Języki i paradygmaty programowania: Laboratorium nr 12

Podstawowe paradygmaty programowania obiektowego - wprowadzenie. Java - biblioteka standardowa.

2017-2018

mgr inż. Przemysław Walkowiak
dr inż. Michał Ciesielczyk

Instrukcja

W czasie pisania programu pamiętaj o:

1. dbaniu o czytelność kodu (odpowiednie formatowanie kodu, nazewnictwo zmiennych adekwatne do ich znaczenia, komentarze),
2. dbaniu o czytelność interfejsu z użytkownikiem (w sposób jawny pytaj użytkownika jakie dane ma podać oraz opisuj wyniki, które zwracasz),
3. przed fragmentem implementującym poszczególne zadania umieść komentarz: `/*Zadanie X */` oraz wypisz na ekranie analogiczny komunikat (X jest numerem zadania),
4. każde zadanie umieść w oddzielnej klasie z odpowiednimi metodami,
5. zaimplementuj menu wyboru zadania, a następnie wykorzystując pętle **do–while** oraz konstrukcję **switch** wykonaj odpowiedni fragment kodu,
6. w zadaniach wymagających udzielenia komentarza bądź odpowiedzi, należy umieścić go w kodzie programu (np. w postaci komentarza albo wydrukować na ekranie),
7. w zadaniach polegających na zaprojektowaniu klasy należy utworzyć jej instancję i wykorzystać zaimplementowaną funkcjonalność.

Zadania

Zadanie 1

Napisz program mierzący czas tworzenia na wirtualnej maszynie Javy (JVM) listy n-elementowej zawierającej liczby całkowite (od 1 do n) z wykorzystaniem:

(a) tablic,

(b) kolekcji `ArrayList`,

(c) kolekcji `ArrayList`, podając jej maksymalny rozmiar podczas tworzenia,

(d) kolekcji `LinkedList`.

Dostosuj wartość n do możliwości swojego komputera (tak by obliczenia nie trwały dłużej niż kilkanaście sekund). Jakie różnice zauważyłeś dla poszczególnych rodzajów kolekcji? Jak sądzisz – co jest przyczyną tych różnic?

Dodatkowe informacje:

- Do mierzenia czasu wykonania możesz wykorzystać metodę System.currentTimeMillis lub System.nanoTime. Przykładowo:

```
long startTime = System.nanoTime();
// ... the code being measured ...
long estimatedTime = System.nanoTime() - startTime;
```

Zadanie 2

Napisz program w Javie mierzący czas wyznaczania silni liczby naturalnej n (wskazówka: użyj typu `BigInteger`). Wynik wyświetl w notacji naukowej, a wartość n dostosuj do możliwości swojego komputera.

Jaką największą wartość silni udało Ci się wyznaczyć? Dla jakiego n i w jakim czasie? Dlaczego zastosowanie jednego z typów prostych (np. `long`) mogłoby być nieodpowiednie?

Wskazówka Użyj algorytmu iteracyjnego do wyznaczenia silni.

Zadanie 3

Napisz program zliczający liczbę unikalnych słów w podanym pliku tekstowym (ignorując znaki interpunkcyjne, wielkość liter, oraz słowa krótsze niż 3 znaki). Zastanów się z jakiej struktury danych do przechowywania słów najlepiej skorzystać - swój wybór uzasadnij w komentarzu.

Uruchom swój program i wczytaj plik tekstowy *macbeth.txt*. W pliku znajduje się ok. 3145 unikalnych słów (wynik może się nieznacznie różnić w zależności od sposobu tokenizacji tekstu).

Wskazówka 1 Do wczytywania całego pliku tekstowego możesz skorzystać z metody `Files.readAllLines`, np.:

```
Files.readAllLines(new File("macbeth.txt").toPath())
```

Wskazówka 2 Do podziału tekstu na poszczególne słowa możesz wykorzystać metodę String.split podając odpowiednie wyrażenie regularne, np.: `"\\W+"`

Wskazówka 3 Zwróć uwagę, że klasa `String` posiada zdefiniowane metody zwracające długość napisu oraz zamieniające wszystkie litery na małe lub wielkie.

Dodatkowe informacje:

- Zmiana wielkości liter w napisie na małe: String.toLowerCase.

Zadanie 4*

Wykorzystując Java Streams API, zmodyfikuj program z poprzedniego zadania. Przetestuj swoją implementację ponownie zliczając słowa w pliku tekstowym *macbeth.txt*.

Wskazówka 1 Do wczytywania strumienia linii z pliku tekstowego możesz skorzystać z metody `Files.lines`

Wskazówka 2 Do przekształcenia każdej linii w zbiór tokenów możesz wykorzystać metodę `Stream.flatMap` oraz wyrażenia lambda, np.:

```
stream.flatMap((line) -> Arrays.stream(line.split("\\W+"))
```

Wskazówka 3 Do filtrowania elementów oraz ich przekształcania możesz wykorzystać odpowiednio metody `Stream.filter` oraz `Stream.map`.

Wskazówka 4 Do zliczania unikalnych słów możesz wykorzystać metody `Stream.distinct` oraz `Stream.count`.

Dodatkowe informacje:

- Understanding Java 8 Streams API
- Java SE 8: Lambda Quick Start

Zadanie 5

Zmodyfikuj program z zadania 3, w taki sposób aby zliczał liczbę wystąpień każdego słowa. Zastanów się z jakiej struktury danych powinieneś skorzystać?

Wyświetl na ekranie:

a) liczbę wystąpień słowa *macbeth* (286), oraz
b) 20 najczęściej występujących słów razem z ich liczbą wystąpień.

Wskazówka 1 Aby wyznaczyć listę najczęściej występujących słów zamień swoją strukturę na listę par (*słowo*, *liczba wystąpień*), a następnie posortuj malejąco względem liczby wystąpień.

Zadanie 6*

Zmień typ pola stanowisko klasy `Pracownik` z poprzednich zajęć na typ wyliczeniowy (ang. enumerate) zawierający stałe (finalne) pola takie jak:

- nazwa stanowiska,
- jego poziom w hierarchii (np. 1 – Dyrektor, 2 – Kierownik, itd.), oraz
- krótki opis tekstowy.

Dodatkowe informacje:

- Typy wyliczeniowe: `https://docs.oracle.com/javase/tutorial/java/javaOO/enum.html`