Programski jeziki - JAVA

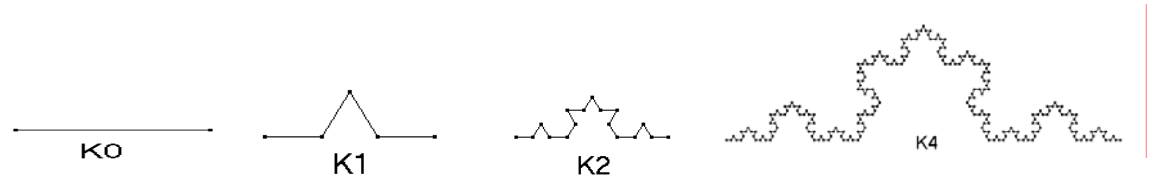
Ime in priimek: _____

Točk: _____ / 120 (100% = 100 točk)

[20 T] Dan je razred `LogoZelva` z objektnimi metodami `public void fd(int d)`, `public void bk(int d)`, `public void right(int kot)` in `public void left(int kot)`. Prva premakne želvo naprej za d enot (pri tem se seveda nariše črta dolžine d), druga jo premakne nazaj za d enot (tudi tu nastane črta dolžine d). Smer želve se pri obeh ukazih ne spremeni. Metoda `right` spremi smer želve za dani parameter stopinj v desno, metoda `left` pa obrne smer želve v levo. Pri teh dveh metodah se položaj želve ne spremeni, pa tudi ne nariše se nič.

Sestavi statično metodo `public static void crtaK(LogoZelvo zelva, int n, int d)` ter nariše sliko, kot je prikazano na spodnjih slikah. Prvi parameter je objekt tipa `LogoZelva` s katerim rišemo, drugi parameter določa red slike (spodnje slike so redov 0, 1, 2 in 4) in tretji dolžino črte (vodoravno). Deli črte K1 so tretjine vodoravne dolžine.

Namig: preprosta rešitev je rekurzivna (premisli, kako sliko reda *n* dobiš iz slik nižjega reda).




```
// Izris kochovo krivuljo reda n in dolzine d
// Pri risanju uporabi logozelvo
// Logozelva je po koncu metode na koncu krivulje in
// gleda v isto smer kot na zacetku

public static void crtaK(LogoZelva zelva, int n, int d)  {
  //ustavitveni pogoj - ce je stopnja enaka 0, narisemo le črto dolzine d
  if(n == 0) {
    zelva.fd(d);
  }
  else {
    //iz slik opazimo, da je krivulja stopnje n sestavljena iz štirih
    //kochovih krivulj stopnje (n-1) dolzine d/3.
    //Narisemo vsako posebej, pri tem pa pazimo na smer zelve.
    crtaK(zelva, n - 1, d / 3);
    zelva.left(60);
    crtaK(zelva, n - 1, d / 3);
    zelva.right(120);
    crtaK(zelva, n - 1, d / 3);
    zelva(left(60);
    crtaK(zelva, n - 1, d / 3);
  }
}
```

[20 T] Janez in Peter sta izumila poseben kodni sistem. Najprej si pošljeta tabelo nizov. Nato si za vsako sporočilo izmenjata seznam števil, izmed katerih vsaka predstavlja besedo, ki je na seznamu nizom številčena s tem številom. Pomagaj jima pri hitrejšem dekodiranju in napiši metodo `public static String dekodiraj(String[] nizi, int[] koda)`, ki sprejme seznam nizov in kodo ter vrne odkodirani niz. Tako del programa

```
String[] nizi
  = {"to","je","res","odlična","metoda","za","kodiranje","metka","ni","slaba","je","dobra"};
```

```
int mojeSporocilo = {2, 9, 4, 5, 6};

System.out.println(dekodiraj(nizi, mojeSporocilo);
```

izpiše niz "res slaba metoda za kodiranje".

```
// Metoda, ki dekodira sporocilo. Dobi seznam besed in kodo,
// ki je predstavljena s tabelo
// celih stevil. Odkodiran niz je sestavljen iz besed,
// ki se v tabeli besed nahajajo na mestu, ki
// ga določa številka iz tabele kod.

public static String dekodiraj(String[] nizi, int[] koda) {
	//na začetku je dekodirani niz prazen, nato pa ga sproti gradimo
	String dekodirano = "";
	int stevec = 0;
	//ideja: za vsako število iz seznama koda bomo tabeli dekodiraj
	//dodali besedo, ki je v tabeli nizi oštevilčena s tem številom
	while(stevec < koda.length) {
		dekodirano = dekodirano + nizi[koda[stevec]] + " ";
		stevec = stevec + 1;
	}
	//vrnemo dekodiran niz
	return dekodirano;
}
```

