Specyfikacja implementacyjna – Gra w życie

Krzysztof Dąbrowski i Jakub Bogusz

12 marca 2019

Spis treści

1 Podział na moduły

Program będzie podzielony na współdziałające moduły. Pozwoli to na łatwiejszą modyfikację programu oraz dodawanie nowych funkcjonalności.

DIAGRAM MODÓŁÓW

Spis modułów

- GameOfLife
- ArgumentsParser
- BoardHandler
- Simulator
- Save
- Loader
- GraphicsGenerator

1.1 GameOfLife

Główny moduł kontrolujący przepływ sterowania i danych między pozostałymi modułami.

1.1.1 Funkcje

`int main(int argc, char** args)` – Punkt startowy programu. Z niej wywoływane będą kolejne funkcje. Przyjmować będzie 2 argumenty – argumenty wsadowe programu:

`int argc` – ilość argumentów,

`char** args` – tablica napisów – faktycznych argumentów wywołania programu.

1.2 ArgumentsParser

Moduł odpowiadający za interpretacje podanych wsadowo argumentów programu, konwersji ich oraz zapisu do utworzonej w tym celu struktury.

1.2.1 Funkcje

`params parseArgs(int argc, char** argv)` – będzie przetwarzać argumenty wsadowe programu podane w postaci flag na strukturę zawierającą ustawienia symulacji. Przyjmować będzie 2 argumenty – argumenty wsadowe programu:

`int argc` – ilość argumentów,

`char** args` – tablica napisów, będących faktycznymi argumentami wywołania programu.

Zwracać będzie strukturę zawierająca ustawienia symulacji.

1.2.2 Struktury

Struktura zawierająca ustawienia symulacji:

```
typedef struct{
     int help;
     char* file;
     char* output_dest;
     char* type;
     int number_of_generations;
     int step;
     int delay;
```

}params;

help – informuje o tym czy ma być wyświetlana pomoc,
file – informuje o tym jaka jest ścieżka do pliku wejściowego,
output_dest – informuje o tym jaka jest ścieżka dla pliku/plików wyjściowych,
type – informuje o tym jaki jest typ zwracanych wyników,
number_o_generations – informuje o tym ile pokoleń komórek zostanie wygenerowane,
step – informuje o tym co ile pokoleń zapisywane będą dane wyjściowe,
delay – informuje w jakich odstępach czasu mają się wyświetlać kolejne generacje komórek.

1.3 Simulator

Moduł odpowiadający za przeprowadzenie właściwej symulacji zgodnie z regułami gry w życie.

1.3.1 Funkcje

board simulate(board b, params p) – będzie przeprowadzać symulacje całej gry w życie:

> board b – struktura zawierająca stan planszy,
> params p – struktura zawierająca ustawienia symulacji.

Zwracać będzie strukturę zawierająca końcowy stan planszy.

board nextGen(board b) – będzie generować planszę odpowiadającą następnemu pokoleniu komórek. Jako argument będzie przyjmować:

> board b – struktura zawierająca stan planszy,

Zwracać będzie strukturę zawierająca stan planszy w następnym pokoleniu.

1.4 Loader

Moduł odpowiedzialny za wczytanie planszy początkowej z pliku tekstowego i zapisanie jej do struktury.

1.4.1 Funkcje

board load(char* path) – będzie przetwarzać plik wejściowy i zapisywać go do struktury. Jako argument będzie przyjmować:

`char* path` – ścieżka do pliku wejściowego,

Zwracać będzie strukturę opisującą początkowy stan planszy.

`int* getSize(char* path)` – będzie wczytywać rozmiar planszy zapisanej w pliku wejściowym:

`char* path` – ścieżka do pliku wejściowego,

Zwracać będzie dwuelementowy wektor zawierający rozmiary planszy początkowej.