Specyfikacja implemetnacyjna automatu komórkowego

`GameOfLife`: Gra w życie Johna Conwaya

Aleksandra Michalska, Natalia Olszweska

09.03.2021

Spis treści

1 Informacje ogólne

1.1 Opis dokumentu

Dany dokument koresponduje z poprzednią specyfikacją - Secyfikacją funkcjonalną

1.2 Środowisko implementacyje

Docelowa gra została napisana w środowiskach Linuxowych, w języku C.

1.3 Informacje o wyświetlanych generacjach

Wyświetlana postać generacji została utowrzona z dwukolorowych kwadraów. Stan żywy komórki został oznaczony białym kolorem, natomiast stan martwy kolorem czarnym. Kolorowe kwadraty zostały wyświetlone przy pomocy Unicodu, żywa - u2b1c, martwa - u2b1b.

2 Opis modułów

2.1 Main

W module głównym, dalej *Main*, została zaimplementowana podstawowa część programu. Moża tam odczytać postać struktury pojedyńczej *generacji*. *Main* odpowiada za obsługę parametrów, czytanie pliku wejściowego oraz zapis końcowej genracji, a także uruchomienie odpowiedniego trybu programu (SBS, FAST). Moduł główny nawiązuje do modułu *Modes*.

2.1.1 Struktura generacji

Struktura *generacji* zawiera:

- zmienną **gen** typu **int**** - dwuwymiarową tablicę zawierającą liczby z zakresu 0-1, oznaczające stan danej komórki.
- zmienną **r** typu **int** - oznaczającą liczbę wierszy tablicy.
- zmienną **c** typu **int** - oznaczającą liczbę kolumn tablicy.
- zmienną **Nr** typu **int** - oznaczającą liczbę kolejnej generacji.

2.1.2 Obsługa parametrów wejściowych

W module *Main* funkcja *main* pobiera parametry z wywołania wsadowego. Parametry te następnie są porządkowane i sprawdzne dzięki funkcji *Switch*. Dzięki tej funkcji parametry wywołania zostają zapisane do zmiennych programu, które następnie przekazywane są do poszczególnych funkcji. W przypadku błędnego typu lub zapisu danych zwaracane zostają kody błędów. Dla ułatwienia działania programu dla niektórych parametrów zostały utowrzone własne typy (przy pomocy *enum*), np. dla parametru "-m" typ danych *modes* może zawierać wartości *sbs* lub *fast*.

2.1.3 Odczyt i zapis plików

Na podstawie pliku wejściowego utowrzona zostaje początkowa genracja, która następnie zostaje przekazana pozostałym funkcją jako argument. Ostatnia generacja natomiast zostaje przekazana z powrotem do funkcji *main* przez, wybraną funkcje trybu jako wartość zwracana. Zostaje ona następnie zapisana do pliku tekstowego, którego nazwę podano przy uruchomieniu programu.

2.1.4 Uruchomienie odpowiedniego trybu

Zależnie od wybranego trybu (sbs, fast) funkjca *main* wywołuje odpowiednią funkcje *SBS* lub *Fast*.

2.2 Modes

W danym module zostały zaimplementowne funkcje obsługi poszczególnych trybów. Nawiązuje on do modułów *SaveImage* oraz *CreateNew*.

2.2.1 SBS

Funkcja *SBS* jako argumenty przyjmuje:

- zmienną **first** typu **generation** - strukturę zawierającą pierwszą generację.
- zmienną **count** typu **int** - proszoną liczbę generacji do wykonania.
- zmienną **how** typu **neighbour** - wybrany tryb sąsiedztwa (Mf, Ms, Nf, Ns).
- zmienną **toSave** typu **char** - określającą wybrany tryb zapisu.
- zmienną **howManyToSave** typu **int** - określającą ile lub którą generację zapisać zależnie od wybranego trybu zapisu.

Funkcja ta po po pojedyńczym wykonaniu oczekuje na wartoś'c podaną przez użytkownika. W przypadku, gdy wyniesie ona **e**, tryb step by step zostanie wyłączony, i zostanie uruchomiony tryb fast. W przeciwnym wypadku zostanie przedstawiona kolejna generacja. Wartość zwracana jest **ostatnią generacją** typu **generation**.

2.2.2 Fast

Funkcja *Fast* jako argumenty przyjmuje:

- zmienną **first** typu **generation** - strukturę zawierającą pierwszą generację.
- zmienną **count** typu **int** - proszoną liczbę generacji do wykonania.
- zmienną **how** typu **neighbour** - wybrany tryb sąsiedztwa (Mf, Ms, Nf, Ns).
- zmienną **toSave** typu **char** - określającą wybrany tryb zapisu.

