Objektno-orijentisano programiranje, Kolokvijum, Grupa 1
Matematički fakultet
Školska godina 2018/2019

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: **1.5 sat**
Minimalan potreban broj poena da bi se položio kolokvijum: **5 poena**
Potrebno je napisati barem jednu klasu i testirati njen rad u test klasi.
Inicijalini asistenata: `Biljana - BS`, `Anja - AB`, `Ivan - IR`, `Nemanja - NM`, `Rastko - RD`

1. (5 poena) Napraviti klasu `Platforma` koju karakterišu atributi:

   - `ime` (String) - ime platforme.
   - `brojInstrukcijaZaUcitavanjeProcesa` (double) - broj asemblerskih instrukcija za kreiranje procesa na platformi.
   - `memorijaZaProces` (double) - potrebna memorija za rad procesa u megabajtima.

   Implementirati:

   - Konstruktor koji prihvata sva tri atributa.
   - Konstruktor kopije.
   - `get` metode za sva tri atributa.
   - `set` metoda za broj instrukcija.
   - `toString` metod za prikaz objekta kao u test primeru u nastavku.

   Napraviti klasu `Main` u kojoj treba instancirati dve platforme i prikazati ih na standardnom izlazu (pozvati toString() metod):

   ```
   Dostupne platforme:
   - ArchLinux broj instrukcija za ucitavanje procesa=8000.0 memorija za proces=20.0MB
   - Windows 10 broj instrukcija za ucitavanje procesa=10000.0 memorija za proces=30.0MB
   ```

2. (3 poena) Napraviti apstraktnu baznu klasu `Program` koju sadrži atribute:

   - `ime` (String).
   - `potrebnoMemorije` (double).
   - `platforma` (Platforma).

   Klasa sadrži i metode:

   - Konstruktor koji prihvata sva tri abtributa.
   - Konstruktor kopije.
   - `abstract double cenaIzvrsavanja()`.
   - `abstract double memorijskoZauzece()`.
   - `get` metode za sva tri atributa.

3. (5 poena) Napraviti klasu `KompiliranProgram` koji nasleđuje klasu `Program` i sadrži atribut `brojInstrukcija` (int) koji predstavlja broj asemblerskih instrukcija za program dobijenih kompilacijom.

   Implementirati:

   - Konstruktor koji prihvata vrednosti za sve atribute.
   - Konstruktor kopije.
   - `get` metod za broj instrukcija.
   - `toString` metod za prikaz kao u nastavku teksta.
   - `cenaIzvrsavanja()` tako da računa cenu kao zbir broja instrukcija za kompiliran program i broja instrukcija za učitavanje procesa za platformu (pretpostavimo da ne postoje instrukcije skoka i da se program izvršava sekvencijalno).
   - `memorijskoZauzece()` tako da računa memorijsko zauzeće kao zbir potrebne memorije za program i potrebne memorije za proces za platformu programa.

   ```
   [kompiliran program] Intellij Idea 700.0MB platforma=ArchLinux brojInstrukcija=3000000
   [kompiliran program] Visual Studio.exe 400.0MB platforma=Windows 10 brojInstrukcija=5000000
   ```

4. (2 poena) Napraviti nabrojivi tip (`Enum`) `SkriptJezik` koji ima dozvoljene vrednosti `Python`, `Perl`, `PHP`, `JavaScript`, `Ruby` i `Lua`.

5. (6 poena) Napraviti klasu `Skripta` koja nasleđuje klasu `Program` i karakteriše se atributima:

   - `jezik` (SkriptJezik) - skript jezik kojim je napisana skripta.
   - `cenaInstrukcije` (`double`) - kolika se cena plaća za interpretaciju jedne instrukcije.
   - `brojLinija` (`int`) - broj linija (instrukcija) koje skripta sadrži.

   Implementirati:

   - Konstruktor koji prihvata vrednosti za sve atribute.
   - Konstruktor kopije.
   - `get` metode za sve atribute.
   - `set` metod za atribut `brojLinija`.
   - `toString` metod za prikaz kao u nastavku teksta.
   - `cenaIzvrsavanja()` tako da vraća zbir broja instrukcija za učitavanje procesa za platformu i proizvoda broja linija skripta i cene izvršavanje instrukcije.
   - `memorijskoZauzece()` tako da vraća količinu potrebne memorije za program.

```
[skripta] inicijalizacijaBaze.lua 0.2MB platforma=ArchLinux brojLinija=100 cenaInstrukcije=1200.0
[skripta] zaustaviWinUpdate.py 5.0MB platforma=Windows 10 brojLinija=300 cenaInstrukcije=1000.0
```

6. (4 poena) U klasi `Main` napraviti objekte kao u test primeru u nastavku i smestiti ih u niz programa tipa `Program[]`. Korisnik unosi komande interaktivno (dok se ne izađe iz programa) i to:

   - `izlaz` - izlaz iz programa
   - `mem` - polimorfno se poziva metod `memorijskoZauzece()` i rezultat prikazuje korisniku
   - `cena` - polimorfno se poziva metod `cenaIzvrsavanja()` i rezultat prikazuje korisniku
   - `svi` - prikazuju se svi dostupni programi (pozvati `toString()`)
   - `pomoc` - prikazuje se pomoćni meni koji prikazuje korisniku dostupne opcije (isti tekst koji se prikazuje pri pokretanju programa)

```
Dostupne platforme:
- ArchLinux broj instrukcija za ucitavanje procesa=8000.0 memorija za proces=20.0MB
- Windows 10 broj instrukcija za ucitavanje procesa=10000.0 memorija za proces=30.0MB

Dostupne opcije:
izlaz - izlazak iz programa
mem - prikaz memorijskog zauzeca
cena - prikaz cene izvrsavanja
svi - prikaz dostupnih programa
pomoc - prikaz ovog menija

svi
- [skripta] inicijalizacijaBaze.lua 0.2MB platforma=ArchLinux brojLinija=100 cenaInstrukcije=1200.0
- [skripta] zaustaviWinUpdate.py 5.0MB platforma=Windows 10 brojLinija=300 cenaInstrukcije=1000.0
- [kompiliran program] Intellij Idea 700.0MB platforma=ArchLinux brojInstrukcija=3000000
- [kompiliran program] Visual Studio.exe 400.0MB platforma=Windows 10 brojInstrukcija=5000000

mem
- 0.2MB
- 5.0MB
- 720.0MB
- 430.0MB

cena
- 128000.0
- 310000.0
- 3008000.0
- 5010000.0

izlaz
```