Specyfikacja funkcjonalna – Gra w życie

Krzysztof Dąbrowski i Jakub Bogusz

5 marca 2019

Spis treści

1 Cel projektu **2**

2 Opis ogólny problemu **3**
2.0.1 Struktury 3

3 Działanie programu **5**
3.1 Komunikacja z użytkownikiem 5
3.2 Tryb wsadowy 5
3.3 Przykłady wywołania 6
3.4 Plik wejściowy 6
3.4.1 Przykład 6
3.4.2 Format pliku 7

4 Wyniki działania programu **8**

5 Sytuacje wyjątkowe **9**
5.1 Zmiana domyślnego zachowania 9
5.2 Błędy 9
5.2.1 Błędy pliku wejściowego 9
5.2.2 Błędy losowe 10

Rozdział 1

Cel projektu

Celem projektu jest implementacja gry w życie w języku C. Gotowy program ma przeprowadzać symulację kolejnych pokoleń oraz generować na ich podstawie pliki graficzne przedstawiające etapy symulacji. Generacja odbywa się na podstawie parametrów podanych przez użytkownika w postaci flag. Program działa wyłącznie w trybie wsadowym, co oznacza, że cały proces symulacji odbywać się bez ingerencji użytkownika. Aplikacja może generować jeden z trzech rodzajów plików wyjściowych: obrazów, animacji lub plików tekstowych.

Rozdział 2

Opis ogólny problemu

Gra w życie jest automatem komórkowym wymyślonym przez brytyjskiego matematyka John Horton Conway w 1970 roku. Polega na symulacji kolejnych pokoleń życia komórek według następujących zasad.

Stany Komórka może znajdować się w jednym z dwóch stanów:

- żywa,
- martwa.

Pokolenie to stan wszystkich komórek w danej chwili. Gdy stan pokolenia jest ustalony, możliwe jest utworzenie nowego (potomnego) pokolenia komórek, powstających według poniższych zasad.

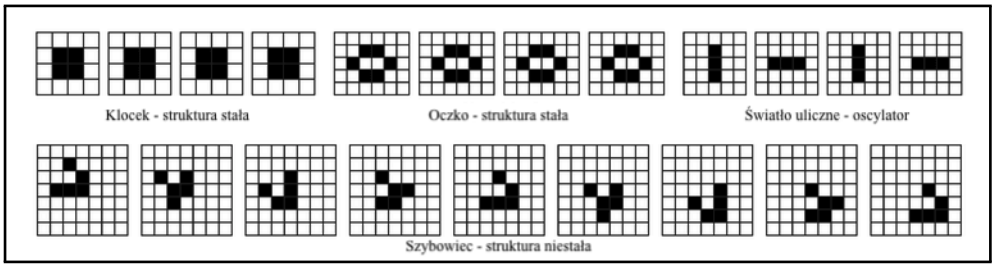
Reguły Następne pokolenie generowane jest zgodnie z regułami:

- Jeżeli komórka była martwa i miała dokładnie 3 żywych sąsiadów, w następnym pokoleniu staje się żywa,
- Jeżeli komórka była żywa to pozostaje żywa jeśli miała dwóch lub trzech żywych sąsiadów. W przeciwnym razie staje się martwa.

2.0.1 Struktury

Symulacja przeprowadzona zgodnie z powyższymi regułami może prowadzić do powstania ciekawych obiektów zwanych strukturami.

Reguły symulacji umożliwiają również tworzenie dużo bardziej skomplikowanych struktur (jak na przykład maszyna Turinga - https://youtu.be/My8AsV7bA94).



Rysunek 2.1: Przykłady struktur

Rozdział 3

Działanie programu

3.1 Komunikacja z użytkownikiem

Program działa w trybie wsadowym. Oznacza to, że użytkownik podaje jedynie argumenty początkowe dla symulacji wraz z uruchomieniem programu (w formie flag), a następnie program automatycznie przetwarza dane i generuje wyniki.

