# Zad. nr 1: Ewolucja czyli niech programy piszą się same

Wer. 1.01 (10 pkt.)
Zadane: 26 IV 2021

## Wstęp

W tym zadaniu zajmiemy się ewolucją. Wprawdzie ewolucja miała do dyspozycji całą powierzchnię Ziemi i trwała kilka miliardów lat, a my do końca semestru nie mamy nawet skromnego miliona, ale mimo to spróbujemy sobie poradzić. Przy pomocy Javy oczywiście!

Zasymulujmy ewolucję kodu robów (nie będziemy tu wnikać, czy chodzi o kod programów robotów, czy o kod DNA robaków, czy jeszcze co innego).

## Plansza (świat)

Symulacja odbywa się w (*ile_tur*) turach, Świat robów to prostokątna plansza składająca się z *rozmiar_planszy_x*rozmiar_planszy_y* pól. Roby znajdują się na pojedynczych polach i mogą przechodzić między nimi. Zakładamy, że plansza ma posklejane ze sobą brzegi, tak więc po ostatniej kolumnie/wierszu następuje pierwsza/pierwszy i analogicznie przed pierwszą kolumną/pierwszym wierszem. Wynika stąd, że każde pole ma tyle samo sąsiadów.

Na początku symulacji w losowych (nie koniecznie różnych) miejscach planszy umieszczanych jest *pocz_ile_robów* robów, każdy wyposażony w początkowy program *pocz_progr* i *pocz_energia* jednostek energii.

### Pola

Na jednym polu może być dowolnie wiele (w tym zero) robów. Każde pole planszy jest jednego z dwu rodzajów: puste lub żywieniowe. W tym drugim przypadku jest na nim na początku symulacji pożywienie. Jeśli na takie pole wejdzie rob, to zjada to pożywienie (w całości), co daje mu energię do życia (*ile_daje_jedzenie*). Pole zaczyna się regenerować, w tym czasie wchodzące roby nie mogą się pożywić. Po *ile_rośnie_jedzenie* turach na polu znów pojawia się pożywienie. Zje je pierwszy rob, który wejdzie na to pole (roby, które stoją na polu w momencie pojawienia się pożywienia nie zjadają go, w celu zjedzenia pożywienia trzeba wejść na pole - promujemy ruch i zdrowy tryb życia :) ). Również roby, które na początku symulacji zostaną umieszczone na polach z pożywieniem, nie mogą go zjeść.

## Roby

Każdy rob znajduje się na jednym polu planszy i jest skierowany w jednym z czterech kierunków (góra, prawo, dól, lewo). Każdy rob ma swój program. W każdej turze wykonuje go instrukcja po instrukcji (p. spis instrukcji) do końca (chyba że skończy mu się energia). Wykonanie każdej instrukcji zużywa jedną jednostkę energii roba. Samo przeżycie jednej tury (nawet z pustym programem) wymaga *koszt_tury* jednostek energii.

Jedynym sposobem uzupełnienia energii jest wejścia na pole z pożywieniem (tak jak napisano wcześniej, jeśli na polu jest jedzenie, to zjedzenie go daje wtedy robowi *ile_daje_jedzenie* jednostek energii). Jeśli w którymkolwiek momencie (np. w trakcie wykonywania programu) stan zapasu energii roba spadnie poniżej zera, to rob przestaje działać/umiera i od następnej tury nie ma go już na planszy.

### Powielanie

W każdej turze rob może (z prawdopodobieństwem *pr_powielenia*) powielić się. Powielenie oznacza, że w tym samym miejscu pojawia się nowy rob, który ma zmutowany program rodzica i część (*ułamek_energii_rodzica*) jego energii (o tę energię maleje oczywiście zapas energii rodzica). Powielenie jest możliwe tylko wtedy, gdy potencjalny rodzic ma co najmniej *limit_powielania* jednostek energii. Nowopowstały rob jest skierowany w przeciwną stronę niż rodzic. Wykonywanie programu roba oraz zużywanie energii zaczyna się w następnej po powstaniu kolejce.

### Mutacje

Kod programu nowego roba może być inny niż kod rodzica. Z prawdopodobieństwem *pr_usunięcia_instr* nowy program ma usuniętą ostatnią instrukcję (o ile oczywiście kod rodzica nie był pusty). Z prawdopodobieństwem *pr_dodania_instr* nowy program ma na końcu kodu jedną, losowo wybraną z dostępnych (*spis_instr*), instrukcję więcej. Z *pr_zmiany_instr* jedna z instrukcji programu (o ile oczywiście w ogóle jakieś są) jest zmieniona na jedną z dostępnych (być może tę samą). Wszystkie mutacje mogą się zdarzyć przy tworzeniu kodu programu (w podanej kolejności), może też się zdarzyć, że żadna nie zajdzie.

