2. laboratorijska vježba – Proceduralna paradigma (6 bodova)

Zadatak 1.

Napišite program `Rectangle` (smjestite ga u paket `hr.fer.oop.lab2.prob1`). Program treba omogućiti da korisnik unese dva podatka - širinu i visinu pravokutnika na dva načina: kao argumente naredbenog retka te čitanjem podataka sa standardnog ulaza. Na temelju dobivenih podataka program računa i ispisuje površinu i opseg pravokutnika. Evo pravila za slučaj kad program podatke dobije kao argumente naredbenog retka.

1. Program provjerava da li je dobio dva podatka, ako broj podataka ne odgovara o tome obavještava korisnika i prekida izvođenje.
2. Ako broj podataka odgovara, izvodi se račun i ispis.

Evo pravila za slučaj kad program čita sa standardnog ulaza.

1. Ako korisnik unese bilo što, možete pretpostaviti da je to broj (ne morate sada brinuti o upravljanju pogreškama; to ćemo raditi kad naučimo kako).
2. Ako ništa nije uneseno, program to dojavljuje i traži korisnika da ponovno unese podatke.
3. Ako korisnik zada negativan broj, dojavite to i tražite korisnika da ponovno unese podatke.
4. Nemojte raditi copy&paste dijelova koda koji čita širinu i visinu (praktički isti dio koda, samo s različitim porukama) – ili izdvojite taj kod u novu metodu s prikladnim argumentima, ili koristite polja i petlje.

Primjer prikazuje očekivano ponašanje programa (korisnički unos prikazan je crvenim).

```
C:\oop\prob1> java -cp bin hr.fer.oop.lab2.prob1.Rectangle
Please provide width:
The input must not be blank.
Please provide width: -43
The width must not be negative.
Please provide width: 10
Please provide height: -10
The height must not be negative.
Please provide height: 20.5
You have specified a rectangle of width 10.0 and height 20.5. Its area is 205.0
and its perimeter is 61.0.
C:\oop\prob1> java -cp bin hr.fer.oop.lab2.prob1.Rectangle 25
Invalid number of arguments was provided.
C:\oop\prob1> java -cp bin hr.fer.oop.lab2.prob1.Rectangle 25 10
You have specified a rectangle of width 25.0 and height 10.0. Its area is 250.0
and its perimeter is 70.0.
```

Napomena: ako za čitanje s tipkovnice koristite `Scanner`, kada pročitate redak koristite metodu `trim()` kako biste uklonili praznine ispred i iza podatka; koristite metodu `isEmpty()` kako biste utvrdili je li ono što je preostalo prazan redak. Ove metode pripadaju razredu `String`. Pogledajte dokumentaciju:

http://docs.oracle.com/javase/8/docs/api/java/lang/String.html

Zadatak 2.

Dovršite sljedeći program:

```
package hr.fer.oop.lab2.prob2;

class TreeNode {
    TreeNode left;
    TreeNode right;
    String data;
}

class TreeProgram {

  public static void main(String[] args) {
      TreeNode node = null;

      node = insert(node, "Jasna");
      node = insert(node, "Ana");
      node = insert(node, "Ivana");
      node = insert(node, "Anamarija");
      node = insert(node, "Vesna");
      node = insert(node, "Kristina");

      System.out.println("Writing tree inorder:");
      writeTree(node);

      node = reverseTreeOrder(node);

      System.out.println("Writing reversed tree inorder:");
      writeTree(node);

      int size = sizeOfTree(node);
      System.out.println("Number of nodes in tree is "+size+".");

      boolean found = containsData(node, "Ivana");
      System.out.println("Searched element is found: "+found);
  }

  static boolean containsData(TreeNode treeRoot, String data) {
    // ...
  }

  static int sizeOfTree(TreeNode treeRoot) {
    // ...
  }

  static TreeNode insert(TreeNode treeRoot, String data) {
    // ...
  }

  static void writeTree(TreeNode treeRoot) {
    // ...
  }

  static TreeNode reverseTreeOrder(TreeNode treeRoot) {
    // ...
  }
}
```

Napomena:
Neka metoda `insert` pretpostavlja da lijevo u stablu smješta manje vrijednosti a desno veće. Provjeru koji je od stringova veći možete obaviti njihovom metodom compareTo: provjerite u dokumentaciji (https://docs.oracle.com/javase/8/docs/api/java/lang/String.html) kako se ista koristi.

