Sprawozdanie końcowe

Mateusz Nowak

15 kwietnia 2014

1 Wprowadzenie

1.0.1 Specyfikacja funkcjonalna

1.0.2 Przeznaczenie

Celem programu jest symulacja automatu komórkowego (Gra w życie). Program wykonany jest w języku C. Projekt zakłada, że wczytywana siatka jest dwuwymiarowa, a każda z uporządkowanych komórek znajduję się w jednym stanie (jest żywa albo martwa). Użytkownik poprzez odpowiednią komendę zadaje ilość generacji. Nowy stan komórki jest uzależniony od jej obecnego stanu i stanu jej sąsiadów.

1.0.3 Wywołanie

Program wywołujemy pisząc: ./game_life
-r nazwa_danych
-k nazwa_pliku_do_zapisu
-m nazwa_obrazków
-n ilość_wywołań_programu
-f częstotliwość_wydruków
-o .format_obrazków
Format obrazów to .bmp, .pcx, .tga

1.0.4 Dane wejściowe

Plik tekstowy, o postaci
$m\ n\ x\ y$
$\mathrm{a}_{11}...a_{1y}$
$a_{21}...a_{2y}$
...
$a_{x1}...a_{xy}$

Plik z danymi jest podawany jako argument wywołania programu. Pierwszy rząd zawiera informacje o ilości wierszy, ilości kolumn, stanie komórki żywej oraz stanie komórki martwej. Kolejne wiersza są siatką o wymiarach powyższych z podanymi stanami komórek.

1.0.5 Format wyników

Program po każdej generacji tworzy dane wyjściowe w formie plików graficznych. Możliwe jest również zapisanie bieżącej konfiguracji do pliku zewnętrznego o identycznej strukturze z plikiem wejścia, dzięki czemu możliwe jest jego ponowne wczytanie i rozpoczęcie generacji.

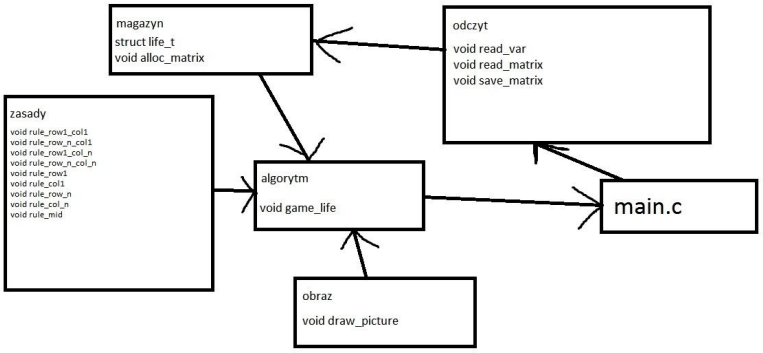
1.1 Specyfikacja implementacyjna

1.1.1 Podział programu

Program został podzielony na 6 modułów współpracujących ze sobą:

- main
- algorytm
- obraz
- zasady
- magazyn
- odczyt

Każdy z tych modułów zawiera szereg funkcji które zostaną pokrótce opisane.



1.1.2 Moduł magazyn

Moduł ten składa się z dwóch plików: *magazyn.h* i *magazyn.c* Plik *magazyn.h* zawiera strukturę:
typedef struct {
int rows; //ilość wierszy
int cols; //ilość kolumn
int **tab;//macierz o rozmiarach rows x cols
int dead; //stan dla komórki martwej
int live; //stan dla komórki żywej
}*life_t;
Zawiera również funkcję: void alloc_matrix(life_t t);, której zadaniem jest alokacja pamięci dla macie- rzy z podanej struktury.
Struktura zadeklarowana w tym pliku wykorzystywana jest przez cały program (przez wszystkie jego składniki). Plik *magazyn.c* zawiera zapis funkcji alloc_matrix.

1.1.3 Moduł odczyt

Moduł zawiera dwa pliki: *odczyt.h* i *odczyt.c* Poza odniesieniem do pliku *magazyn.h* plik *odczyt.h* zawiera 4 funkcje: void read_var(FILE *in, life_t t); zajmuje się odczytem z podanego pliku roz- miaru macierzy i stanów komórki
void read_matrix(FILE *in, life_t t); zajmuje się odczytem z podanego pliku samej macierzy
void save_matrix(FILE *in, life_t t); zajmuje się zapisem do podanego pliku macierzy
void print_matrix(life t t); zajmuje się wydrukiem na ekran macierzy
Plik *odczyt.c* zawiera zapis funkcji podanych w *odczyt.h*

