1. laboratorijska vježba: prvi dio zadataka

Zadatak 1.

Napišite program `Rectangle` (smjestite ga u paket `hr.fer.oop.lab1.topic1.prob1`). Program treba tražiti korisnika unos širine i visine pravokutnika (jedan unos po retku, program čita sa standardnog ulaza). Program treba izračunati i ispisati površinu i opseg pravokutnika. Ako podatke dobije kao argumente naredbenog retka, tada ništa ne pita korisnika već koristi te podatke. Evo pravila za slučaj da 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.

Evo primjera koji prikazuje očekivano ponašanje programa (korisnički unos prikazan je crvenim).

```
C:\oop\prob1> java -cp bin hr.fer.oop.lab1.topic1.prob1.Rectangle
Please provide width:
Nothing was given.
Please provide width: -43
Width is negative.
Please provide width: 10
Please provide height: -10
Height is negative.
Please provide height: 20.5
You have specified a rectangle with 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.lab1.topic1.prob1.Rectangle 25
Invalid number of arguments was provided.
C:\oop\prob1> java -cp bin hr.fer.oop.lab1.topic1.prob1.Rectangle 25 10
You have specified a rectangle with 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 `BufferedReader`, 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.lab1.topic1.prob2;

class TreeProgram {

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

  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) {
    // ...
  }
}
```

Zadatak 3.

Napišite program `HofstadterQ` (smjestite ga u paket `hr.fer.oop.lab1.topic1.prob3`). Program računa *i*-ti broj Hofstadterovog Q-slijeda (pronađite definiciju na Internetu; u dokumentaciju obavezno uključite link na stranicu na kojoj ste pronašli definiciju). Koristite tip `long` za izračune. Program kao argument naredbenog retka prihvaća *i*. Taj argument mora biti pozitivan; ako nije, prijavite pogrešku i prekinite program.

Primjer uporabe:

```
C:\oop\prob3> java -cp bin hr.fer.oop.lab1.topic1.prob3.HofstadterQ 10
You requested calculation of 10. number of Hofstadter's Q-sequence. The requested
number is 6.
```

Zadatak 4.

Napišite program `Roots` (smjestite ga u paket `hr.fer.oop.lab1.topic1.prob4`). 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\prob4> java -cp bin hr.fer.oop.lab1.topic1.prob4.Roots 3 4 2
You requested calculation of 2. roots. Solutions are:
1) 2 + 1i
2) -2 - 1i
```

Problem 5.

Napišite program `PrimeNumbers` (smjestite ga u paket `hr.fer.oop.lab1.topic1.prob5`). Program prihvaća jedan argument preko naredbenog retka: broj *n* (*n*>0) te računa i ispisuje prvih *n* prostih brojeva. Pretpostavite da je broj 2 prvi prosti broj.

Primjer uporabe:

```
C:\oop\prob5> java -cp bin hr.fer.oop.lab1.topic1.prob5.PrimeNumbers 4
You requested calculation of 4 prime numbers. Here they are:
1. 2
2. 3
3. 5
4. 7
```

Zadatak 6.

Napišite program `NumberDecomposition` (smjestite ga u paket `hr.fer.oop.lab1.topic1.prob6`). 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:\prob6> java -cp bin hr.fer.oop.lab1.topic1.prob6.NumberDecomposition 84
You requested decomposition of number 84 onto prime factors. Here they are:
1. 2
2. 2
3. 3
4. 7
```

Napomena. Slobodno o ovim zadatcima razgovarajte s Vašim kolegama *prije* no što ih počnete rješavati. Jednom kada pokrenete Vaš omiljeni IDE i krenete kodirati, konzultacije s drugima (osim s nastavnim osobljem kolegija) smatrat će se varanjem. Ne smijete koristiti nikakve prethodno razvijene biblioteke (bilo tuđe, bilo Vaše); smijete koristiti samo standardne biblioteke koje su uključene u JRE. Međutim, pri rješavanju ovih zadataka ne smijete koristiti niti razrede koji čine Javin okvir kolekcija (engl. *Java Collection Framework*) ili koji su izvedeni iz njega; ako niste sigurni smijete li nešto koristiti, pitajte nas. Dokumentirajte kod. Ako već znate nešto o objektno-orijentiranom oblikovanju koda (primjerice, uporaba konstruktora i slično), pri rješavanju ovih zadataka nemojte koristiti to znanje – ideja je za početak izvježbati se u pisanju proceduralnog koda (kao da pišete u C-u).

Da biste riješili ove zadatke, za svaki zadatak napravite po jedan projekt (u Eclipse-u ili NetBeansu). Kad ste gotovi, za svaki zadatak koji ste rješavali eksportajte samo sadržaj direktorija `src` kao ZIP-arhivu. Provjerite sadržaj ZIP arhive koju ste napravili: prilikom otvaranja, u njoj se mora kao vršni direktorij nalaziti direktorij `src`. Potom se prijavite na Ferka (prijavu radite isptim podatcima koje koristite i za FERWeb); otiđite na stranicu kolegija OOP i odaberite "Komponente kolegija", "Prva laboratorijska vježba". Tamo ćete za svaki od ovdje zadanih 6 zadataka pronaći mjesto gdje trebate uploadati napravljenu ZIP arhivu. Ako ste rješili svih 6 zadataka, radite 6 uploada. Upload je moguće napraviti do ponedjeljka, 20. listopada do 07:30 ujutro. Nemojte zaboraviti zaključati upload jer ga inače nećemo priznati.