Četrtek 23. 6. 2022 ob 14:00 v PR 3.10-11 ali na daljavo

1. pisni izpit pri Programiranje 2 (Praktična matematika)

Čas reševanja pisnega izpita je **120 minut**. Pisni izpit lahko rešujete v učilnici ali na daljavo, dočim pa morate biti celoten čas izpita **vidni preko kamere** na Zoom! Dovoljena je uporaba poljubnega gradiva, literature, spletnih virov ter zapiskov in rešitev iz predavanj ali vaj. **Prepovedana je kakršnakoli komunikacija** v realnem času!

Pisni izpit je sestavljen iz **štirih enakovrednih nalog**. Prvi dve nalogi zahtevata rešitev v **programskem jeziku** Python 3, zadnji dve nalogi pa zahtevata rešitev v **programskem jeziku** Java 8. Sestavljeni programi ne smejo uporabljati modulov, ki niso skladni s programskim jezikom Python 3, oziroma knjižnic, ki niso del programskega jezika Java 8.

Vso **gradivo potrebno za reševanje nalog** je dostopno na spletni učilnici. Sestavljene **programe oddate na spletni učilnici** kot je zavedeno v razdelkih ***Kaj in kako oddam?***. Pazite, da se natančno držite navodil oddaje!

---

1. Fibonaccijevo zaporedje (≈5 vrstic Python kode)

V programskem jeziku Python 3 sestavite program za **izračun Fibonaccijevega zaporedja števil**. Naj bo ničto Fibonaccijevo število $F_0 = 0$ in prvo Fibonaccijevo število $F_1 = 1$. Potem je n-to Fibonaccijevo število F_n za $n \geq 2$ definirano kot

$$F_n = F_{n-1} + F_{n-2}.$$

Na zaslon **izpišite prvih sto Fibonaccijevih števil**, pri čimer pa pazite, da direktna implementacija zgornje formule z uporabo rekurzije omogoča izračun zgolj nekje prvih petdeset števil. Zato števila F_n izračunajte po vrsti z uporabo že vnaprej izračunanih števil F_{n-1} in F_{n-2} (tj. dinamično programiranje).

Primer izpisa sestavljenega programa je prikazan spodaj.

```Bash
F_0 = 0
F_1 = 1
F_2 = 1
F_3 = 2
F_4 = 3
F_5 = 5
...
```

```Bash
...
F_95 = 31,940,434,634,990,099,905
F_96 = 51,680,708,854,858,323,072
F_97 = 83,621,143,489,848,422,977
F_98 = 135,301,852,344,706,746,049
F_99 = 218,922,995,834,555,169,026
F_100 = 354,224,848,179,261,915,075
```

Kaj in kako oddam?

Sestavljen program shranite v **datoteko** `fibonacci.py`, ki jo **oddate na** spletni učilnici. Ne pričakuje se, da je programska koda opremljena s komentarji. Pazite pa, da program ne vsebuje napak, kar pomeni, da se ukaz `python fibonacci.py` uspešno izvede!

---

2. Zemljevid svetovnih mest (≈10 vrstic Python kode)

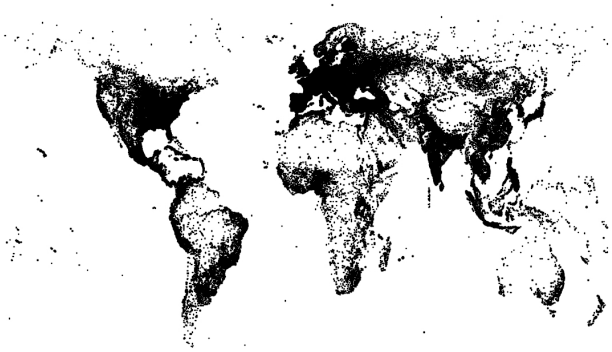
Na naslovu https://lovro.fri.uni-lj.si/api/cities je dostopna datoteka s **seznamom svetovnih mest** v JSON formatu. Za vsako mesto je podano njegovo ime, država ter zemljepisna širina in dolžina.

