

Zadaća 1

Razvoj softvera

Dr.sc. Emir Mešković, vanr.prof.
As. Bakir Agić, bach.ing.el.

April 01, 2025

Sadržaj

1 Problem 1	2
1.1.....	2
1.2.....	2
1.3.....	2
1.4.....	3
1.5.....	3
1.6.....	3
2 Problem 2	4
3 Problem 3	4
4 Problem 4	5
5 Problem 5	6
6 Problem 6	6
6.1.....	6
6.2.....	7
7 Problem 7	7
8 Problem 8	8

1 Problem 1

1.1

Da li će se kompajliranje i izvršavanje naredne klase uspješno obaviti? Dati detaljno obrazloženje odgovora.

```
public class Assignment {  
    public static void main(String[] args) {  
        int a, b, c;  
        b = 10;  
        a = b = c = 20;  
        System.out.println(a);  
    }  
}
```

1.2

Šta će biti ispisano nakon izvršenja sljedećeg segmenta Java programa? Dati detaljno obrazloženje odgovora.

```
int[] niz = { 4, 8, 16 };  
int i=1;  
niz[++i] = --i;  
System.out.println(niz[0] + niz[1] + niz[2]);
```

1.3

Šta će biti rezultat kompajliranja sljedećeg programa?

```
public class MyClass {  
    long var;  
    public void MyClass(long param) {  
        var = param; } // (1)  
    public static void main(String[] args) {  
        MyClass a, b;  
        a = new MyClass(); // (2)  
        b = new MyClass(5); // (3)  
    }  
}
```

- a) kompajler će javiti grešku na (1), jer konstruktor ne može specificirati povratnu vrijednost..
- b) kompajler će javiti grešku na (2), jer klasa nema podrazumjevani konstruktor.
- c) kompajler će javiti grešku na (3), jer klasa nema konstruktor koji uzima jedan argument tipa int.
- d) Program će se uspješno kompajlirati.

1.4

Koji iskaz je tačan u pogledu pristupa članovima?

- a. Privatnim članovima se uvijek može pristupiti unutar istog paketa
- b. Privatnim članovima se može pristupiti iz koda klase kojoj pripadaju.
- c. Članu sa podrazumjevanim pristupom može se pristupiti iz svake podklase klase u kojoj je definisan.
- d. Privatnim članovima se uopšte ne može pristupiti.

1.5

Šta se dešava tokom izvršenja sljedećeg programa?

```
int i = 0;  
int[] a = {3,6};  
a[i] = i = 9;  
System.out.println(i + " " + a[0] + " "  
+ a[1]);
```

- a) Ispisuje "9 9 6"
- b) Ispisuje "9 0 6"
- c) Ispisuje "9 3 6"
- d) Ispisuje "9 3 9"

1.6

Magični kvadrat n-tog reda je kvadratna shema $n \times n$ brojeva od 1 do n^2 u kojoj je zbir u svakom redu, koloni i dijagonali jednak.

2	7	6	→ 15
9	5	1	→ 15
4	3	8	→ 15
← 15	↓ 15	↓ 15	↓ 15

Potrebno je iz dva tekstualna fajla u prilogu zadaće učitati cijele brojeve i provjeriti da li su zadani kvadrati magični.

Napomena: Za čitanje podataka iz fajla koristiti klasu `BufferedReader` i njen metod `readLine()`. Za podjelu linije teksta na niz stringova prema tabu (`\t`) koristiti

`split()` metod klase `String`. Za konverziju `String` objekta u njegov `int` ekvivalent koristiti statički metod `parseInt` klase `Integer`.

2 Problem 2

Polinom realne varijable x je izraz oblika $a_n x^n + a_{n-1} x^{n-1} + \dots + a_1 x + a_0$ gdje su a_0, \dots, a_n realne konstante koje se nazivaju koeficijenti. Npr. $2.5x^3 + 1.2x - 5$ je polinom od x sa $a_3 = 2.5$, $a_2 = 0$, $a_1 = 1.2$, $a_0 = -5$. Potrebno je razviti klasu pod nazivom *Polinom* koja podržava simboličku manipulaciju polinomima fiksne varijable x , tj. svi polinomi su polinomi iste varijable x . Klasa treba podržati operacije za:

1. Konstruisanje polinoma sa datim nizom koeficijenata. Npr., za dati niz realnih brojeva $\{2.5, 0.0, 1.2, -5.0\}$ operacija konstruiše objekat *Polinom* koji predstavlja $2.5x^3 + 1.2x - 5$. Ovo je konstruktor.
2. Sabiranje polinoma. Npr. ukoliko *this* objekat predstavlja $2.5x^3 + 1.2x - 5$ onda dodavanje drugog objekta koji predstavlja $7x^4 + 2x^3 + x^2 + 3$ *this* objektu mijenja *this* objekat tako da predstavlja $7x^4 + 5.5x^3 + x^2 + 1.2x - 2$. Primjetite da stepen (najveći eksponent) dva polinoma ne mora biti jednak.
3. Množenje polinoma. Npr. ukoliko *this* objekat predstavlja $2x^2 + x - 5$ onda njegovo množenje sa drugim objektom koji predstavlja $3x^3 + x^2 + 3$ mijenja *this* objekat tako da predstavlja $6x^5 + 5x^4 - 14x^3 + x^2 + 3x - 15$. Ponovo stepen dva polinoma ne mora biti jednak.
4. Izračunavanje polinoma za datu vrijednost x . Npr. ukoliko *this* objekat predstavlja $2x^2 + x - 5$ onda njegovo izračunavanje za $x = 2$ daje 5.
5. Konverzija *Polinom* objekta u string tako da *Polinom* objekat može biti prikazan. Ukoliko *this* objekat predstavlja $3x^3 + x^2 - 1$ onda bi metod trebao vratiti „ $3*x^3 + x^2 - 1$ “ mada je prihvatljivo i „ $3*x^3 + x^2 0*x^1 + (-1)$ “

3 Problem 3

Napisati javnu klasu `Queue` koja služi kao kontejner za cijele brojeve. Klasa radi po FIFO (first-in-first-out) principu i treba biti implementirana pomoću povezane liste (linked-list). Klasa treba da sadrži metode: `push(a)` – koji dodaje broj u red, `pop()` – koji izbacuje broj koji je prvi dodan u red (ujedno vraća i njegovu vrijednost), `end()` – koji vraća vrijednost broja koji je prvi dodan u red (bez njegovog izbacivanja iz reda) i `isEmpty()` – koji vraća `true` ako je red prazan. Napisati program koji puni `Queue` objekat sa listom cijelih brojeva koja je proslijeđena kao argument programu. Program zatim, koristeći `pop` metod, ispisuje sve unijete elemente svaki u zasebnoj liniji npr:

```
$ java Queue 4 5 6 19 2 3 123
4
5
6
19
2
```

3
123

Napomena: Klasa `Queue` treba da definiira samo navedene metode. Pomoćne klase trebaju biti implementirane u istom fajlu. Za konverziju `String` objekta u njegov `int` ekvivalent koristiti statički metod `parseInt` klase `Integer` npr:

```
String a = "1234";  
int p = Integer.parseInt(a);
```

4 Problem 4

Potrebno je implementirati tzv. „drevni egipatski“ metod množenja. Algoritam je sljedeći: Ukoliko su *A* i *B* dva cijela broja (samo cijela broja) koja je potrebno pomnožiti, onda se *A* množi sa 2, a *B* dijeli sa 2, sve dok *B* više nije moguće podijeliti, tj. dok njegova vrijednost ne bude 0 (zapamtiti, riječ je o cjelobrojnom dijeljenju). Tokom svakog koraka, kad god je *B* neparan broj, odgovarajuća *A* vrijednost iz tog koraka se dodaje proizvodu koji se generiše. Na kraju, suma *A* vrijednosti kada je *B* bio neparan broj predstavlja proizvod. Npr. ukoliko su dva cijela broja koja je potrebno pomnožiti 34 i 19, operacije bi bile:

A	B	Komentar
34	19	Dodaj A proizvodu, B je neparan
68	9	Dodaj A proizvodu, B je neparan
136	4	Ignoriši vrijednost A, B je paran
272	2	Ignoriši vrijednost A, B je paran
544	1	Dodaj A proizvodu, B je neparan

Sabrati sve *A* vrijednosti za koje je *B* neparan i dobija se: $34 + 68 + 544 = 646 \Rightarrow$ Konačni proizvod

Specifikacije:

- korisnik bi trebao unijeti dva cijela broja razdvojena razmakom
- prilikom izračunavanja proizvoda prikazati kako algoritam napreduje, što osigurava da se koristi zahtjevani algoritam
- nakon izračunavanja proizvoda, indicirati njegov znak (pozitivan, negativan ili nula)
- program treba dati korektne rezultate za sve cijele brojeve, tj. kako za pozitivne, tako i za negativne brojeve.

Primjeri poziva programa dati su u nastavku:

```
$ java proizvod 22 33  
A = 22 i B = 33  
B je neparno, dodajemo A za kreiranje proizvoda: 22  
A = 44 i B = 16  
A = 88 i B = 8  
A = 176 i B = 4
```

```

A = 352 i B = 2
A = 704 i B = 1
B je neparno, dodajemo A za kreiranje proizvoda: 726
Proizvod je pozitivan
Proizvod dva broja je: 726
=====
$ java proizvod -22 33
A = -22 i B = 33
B je neparno, dodajemo A za kreiranje proizvoda: -22
A = -44 i B = 16
A = -88 i B = 8
A = -176 i B = 4
A = -352 i B = 2
A = -704 i B = 1
B je neparno, dodajemo A za kreiranje proizvoda: -726
Proizvod je negativan
Proizvod dva broja je: -726
=====
$ java proizvod -22 -33
A = -22 i B = 33
B je neparno, dodajemo A za kreiranje proizvoda: -22
A = -44 i B = 16
A = -88 i B = 8
A = -176 i B = 4
A = -352 i B = 2
A = -704 i B = 1
B je neparno, dodajemo A za kreiranje proizvoda: -726
Proizvod je pozitivan
Proizvod dva broja je: 726

```

Napomena: Za konverziju String objekta u njegov int ekvivalent koristiti statički metod `parseInt` klase `Integer` npr:

```

String a = "1234";
int p = Integer.parseInt(a);

```

5 Problem 5

Detaljno objasnite razlike između Java interface-a i apstraktnih klasa.

