Matematički fakultet - OOP.

Objektno programiranje, vezbanje 04

Nemanja Mićović

1. Napraviti klasu `Student` koja opisuje studente Matematičkog fakulteta.

   Od privatnih polja klasa sadrži:

   - `String ime`, ime studenta
   - `String prezime`, prezime studenta
   - `String indeks`, indeks u obliku mXYYZZZ (X oznaka smera, YY godina upisa, ZZZ broj indeksa)
   - `double prosek`, trenutni prosek

   Konstruktor za klasu je oblika:

   - `Student(String ime, String prezime, String indeks, double prosek)`

   Obezbediti javno dostupne metode:

   - `String getIme()`, vraća ime studenta
   - `String getPrezime()`, vraća prezime studenta
   - `int getBrojIndeksa()`, vraća broj indeksa studenta
   - `int getGodinaUpisa()`, vraća godinu upisa studenta
   - `String getSmer()`, vraća smer studenta
   - `void setIndeks(String indeks)`, postavlja indeks studenta na prosleđenu vrednost

   Obezbediti implementaciju funkcije `String toString()` koja studenta ispisuje kao:

   ```
   > Petar Peric, me14123, 7.2
   ```

2. Napraviti klasu `Autor` koja opisuje autora neke knjige. Od privatnih polja, klasa sadrzi:

   - `String ime`, ime autora
   - `String prezime`, prezime autora
   - `int godinaRodjenja`, godina u kojoj je autor rodjen

   Konstruktor za klasu je oblika:

   - `Autor(String ime, String prezime, int godinaRodjenja)`

   Od metoda, klasa sadrzi:

   - `String getIme()`
   - `String getPrezime()`
   - `String getGodinaRodjenja()`
   - `int getGodine()`, izračunati na osnovu trenutne godine

   Obezbediti implementaciju funkcije `String toString()` koja autora ispisuje kao:

   ```
   > Autor: Robert Dzordan
   > Godina rodjenja: 1948
   ```

3. Napraviti klasu `Knjiga` koja opisuje knjigu. Od privatnih polja, klasa sadrzi:

   - `String naslov`, naslov knjige
   - `Autor autor`, autora knjige
   - `double ocena`, prosecna ocena knjige, najmanje 0, najvise 10

   Konstruktor za klasu je oblika:

   - `Knjiga(String naslov, Autor autor, double ocena)`

   Od metoda, klasa sadrzi:

   - `String getNaslov()`
   - `Autor getAutor()`
   - `double getOcena()`
   - `String toString()`, koji vraca stringovnu reprezentaciju knjige.

Obezbediti funkciju `String toString()` koja knjigu ispisuje kao:

```
> Knjiga: Zenica Sveta
> Ocena: 9.7
> Autor: Robert Dzordan
> Godina rodjenja: 1948
```

Napisati funkciju `main` u kojoj je potrebno napraviti sledece knjige[1]:

- Robert Dzordan, Zenica Sveta, 9.7
- Daniel Suarez, Demon, 9.5
- Daniel Suarez, Sloboda, 9.1

A pomenuti autori su:

- Robert Dzordan rodjen 1948.
- Daniel Suarez rodjen 1964.

4. Napraviti klasu `SigurniScanner` koja omogućava lakše korišćenje klase `Scanner`.

   Od privatnih polja klasa sadrži:

   - `Scanner sc`

   Konstruktor je oblika:

   - `SigurniScanner()`, instancira `sc` i vezuje ga na `System.in`

   Klasa sadrži sledeće metode:

   - `int sledeciInt()`, vraća sledeći `int` ako postoji
   - `double sledeciDouble()`, vraća sledeći `double` ako postoji
   - `String sledecaRec()`, vraća sledeću reč ako postoji
   - `String sledecaLinija()`, vraća sledeću liniju ako postoji
   - `char sledeciKarakter()`, vraća sledeći karakter ako postoji

   Obezbediti da funkcije **sigurno** čitaju standardni ulaz. U slučaju da detektuju grešku, potrebno je korisniku ispisati obaveštenje o grešći na standardni izlaz i izaći iz programa.

5. Napraviti klasu `Program` koja opisuje računarski program. Računarski program se sastoji iz svog mašinskog koda (binarno zapisan), imena izvršive datoteke, svoje `minor` i `major` verzije, veličine u megabajtima (realan broj).

   Na primer:

   ```
   > ime: vim
   > sadrzaj: 10101011111011101101010110101101010110101010101101001010101
   > major: 7
   > minor: 3
   > velicina: 2.8 MB
   ```

   Obezbediti funkcije:

   - `int brojJedinicnihBitova()`, vraća broj bitova sadržaja programa koji su 1
   - `String getVerzija()`, vraća verziju programa u oblikut `MAJOR.MINOR`, npr. `7.3`
   - `double getVelicinaKilobajta()`, vraća veličinu programa u kilobajtima
   - `String getMasinskiKod()`, vraća sadržaj mašinskog koda programa
   - `void setMasinskiKod(String kod)`, postavlja sadržaj mašinskog koda programa
   - `String toString()`, vraća sadržaj programa kao u primeru

   Miki Hakerović, stručnjak za bezbednost kompanije `Best Security Solutions D.O.O. Inc` testira bezbednost softvera koji je dobila njegova kompanija. On želi da u mašinskom kodu programa koji je dobio podmetne sekvencu `001001001` odmah posle svake sekvence `111` jer je otkrio da na taj način može naterati program da pročita sadržaj memorijske adrese koju želi, što je veliki sigurnosti rizik.