Metodu `containsData` napišite uz pretpostavku da je stablo složeno po načelu: lijevo manje vrijednosti, desno veće. U kojoj složenosti radi takva metoda? Hoće li ta metoda pronalaziti elemente u stablu dobivenom pozivom metode `reverseTreeOrder`? Napišite metodu `containsData2` koja element (ako postoji) pronalazi garantirano, neovisno o trenutnom poretku elemenata u stablu. U kojoj složenosti radi takva metoda? Zašto?

Zadatak 3.

Napišite program `Roots` (smjestite ga u paket `hr.fer.oop.lab2.prob3`). Program kao argumente naredbenog retka prihvaća tri podatka: realni dio kompleksnog broja, imaginarni dio kompleksnog broja te korijen koji je potrebno izračunati (korijen se zadaje kao prirodni broj veći od 1). Program računa i ispisuje sve korijene zadanog kompleksnog broja (također u obliku realni dio plus imaginarni dio). U slučaju da zatrebate trigonometrijske funkcije (ili slične), poslužite se metodama razreda `Math` – dokumentacija je ovdje:
http://docs.oracle.com/javase/8/docs/api/java/lang/Math.html

Primjer uporabe:

```
C:\oop\prob3> java -cp bin hr.fer.oop.lab2.prob3.Roots 3 4 2
You requested calculation of 2. roots. Solutions are:
1) 2 + 1i
2) -2 - 1i
```

Zadatak 4.

Napišite program `PrimeNumbers` (smjestite ga u paket `hr.fer.oop.lab2.prob4`). Program prihvaća jedan argument preko naredbenog retka: broj n (n>0) te računa i uzlazno ispisuje prvih n prostih brojeva. Pretpostavite da je broj 2 prvi prosti broj.

Primjer uporabe:

```
C:\oop\prob4> java -cp bin hr.fer.oop.lab2.prob4.PrimeNumbers 4
You requested calculation of first 4 prime numbers. Here they are:
1. 2
2. 3
3. 5
4. 7
```

Zadatak 5.

Napišite program `PrimeFactorization` (smjestite ga u paket `hr.fer.oop.lab2.prob5`). Program prihvaća jedan argument preko naredbenog retka: prirodni broj veći od 1. Potom računa i ispisuje rastav tog broja na proste faktore.

Primjer uporabe:

```
C:\oop\prob5> java -cp bin hr.fer.oop.lab2.prob5.PrimeFactorization 84
You requested decomposition of number 84 into prime factors. Here they are:
1. 2
2. 2
3. 3
4. 7
```

Zadatak 6.

Napišite program `Shapes` (smjestite ga u paket `hr.fer.oop.lab2.prob6`) koji će na izlaznoj jedinici uz pomoć standardnih znakova iz ASCII tablice nacrtati četiri oblika: pješčani sat, geometrijski lik, šalicu i kapu kako je prikazano:

```
+--------+
\        /
 \______/

  ______
 /      \
/        \
+--------+

  ______
 /      \
/        \
\        /
 \______/

\        /
 \______/
+--------+

  ______
 /      \
/        \
+--------+
```

U programu nemojte koristiti *new* operator (tj. nemojte kreirati objekte). Također, u programu smije postojati više poziva naredbe za ispis na izlaznu jedinicu (*System.out.println*), ali ne smiju postojati dva poziva s identičnom vrijednošću parametra. Identične dijelove različitih likova potrebno je iscrtavati istom metodom.