Specyfikacja funkcjonalna automatu komórkowego

`GameOfLife`: Gra w życie Johna Conwaya

Aleksandra Michalska, Natalia Olszweska

09.03.2021

Spis treści

1 Opis ogólny

1.1 Nazwa programu

Nazwa programu to *"GameOfLife"*.

1.2 Wstęp teoretyczny

Gra w życie Johna Conwaya jest automatem komórkowym, czyli systemem składającym się z pojedynczych komórek. Każda taka komórka znajduje się w jednym ze skończonej liczby stanów (może być martwa lub żywa).

1.3 Cel projektu

Program ma na celu wyświetlanie kolejnych generacji gry w życie przy użyciu konsoli systemowej. Program może działać zarówno w trybie interaktywnym jak i wsadowym. Wybrane obrazy generowane przez program zapisywane mogą być do pliku o rozszerzeniu graficznym.

1.4 Cel dokumentu

Dokument ma na celu przybliżenie kożystania z programu jego użytkownikowi docelowemu.

1.4.1 Użytkownik docelowy

Program jest powszechnie dostępny oraz dedykowany jest dla każdego użytkownika.

2 Opis funkcjonalności

2.1 Możliwości programu

Program może działać w dwóch trybach: `step-by-step` oraz `fast`. Użytkownik może wybrać ile generacji obrazu chce zobaczyć oraz które z nich należy zapisać do pliku graficznego. Dostępna jest opcja ustawienia trybu planszy

2.2 Argumenty wywołania programu

Do poprawnego działania programu potrzebne jest podanie na wejściu wszystkich parametrów podanych poniżej (istnieje moliwość wyboru alternatywnych opcji). Program *GameOfLife* akceptuje następujące argumanty wywołania:

- `-in filein.txt` nazwa pliku z danymi wejściowymi
- `-out fileout.txt` nazwa pliku do którego zapisywana będzie końcowa generacja programu
- `-n 7` ilość generacji do wyświetlenia
- `-s(o5 || f5)` :

  - `-s o5` "save one" - zapisuje `5`- tą generację obrazu do pilku graficznego o rozszerzeniu .bmp (program sam przypisuje nazwę obrazowi)
  - `-s f5` "save first" - zapisuje pierwsze `5` obrazów do plików graficznych o rozszerzeniach .bmp(program sam przypisuje nazwy obrazom)

- `-m(sbs || fast)`:
  - `-m sbs` "step-by-step mode" - tryb krok po kroku; użytkownik naciskając dowolny klawisz poza klawiszem `f` przechodzi do kolejnej generacji. Istnieje możliwość przejścia z trybu `sbs` do trybu `fast` naciskając klawisz `f`.
  - `-m fast` "fast mode" - tryb szybki; kolejne generacje wyświetlają się automatycznie.

- `-how(Ms || Mf || Ns || Nf)`:
  - `M` liczba sąsiadów określana za pomocą sąsiedztwa Moore'a
  - `N` liczba sąsiadów określana za pomocą sąsiedztwa von Neumanna
  - `s` "sphere world" świat działający jak sfera tzn. komórki nie mogą spadaść z brzegów
  - `f` "flat world" świat działajacy jak płaska plansza tzn. komórki mogą spadaść z brzegów

    Przykładowo: `-how Ms` oznacza sferyczny świat i sąsiedztwo Moore'a.

2.3 Jak korzystać z programu?

Aby uruchomić program użytkownik powinien utworzyć plik wejściowy z danymi (jak opisano w sekcji "Dane wejściowe") i następnie, w terminalu, wybrać parametry uruchomienia (jak opisano w sekcji "Argumenty wywołania programu").

Przykładowe uruchomienie programu:

```
./game -in input.txt -out output.txt -n 20 -s o4 -m fast -how Nf
```

oznacza, że:

- dane czytane będą z pliku o nazwie input.txt
- wynikowe dane zostaną zapisane do pliku output.txt
- wyświetlone zostanie 20 generacji
- obraz o numerze 4 zostanie zapisany w postaci pliku graficznego o rozszerzeniu .bmp
- wybrany został tryb `fast`
- wybrane zostało sąsiedztwo von Neumanna i świat w postaci plaskiej planszy

3 Format danych i struktura plików

3.1 Dane wejściowe

Dane wejściowe są przekazywane do programu w pliku tekstowym o rozszerzeniu .txt . W pilku powinny znajdować się następujące dane:

