Torek 27. 6. 2023 ob 15:00 v PR 3.10-11 (FMF)

1. pisni izpit pri Programiranje 2 (Praktična matematika)

Čas reševanja pisnega izpita je **120 minut**. Pisni izpit lahko rešujete na fakultetnih računalnikih ali lastnih prenosnikih. Reševanje izven učilnice ni dovoljeno! Dovoljena je uporaba poljubnega gradiva, literature in spleta, dočim pa je **prepovedana kakršnakoli komunikacija ali uporaba orodij umetne inteligence**!

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.

Vse sestavljene **programe stisnite v datoteko** `<vpisna>.zip`, ki jo **oddate na** spletni učilnici kot je zavedeno v razdelkih ***Kaj oddam?***. Ne pričakuje se, da je programska koda opremljena s komentarji. Pazite pa, da se natančno držite navodil!

---

1. Prepoznava imenskih entitet (≈15 vrstic Python kode)

Na spletni strani https://www.gutenberg.org/files/11/11-h/11-h.htm je dostopna zgodba ***Alica v čudežni deželi*** v angleškem jeziku. Z uporabo knjižnice `requests` najprej **preberite vsebino spletne strani** v formatu HTML. Nato iz dobljenega besedila odstranite vse narekovaje `"` in zamenjajte vsa zaporedja belih znakov (npr. presledek, tabulator, nova vrstica) z zgolj enim presledkom.

Vaša naloga je, da iz besedila **izluščite imenske entitete**. Pri tem imensko entiteto definiramo kot zaporedje ene ali več besed, ki se začnejo z veliko začetnico, vendar se *ne* pojavijo na začetku stavka. Ali se beseda pojavi na začetku stavka lahko ugotovite tako, da preverite ali se prejšnja beseda konča z enim izmed znakov `.!?>;`.

Preštejte število pojavitev vsake imenske entitete in **izpišite 15 najpogostejših entitet** skupaj s številom pojavitev. Pričakovan izpis sestavljenega programa je prikazan spodaj.

```
Alice (286x)
Queen (75x)
King (61x)
Mock Turtle (56x)
Project Gutenberg (56x)
Gryphon (55x)
...
```

Kaj oddam?

Sestavljen program **shranite v datoteko** `alice.py`. Pazite, da program ne vsebuje napak, kar pomeni, da se ukaz `python alice.py` uspešno izvede!

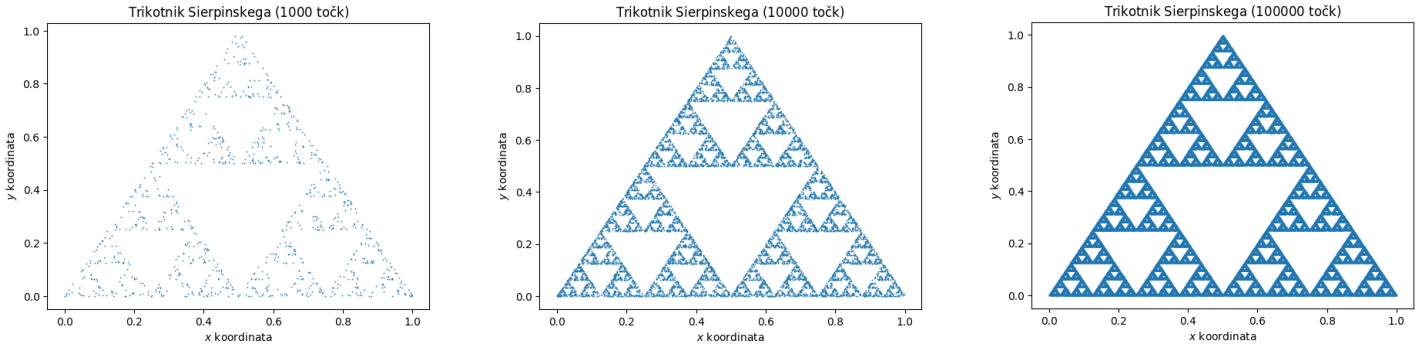
---

2. Trikotnik Sierpinskega (≈15 vrstic Python kode)

Vaša naloga je, da **izrišete trikotnik Sierpinskega**, ki ga definiramo kot zaporedje točk P_i ustvarjenih po naslednjem postopku.

Naj bodo $A = (0, 0)$, $B = (1, 0)$ in $C = (0.5, 1)$ oglišča trikotnika. Prva točka trikotnika naj bo enaka $P_1 = A$. Vsaka naslednja točka P_i za $i > 1$ je definirana kot **razpolovišče daljice med prejšnjo točko** P_{i-1} **in naključno izbranim ogliščem** A, B ali C. Zadnji korak ponavljajte dokler ne ustvarite vsaj 100000 točk P_i.

Z uporabo knjižnice `matplotlib.pyplot` ustvarjene **točke** P_i **izrišite na grafu** kot je prikazano spodaj. Grafu dodajte tudi naslov in oznake osi ter ga shranite v datoteko `sierpinski.pdf`.



Kaj oddam?

Sestavljen program **shranite v datoteko** `sierpinski.py`, dočim ni potrebno oddati ustvarjene slike `sierpinski.pdf`. Pazite, da program ne vsebuje napak, kar pomeni, da se ukaz `python sierpinski.py` uspešno izvede!

---

3. Elementarne funkcije (4 Java datoteke)

Najprej **sestavite vmesnik** `Function`, ki naj predstavlja poljubno odvedljivo realno funkcijo $f(x)$ in definira naslednje funkcije:

- `String function()`, ki vrne enolično predstavitev funkcije $f(x)$ (glej spodaj);
- `double function(double x)`, ki izračuna vrednost funkcije $f(x)$ pri podanem x;
- `String derivative()`, ki vrne enolično predstavitev odvoda funkcije $f'(x)$ (glej spodaj); in
- `double derivative(double x)`, ki izračuna vrednost odvoda funkcije $f'(x)$ pri podanem x.