3.2 Tryb wsadowy

Argumenty

- `-h / --help`
  Wyświetlenie pomocy,
- `-f [nazwa pliku] / --file plik [nazwa pliku]`
  Łańcuch znaków będący nazwą pliku z wejściowym stanem planszy zgodny z formatem; wyklucza się z flagą -s
- `-o [ścieżka] / --output_dest [ścieżka]`
  Łańcuch znaków będący ścieżką do folderu, w którym zostaną zapisane wyniki symulacji. Domyślnie pliki będą generowane w folderze o nazwie będącej aktualną datą i godziną wywołania programu,
- `-t [gif | png | txt] / --type [gif | png | txt])`
  Łańcuch znaków reprezentujący typ generowanych rezultatów. Domyślnie gif,
- `-n [liczba] / --number_of_generations [liczba]`
  Liczba pokoleń do wygenerowania. Domyślnie 15,
- `-p [liczba] / --step [liczba]`
  Liczba decydująca o tym, co który stan symulacji będzie zapisywany. Domyślnie 1,

- `-s [LICZBAxLICZBA] / --size [LICZBAxLICZBA]`
  Łańcuch znaków o formacie "XxY" (X – szerokość planszy, Y – wysokość planszy), będący wymiarami losowo generowanej planszy początkowej. Wyklucza się z -f,

- `-d [liczba] / --delay [liczba]`
  Podanie tego argumentu spowoduje wyświetlanie w konsoli kolejnych pokolenie symulacji. *Liczba* ta będzie oznaczać czas w milisekundach między wyświetleniem poszczególnych pokoleń. Wartość -1, co oznacza manualne przechodzenie do następnego pokolenia klawiszem ENTER lub zapisanie wyświetlanego pokolenia w pliku.txt wpisując „save".

Wywołanie programu bez żadnego argumentu przyjmuje flagę –size 10, i wartości domyślne innych parametrów.

Wywołanie programu z błędnymi argumentami lub niewystarczającą ich liczbą wyświetli pomoc wraz z przykładami wywołania i ich opisami.

3.3 Przykłady wywołania

`gra_w_zycie -s 5x8 -d -1`
Program utworzy losową planszę o 5 komórkach szerokości i 8 wysokości i przeprowadzi symulację wyświetlając kolejne pokolenia w konsoli, pomiędzy którymi użytkownik będzie przełączać się wciskając ENTER oraz będzie mógł zapisać aktualnie wyświetlane pokolenie wpisując „save" i zatwierdzając klawiszem ENTER.

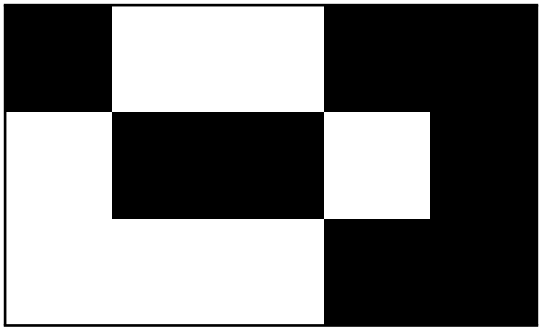
`gra_w_zycie -f dane_wejsciowe.txt -p 3 -t png -n 21`
Program wyczyta początkową planszę z pliku o nazwie „dane_wejsciowe.txt", utworzy nowy folder o nazwie będącej dzisiejszą data oraz godziną i zapisze do niego 7 obrazów w formacie .png (co 3 pokolenie).

3.4 Plik wejściowy

Plik wejściowy pozwala na wczytanie stanu planszy. Dzięki temu użytkownik ma kontrolę nad początkiem symulacji, oraz może kontynuować symulacje z zapisanego wcześniej etapu.

3.4.1 Przykład

5 3	– rozmiar (x y)
1 0 0 1 1	– Wartości poszczególnych komórek
0 1 1 0 1	– 1 - żywa
0 0 0 1 1	– 0 - martwa



Rysunek 3.1: Grafika wygenerowana na podstawie przykładowej planszy

3.4.2 Format pliku

Kodowanie

Ponieważ plik powinien zawierać tylko liczby arabskie i odstępy możliwe jest dowolne kodowanie kompatybilne z ASCII.

Sugerowane kodowania to: ASCII, UTF-8, ISO 8859, Windows-1250

Opis formatu

Plik w pierwszej linii powinien zawierać 2 liczby. Pierwsza z nich oznacza rozmiar planszy w poziomie, druga w pionie.
Następnie plik powinien zawierać tyle linii jaki został podany rozmiar w pionie. W każdej z tych linii powinno być tyle 0 lub 1 ile wynosi rozmiar w poziomie. Zero oznacz komórkę martwą, a jeden komórkę żywą.

