Programiranje 2 – pisni izpit

31. 5. 2022 ob 10.00 (čas pisanja: 90 minut)

1. naloga

V tabeli imamo shranjena cela števila. Spodnji program dopolni z metodo `premakni()`, ki ciklično premakne elemente tabele tako, da je na prvem mestu v tabeli najmanjši element, vrstni red elementov pa se pri tem ne spremeni.

Pri implementaciji metode ne smeš ustvariti in uporabiti nobene nove tabele in ne smeš uporabljati nobenih dodatnih Javanskih razredov (npr. za delo s tabelami ali nizi).

```
public class Naloga11 {

   // NAREDI: napiši manjkajočo metodo

   public static void main(String[] args) {
    int[] tabela = new int[args.length];
    for (int i = 0; i < tabela.length; i++)
      tabela[i]=Integer.parseInt(args[i]);

    System.out.printf("Vhodna tabela:    %s\n", Arrays.toString(tabela));
    premakni(tabela);
    System.out.printf("Rotirana tabela:  %s\n", Arrays.toString(tabela));
   }
}
```

Primer: Ob klicu programa

```
java Naloga11 3 5 -1 404 11 5 -5 9
```

naj program izpiše

```
Vhodna tabela:   [3, 5, -1, 404, 11, 5, -5, 9]
Rotirana tabela: [-5, 9, 3, 5, -1, 404, 11, 5]
```

Pravilnost delovanja programa lahko preveriš s spletnim preverjevalnikom.

2. naloga

Podatki o študentih in njihovih ocenah so zapisani v treh datotekah:

1. seznam študentov (podatki: vpisna številka in ime),
2. seznam predmetov (podatki: kratica predmeta),
3. seznam ocen (podatki: vpisna številka, kratica predmeta in ocena).

Primer vsebine treh datotek

studenti.txt

```
6300001:Micka Kovačeva
6300002:Lolek Bolek
```

predmeti.txt

```
P2
APS2
OS
```

ocene.txt

```
6300004:P2:10
6300003:APS2:5
6300003:ORS:8
6300002:OS:5
6300001:P2:10
6300003:APS2:5
6300002:ORS:8
6300001:APS2:9
6300002:OS:7
6300001:APS2:5
```

Naloga: napiši program Naloga12, ki za vse študente iz seznama študentov izpiše **vse ocene** pri predmetih, ki so navedeni v seznamu predmetov. Študenti in predmeti naj bodo izpisani v istem vrstnem redu, kot so navedeni v vhodnih datotekah, ocene študenta pa naj bodo urejene po velikosti od najmanjše proti največji. Če študent pri posameznem predmetu ni prejel ocene, naj program izpiše znak "/".

Primer: Ob klicu programa

```
java Naloga12 studenti.txt predmeti.txt ocene.txt
```

naj program izpiše

```
Micka Kovačeva
     P2: 10
   APS2: 5, 9
     OS: /
Lolek Bolek
     P2: /
   APS2: /
     OS: 5, 7
```

3. naloga

Napiši program, ki v podanem direktoriju in vseh njegovih poddirektorijih poišče vse datoteke, ki so večje od podane velikosti. Začetni direktorij in velikost (v MB, celo število) sta po vrsti podana kot argumenta programa. Program naj izpiše vse najdene datoteke, urejene padajoče po velikosti (velikost zapiši v MB na eno decimalko natančno). Če ima več datotek enako velikost, naj jih uredi po abecedi po imenu (brez poti), če pa imajo tudi imena enaka, jih uredi po abecedi po celem imenu skupaj s potjo.

Primer: klic programa `java Naloga13 viri 5` poišče vse datoteke, večje od 5 MB, ki se nahajajo v direktoriju `viri` ali njegovih poddirektorijih. Izpiše ustrezno urejen seznam vseh najdenih datotek, npr.:

```
viri\arhiv\Dat1.txt (17 MB)
viri\arhiv\Dat3.pdf (17 MB)
viri\arhiv\P2\Dat3.pdf (17 MB)
viri\arhiv\tmp\Dat3.pdf (17 MB)
viri\arhiv\P2\Dat2.pdf (16 MB)
viri\arhiv\P2\resitve_nalog.java (12 MB)
viri\arhiv\P2\Dat1.pdf (11 MB)
```