## Spis instrukcji

W kodzie programów mogą występować tylko instrukcje ze *spis_instr*. Ten spis w kolejnym uruchomieniu symulacji może zawierać tylko część z wymienionych poniżej instrukcji (ale może zawierać wszystkie, o ile gdzie indziej nie jest podane inaczej):

- l (lewo) obróć się o 90 stopni w lewo,
- p (prawo) obróć się o 90 stopni w prawo,
- i (idź) idź do przodu o jedno pole (jeśli tam jest pożywienie, to je zjedz),
- w (wąchaj) sprawdź, czy któraś z (czterech) sąsiednich komórek ma pożywienie, jeśli tak, to zwróć się w jej stronę (bez przechodzenia),
- j (jedz) sprawdź, czy któraś z (ośmiu) sąsiednich komórek (także te na ukos) ma pożywienie, jeśli tak, to przejdź tam i zjedz, wpp nic nie rób.

#### Co program ma robić

Program powinien najpierw utworzyć planszę na podstawie pliku plansza.txt.

Następnie powinien wykonać symulację zgodnie z zadanymi parametrami. Program powinien wypisywać w trakcie symulacji co zadaną liczbę tur (*co_ile_wypisz* ) oraz przed i po zakończeniu symulacji tekstowy opis bieżącego stanu symulacji (zawierający m.in. stany wszystkich robów).

Do tego po każdej turze powinien wypisać (w jednym wierszu, w podanej kolejności) podstawowe dane o stanie symulacji: numer tury, liczba robów, liczba pól z żywnością, minimalna, średnia, maksymalna długość programu robów, minimalna, średnia i maksymalna energia roba, minimalny, średni i maksymalny wiek roba, np:

```
245, rob: 120, żyw: 340, prg: 3/4.56/78, energ: 1/4.34/26, wiek: 1/12.46/78
```

## Implementacja

Należy zaimplementować program zgodnie z podaną specyfikacją. Jako wynik należy przesłać spakowane archiwum zip zawierające wszystkie pliki .java (i tylko takie pliki, np. bez plików .class) potrzebne do skompilowania i uruchomienia tego programu oraz podkatalog plansze z przykładowymi . Pliki powinny być w podkatalogach odpowiadających pakietom. Program powinien kompilować się poleceniem:

```
javac zad1/Symulacja.java
```

(\ zamiast / pod Windows) i dawać wykonywać poleceniem:

```
java zad1/Symulacja plansza.txt parametry.txt
```

Wszystkie parametry są ścieżkami do plików z danymi:

- *plansza.txt* zawiera plan planszy (spacja - wolne pole, x - pole z pożywieniem, inne znaki nie są dozwolone). Jeden wiersz odpowiada jednemu wierszowi planszy. Wszystkie wiersze muszą mieć tę samą długość. Liczba wierszy i długość wiersza muszą być dodatnie.
- *parametry.txt* zawiera wiersze z poszczególnymi parametrami symulacji[1] (po jednym w wierszu). Jeden wiersz zawiera nazwę parametru (taką jak w treści zadania) jedną spację i wartość parametru. Kolejność wierszy jest dowolna, każdy parametr (poza rozmiarem planszy, który wynika z pierwszego pliku) musi wystąpić dokładnie raz.

Należy sprawdzić poprawność danych (liczba parametrów programu, warunki podane powyżej). Można założyć (zwn. że tego jeszcze nie przerabialiśmy na zajęciach, a nie dlatego, żeby to było rozsądne założenie), że te ścieżki są poprawne i prowadzą do plików, które da się otworzyć. Do czytania danych należy użyć klasy Scanner. Warto zwrócić uwagę na operacje *next** i *hasNext**). Ciekawy są też konstruktory - można np. mieć w programie dwa różne skanery, np. jeden czytający z pliku, drugi z napisu. Zakładamy kodowanie UTF-8 (nie ma to znaczenia dla poprawnego pliku *plansza.txt*).

Prosimy zwrócić uwagę na obiektowość rozwiązania!

## Możliwe modyfikacje i rozszerzenia

Można to zadanie rozbudowywać (nie można ograniczać). Można dodać inne sposoby zachowań, zaimplementować dodatkowe instrukcje, wprowadzić dodatkowe parametry, zaimplementować wymieranie dinozaurów, czyli zanikanie pożywienia co jakiś czas itp., itd. Zachęcamy do rozszerzania programu, choć nie jest to oczywiście wymagane. Nie jest też oddzielnie punktowane.

Zachęcamy też do eksperymentowania z parametrami (pliki plansza i parametry) programu - nie co dzień ma się okazję śledzić przebieg ewolucji! Jeśli jakiś zestaw parametrów da Państwa zdaniem ciekawe wyniki, to zachęcam do dzielenia się nimi na specjalnym forum Moodle'a (w tym samym module Moodle'a).

Historia zmian:

- wer. 1.0 [26 IV 2021]
- wer. 1.01 [12/13 V 2021]: zmiana frazy "*i wybrać sposób wypełnienia planszy pożywieniem*" na "*na podstawie pliku plansza.txt*" oraz usunięcie frazy "(w kodzie programu)" z części *Co program ma robić.*

[1] Początkowy program robów też jest parametrem.