- zmienną **howManyToSave** typu **int** - określającą ile lub którą generację zapisać zależnie od wybranego trybu zapisu.

Funkcja działa w trybie szybkim. Kolejne generacje są wyświetlane po sobie bez możliwości ingerencji użytkownika. Wartość zwracana jest **ostatnią generacją** typu **generation**.

2.3 SaveImage

W tej części programu zostały utworzone funkcje zapisu danej *generacji* do postaci obrazu. Dany moduł nie korzysta z żanych innych modułów.

2.3.1 SaveGeneration

Funkcja SaveGeneration jako argumenty przyjmuje:

- zmienną **generationToSave** typu **generation** - strukturę zawierającą generację przeznaczoną do zapisu.
- zmienną **numberOfGeneration** typu **int** - liczbę odpowiadającą kolejnej generacji.

Funkcja ta tworzy zapis generacji w pliku o rozszerzeniu *bmp*, przy pomocy bitmapy. SaveGeneration jest typu *void*.

2.4 CreateNew

W module znajduje się funkcja tworząca nową *generacje*, według zasad opisanych również w danym module. Nawiązuje do modułu *HelpCreate*.

2.4.1 Rules

Funckja *Rules* opisuje zasady umierania i ożywania komórek. Jako argumenty przyjmuje:

- zmienną **howManyNeighbours** typu **int** - liczba sąsiadów komórki.
- zmienną **isAlive** typu **int** - określającą stan komórki, gdzie 0 - martwa, 1 - żywa.

Funkcja zwraca zmeinną typu **int** określającą stan komórki w nowej generacji.

2.4.2 New

Funkcja *New* przyjmuje argumenty:

- zmienną **oldGeneration** typu **generation** - struktura starszej generacji.
- zmienną **how** typu **neighbour** - określającą typ sąsiedztwa.

Dzięki użyciu funkcji z modułu *HelpCreate* funkcja *New* tworzy tablicę sąsiedztwa, z której kożysta wraz z funkcją *Rules* do utworzenia nowej generacji. Funkcja zwraca nową generację typu **generation**.

2.5 HelpCreate

W tym module zostały zaimplementowane funkcje pomocnicze przy tworzeniu nowej *generacji*. Nawiązuje do modułu *Neighbor*.

2.5.1 CreateNeighbourhood

Funkcja tworzy tablicę sąsiedztwa. Przyjmuje argumenty:

- zmienną **whichGeneration** typu **generation** - struktura generacji przeznacoznej do utworzenia sąsiedztwa.
- zmienną **how** typu **neighbour** - określającą typ sąsiedztwa.

Funkcja powodułe się do modułu *Neighbor* w celu obliczenia liczby sąsiadów dla każdej komórki. Zwracana jest tablica typu **int**** gdzie wartość w danej komórce odpowiada liczbie sąsiadów w wybranym sąsiedztwie.

2.6 Neighbor

W tej części napisane zostały funkcje potrzebne do sprawdzania liczby sąsiadów pojedyńczej komórki. Ten moduł nie powołuje się na inne moduły.

2.6.1 MooreFlatWorld

Funkcja liczy liczbę sąsiadów danej komórki przy według zasad sąsiedztwa Moore'a oraz założenia płaskiego świata. Oznacza to, że żywa komórka która znajdzie się na brzegu planszy umiera.

Funkcja przyjmuje argumenty:

- zmienną **worldGeneration** typu **generation** - struktura generacji do policzenia sąsiedztwa.
- zmienną **row** typu **int** - indeks wiersza komórki, której chcemy policzyć sąsiedztwo.
- zmienną **column** typu **int** - indeks kolumny komórki, której chcemy policzyć sąsiedztwo.

Funkcja zwraca liczbę typu **int** sąsiadów komórki o podanych indeksach.

2.6.2 MooreSphereWorld

Funkcja liczy liczbę sąsiadów danej komórki przy według zasad sąsiedztwa Moore'a oraz założenia kulistego świata. Oznacza to, że dla komórki znajdującej się na brzegu tablicy, sąsiad, którego szukalibyśmy poza rozmiarem tablicy będzie pojawiał się po przeciwnej stronie tablicy. W ten sposób tablica jest zawijana.

Funkcja przyjmuje argumenty:

- zmienną **worldGeneration** typu **generation** - struktura generacji do policzenia sąsiedztwa.