4. naloga

Napiši razred, ki ga bomo uporabljali kot množico za shranjevanje znakov (malih črk) angleške abecede. Razred naj se imenuje `MnozicaZnakov` in naj implementira vmesnik `Mnozica`:

```
interface Mnozica {
  void add(char c);
  void remove(char c);
  void flip(char c);
  boolean contains(char c);
  boolean isEmpty();
}
```

Metode vmesnika imajo naslednji pomen:

- `add()` ... dodaj znak v množico,
- `remove()` ... odstrani znak iz množice,
- `flip()` ... če je znak v množici, ga odstrani, sicer ga dodaj,
- `contains()` ... vrne `true`, če je znak v množici, `false` sicer,
- `toString()` ... vrne niz, ki predstavlja elemente množice.

Vse elemente množice shrani v enem celem številu; `i`-ti bit tega števila naj bo prižgan takrat in le takrat, ko je `i`-ta črka vsebovana v množici. Pri tem je 1. črka a, 2. črka b, 3. črka c …

Program bomo poganjali s pomočjo spodnje metode main(); te metode ni treba spreminjati, lahko pa z njo preverjaš pravilnost delovanja razreda `MnozicaZnakov`.
Vmesnik `Mnozica` in razred `MnozicaZnakov` dodaj v datoteko `Naloga14.java` in poskrbi, da bo program `Naloga14` prevedljiv in se ga bo dalo izvesti.

```
  public static void main(String[] args) {
    MnozicaZnakov mn = new MnozicaZnakov();

    for (int i = 0; i < args.length; i++) {
      String arg = args[i];
      if (arg.length() != 2) continue;
      switch (arg.charAt(0)) {
        case '+': mn.add   (arg.charAt(1)); break;
        case '-': mn.remove(arg.charAt(1)); break;
        case 'x': mn.flip  (arg.charAt(1)); break;
      }
    }
    System.out.println(mn.toString());
  }
```

Programiranje 2 – pisni izpit

16. 6. 2022 ob 14.00 (čas pisanja: 90 minut)

1. naloga

V razredu `Naloga21` napiši metodo `izpisi()`, ki besede danega besedila opremi s številkami (v oklepaju zapiše dolžino besede) in tako opremljeno besedilo izpiše na zaslon.

Primer: ob klicu `izpisi("Java je super")` naj metoda izpiše

```
Java(4) je(2) super(5)
```

Metoda `izpisi()` naj upošteva, da ločila (presledek, vejica in pika) ne sodijo k besedi in zato ne prispevajo k njeni dolžini. Ob klicu `izpisi("Bravo, zelo dobro.")` je pravilen izpis tak:

```
Bravo(5), zelo(4) dobro(5).
```

Izhodno besedilo naj bo oblikovano tako, da bodo v eni izpisani vrstici največ 4 besede. Primer: Ob klicu `izpisi("Prva, druga, tretja, cetrta, peta, sesta.")` je pravilen izpis tak:

```
Prva(4), druga(5), tretja(6), cetrta(6),
peta(4), sesta(5).
```

Metodo `izpisi()` napiši v razred `Naloga21`, v katerem naj bo tudi `main()` metoda

```
public static void main(String[] args) {
    izpisi(args[0]);
}
```

Pravilnost programa preveri s spletnim preverjevalnikom.

2. naloga

Mobilne telefonske številke smo zaradi varčevanja s prostorom shranili v datoteko v formatu P2. Datoteka v formatu P2 je binarna datoteka, v kateri sta prva dva bajta (določata podpis) enaka 0x50 in 0x32, sledijo pa po 4 bajti za vsako mobilno številko. Mobilna številka je zapisana brez vodilne ničle pri omrežni skupini, tako da v vsakem bajtu zapišemo dve zaporedni cifri. Prva cifra se shrani v prve 4 bite bajta, druga pa v zadnje 4 bite bajta.

Primer: za mobilno številko 041 113 113 se zapiše osem števk (brez začetne 0) po parih (41 11 31 13), kar bi v šestnajstiškem zapisu izgledalo takole: 0x41 0x11 0x31 0x13.

Hex dump datoteke z zapisanima mobilnima številkama 041 123 456 in 040 335 992 pa je naslednji:

```
50 32 41 12 34 56 40 33 59 92
```

Napiši program `Naloga22`, ki prebere datoteko P2 (njeno ime je podano kot argument programa) in na standardni izhod izpiše vse mobilne številke, ki so zapisane v njej. Pri tem naj vsako številko izpiše v svoji vrstici, oblika izpisane številke pa naj bo naslednja (za zgornji primer datoteke; pazi na presledke):

```
041 123 456
040 335 992
```

Pravilnost programa preveri s spletnim preverjevalnikom.

