Polsemestrálny test
praktická časť

Ústav informatiky
Prírodovedecká fakulta
UPJŠ v Košiciach

Potrebný zdrojový kód nájdete na stránke, resp. v moodli. Evaluátor vyžaduje tieto názvy tried a tiež názvy metód, ktoré sú zadané. Všetky ostatné názvy premenných, vnorených tried a ostatných pomocných metód si môžete upraviť.

1. Slovíčka z nemčiny (5b) Mišo sa učí nemecké slovíčka. Niektorý deň sa naučí nové slovíčka, iný deň si zase opakuje tie, ktoré sa už učil. Často si opakuje niektoré slovíčka viackrát. Mišo má za sebou sériu dní, počas ktorých sa každý deň venoval učeniu alebo opakovaniu slovíčok. Každý deň si poznačil hodnotu po koľkých dňoch si opakuje slovíčka, prípadne 0 ak sa učí nové slovíčka.

Séria môže vyzerať nasledovne: 1, 4, 2, 0, 0, 0. Interpretujeme to nasledovne (od konca): pondelok, utorok a stredu sa učil nové slovíčka. Vo štvrtok si opakoval slovíčka spred 2 dní, teda z utorka. V piatok si opakoval slovíčka spred 4 dní (teda z pondelka). V sobotu si opakoval to, čomu sa venoval 1 deň dozadu, čiže si zopakoval to čo v piatok. Slovíčka zo stredy si neopakoval vôbec.

Mišo nechce prerušiť sériu, ale nie vždy sa mu chce učiť. Preto si vytvoril jeden *fake* deň, kedy sa v skutočnosti nič neučil a následne vždy keď sa mu nechcelo učiť, si poznačil, že si opakoval slovíčka z daného dňa.

Do triedy `SpajanyZoznam` pridajte metódu `odstranFake`, ktorá modifikuje spájaný zoznam tak, že odstráni všetky záznamy s fake hodnotou. Môžete očakávať korektný vstup, kde na prvom mieste (posledný deň aktuálnej série) je fake deň. Ostatné hodnoty ostávajú nezmenené - aj keď vo výsledku nezodpovedajú jednotlivým dňom a nie je možné rekonštruovať postupnosť učenia, ale aspoň poskytujú informáciu po koľkých dňoch si opakuje slovíčka.

```
public void odstranFake()
```

Metóda nech pracuje v lineárnom čase vzhľadom k dĺžke zoznamu, t.j. $O(n)$, kde n je dĺžka zoznamu. Za viac ako jeden prechod spájaným zoznamom je bodový zisk znížený o polovicu.

Metóda `odstranFake` zmení obsah zoznamu takto:

- `[2, 2, 0, 0]` → `[2, 0]`
- `[3, 1, 0, 0]` → `[1, 0]`
- `[2, 4, 2, 0, 2, 0, 0]` → `[4, 0, 0]`
- `[0, 0, 1, 0, 0]` → `[0, 1, 0, 0]`
- `[0]` → `[]`

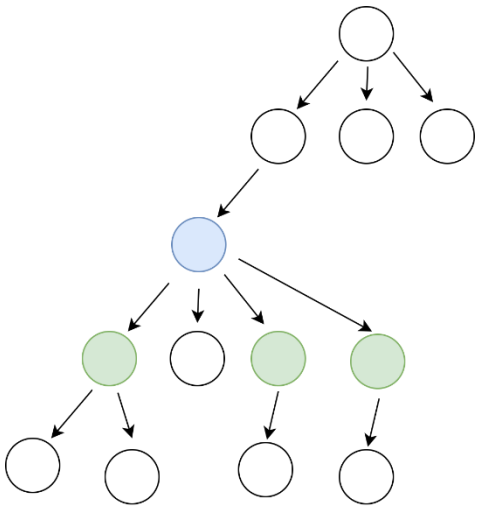
2. Rodokmene z dejepisu (5b) Mišo sa učil aj na dejepis. Dozvedel sa, že pre kráľa v rôznych obdobiach bolo dôležité mať mužského potomka a že to často nebolo jednoduché (viď Henry VIII. alebo Louis XIV.). Niektorí králi mali viacero manželiek, s niektorými nemali deti vôbec, s niektorými mali iba dcéry a občas keď sa narodil chlapec, tak ten zomrel vo veľmi mladom veku.

Ak si zoberieme rodokmeň iba mužských potomkov, na prvý pohľad by sa mohlo zdať, že tá najšťastnejšia osoba je tá, ktorá má najviac synov. Avšak po preštudovaní dejinných udalostí bude Mišo predpokladať, že najštastnejšia osoba v rodokmeni je taká, ktorá má čo najviac synov, ktorí majú aspoň jedného mužského potomka.

Uvažujme triedu `Osoba`, ktorá uchováva meno danej osoby a zoznam potomkov. Doplňte metódu `najstastnejsiDedo`, ktorá vráti koľko synov (majúcich aspoň jedného svojho potomka) má najšťastnejšia osoba.

```
public int najstastnejsiDedo()
```

Príklad: strom na obrázku vráti hodnotu 3, pretože najšťastnejšia osoba (označená modrou farbou) má 3 potomkov, ktorí majú aspoň jedného syna (3 synovia sú označení zelenou farbou).



Poznámka: strom je v evaluátore vypisovaný v tvare ako predpisuje metóda toString, teda vo formáte menoOsoby (dieta0 dieta1 ... dietaN-1)

3. Logické výrazy (5b) Mišo sa na matematike dozvedel, že tautológia je logický výraz, ktorý je vždy pravdivý. Na hodine dostali niekoľko výrazov a jeho úlohou bolo overiť, či je výraz tautológiou alebo nie. Mišo to urobil pomocou tabuľky pravdivostných hodnôt, kde si overil výsledok výrazu po dosadení jednotlivých *true* a *false*.

Vytvorte kód, ktorý pripraví tabuľku pravdivostných hodnôt pre zadaný počet booleovských premenných a zistí, či predložený výraz je tautológia.

```
public boolean jeTautologia(int n)
```

Parameter n označuje počet premenných v logickom výraze (n > 0). Samotný logický výraz vopred nepoznáte. Overenie výrazu pre konkrétne hodnoty premenných môžete vykonať zavolaním statickej metódy:

```
Eval.overVyraz(premenne)
```

kde **premenne** je pole booleanov veľkosti n

V priložených súboroch nájdete aj implementáciu triedy `Eval`, ktorú môžete použiť na otestovanie výsledného kódu. Túto triedu do evaluátora neodovzdávate, reálna implementácia metódy `overVyraz` a teda aj znenie samotných výrazov ostáva pre vás neznáme.