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) Mojster Vahid stoji pred steno velikosti $n \times n$, v škatli pa ima k ploščic velikosti $h_i \times w_i$ (h_i je višina, w_i pa širina i-te ploščice za $i \in \{1, \ldots, k\}$). Na vrh stene bi rad položil čimveč ploščic z najmanjšo višino, pod njimi čimveč ploščic z drugo najmanjšo višino itd. Znotraj ene vrste ploščic so torej vse ploščice enako visoke, višine vrst pa strogo naraščajo. Največ koliko ploščic lahko položi na opisani način?

Vhod: V prvi vrstici vhoda je podano celo število $n \in [1, 10^9]$, v drugi celo število $k \in [1, 10^3]$, nato pa sledi k vrstic s pari celih števil $h_i \in [1, n]$ in $w_i \in [1, n]$ (števili sta ločeni s presledkom).

V primerih 1–10 so vse ploščice enako velike, v primerih 11–25 pa enako visoke. (V primerih 1–25 bo torej na steni samo ena vrsta ploščic.) V primerih 11–20 in 26–35 so ploščice na vhodu urejene po naraščajočih višinah, pri enakih višinah pa po naraščajočih širinah.

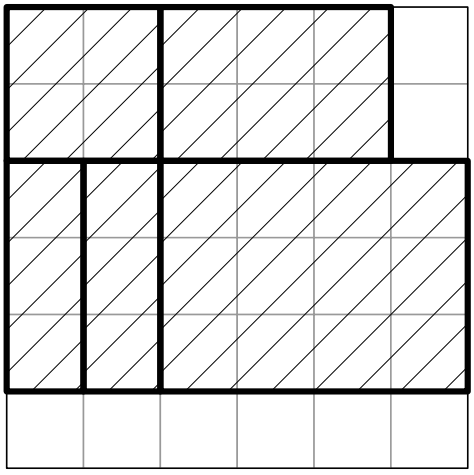
Izhod: Na izhodu izpišite samo iskano število ploščic.

Primer (vhod/izhod):

```
6
7
3 1
2 3
3 4
4 2
3 1
2 3
2 2
```

```
5
```

V gornjem primeru lahko položi vse ploščice razen ploščice 4×2 in ene od obeh 2×3:



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

```
class Sestavina {
    private String naziv;      // naziv sestavine, npr. "moka"
    private double kalorije;   // število kcal na 100 gramov sestavine
    private int oh;            // delež ogljikovih hidratov (v odstotkih)
    private int beljakovine;   // delež beljakovin (v odstotkih)
    private int mascobe;       // delež maščob (v odstotkih)
}
class Jed {
    private String naziv;            // naziv, npr. "prekmurska gibanica"
    private Sestavina[] sestavine;   // vse sestavine, ki tvorijo jed
    private double[] mase;           // mase posameznih sestavin v gramih
    private String vrsta;            // npr. "glavna jed", "sladica"
}
class Obrok {
    private Jed[] jedi;  // jedi, ki sestavljajo obrok
}
interface Primerjalnik {}
```

V vseh objektih razreda `Jed` je tabela `mase` enako dolga kot tabela `sestavine`. Element `mase[i]` podaja maso sestavine `sestavine[i]` v jedi.

Dopolnite sledeče metode:

- `public boolean jeBeljakovinska()` v razredu `Jed` [1–10]:

  Vrne `true` natanko v primeru, če je jed `this` beljakovinska, kar pomeni, da vsaj ena sestavina jedi `this` vsebuje najmanj 10% beljakovin.

- `public double kalorije()` v razredu `Jed` [11–20]:

  Vrne skupno število kcal za jed `this`.

- `public int indeksNajboljKaloricneBeljakovinske()` v razredu `Obrok` [21–30]:

  Vrne indeks najbolj kalorične beljakovinske jedi v tabeli jedi obroka `this`. Če je takih jedi več, naj metoda vrne indeks prve od njih. Če beljakovinskih jedi ni, naj metoda vrne vrednost -1.

- `public double masaSestavine(String naziv)` v razredu `Obrok` [31–40]:

  Vrne skupno maso sestavine (ne jedi!) s podanim nazivom v obroku `this`.