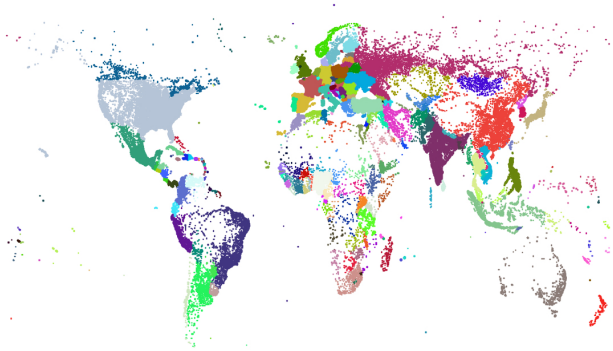
Vsebino datoteke si lahko ogledate v poljubnem urejevalniku besedil, dočim je del vsebine prikazan spodaj.

```JavaScript
[
  {
    "country": "SI",
    "name": "Ljubljana",
    "lat": "46.05108",
    "lng": "14.50513"
  },
  {
    "country": "UA",
    "name": "Kyiv",
    "lat": "50.45466",
    "lng": "30.5238"
  },
  ...
]
```

Vaša naloga je, da v programskem jeziku Python 3 sestavite program, ki najprej s pomočjo knjižnice `requests` **prebere vsebino datoteke** in nato z uporabo knjižnice `matplotlib.pyplot` **izriše zemljevid mest**. Na zemljevidu naj bodo mesta prikazana s posameznimi točkami (x, y), kjer je x zemljepisna dolžina in y zemljepisna širina mesta. Nato dopolnite program tako, da bodo mesta v posamezni državi izrisana z drugo **naključno izbrano barvo**. Program naj končni **graf shrani v datoteko** `cities.png`.

Primer grafa v datoteki `cities.png` je prikazan spodaj desno. Izgled grafa je popolnoma poljuben dokler le-ta zadošča zahtevam naloge.





Kaj in kako oddam?

Sestavljen program shranite v **datoteko** `cities.py`, ki jo **oddate na** spletni učilnici. Ni potrebno oddajati datoteke `cities.png`, poleg tega se ne pričakuje, da je programska koda opremljena s komentarji. Pazite pa, da program ne vsebuje napak, kar pomeni, da se ukaz `python cities.py` uspešno izvede!

---

3. Vektorji v n-razsežnem prostoru (≈30 vrstic Java kode)

V programskem jeziku Java 8 **sestavite razred** `Vector`, ki naj predstavlja **vektor v n-razsežnem prostoru** $\mathbb{R}^n$. Vektor najlažje predstavite s tabelo realnih koordinat vektorja (npr. objektna spremenljivka `double[] coordinates`).

Pripravite tri konstruktorje za razred `Vector`. Prvi konstruktor naj bo brez parametrov in naj ustvari ničelni vektor v trorazsežnem prostoru. Drugi konstruktor naj sprejme število `int n` in ustvari n-razsežni vektor, kjer je vsaka koordinata naključno izbrana iz intervala $[0, 1)$. Tretji konstruktor naj sprejme tabelo `double[] coordinates` s koordinatami vektorja.

Razredu `Vector` **dodajte metodo** `main(String[] args)`, ki naj vključuje spodnji program. Le-tega *ne smete spreminjati*! Zato ustrezno posodobite razred `Vector` tako, da se program uspešno izvede. Pri tem dva vektorja primerjajte tako, da zaporedoma primerjate njune istoležne koordinate. V primeru, da vektorja nista enake dimenzije, predpostavite, da so manjkajoče koordinate enake 0.

```java
// Urejena množica vektorjev
Set<Vector> set = new TreeSet<Vector>();
set.add(new Vector());
set.add(new Vector(3));
set.add(new Vector(new double[] {0.0, 0.0, 0.0}));
set.add(new Vector(new double[] {1.0, 1.0, 1.0}));

// Izpis množice vektorjev
for (Vector vector: set)
    System.out.println(vector);
```

Primer izpisa zgornjega programa je prikazan spodaj.

```bash
[0.000, 0.000, 0.000]
[0.913, 0.227, 0.156]
[1.000, 1.000, 1.000]
```

Kaj in kako oddam?

Sestavljen program shranite v **datoteko** `Vector.java`, ki jo **oddate na** spletni učilnici. Ne pričakuje se, da je programska koda opremljena s komentarji. Pazite pa, da program ne vsebuje napak, kar pomeni, da se ukaza `javac Vector.java` in `java Vector` uspešno izvedeta!