[20 T] Sestavi metodo `String opisnaOcena(int t, int maxTock)`, ki sprejme število točk `t`, ki jih je dosegel študent in maksimalno število možnih točk `maxTock` in vrne opisno oceno "nezadostno" (0 – 49,999 %), "zadostno" (50 % - 59,999 %), "dobro" (60 % - 69,999 %), "prav dobro" (70 % - 89,999 %) ali "odlično" (90 % ali več).

Na primer `opisnaOcena(63, 100)` vrne niz "dobro".

```
//Vrne opisno oceno
public static String opisnaOcena(int t, int maxTock) {
	//izracunamo dosezene procente in glede na zahteve vrnemo niz
	double procent = 1.0*t / maxTock;
	if(procent < 0.5) {
		return "nezadostno";
	}
	else if(procent < 0.6) {
		return "zadostno";
	}
	else if(procent < 0.7) {
		return "dobro";
	}
	else if(procent < 0.9) {
		return "prav dobro";
	}
	else {
		return "odlicno";
	}
}
```

[20 T] V Sloveniji je registrska številka (zaenkrat še) sestavljena iz dveh nizov. Prvi niz je območje in vsebuje dva znaka, drugi niz pa je registracija in vsebuje pet poljubnih znakov. Na primer, registrska številka LJ V1-02E sestoji iz območja LJ in registracije V102E.
(a) Definiraj razred `Registracija`, ki vsebuje podatke o registrski številki avtomobila. Ker se ta načeloma nikoli ne spremeni, naj bo v razredu na voljo le konstruktor, ki sprejme oba niza ter metoda

`toString`, ki dano registrsko številko izpiše v obliki `xy_ab-cde`, kjer sta `xy` črki območja, `abcde` pa ustrezni znaki registracije.

(b) V razredu napiši še objektno metodo boolean veljavnoObmocje(), ki vrne true, če ima registerska številka veljavno območje. V Sloveniji so veljavna območja LJ, KR, KK, MB, MS, KP, GO, CE, SG, NM in PO.

Primer uporabe:

```
Registracija x = new Registracija("MM", "V102E");
boolean b = x.veljavnoObmocje(); // b == false
```

```
//Razred s podatki o registrskih stevilkah
public class Registracija {
    //veljavna obmocja - vsa enaka za vse tablice
    private final static String veljavnaObmocja = "LJ KR KK MB MS KP GO CE SG NM PO";
    //imemo dve spremenljivki - ombocje in registracija
    private String obmocje;
    private String registracija;

    //konstruktor - sprejme dva niza in nastavi obmocje in registacijo
    //paziti moramo na pravilnost podatkov!!
    public Registracija(String obmocje, String registracija)  {
        //ce je obmocje sestavljeno iz dveh znakov
        if(obmocje.length() == 2) {
                this.obmocje = obmocje;
        }
        //drugace obmocje nastavimo na LJ
        else {
                this.obmocje = "LJ";
        }

        //ce je registracija sestavljena iz petih znakov
        if(registracija.length() == 5) {
                this.registracija = registracija;
        }
        //drugace registracijo nastavimo na xxxxx
        else {
                this.registracija = "XXXXX";
        }
    }


    //Metoda pove, ce je obmocje veljavno. To izvemo tako, da pogledamo, ce je
    //niz obmocje vsebovan v nizu veljavnaObmocja
    public boolean veljavnoObmocje() {
        int index = veljavnaObmocja.indexOf(this.obmocje);
        //ce je index vecji ali enak nic, je obmocje pravilno
        if(index >= 0)
                return true;
        else
                return false;
    }


    //metoda vrne registracijo v obliki: OBMOCJE_AB-CDE, kjer so ABCDE znaki v registraciji
    public String toString() {
        return this.obmocje + "_" + registracija.substring(0,2) + "-" +
        registracija.substring(2,5);
    }
}
```

