraju linije). `Primer datoteke (kartice.txt)`:

   ```
   P, 123, 2, P2, da
   N, 2002, 1, 70, -
   P, 3210, 1, A1, da
   P, 401, 1, P13, da
   N, 2501, 2, 270, +
   P, 5001, 1, P3, ne
   N, 2020, 2, 180, -
   ```

6. U klasu `Nepersonalizovana` dodati metod `boolean ocitajKartu()` za očitavanje karte. Ukoliko je karta već očitana ili nema dovoljno kredita metod vraća `false`, inače ažurira polja `kredit` i `ocitana` i vraća `true`.

7. U klasu `GSPVozilo` dodati metode:

   - `BusPlus noviPutnik(int vrstaKartice)` koji u mapu `kartice` dodaje karticu novog putnika i ujedno je vraća. Argument `vrstaKartice` može uzeti vrednost 1 ili 2 (ne vršiti proveru) i označava da li je nova kartica personalizovana (vrednost 1) ili nepersonalizovana (vrednost 2). Kartica se generiše slučajnim izborom. Id kartice se postavlja na prvi sledeći slobodan počevši od vrednosti `noviId` i u skladu sa tim i ažurira `noviId`. Zona se postavlja na vrednost `vrstaKartice`. Ako je kartica personalizovana kategorija se bira slučajnim izborom tako što se izabere slučajan ceo broj na osnovu kojeg se metodom `Kategorija.izBroja()` vraća element nabrojivog tipa. Kartica sa jednakom verovatnoćom može imati dopunu ili ne. Ako je kartica nepersonalizovana kredit se bira slučajnim izborom tako da maksimalno može da uzme vrednost dvostruke cene karte za jednu vožnju. Kartica inicijalno nije očitana. Metodom `ocitajKartu()` izvršiti njeno očitavnje.
   - `String kontrola()` koji vraća spisak kartica kao što je prikazano na slici 2 (za svaku karticu u mapi po jedna linija kao rezultat poziva metoda `toString` za tu karticu kojoj prethodi znak `+`, osim ako kartica nema dopunu ili nije očitana kada joj prethodi znak `-`). Id kartice koja nema dopunu ili nije očitana dodaje se u listu `nevalidne`.
   - `boolean izbaciPutnike()` koji iz mape uklanja kartice čiji id pripada listi `nevalidne`. Ukoliko nema nevalidnih kartica vratiti `false` u protivnom vratiti `true`.

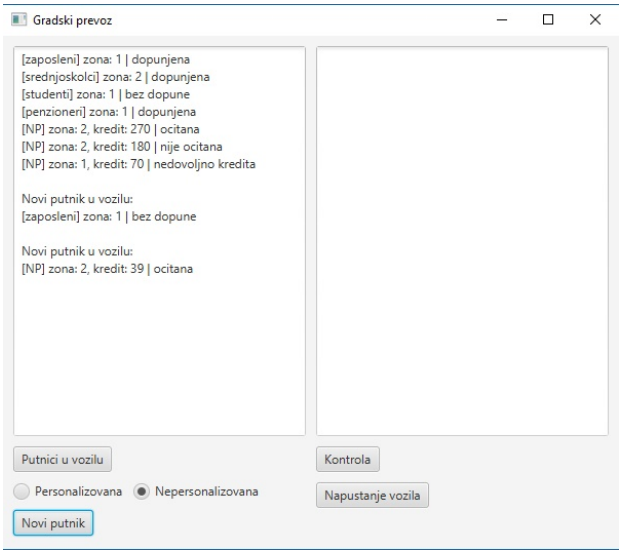
   Pretpostaviti da će se u mapi naći **barem jedna** kartica.

8. Napraviti klasu GSPVoziloGUI koja predstavlja `javafx` aplikaciju i izgleda kao na slici 1. Klasa sadrži statičko polje `gsp` (tipa `GSPVozilo`) koje se instancira pri pokretanju programa i polje `kartice` (tipa `List<BusPlus>`) sa vrednošću `null`. Obezbediti da u svakom trenutku može biti selektovano tačno jedno radio dugme i da je na početku odabrano `Personalizovane`. Onemogućiti unos teksta sa tastature u `TextArea` elemente.

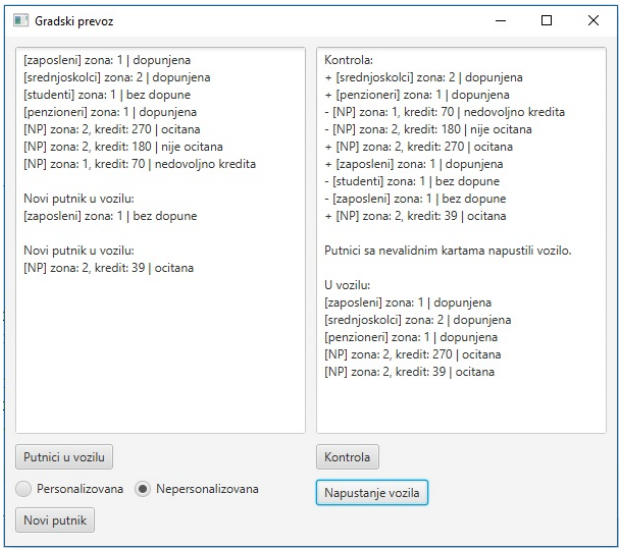
   Za implementaciju događaja je poželjno koristiti prethodno implementirane metode u klasi `GSPVozilo`.

   Na klik dugmeta:

   - **Putnici u vozilu** u levi `TextArea` element se upisuju podaci o svim BusPlus karticama pročitanim iz datoteke **"kartice.txt"** (metodom `putniciUVozilu` klase `GSPVozilo`). Kartice se prikazuju sortirano tako što se prvo prikazuju personalizovane a zatim nepersonalizovane. Personalizovane kartice sortirati prema oznaci kategorije rastuće a nepersonalizovane prema iznosu kredita opadajuće. Ukoliko nije instancirano polje `gsp` u levi `TextArea` ispisati poruku `Vozilo je u kvaru!` a ukoliko datoteka nije uspešno učitana poruku `Vozilo je trenutno bez putnika!`
   - **Novi putnik** na osnovu izabranog radio dugmeta određuje se vrsta karte novog putnika i poziva metod `noviPutnik(vrstaKarte)` klase `GSPVozilo`. Za personalizovane kartice vrsta je 1, a za nepersonalizovane 2. U levi `TextArea` element ispisati podatke o kartici novog putnika.
   - **Kontrola** vrši se kontrola karata i u desni `TextArea` element ispisuje rezultat kontrole.
   - **Napustanje vozila** putnici sa nevalidnim karticama, ukoliko ih ima, napuštaju vozilo o čemu se izveštava u desnom `TextArea` elementu i u nastavku se prikazuje sortirani spisak kartica putnika preostalih u vozilu. Inače, ispisuje se poruka `Sve kartice su validne!`



Slika 1: Prikaz kartica postojećih i novih putnika



Slika 2: Kontrola kartica i napuštanje vozila