- `w` ilość wierszy $W = \{w \in \mathbb{Z} : 3 \leq w \leq 30\}$
- `k` ilość kolumn $K = \{k \in \mathbb{Z} : 3 \leq k \leq 30\}$
- wypełnienie każdej komórki: 0 (komórka martwa) lub 1 (komórka żywa)

  Przykladowe dane z pliku wejsciowego, wypełniające tabelę o 4 wierszach i 4 kolumnach:

4	4		
0	0	1	0
0	1	0	1
1	0	0	0
0	1	1	0

Tabela 1: Przykładowe dane wejściowe

Generują poniższy obraz początkowy:

3.2 Dane wyjściowe

Dane wyjściowe z programu zapisywane są w postaci pliku tekstowego o rozszerzeniu .txt w takiej samej postaci jak dane w pliku wejściowym (patrz: Tabela 1).

W zależności od użytych przy wywolaniu programu parametrów, zapisywane mogą być także wybrane generacje programu do pliku graficznego o rozszerzeniu .bmp.

4 Scenariusz działania programu

4.1 Scenariusz ogólny

Scenariusz ogólny zakłada brak ingerencji użytkownika w trakcie pracy programu po jego uruchomieniu.

1. Tworzenie pliku wejściowego

   Użytkownik tworzy poprawny plik wjeściowy z danymi (patrz: sekcja 3.1).

2. Uruchomienie

   Użytkownik uruchamia program przy pomocy konsoli systemowej podając parametry:

```
./game -in input.txt -out output.txt -n 10 -s o2 -m fast
-how Ms
```

3. Koniec pracy programu

   Po zakończeniu pracy programu utworzony zostaje plik `output.txt` z danymi ostatniej generacji oraz obraz drugiej generacji.

4.2 Scenariusz szczegółowy

Scenariusz szczegółowy zakłada uruchomienie programu w trybie `step-by-step`.

1. Tworzenie pliku wejściowego

   Użytkownik tworzy poprawny plik wjeściowy z danymi (patrz: sekcja 3.1).

2. Uruchomienie

   Użytkownik uruchamia program przy pomocy konsoli systemowej podając parametry:

   ```
   ./game -in input.txt -out output.txt -n 10 -s f2 -m sbs -how
   Ms
   ```

3. Praca programu w trybie `step-by-step`

   Użytkownik wyświetla kolejne generacje programu wybierając dowolny klawisz poza klawiszem `f`.

4. Zmiana trybu pracy programu

   Znajdując się w trybie `step-by-step`, po wyświetleniu kilku generacji, użytkownik przechodzi do trybu `fast` wybierając klawisz `f`. Użytkownik nie może zmienić ponownie trybu pracy programu.

5. Koniec pracy programu

   Po zakończeniu pracy programu utworzony zostaje plik `output.txt` z danymi ostatniej generacji oraz obrazy pierwszych dwóch generacji.

4.3 Scenariusz w przypadku błędnego uruchomienia

4.3.1 Nieprawidłowy plik z danymi wejściowymi

1. Tworzenie pliku wejściowego Użytkownik tworzy plik z niepoprawnymi danymi np. brak podanej ilości kolumn lub wierszy, nieprawidłowe dane tabeli etc.

2. Uruchomienie Użytkownik próbuje uruchomić program przy użyciu niepoprawnie stworzonego pliku.

3. Koniec pracy programu Program nie daje się uruchomić, wypisuje odpowiedni komunikat błędu i kończy pracę.

4.3.2 Nieprawidłowe argumenty wywołania

1. Tworzenie pliku wejściowego Użytkownik tworzy poprawny plik wjeściowy z danymi (patrz: sekcja 3.1).

2. Uruchomienie Użytkownik próbuje uruchomić program przy pomocy konsoli systemowej podając przykładowe nieprawidłowe parametry:

```
./game -in input.txt -out output.txt -n 10 -s f2 -m sbs
                    -incorrect parameter
```

3. Koniec pracy programu Program nie daje się uruchomić, wypisuje odpowiedni komunikat błędu i kończy pracę.

4.4 Komunikaty błędów

W zależności od rodzaju błędu wyświetlany będzie odpowiedni komunikat:

1. W przypadku nieprawidłowych danych w pliku wejściowym wyświetlany będzie komunikat: `FileError` i parca programu zostanie przerwana.

2. W przypadku podania nieprawidłowych parametrów wywołania programu (nieistniejący parametr, zbyt mało paarametrów, zbyt dużo parametrów), wyświetlany będzie komunikat: `ParametersError` i praca programu zostanie przerwana.