3. naloga

a) Napiši razred `Matrika`, ki predstavlja matriko celih števil. Razred naj ima tri atribute: dva določata velikost matrike, tretji pa vrednosti. V razred dodaj konstruktor, ki prejme 2D tabelo vrednosti in nastavi vrednosti matrike. Dodaj tudi metodo `toString()`, ki vrne niz z velikostjo matrike, kateri sledijo vse vrednosti. Po potrebi dodaj tudi druge konstruktorje ali metode.

   Primer izpisa matrike: `System.out.println(m2.toString());`

```
Matrika 3 x 5:
 7  8 10  1  1
 1  1  1  1  0
 0  1  1  1  5
```

b) Razredu `Matrika` dodaj metodo `Matrika vsota(Matrika m1, Matrika m2)`, ki prejme dve matriki in vrne novo matriko, ki je vsota podanih dveh matrik. Če se dimenzije podanih matrik ne ujemajo, naj metoda vrne `null`.

c) Razredu `Matrika` dodaj metodo `void pomnozi(int skalar)`, ki prejme skalar, s katerim pomnoži matriko.

Delovanje metod lahko preveriš v metodi `main()` razreda `Naloga23` z naslednjimi matrikami:

```
Matrika m1 = new Matrika(new int[][]{{1, 2, 3, 4, 5}, {5, 4, 3, 2, 1},
{10, 11, 12, 13, 14}});
Matrika m2 = new Matrika(new int[][]{{7, 8, 10, 1, 1}, {1, 1, 1, 1,
0}, {0, 1, 1, 1, 5}});
Matrika m3 = new Matrika(new int[][]{{0, 1, 1}, {1, 1, 1}, {0, 1,
1}});
```

Nasvet: Vsoto matrike dobimo tako, da seštejemo istoležne elemente obeh matrik. Matriko množimo s skalarjem tako, da z njim pomnožimo vsak element matrike.

4. naloga

V datoteko `Naloga24.java` napiši razred `Sonce`, ki s pomočjo knjižnice `stdlib` nariše okroglo sonce rumene barve z `2*n` kraki, ki so enakomerno razporejeni po celotnem obodu sonca. Dolžina krakov naj bo enaka polmeru sonca, polovica krakov naj se dotika površine, druga polovica pa naj bo od površine oddaljena za ⅓ polmera `r` sonca. Središče sonca naj bo v točki `(x, y)`.
Razred `Sonce` naj vsebuje konstruktor, ki prejme štiri parametre (x, y, r, n) in metodo `narisi()`, ki sonce dejansko nariše.

Primer: Ob klicu metode `main()`

```
public static void main(String[] args) {
    StdDraw.setScale(-100,100);
    StdDraw.setPenRadius(0.01);
    Sonce s = new Sonce(0,0,30, 10);
    s.narisi();
}
```

naj program izriše tako sliko:

Programiranje 2 – pisni izpit

25. 8. 2022 ob 10.00 (čas pisanja: 90 minut)

1. naloga

Imamo tabelo imen študentov `String[] studenti` in tabelo njihovih ocen pri predmetu `int[] ocene`. Zaporedje študentov in ocen je enako, torej velja: študent `studenti[i]` je dobil oceno `ocene[i]`. V razredu `Naloga31` napiši metodo

```
  private static int[] uredi(String[] studenti, int[] ocene),
```

ki prejme tabelo študentov in tabelo ocen ter vrne tabelo indeksov študentov, ki je urejena po padajočih ocenah, v primeru enakih ocen pa po abecedi imen študentov.
Pri tem tabel `studenti` in `ocene` ne smeš spremeniti, pri rešitvi pa lahko uporabiš le eno dodatno tabelo (tisto, ki jo metoda vrne).

Dopolni tudi metodo `main()`

```
public static void main(String[] args) {
  // NAREDI: na podlagi arguemntov ustvari tabeli studenti in ocene
  int[] urejeni = uredi(studenti, ocene);
  // NAREDI: izpiši študente v urejenem vrstnem redu
}
```

tako, da bo program `Naloga31` ob klicu

```
java Naloga31 3 Micka Ana Polde 8 10 9
```

izpisal

```
 1 Ana, ocena 10
 2 Polde, ocena 9
 3 Micka, ocena 8
```

(prvi argument pove, koliko imen in ocen bo sledilo, nato sledjo najprej imena, nato ocene).