- zmienną **row** typu **int** - indeks wiersza komórki, której chcemy policzyć sąsiedztwo.
- zmienną **column** typu **int** - indeks kolumny komórki, której chcemy policzyć sąsiedztwo.

Funkcja zwraca liczbę typu **int** sąsiadów komórki o podanych indeksach.

2.6.3 NeumannFlatWorld

Funkcja liczy liczbę sąsiadów danej komórki przy według zasad sąsiedztwa vun Neumanna'a oraz założenia płaskiego świata.
Funkcja przyjmuje argumenty:

- zmienną **worldGeneration** typu **generation** - struktura generacji do policzenia sąsiedztwa.
- zmienną **row** typu **int** - indeks wiersza komórki, której chcemy policzyć sąsiedztwo.
- zmienną **column** typu **int** - indeks kolumny komórki, której chcemy policzyć sąsiedztwo.

Funkcja zwraca liczbę typu **int** sąsiadów komórki o podanych indeksach.

2.6.4 NeumannSphereWorld

Funkcja liczy liczbę sąsiadów danej komórki przy według zasad sąsiedztwa von Neumanna'a oraz założenia kulistego świata.
Funkcja przyjmuje argumenty:

- zmienną **worldGeneration** typu **generation** - struktura generacji do policzenia sąsiedztwa.
- zmienną **row** typu **int** - indeks wiersza komórki, której chcemy policzyć sąsiedztwo.
- zmienną **column** typu **int** - indeks kolumny komórki, której chcemy policzyć sąsiedztwo.

Funkcja zwraca liczbę typu **int** sąsiadów komórki o podanych indeksach.

3 Testowanie

3.1 Użyte narzędzia

Szczególną uwagę przy testach zwrócimy na mazania pamięci. W tym celu program zostanie przetestowany przy pomocy dwóch narzędzi:

- valgrind,
- gdb.

3.2 Wygląd testów

Testowanie zostanie zaimplementowane w trakcie tworzenia kawałków kodu. Np. W trakcie pisania kodu zostanąną sprawdzone poprawne kontrolowanie i zwracanie kodów błędów.

Ponadto niektóre części programu zostaną przetestowane w osobnych plikach. Np. Zapis obrazów z rozszerzeniem bmp.

3.3 Najważniejsze elementy do przestesowania

Kluczowym elementem w testach programu będą sprawdziany luk pamięci. Przy pomocy narzędzi wymienionych w podpunkcie pierwszym zostaną naprawione uszczerki.

Jednak to nie jedyne elementy warte uwagi. Zostaną przetestowane wszelkie możliwe błędy w wprowadzaniu danych wejściowych lub też ich pominięciu.

Ważna jest również poprawa wszelkiej estetyki wyświetalnych informacji.

3.4 Opis testów kluczowych

1. Sprawdzenie działania programu w podania poprawych flag i parametrów. Efekt oczekiwany jest znany.
2. Sprawdzenie działania programu w przypadku podania pliku wejściowego bez podanej liczby kolumn/wierszy. Oczekiwana wartość: FileError.
3. Sprawdzenie działania programu w przypadku podania pliku wejściowego z błędnymi danymi tablicy. Oczekiwana wartość: FileError.
4. Sprawdzenie działania programu w przypadku błędnych flag. Oczekiwana wartość: ParametersError.
5. Sprawdzenie działania zapisu obrazu dla trybu "-s o5". Oczekiwana wartość: zapis piątej generacji.
6. Sprawdzenie działania zapisu obrazów dla trybu "-s f5". Oczekiwana wartość: zapis pierwszych pięciu generacji.
7. Sprawdzenie działania sąsiedztwa dla trybu "-how Ms". Oczekiwana wartość: zwrócona liczba sąsiadów.
8. Sprawdzenie działania sąsiedztwa dla trybu "-how Mf". Oczekiwana wartość: zwrócona liczba sąsiadów.
9. Sprawdzenie działania sąsiedztwa dla trybu "-how Ns". Oczekiwana wartość: zwrócona liczba sąsiadów.
10. Sprawdzenie działania sąsiedztwa dla trybu "-how Nf". Oczekiwana wartość: zwrócona liczba sąsiadów.
11. Sprawdzenie poprawności działania trybu "fast".
12. Sprawdzenie poprawności działania trybu "sbs". Oczekiwanie na interakcje użytkownik.
13. Sprawdzenie poprawności działania trybu "sbs". Opcja przejścia do trybu "fast".

4 Diagram modułów