Nato **sestavite tri razrede**, ki so implementacije vmesnika `Function`:

- `Linear`, ki predstavlja linearno funkcijo $f(x) = x$;
- `Power`, ki predstavlja potenčno funkcijo $f(x) = x^n$ za izbran n; in
- `Exponential`, ki predstavlja eksponentno funkcijo $f(x) = e^x$;

Na koncu **vmesniku** `Function` **dodajte metodo** `main(String[] args)`, ki naj vključuje spodnji program. Le-tega *ne* smete spreminjati! Zato ustrezno posodobite vmesnik in razrede, da se komentirana vrstica pravilno izvede. Pri tem naj bodo funkcije urejene naraščajoče pri $x = 1$, tj. najprej glede na vrednost funkcije $f(1)$ in nato glede na vrednost odvoda $f'(1)$.

```java
Function[] functions = new Function[] { new Exponential(), new Power(3), new Linear() };

Arrays.sort(functions); // Urejanje funkcij

for (Function function: functions) {
    System.out.println(function.function());
    System.out.println("f(1) = " + function.function(1.0));
    System.out.println(function.derivative());
    System.out.println("f'(1) = " + function.derivative(1.0));
    System.out.println();
}
```

Pričakovan izpis zgornjega programa je prikazan spodaj.

```
f(x) = x
f(1) = 1.0
f'(x) = 1
f'(1) = 1.0

f(x) = x^3
f(1) = 1.0
f'(x) = 3*x^2
f'(1) = 3.0

f(x) = e^x
f(1) = 2.718281828459045
f'(x) = e^x
f'(1) = 2.718281828459045
```

Kaj oddam?

Sestavljen program **shranite v datoteke** `Function.java`, `Exponential.java`, `Power.java` in `Linear.java`. Pazite, da program ne vsebuje napak, kar pomeni, da se ukaza

`javac Function.java` in `java Function` uspešno izvedeta!

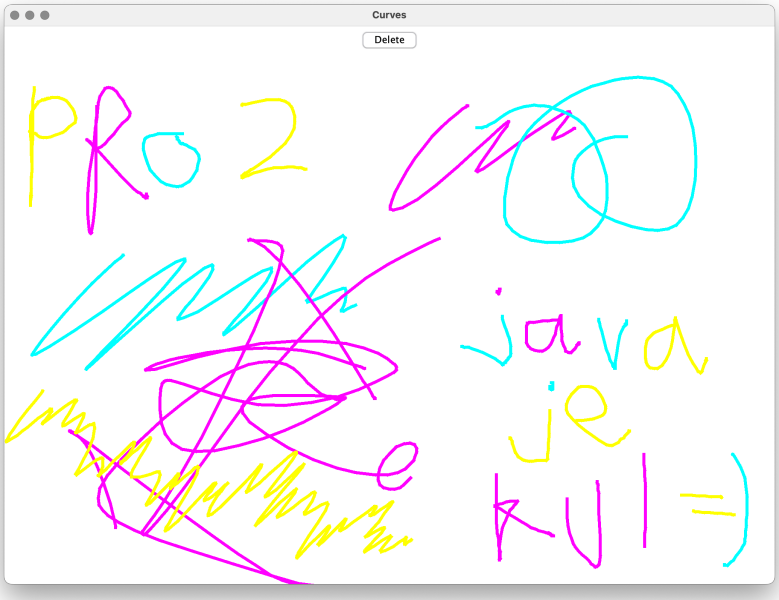
---

4. Risanje krivulj (≈25 vrstic Java kode)

Sestavite **enostaven grafični vmesnik**, ki naj vsebuje dva panela (glej spodaj).

- Osrednji panel naj bo namenjen **risanju krivulj z miško**. Uporabnik s pritiskom miškinega gumba nad panelom začne z risanjem nove krivulje. Krivuljo riše z vlečenjem miške, dokler drži miškin gumb. Vsaka krivulja naj bo izrisana z naključno izbrano barvo izmed `Color.CYAN`, `Color.YELLOW` in `Color.MAGENTA`.
- Zgornji panel naj vsebuje gumb `Delete`, ki **zbriše vse narisane krivulje**.

Primer izgleda grafičnega vmesnika z nekaj krivuljami je prikazan spodaj.



Priporoča se, da kot osnovo za delo uporabite spodnji program, pri čimer krivulje predstavite kot seznam `List<Curve> curves`.

Java

```
public class Curves extends JFrame {

    public Curves() {
        super();

        setTitle("Curves");
        setLayout(new BorderLayout());
        setSize(new Dimension(1024, 768));
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
```

```
        JPanel panel = new JPanel() {
            @Override
            public void paint(Graphics g) {
                super.paint(g);

                ... // risanje krivulj
            }
        };
        panel.setBackground(Color.WHITE);
        add(panel, BorderLayout.CENTER);

        ... // poslušalci dogodkov miške

        JPanel console = new JPanel();
        console.setBackground(Color.WHITE);
        add(console, BorderLayout.NORTH);

        ... // gumb za brisanje krivulj
    }

    public static void main(String[] args) {
        new Curves().setVisible(true);
    }
}

class Curve {

  public List<Point> points;

  public Color color;

  public Curve(int x, int y, Color color) {
    points = new ArrayList<Point>();
    points.add(new Point(x, y));

    this.color = color;
  }
}
```

Kaj oddam?

Sestavljen program **shranite v datoteko** `Curves.java`. Pazite, da program ne vsebuje napak, kar pomeni, da se ukaza `javac Curves.java` in `java Curves` uspešno izvedeta!