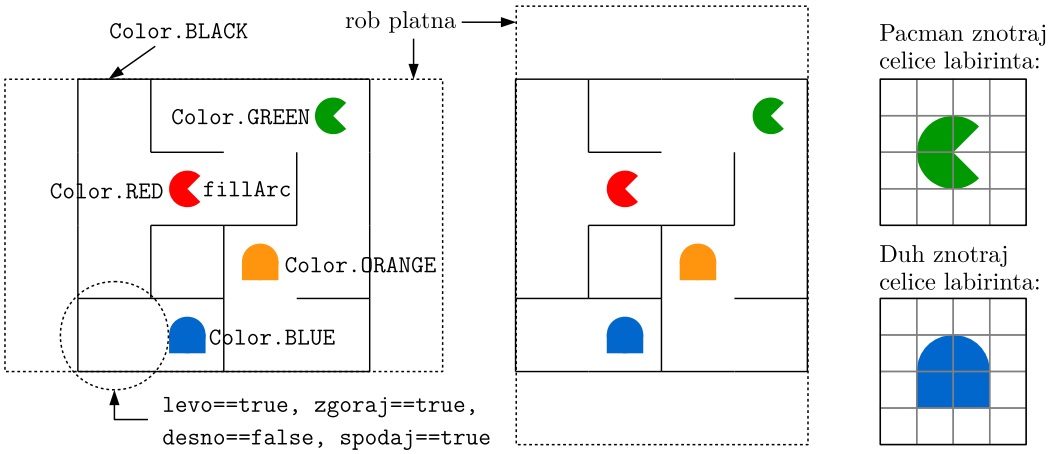
Napišite še razred `PrimerjalnikBK`, in to tako, da bo metoda `urediJedi` v razredu `Obrok`, če ji podamo objekt tega razreda, jedi obroka `this` uredila najprej po »beljakovinskosti« (v urejeni tabeli bodo najprej nanizane vse nebeljakovinske, nato pa vse beljakovinske jedi), obe kategoriji posebej pa bo naraščajoče uredila po številu kalorij [primeri 41–50].

(3) Dopolnite metodo `narisi` v razredu `Igra` tako, da bo na podlagi tabele `celice` narisala labirint, sestavljen iz enako velikih kvadratnih celic, na podlagi tabele `figure` pa posamezne figure (*pacmane* in *duhove*) znotraj labirinta. Tabela `celice` je kvadratna tabela tipa `Celica[][]`, v kateri element na indeksih `[`i`][`j`]` pove, na katerih stranicah je celica v vrstici z indeksom i in stolpcu z indeksom j obdana s steno. Tabela `figure` (tipa `Figura[]`) pa vsebuje podatke o posameznih figurah. Podatki so predstavljeni z objekti razredov `Pacman` in `Duh`.

```
class Celica {    // notranji razred v razredu Igra
    // vrednost true pove, da se na pripadajoči stranici celice nahaja stena
    private boolean levo, zgoraj, desno, spodaj;
}
abstract class Figura {    // notranji razred v razredu Igra
    private int vrstica;   // indeks vrstice celice, kjer se nahaja figura this
    private int stolpec;   // indeks stolpca celice, kjer se nahaja figura this
    private Color barva;   // barva figure this
}
class Pacman extends Figura {}   // objekt tega razreda predstavlja pacmana
class Duh extends Figura {}      // objekt tega razreda predstavlja duha
```

Vsi pacmani in duhovi se nahajajo na veljavnih in medsebojno različnih položajih. Velikost tabele `celice` znaša vsaj 1×1. Podatki o celicah labirinta so konsistentni.

Labirint naj zavzema celotno krajšo stranico platna, po daljši pa naj bo narisan na sredini platna. Pri risanju se zgledujte po sledeči sliki, ki se nanaša na javne testne primere 7–10 ter na tabeli `CELICE` in `FIGURE` v razredu `Igra`:



Poleg metode `narisi` dopolnite tudi sledeči metodi:

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

  Vrne dolžino stranice celice v odvisnosti od širine (`wp`) in višine (`hp`) platna.

- `public double[] zgorajLevo(double wp, double hp)` [6–10]:

  Vrne tabelo z dvema elementoma, ki podajata koordinati (najprej x, nato y) zgornjega levega kota labirinta.

V primerih 11–20 velja `wp == hp`. V primerih 11–30 velja `figure.length == 0`, kar pomeni, da se vam za do 60% točk s pacmani in duhovi ni treba ukvarjati.