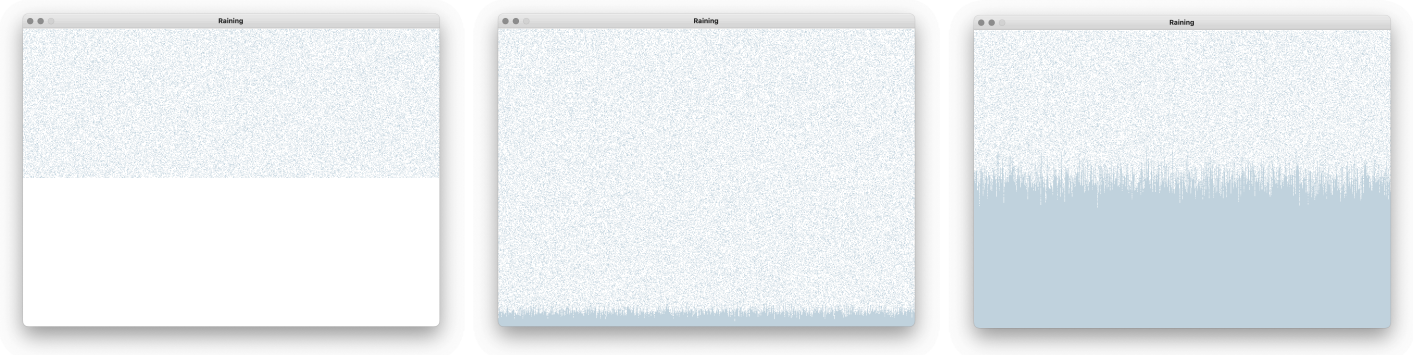
---

4. Simulacija dežnih kapelj (≈25 vrstic Java kode)

V programskem jeziku Java 8 sestavite **enostaven grafični vmesnik**, ki naj vsebuje zgolj en panel fiksne velikosti. Le-ta naj bo **namenjen simulaciji dežnih kapelj**, ki se **naključno pojavljajo** na vrhu panela in **padajo proti dnu**. Vsaka dežna kaplja naj ustreza **enemu pikslu panela**. Dežne kaplje najpreprosteje predstavite z dvodimenzionalno tabelo logičnih vrednosti `boolean[][] raindrops` enakih dimenzij kot je velikost panela.

Simulacija naj posodobi stanje dežnih kapelj vsakih 5 milisekund. Na vsakem koraku najprej v vsak piksel **na vrhu panela dodajte dežno kapljo** z verjetnostjo 25%. Nato **premaknite vse dežne kaplje** za en piksel proti dnu panela v kolikor je to mogoče. Pazite, da dežne kaplje premikate **od spodaj navzgor** tako, da se le-te “zbirajo” na dnu panela. Simulacija naj se zaključi, ko je **polovica panela pod vodo**.

Primer izgleda grafičnega vmesnika je prikazan spodaj. Pri tem je **izgled grafičnega vmesnika popolnoma poljuben** dokler le-ta zadošča zahtevam naloge.



Priporoča se, da **kot osnovo** za razvoj **uporabite spodnji program**.

Java

```
public class Raining {

    public static void main(String[] args) throws InterruptedException {
        JFrame frame = new JFrame("Raining");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(new Dimension(800, 600));
        frame.setResizable(false);

        JPanel panel = new JPanel() {
            private static final long serialVersionUID = 1L;
            @Override
            public void paint(Graphics g) {
                super.paint(g);
                Graphics2D graphics = (Graphics2D)g;

                ... // izris dežnih kapelj
            }
        };
        panel.setBackground(Color.WHITE);
        frame.add(panel);

        frame.setVisible(true);

        ... // inicializacija dežnih kapelj

        while (true) {
            ... // dodajanje dežnih kapelj

            ... // premikanje dežnih kapelj

            panel.repaint();
            Thread.sleep(5);
        }
    }
}
```

Kaj in kako oddam?

Sestavljen program shranite v **datoteko** `Raining.java`, ki jo **oddate na** spletni učilnici. Ne pričakuje se, da je programska koda opremljena s komentarji. Pazite pa, da program ne vsebuje napak, kar pomeni, da se ukaza `javac Raining.java` in `java Raining` uspešno izvedeta!