6 Problem 6

6.1

Deklarisati interfejs pod nazivom `Funkcija` koji ima metod pod nazivom `izracunaj` koji uzima `int` parametar i vraća `int` vrijednost. Kreirati klasu `Kvadrat` koja implementira interfejs `Funkcija`. Neka implementacija metoda `izracunaj()` vraća vrijednost dobijenu kvadriranjem `int` argumenta.

U klijentu, kreirati metod koji prihvata proizvoljan niz `int` vrijednosti kao parametar i vraća niz koji ima istu dužinu, ali vrijednost elementa u novom nizu je kvadrat vrijednosti odgovarajućeg elementa u nizu proslijeđenom kao parametar. Neka implementacija ovog metoda kreira instancu od `Kvadrat` i koristi ovu instancu za izračunavanje vrijednosti u nizu.

6.2

Ponovo napisati metod iz prethodnog zadatka koji radi sa nizovima: metod bi sada trebao kao argument uzeti referencu na interfejs `Funkcija` i koristiti je umjesto kreiranja instance `Kvadrat`. Kreirati klasu pod nazivom `Ispis` koja implementira interfejs `Funkcija`, ima metod koji jednostavno ispisuje `int` vrijednost datu kao argument i vraća tu vrijednost.

Napisati program koji kreira niz `int` vrijednosti od 1 do 10 i radi sljedeće:

- Ispisuje niz koristeći instancu `Ispis` klase i metod opisan ranije.
- Kvadrira vrijednosti u nizu i ponovo ispisuje vrijednosti, korištenjem `Kvadrat` i `Ispis` klase i metoda opisanih iznad.

7 Problem 7

Data je sljedeća Java klasa:

```
class Tester
{
    public static void main(String[] args)
    {
        Oblik[] a = new Oblik[2];
        a[0] = new Krug(3);
        a[1] = new Pravougaonik(2,3);
        for (int k = 0; k < a.length; ++k)
            System.out.println(a[k].povrsina());
    }
}
```

Gdje `Krug` i `Pravougaonik` definišu geometrijska tijela. `Krug` se inicijalizira sa poluprečnikom a `Pravougaonik` sa svojim stranicama.

Na osnovu gornjeg koda u kakvoj relaciji su `Krug` i `Pravougaonik` sa `Oblik`-om? Koja je minimalna definicija za `Oblik` da bi gornji program ispravno radio? U skladu sa vašim odgovorom u jednom java fajlu definirati `Oblik`, `Krug` i `Pravougaonik` te klasu `Tester`, kako bi gornji program ispravno računao površine.

8 Problem 8

Data je sljedeća Java klasa:

```
import zadaca2.MojStack;
import zadaca2.StackList;
import zadaca2.StackArray;

public class TestStack
{
    public static void main(String[] args)
    {
        MojStack a = new StackArray(2);
        a.push("String");
        a.push(new Integer(2));
        a.push(new Double(2.1));
        a.push(new Integer(10));
        while(!a.isEmpty())
            System.out.println(a.pop());
        try
        {
            a.pop();
        }
        catch (Exception e)
        {
            e.printStackTrace();
        }
        a = new StackList();
        a.push(new Integer(10));
        System.out.println(a.pop());
        a.pop();
    }
}
```

Gdje `StackList` implementira kontejner tipa `stack` pomoću vezane liste. `StackArray` implementira kontejner `stack` pomoću Java niza, a inicijalizira se sa početnim kapacitetom niza. Kapacitet niza povećava se dva puta svaki put kada `stack` dostigne maksimalni trenutni kapacitet. Definirati `MojStack`, `StackList` i `StackArray` kako bi gornji program bilo moguće kompajlirati te da program proizvede izlaz identičan slijedećem:

```
$ java TestStack
10
2.1
2
String
java.lang.RuntimeException: Stack je prazan
    at zadaca2.StackArray.pop(StackArray.java:16)
    at TestStack.main(TestStack.java:18)
10
```



```
Exception in thread "main" java.lang.RuntimeException: Stack je  
prazan  
    at zadaca2.StackList.pop(StackList.java:34)  
    at TestStack.main(TestStack.java:27)
```

Napomena: obratiti pažnju na pripadnost gornjih klasa paketima, tip elemenata u kontejnerima te tip generirane greške u ispisu programa. Brojevi linija ispisa StackTrace-a ne moraju biti isti. Sve definirani tipovi trebaju imati minimum funkcionalnosti da bi zadovoljili rad gornjeg programa.