Cvičenia č. 3, úloha č. 2

Stiahnite si kostru k tejto úlohe, ktorá obsahuje dve hotové triedy `Sequence` a `CommandInterpreter` v balíku `sequences`. Trieda `Sequence`:

- Je *abstraktnou* triedou reprezentujúcou nekonečnú postupnosť celých čísel $a_0, a_1, a_2, \ldots$ indexovanú od nuly.
- Obsahuje *abstraktnú* metódu `getValue`, ktorá dostane ako argument celé číslo `index` (môžete predpokladať, že je toto číslo nezáporné) a na výstupe vráti prvok danej postupnosti s indexom `index`.
- Obsahuje tiež implementovanú metódu `getValues`, ktorej jediným argumentom je pole celých čísel `indices`. Táto metóda vráti na výstupe pole celých čísel rovnakej dĺžky, obsahujúce hodnoty reprezentovanej postupnosti na indexoch daných poľom `indices`. Pri výpočte týchto hodnôt volá metóda `getValues` abstraktnú metódu `getValue`. Správanie metódy `getValues` tak závisí od implementácie metódy `getValue` v konkrétnych podtriedach triedy `Sequence` – ide o ukážku polymorfizmu.

Vašou úlohou bude implementovať tri konkrétne podtriedy triedy `Sequence` – všetky by mali byť súčasťou balíka `sequences` a mali by byť uložené v samostatných súboroch:

- Triedu `ArithmeticSequence` reprezentujúcu aritmetickú postupnosť s daným počiatočným (čiže nultým) členom a danou diferenciou (čiže rozdielom medzi každými dvoma jej po sebe idúcimi členmi). Táto trieda by mala poskytovať konštruktor, ktorý dostane dva celočíselné argumenty udávajúce počiatočný člen a diferenciu vytváranej postupnosti (v tomto poradí). Ďalej by mala vhodne prekryť metódu `getValue` z triedy `Sequence`.
- Triedu `GeometricSequence` reprezentujúcu geometrickú postupnosť s daným počiatočným (čiže nultým) členom a daným kvocientom (každý nasledujúci člen postupnosti je daný súčinom kvocientu s predchádzajúcim členom). Táto trieda by opäť mala poskytovať konštruktor, ktorý dostane dva celočíselné argumenty udávajúce počiatočný člen a kvocient vytváranej postupnosti (v tomto poradí). Takisto by mala vhodne prekryť metódu `getValue` z triedy `Sequence`.
- Triedu `FibonacciSequence` reprezentujúcu Fibonacciho postupnosť. Pri tejto triede nie je potrebné explicitne definovať jej konštruktor; postačí aj východzí, automaticky vygenerovaný konštruktor. Trieda by ale mala vhodne prekryť metódu `getValue` z triedy `Sequence`. U *posledného* testovacieho vstupu môžu pomalšie implementácie tejto metódy naraziť na časový limit.[1]

Trieda `CommandInterpreter` už obsahuje metódu `main` načítavajúcu vstup, volajúcu metódu `getValues` pre rôzne postupnosti a polia indexov a vypisujúcu výstup. Vstup pozostáva z niekoľkých príkazov na vytvorenie inštancie niektorej z troch konkrétnych podtried triedy `Sequence`. Každý z týchto príkazov je nasledovaný prirodzeným číslom udávajúcim dĺžku poľa `indices` a napokon samotnými prvkami tohto poľa. Pre načítané pole `indices` metóda `main` zavolá metódu `getValues` vytvorenej postupnosti a výstup tejto metódy vypíše. Príklad vstupu a výstupu je uvedený nižšie.

Na testovač odovzdávajte ZIP archív obsahujúci priečinok `sequences` a v ňom zdrojové súbory všetkých tried tohto balíka (vrátane tých z kostry). Hotové triedy `Sequence` a `CommandInterpreter` nemeňte. Na testovači sa bude spúšťať trieda `CommandInterpreter`.

Príklad vstupu:

```
ARITHMETIC 2 3
7 0 1 2 3 4 5 6
GEOMETRIC 2 3
6 0 1 2 3 4 5
FIBONACCI
7 0 1 2 3 4 5 6
FIBONACCI
4 0 20 0 30
```

Príklad výstupu:

```
2 5 8 11 14 17 20
2 6 18 54 162 486
0 1 1 2 3 5 8
0 6765 0 832040
```

[1] S rôznymi spôsobmi výpočtu Fibonacciho čísel ste sa už stretli v zimnom semestri.