Primer izpita pri predmetu Programiranje I

Tik pred začetkom ...

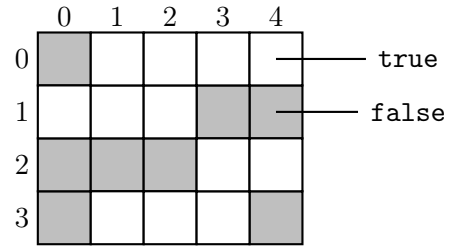
- Na voljo imate 90 minut časa.
- Ni nujno, da se vam bo prva naloga zdela najlažja, tretja pa najtežja.
- Pazite, da se ne boste predolgo ukvarjali s težjimi primeri oz. podnalogami pri nalogi X, pri tem pa povsem pozabili na nalogo Y.
- Navodila za testiranje in oddajo najdete v datotekah `README.txt` v mapah, ki pripadajo posameznim nalogam.
- Na učilnico oddajte točno tisto, kar piše v navodilih. Oddajajte sproti!
- Sproti testirajte!
- Pazite na zahrbtne napake, kot so npr. zamenjava indeksov `i` in `j`, podpičje na koncu glave zanke, ...
- Najboljši zatiralec hroščev in izjem je `System.out.println`/`printf`.
- Dopolnite že pripravljene razrede. Seveda lahko vanje po potrebi dodajate tudi pomožne metode.
- Goljufe čaka Dantejev pekel.

Držimo pesti!

(1) V razredu `Park` dopolnite spodaj navedene metode. **Pozor:** morda se vam bosta nalogi (c) in (d) zdeli lažji od naloge (b).

(a) `public static int steviloProstih(boolean[][] p)` [1–12]:

Vrne število prostih parkirnih mest na pravokotnem parkirišču. Neprazna pravokotna tabela `p` podaja razpoložljivosti posameznih mest (`true`: prosto; `false`: zasedeno). V sledečem primeru bi metoda vrnila vrednost 12.



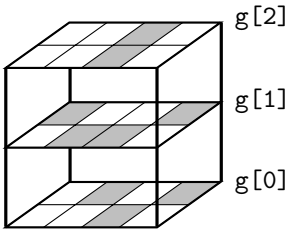
(b) `public static boolean zaporedje(boolean[][] p, int k)` [13–25]:

Vrne `true` natanko v primeru, če se v vsaj eni vrstici parkirišča nahaja vsaj `k` zaporednih prostih mest ($\texttt{k} \geqslant 1$). V gornjem primeru bi metoda vrnila `true` za vse $\texttt{k} \leqslant 4$.

(c) `public static int steviloMest(boolean[][][] g)` [26–37]:

Parkirišče je »zraslo« v garažno hišo v obliki kvadra. Prva dimenzija neprazne pravilne tabele `g` se nanaša na nadstropja, druga in tretja pa na zasedenosti parkirnih mest v posameznih nadstropjih.

Metoda `steviloMest` naj vrne število *vseh* parkirnih mest (prostih in zasedenih) v garažni hiši. V sledečem primeru bi bil rezultat enak 24.



(d) `public static int najboljProstoNadstropje(boolean[][][] g)` [38–50]:

Vrne indeks nadstropja z največ prostimi mesti; če je takih indeksov več, naj vrne najmanjšega. V gornjem primeru bi bil rezultat enak 2.

(2) Podani so sledeči razredi (prikazani so samo atributi):

```
class Recept {    // kuharski recept
    private Korak[] koraki;    // koraki recepta
}
class Korak {    // korak recepta
    private Snov[] vhodi;      // jedilne snovi, ki vstopajo v korak
    private Snov[] izhodi;     // jedilne snovi, ki so rezultat koraka
    private String akcija;     // akcija, npr. "zmesaj"
    private int trajanje;      // trajanje koraka v minutah
}
class Snov {     // jedilna snov
    private String naziv;      // unikaten naziv, npr. "moka" ali "palacinke"
}
```

Dopolnite sledeče metode:

(a) `public int trajanje()` v razredu `Recept` [1–12]:

Vrne trajanje recepta v minutah (torej vsoto trajanj vseh korakov).

(b) `public int prviKorakZAkcijo(String akcija)` v razredu `Recept` [13–25]:

Vrne indeks prvega koraka s podano akcijo oziroma -1, če takšnega koraka v receptu ni.

