Objektno orijentirano programiranje
međuispit

18. studenog 2014.

Ispit nosi ukupno 30 bodova i piše se 120 minuta.

1. zadatak (5 bodova)

Dane su definicije dvije klase. Navedite što će ispisati metoda `main` te objasnite razlog takvom ispisu.

```
public class BaseClass {
    private final String message1 = "message1";
    private static final String message2 = "message2";

    public String getMessage() {
        return message1;
    }
    public static String getStaticMessage() {
        return message2;
    }
}

public class ExtendedClass extends BaseClass {
    private final String message3 = "message3";
    private static final String message4 = "message4";

    @Override
    public String getMessage() {
        return message3;
    }
    public static String getStaticMessage() {
        return message4;
    }
    public static void main(String[] args) {
        ExtendedClass ec = new ExtendedClass();
        System.out.println(ec.getMessage());
        System.out.println(ec.getStaticMessage());

        BaseClass bc = ec;
        System.out.println(bc.getMessage());
        System.out.println(bc.getStaticMessage());
    }
}
```

2. zadatak (5 bodova)
Objasnite što su to konstruktori i čemu služe te na koji način se implementiraju i koriste. Objasnite redoslijed pozivanja konstruktora prilikom stvaranja primjerka izvedenog razreda i ilustrirajte prikladnim odsječkom koda.

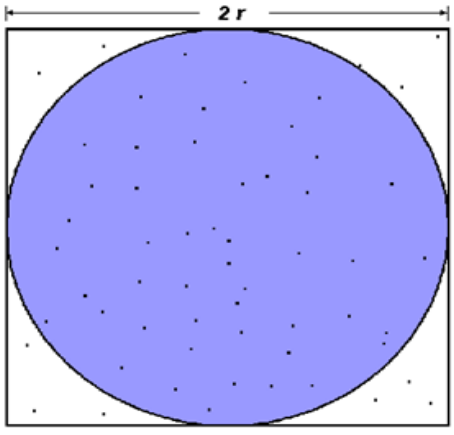
3. zadatak (10 bodova)
Potrebno je osmisliti klase i/ili sučelja koje modeliraju izračun broja PI. Broj PI se može izračunati na mnogo različitih načina, no kao rješenje ovog zadatka je potrebno napraviti dvije implementacije:

1. Jednostavnu implementaciju koja vraća vrijednost broja PI iz Javine matematičke knjižnice
2. Složeniju implementaciju čiji je algoritam izračuna opisan na kraju zadatka

U rješenju zadatka je potrebno napraviti apstraktne klase ili sučelja (sami ocijeniti što je u ovome slučaju potrebno) kojima se na načelnoj razini modelira izračun broja PI. Također, potrebno je te klase/sučelja naslijediti/implementirati na način da se stvore konkretne klase u kojima je implementacija izračuna opisanih točkama 1 i 2.

Napomena: U konačnici, predloženi model mora omogućavati trećim metodama izračun broja PI na način da nisu (i ne moraju biti) svjesne konkretne implementacije algoritma (je li to algoritam 1 ili 2).Ako smatrate da je potrebno, za složeniji izračun se mogu izraditi posebne klase. Ali, to nije nužno, već će biti dovoljno da složeniji izračun implementirate adekvatnim metodama.

<u>Opis složenijeg izračuna broja PI (točka 2 u zadatku gore):</u>
Izračun broja PI može se aproksimirano izračunati metodom koja se temelji na kvadratu i odgovarajućem upisanom krugu. Slučajnim odabirom potrebno je generirati koordinate točaka unutar kvadrata pri čemu PI odgovara umnošku broja 4 i broja točaka unutar kruga podijeljenom s brojem točaka unutar kvadrata.
Napomena: broj nasumičnih točaka koje se ukupno generira treba u glavnom programu biti učitan od strane korisnika i proslijeđen (sami odredite kako) ovome algoritmu. Za određivanje vrijednosti radijusa (r) koristiti konstantu i postaviti je na vrijednost 1000. Posebnu pažnju obratiti na dijelove programa koji mogu prouzrokovati greške (npr. dijeljenje s nulom) te ih osigurati adekvatnim iznimkama.



$$A_S = (2r)^2 = 4r^2$$
$$A_C = \pi r^2$$
$$\pi = 4 \times \frac{A_C}{A_S}$$

Napomena: As je površina kvadrata (u ovisnosti o polumjeru kruga r), a Ac je površina kruga.

4. zadatak (10 bodova)

Razmotrite sljedeći problem. Želimo napisati program koji kao argument naredbenog retka prima jedan argument: direktorij. Program mora u tom direktoriju i svim njegovim poddirektorijima odrediti koliko ima slika. Slike su datoteke čije su ekstenzije "PNG", "JPG" i "GIF" (neovisno o velikim i malim slovima). Na zaslon treba ispisati konačni broj pronađenih slika.

Napišite čitav program koji taj zadatak rješava isključivo uporabom razreda `File`.

Dodatak (predavanja – rad s datotekama)

```
public static void main(String[] args) {
 File dir = new File("d:/tmp/javaPrimjeri");
 File file1 = new File(dir, "readme.txt");
 File file2 = new File("d:/tmp/javaPrimjeri/readme.txt");

 //korisne metode klase File
 File parent = file1.getParentFile();
 boolean exists = file2.exists();
 boolean readable = file2.canRead();
 boolean writeable = file2.canWrite();
 boolean executable = file2.canExecute();
 long fileSize = file2.length();
 boolean isFile = file2.isFile();
 boolean isDirectory = file2.isDirectory();
 boolean isHidden = file2.isHidden();
 String fileName = file2.getName();
 File[] files = dir.listFiles();

 //korisne metode klase String
 boolean contains = fileName.contains("xyz");
 boolean endsWith = fileName.endsWith(".xyz");
 boolean startsWith = fileName.startsWith("xyz");
}
```