   U funkciji main implementirati zamenu koju Miki želi da izvrši i testirati da li zamena radi.

[1] A i pročitati ukoliko volite da čitate.

6. Napraviti klasu `Tacka` koja opisuje tačku u ravni. Tačka je definisana svojim koordinatama `x`, `y` kao i svojim `imenom`.

   Klasa sadrži statičku promenljivu `brojac` koja je inicijalno postavljena na vrednost `1`. Pri konstrukciji nove tačke (kada se izvršava konstruktor), `brojac` se uvećava za jedan. Namena ove promenljive je da oslobodi korisnika potrebe da tačkama daje ime (klasa će im sama dodeliti ime).

   Npr ako je `brojac = 5`, pri pravljenju tacke (bez zadavanja imena). tacka dobija ime X5 i inkrementira se `brojac` koji će nakon toga imati vrednost `brojac = 6`.

   Obezbediti konstruktore:

   - `Tacka()`, postavlja tačku u `(0, 0)` i dodeljuje joj sistemsko ime
   - `Tacka(String ime)`, postavlja tačku u `(0, 0)` i dodeljuje joj prosleđeno ime
   - `Tacka(double x, double y)`, postavlja ime na sistemsko ime
   - `Tacka(double x, double y, String ime)`

   Obezbediti sledeće metode:

   - `double euklidskoRastojanje(Tacka t)`, vraća euklidsko rastojanje do tačke `t`
   - `double getX()`, vraća vrednost `x` koordinate
   - `double getY()`, vraća vrednost `y` koordinate
   - `void translirajHorizontalno(double dx)`, translira tačku horizontalno za `dx`
   - `void translirajVertikalno(double dy)`, translira tačku vertikalno za `dy`

   Implementirati i funkciju `String toString()` koja ispisuje tačku u obliku:

   ```
   > A(2, 3)
   ```

   Napisati klasu `Eksperiment` koja poseduje funkciju:

   - `double minimalnoRastojanje(int n)`

   Funkcija generiše `n` nasumično odabranih tačaka sa ograničenjem:

   $$0 \leq x, y \leq 100$$

   i vraća najmanje rastojanje između generisanih tačaka.

   Generisanje pseudoslučajnih brojeva realizovati koristeći klasu `Random`.

   Napisati funkciju `main` koja sa standardnog ulaza unosi veličinu eksperimenta `n`, instancira objekat klase `Eksperiment`, vrši eksperiment za dato `n` i na standardni izlaz ispisuje pronađeno minimalno rastojanje.

7. Napisati klasu `Complex` koja predstavlja kompleksni broj.

   Kompleksni broj se karakteriše realnim i imaginarnim delom. Implementirati sledeće metode:

   - `double getReal()`, vraća vrednost `realnog` dela
   - `double getImag()`, vraća vrednost `imaginarnog` dela
   - `Complex add(Complex t)`, vraća vrednost `zbira` dva kompleksna broja
   - `Complex subtract(Complex t)`, vraća vrednost `razlike` dva kompleksna broja
   - `Complex multiply(Complex t)`, vraća vrednost `proizvoda` dva kompleksna broja
   - `Complex divide(Complex t)`, vraća vrednost `količnika` dva kompleksna broja

   Obezbediti implementaciju funkcije `String toString()` koja kompleksni broj prikazuje kao u primeru:

   ```
   > 2 + 3i
   > 2 - 4i
   > 1
   > 4i
   ```

8. Napisati klasu `Polinomial` koja predstavlja polinom.

   Polinom stepena n u oznaci $P_n(x) = a_0 + a_1x + a_2x^2 + ... + a_{n-1}x^{n-1} + a_nx^n$ možemo reprezentovati nizom njegovih koeficijenata: `[a0, a1, a2, a3, ..., an-1, an]`

   - Podrazumevani konstruktor treba da konstruiše polinom $P_n(x) = 0$
   - Konstruktor koji prima niz treba da inicijalizuje koeficijente polinoma na prosleđene vrednosti

   Obezbediti metode:

   - `double getCoef(int i)`, vraća vrednost a_i (za gore prikazani polinom $P_n(x)$)

- `int getDeg()`, vraća stepen polinoma, za $P_n(x)$ je to n
- `double evaluate(double x)`, evaluira $P_n(x)$ za prosleđeno x i vraća rezultat
- `Polinomial add(Polinomial t)`, vraća `zbir` dva polinoma
- `Polinomial subtract(Polinomial t)`, vraća `razliku` dva polinoma
- `Polinomial multiply(Polinomial t)`, vraća `proizvod` dva polinoma

Obezbediti implementaciju funkcije `String toString()` koja polinom prikazuje kao u primeru:

```
> 2 + 3x
> 2 - 4x^2
> 2 + x + x^2 - 4x^3
> 1 + x + x^2 + x^3 + x^4 + x^5 + x^6 + x^7 + x^8 + x^9 + x^10
> 1 - x + x^2 - x^3 + x^4 - x^5 + x^6 - x^7 + x^8 - x^9 + x^10
> 0
> - 1 + x^21
```

Hornerova šema (dodatni zadatak) Implementirati metod `double hornerEval(double x)` koji evaluira polinom za dato `x` koristeću Hornerovu šemu. Standardno evaluiranje polinoma zahteva najviše n sabiranja i $(n^2+n)/2$ množenja. Hornerova metoda nam omogućava da imamo svega n sabiranja i n množenja.

Implementacija treba da bude rekurzivna.

$$P_n(x) = a_0 + x(a_1 + x(a_2 + ... + x(a_{n-1} + a_n x)...))$$