[20 T] Napiši program `Razmakni.java`, ki iz ukazne vrstice sprejme argument in ga izpiše tako, da vrine med znake besede presledke. Primeri:

```
> java Razmakni prestolonaslednik
p r e s t o l o n a s l e d n i k
> java Razmakni Zakuga?
Z a k u g a ?
> java Razmakni brez muje se se cevelj ne obuje
b r e z
> java Razmakni "brez muje se se cevelj ne obuje"
b r e z   m u j e   s e   s e   c e v e l j   n e   o b u j e
```

Opomba: pred prvim in za zadnjim znakom program ne izpiše presledka.

```
public class Razmakni {


    public static void main(String[] argumenti) {
      //Ideja: prvi niz iz tabele argumenti razmaknemo

      //Najprej preverimo, da sploh imamo argument. Ce ga nimamo,
      //izpisemo napako
      if(argumenti.length == 0) {
        System.out.println("Vnesti moras vsaj en argument!");
        return;
      }


      String razmakni = argumenti[0];
      //Sprehodimo se po znakih iz niza razmakni in izpisemo
      //vsako crko posebej. Za vsako crko izpisemo tudi presledek, razen ce je
      //crka zadnja v besedi.

      int stevec = 0;
      while(stevec < razmakni.length() )  {
        //izpisemo crko
        System.out.print(razmakni.charAt(stevec));
        //ce crka ni zadnja, izpisemo tudi presledek
        if(stevec != razmakni.length() - 1) {
               System.out.print(" ");
        }
        stevec = stevec + 1;
      }
      //na koncu se premaknemo se novo vrstico
      System.out.println("");
    }
}
```

[20 T] Karl Gauss pri pouku ni bil priden, zato mu je profesorica dala izračunati vsoto prvih 100 naravnih števil. Ker jo je presenetil s hitrim odgovorom, ga je sklenila bolj zaposliti. Sklenila je, naj izračuna vsoto cifer vseh števil, ki jih bo napisala na tablo. Mladi Karl pa ni želel zapravljati časa z računanjem na pamet, zato te je prosil, da mu napiši program v Javi. Gauss bo števila sproti vpisoval v računalnik. Vnos se zaključi s številko 0.
Če bo torej Gauss vnesel števila 123, 56, 11, 0, mora tvoj program kot odgovor vrniti 19 (1 + 2 + 3 + 5 + 6 + 1 + 1).

Namig: Sestavi metodo, ki za dano celo število vrne vsoto njegovih števk. Nato to metodo uporabi za vsako vnešeno število!

```
import javax.swing.*;

public class Gauss {

    //metoda vrne vsoto stevk v stevilo stevilo
    public static int vsotaStevk(int stevilo) {
      // ideja: vsoto postavimo na 0. Nato pogledamo zadnjo stevko
      // stevila (ce ni ze enako 0)
      // (zadnjo stevko dobimo kot ostanek pri deljenju z deset),
      // jo pristejemo vsoti in stevilo odbijemo zadnjo stevko (deljimo ga z 10)
      // Na koncu vrnemo vsoto

      int vsota = 0;
      while(stevilo > 0) {
        int stevka = stevilo % 10;
        vsota = vsota + stevka;
        stevilo = stevilo / 10;
      }
      return vsota;
    }
```

```
    public static void main(String[] args) {
      //beremo stevila iz vhoda toliko casa, dokler stevilo ni enako 0
      String stevilo;
      int vsota = 0;

      while(!(stevilo = JOptionPane.showInputDialog("Vnesi stevilo")).equals("0")) {
        //pretvorimo niz  stevilo
        int stev = Integer.parseInt(stevilo);
        //izracumano vsoto stevk stevila stev
        int vsotaStevk = vsotaStevk(stev);
        //in jo pristejemo v vsoti
        vsota = vsota + vsotaStevk;
      }
      //Izpisimo vsoto
      System.out.println("Vsota stevk je " + vsota + ".");
    }
}
```