(c) `public boolean naIzhodu(Recept recept)` v razredu `Snov` [26–37]:

Vrne `true` natanko v primeru, če snov `this` nastopa na izhodu vsaj enega koraka podanega recepta.

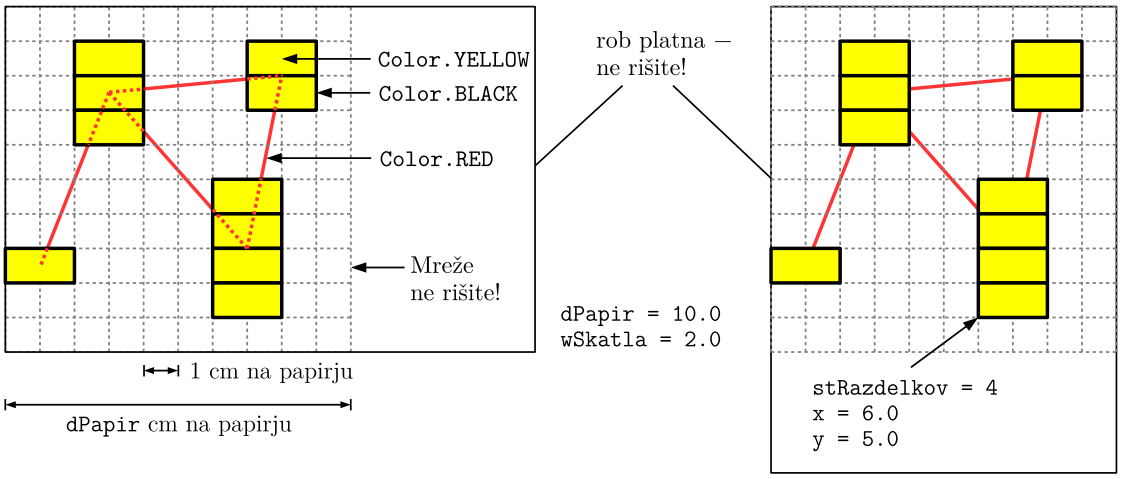
(d) `public int steviloVstopnihSnovi()` v razredu `Recept` [38–50]:

Vrne število snovi, ki se pojavljajo *samo* na vhodih v korake recepta. Lahko predpostavite, da vsaka snov nastopa na vhodu v kvečjemu enem koraku recepta.

(3) Na kvadratnem listu papirja s stranico `dPapir` cm je narisan diagram s »škatlami« in povezavami med njimi. Vse škatle so široke `wSkatla` cm, po višini pa so razdeljene na razdelke višine 1 cm. Povezave povezujejo središča škatel. Podatki o škatlah in povezavah so zbrani v tabelah `skatle` (tipa `Skatla`) in `povezave` (tipa `Povezava`):

```
private static class Skatla {
    private int stRazdelkov;   // število razdelkov (≥ 1)
    private double x, y;       // odmik (v cm) zgornjega levega kota škatle
                               //       od zgornjega levega kota lista papirja
}
private static class Povezava {
    private Skatla prva, druga;    // škatli, ki ju povezava povezuje
}
```

Dopolnite metodo `narisi` v razredu `Diagram` tako, da bo list z diagramom »preslikala« na platno. List naj bo postavljen v zgornjem levem kotu, zavzema pa naj celotno krajšo stranico platna. Zgledujte se po sledeči sliki, ki se nanaša na primer v razredu `Diagram` in na javna primera 8–9:



Dopolnite tudi sledeči metodi:

- `public double sirinaSkatle(double wp, double hp)` [1–5]:

  Na podlagi širine in višine platna vrne širino škatle v slikovnih pikah (»pikslih«) — ne v centimetrih!

- `public double[] sredisceSkatle(Skatla sk, double wp, double hp)` [6–15]:

  Vrne koordinati v »pikslih« (v obliki tabele $\{x, y\}$) središča podane škatle.

Namig: Kaj se vam splača narisati prej — škatle ali povezave?

Opombe: Škatle se med seboj ne prekrivajo. Za primere 16–25 velja `wp == hp`, za primere 16–35 pa (tudi) `povezave.length == 0`.