Rozdział 4

Wyniki działania programu

Wyniki działania programu będą zależeć od preferencji użytkownika - od podanych argumentów.

- wyświetlić wybraną ilość pokoleń w konsoli,
- wygenerować plik lub pliki .png z reprezentacjami graficznymi kolejnych pokoleń,
- wygenerować plik .gif przedstawiający życie cywilizacji,
- wygenerować plik lub pliki .txt reprezentujący konkretny stan cywilizacji, mogący służyć za plik wejściowy.

Rozdział 5

Sytuacje wyjątkowe

Czasem działanie programu może ulec zmianie na skutek nieprawidłowych danych wejściowych, niestandardowych ustawień wprowadzonych przez użytkownik lub z przyczyn losowych. Ten rozdział opisuje jak program zachowa się w takiej sytuacji, oraz co może ją wywołać.

5.1 Zmiana domyślnego zachowania

Zbyt szeroka plansza W przypadku gdy użytkownik włączy wyświetlanie kolejnych stanów w konsoli ale rozmiar planszy będzie zbyt szeroki by możliwe było jej wyświetlenie bez zawijania wierszy program wyświetli komunikat o niemożliwości wyświetlenia planszy w konsoli. Kolejne pokolenia nie będą wyświetlanie oknie wiersza poleceń ale generacja plików wynikowych nie ulegnie zmianie.
Treść komunikatu: „Wybrana szerokość planszy jest zbyt duża by możliwe było wyświetlenie kolejnych pokoleń w oknie konsoli."

5.2 Błędy

Opis błędów, które mogą wystąpić w trakcie działania programu.

5.2.1 Błędy pliku wejściowego

Podany plik nie istnieje Jeśli ścieżka podana przez użytkownika jest błędna program wyświetli komunikat o braku możliwości otwarcia wskazanego pliku i zakończy pracę.
Treść komunikatu: „Nie udało się otworzyć wskazanego pliku."

Pusty plik W przypadku gdy plik wskazany przez użytkownika będzie pusty program powiadomi o tym i zakończy pracę.
Treść komunikatu: „Wskazany plik wejściowy jest pusty."

Rozmiar planszy nie będący liczbą Jeśli w pierwszej lini pliku znajdować się będą wartości inne niż liczby program nie będzie w stanie wczytać rozmiaru planszy. W takiej sytuacji wyświetli odpowiedni komunikat i zakończy pracę.
Treść komunikatu: „Nie udało się wczytać rozmiaru planszy."

Niedodatni rozmiar planszy Jeśli jeden z wymiarów planszy nie będzie dodatnią liczbą całkowitą program zasygnalizuje błąd i zakończy pracę.
Przykładowy komunikat: „Szerokość musi być większa od 0. Podana szerokość to -5."

Brak nowej linii po rozmiarze planszy W przypadku gdy po wysokości planszy w pliku będzie inny znak niż przejście do nowej lini program zasygnalizuje błąd i zakończy pracę.
Treść komunikatu: „Spodziewany koniec lini po wymiarze planszy."

Błąd przy wczytywaniu stanu komórki Jeśli nie uda się wczytać stanu komórki, na przykład ponieważ w pliku jest za mało linii lub jedna z linii jest zbyt krótka, program wypisze, w którym miejscu pliku wystąpił błąd i zakończy pracę.
Przykładowy komunikat: „Wystąpił błąd przy próbie przeczytania znaku w lini: 3 kolumnie: 8."

Nieprawidłowy znak w pliku Jeśli podczas czytania pliku program napotka nieprawidłowy znak wypisze na jakiej pozycji w pliku napotkany został nieprawidłowy znak, jaki to znak, oraz czego spodziewał się program.
Przykładowy komunikat: „Niewspierany znak napotkany w lini: 2 kolumnie: 1. Spodziewana wartość: 0 lub 1. Napotkana wartość: T"

5.2.2 Błędy losowe

Brak pamięci operacyjnej Gdyby w systemie zabrakło pamięci program nie będzie w stanie funkcjonować poprawnie. Program wyświetli komunikat o błędzie i przerwie pracę.
Treść komunikatu: „Program nie uzyskał pamięci od systemu operacyjnego. Spróbuj uruchomić program ponownie za pewien czas."