Cvičenia č. 6, úloha č. 1

Priložený ZIP archív obsahuje triedy pre grafy z prednášky, ako aj kostru triedy `Graphs` – všetky tieto triedy sú súčasťou balíka `graphs`. Doprogramujte do triedy `Graphs` telá nasledujúcich troch statických metód:

- Metódy `public static Iterable<Integer> getPredecessors(DirectedGraph g, int v)`, ktorá vráti všetkých *predchodcov* vrcholu `v` grafu `g` v podobe inštancie typu `Iterable<Integer>`, ktorej iterátor každého z týchto predchodcov vráti práve raz (v ľubovoľnom poradí).[1]

  V prípade, že je za `g` dosadená referencia `null`, alebo graf `g` neobsahuje vrchol `v`, malo by dôjsť k vyhodeniu výnimky typu `IllegalArgumentException`.

- Metódy `public static int loopCount(DirectedGraph g)`, ktorá vráti *počet slučiek* v grafe `g`.

  V prípade, že je `g` rovné `null`, malo by dôjsť k vyhodeniu výnimky typu `IllegalArgumentException`.

- Metódy `public static int isolatedVertexCount(DirectedGraph g)`, ktorá vráti *počet izolovaných vrcholov* grafu `g`. Vrchol grafu pritom nazývame *izolovaným*, ak nie je počiatočným ani koncovým vrcholom žiadnej hrany.

  V prípade, že je `g` rovné `null`, malo by dôjsť k vyhodeniu výnimky typu `IllegalArgumentException`.

V triede `Graphs` je už hotová metóda `main`, ktorá z konzoly načíta jeden z reťazcov `LISTS` alebo `MATRIX` zodpovedajúci implementácii grafu a za ním orientovaný graf zvolenej implementácie. Následne načítava príkazy zodpovedajúce volaniam metód opísaných vyššie a vypisuje na konzolu ich výstupy. Príklad vstupu a výstupu možno nájsť nižšie (na jeho plné pochopenie môže byť užitočné pozrieť si implementáciu metódy `main`). Metódu `main`, ani metódu `readDirectedGraph` z prednášky, ktorá je v nej použitá, nemodifikujte.

Na testovač odovzdávajte iba súbor `Graphs.java` obsahujúci kód vami doplnenej triedy `Graphs` v balíku `graphs`. Zvyšné triedy balíka `graphs` budú k vašej triede na testovači priložené; spúšťať sa na testovači bude metóda `main` triedy `Graphs`.

Príklad vstupu:

```
LISTS
7 8
0 1
0 2
0 3
1 2
2 3
2 5
5 3
6 6
getPredecessors 0
getPredecessors 1
getPredecessors 2
getPredecessors 3
getPredecessors 4
getPredecessors 5
getPredecessors 6
getPredecessors 7
loopCount
isolatedVertexCount
```

Príklad výstupu:

```
Predchodcovia vrcholu 0: [].
Predchodcovia vrcholu 1: [0].
Predchodcovia vrcholu 2: [0, 1].
Predchodcovia vrcholu 3: [0, 2, 5].
Predchodcovia vrcholu 4: [].
Predchodcovia vrcholu 5: [2].
Predchodcovia vrcholu 6: [6].
Vynimka typu IllegalArgumentException.
Pocet sluciek v grafe: 1.
Pocet izolovanych vrcholov grafu: 1.
```

[1] Môže teda ísť napríklad o zoznam všetkých predchodcov vrcholu `v`, v ktorom sa žiaden z prvkov neopakuje.