Namig: preprost način urejanja tabel je tak, da poiščeš največji element tabele, ga postaviš na prvo mesto (zamenjaš mesti prvega elementa in največjega elementa) ter postopek ponoviš s preostalimi elementi tabele, brez prvega elementa.

2. naloga

V podani binarni datoteki je velikost datoteke (to je skupno število vseh bajtov v datoteki) zapisana v prvih nekaj bajtih na naslednji način: v prvem bajtu je zapisano, koliko naslednjih bajtov določa velikost datoteke, temu sledijo bajti, ki določajo velikost, potem sledi vsebina datoteke.

Primer: če se binarna datoteka začne z bajti

03 00 01 4b a1 15 04 …

potem prvi bajt (0x03) pove, da je velikost zapisana v naslednjih treh bajtih. Ker je vrednost naslednjih treh bajtov enaka 0x00014b, je skupno število vseh bajtov v datoteki enako 331. Podatki a1 15 04 … so vsebina datoteke.

Napiši program `Naloga32`, ki prebere binarno datoteko (ime datoteke je podano v prvem argumentu) in ugotovi, ali se velikost, ki je zapisana na začetku datoteke, ujema z dejanskim številom bajtov v datoteki. Če se velikosti ujemata, naj program velikost izpiše, če pa je število vseh bajtov v datoteki večje ali manjše od zapisane velikosti datoteke, naj program sproži nepreverljivo izjemo `IzjemaNapacneDatoteke` (definiraj tudi ustrezen razred za to izjemo), v nasprotnem primeru pa izpiši “OK”.

Nasvet: Ker je javanski tip `byte` predznačen, lahko v spremenljivko tega tipa shranimo vrednosti od -128 do +127. Če želimo shraniti večjo vrednost (recimo do 255, kar je 0xFF), moramo uporabiti tip int. Pri pretvorbi poskrbimo, da je 24 bitov na levi enakih 0:

```
stevilo = (bajt & 0xFF)
```

3. naloga

Podan je program `Naloga33`, ki zgenerira naključno število točk z naključnimi koordinatami x in y ter jih doda na seznam.

```
public class Naloga33 {
   public static void main(String[] args) {
        Random rnd = new Random();
        ArrayList<Tocka> seznam = new ArrayList<>();
        int n = rnd.nextInt(5) + 2;
        for (int i = 0; i < n; i++) {
            int x = rnd.nextInt(199) - 99;
            int y = rnd.nextInt(199) - 99;
            seznam.add(new Tocka(x, y));
        }
        for (Tocka t : seznam)
            System.out.println(t.toString());
    }
}
```

Program dopolni z razredom `Tocka`, v katerega dodaj ustrezne atribute, konstruktorje in metode, da bo izpis npr. naslednji (pri tem je D razdalja točke od izhodišča, zaokrožena na eno decimalko):

```
Tocka ( 96,-83)  D = 126,9
Tocka ( 72, 77)  D = 105,4
Tocka ( 62, 10)  D = 62,8
Tocka (-83, 32)  D = 89,0
Tocka (  7,-23)  D = 24,0
```

Dodaj tudi komparator `PrimerjajTockePoOddaljenostiPadajoce`, ki omogoča urejanje točk po padajoči oddaljenosti od izhodišča. Metodo `main()` dopolni tako, da točke izpiše urejeno glede na oddaljenost od izhodišča.

4. naloga

Napiši program Naloga34, ki prebere tekstovno datoteko, katere ime je podano v prvem argumentu, ter izpiše njeno vsebino na zaslon v več vrstic tako, da bo vsaka vrstica vsebovala natanko `n` znakov (`n` je drugi argument). Besed v vrstici ne smeš deliti, vsaka beseda mora biti v celoti izpisana v eni vrstici. Da dosežeš željeno širino vrstice, na konec vrstice dodaj primerno število podčrtajev (_).

Primer: Če bo vhodna datoteka vsebovala besedilo "`Danes je res en lep soncen dan.`" in bo drugi argument enak 14 (zahtevan je torej izpis na širino 14), potem naj program izpiše:

```
Danes je res__
en lep soncen_
dan._________
```