1.1.4 Moduł algorytm

Moduł składa się z *algorytm.h* i *algorytm.c* Plik *algorytm.h* zawiera odniesienie do *magazyn.h* i *zasady.h* Zawiera również funkcję void game_life (life_t t,life_t p,char *name); która otrzymuje dwie struktury:
t - struktura główna z wypełnioną tablicą, p - struktura pomocnicza, do któ- rej zapisywane są wyniki pracy funkcji oraz nazwę pliku graficznego do zapisu. Plik *algorytm.c* zawiera zapis funkcji po- danej w *algorytm.h*, która jest głównym algorytmem projektu

1.1.5 Moduł zasady

Moduł ten składa się z *zasady.h* i zapisu funkcji zawartych w *zasady.c* Plik *zasady.h* zawiera odniesienie do *magazyn.h* Zawiera również 9 funckji
void rule_row1_col1 (life_t a, life_t b); // dla prawego górnego rogu
void rule_row_n_col1 (life_t a, life_t b); // dla lewego dolnego rogu
void rule_row1_col_n (life_t a, life_t b); // dla lewego górnego rogu
void rule_row_n_col_n (life_t a, life_t b); //dla lewego dolnego rogu
Powyższe funkcje zajmują się szczególnymi przypadkami położeń w macierzy prostokątnej/kwadratowej.
void rule_row1 (life_t a, life_t b, int x, int y); // pierwszy wiersz
void rule_row_n (life_t a, life t_b, int x, int y); // ostatni wiersz
void rule_col1 (life_t a, life_t b, int x, int y); // pierwsza kolumna
void rule_col_n (life_t a, life_t b, int x, int y); // ostatnia kolumna
4 powyższe funkcje są odpowiedzialne za odpowiedni dobór sąsiadów w po-danych wierszach/kolumnach
void rule_mid (life_t a, life_t b, int x, int y); // środek macierzy

1.1.6 Moduł obraz

Moduł obraz składa się z *obraz.h* i pliku wykonywalnego *obraz.c*. Plik *zasady.h* zawiera odniesienie do *magazyn.h* i *odczyt.h* oraz deklarację funkcji void draw_picture (life_t t, char *picture name); , która drukuje do pliku w formacie np. .bmp zadany obraz.

1.1.7 Moduł main

Moduł main składa się z pliku *main.c*, który jest sercem całego programu. Posiada odniesienia do modułów: *magazyn*, *odczyt*, *algorytm*. Steruje pracą całego programu i podległych mu modułów i im podległych modułów. Sam zawiera bibliotekę *getopt.h* służącą do oczytu z konsoli odpowiednich opcji i funkcji programu.

2 Decyzje projektowe

- Wybór biblioteki graficznej padł na dobrze opisaną i szeroko wspieraną bibliotekę allegro
- Sposób wczytywania planszy do gry- wczytywanie całej planszy zamiast pojedynczych żywych komórek powoduje, że program jest bardziej zrozumiały dla laika
- Podział programu na 6 modułów i ich logiczne powiązanie
- Sposób komunikacji z użytkownikiem poprzez listę komend - program może być wykorzystywany w automatach itp.
- Użycie funkcji tylko typu void (niżej szczegółowy opis)
- Działanie na dwóch strukturach- struktura główna i kopia
- Rezygnacja z tworzenia folderu wewnątrz programu spowodowana nieuniwersalnością rozwiązania- program mógłby działać tylko pod jednym konkretnym typem systemu.

Wybór tylko funkcji void nie jest przypadkowy. Chciałem, żeby program operował tylko na dwóch strukturach,a nie żeby np. deklarował kolejną strukturę w funkcji odczytu. Według mnie taki sposób wykonania powoduje spore uproszczenie w działaniu programu jak też jest łatwiejsze do zrozumienia.

3 Obsługa i działanie programu

Jak to było zapisane w specyfikacji funkcjonalnej program uruchamiamy: ./game_life -r dane -k tesktowy$_w ychodzcy - mobraz - n100 - f5 - o.bmp$

Obsługa komend uruchamiania została oparta o bibliotekę getopt.
Program po otrzymaniu odpowieniej ilości argumentów kolejno robi:
1. Otwiera plik z danymi
2. Wstępnie deklaruje dwie strutkury biorące udział w programie: main i pom
3. Za pomocą funkcji read_var odczytuje wartości z pierwszej linii pliku do strukury, tj. ilość kolumn ilość wierszy, stan żywej komórki, stan komórki martwej, a następnie przypisuje te same wartości strukturze pom.
4. Deklaracja tablic zawartych w strukturach.
5. Pętla główna wykonująca się tyle razy ile było podane w argumencie -n

-przypisanie wartości z -m do statycznej tablicy znaków

-przypisanie wartości z -k do statycznej tablicy znaków

-warunek jeżeli i jest równe wartości z -f to:

- Przypisanie wartości i do tablicy znaków

- Za pomocą funckji strcat kopiowanie kolejnych modułów nazw plików tj. nazwa+numer+format

- Zwiększenie wartości freq o wartość podaną w -f

-wywołanie funkcji game_life odpowiadającej za główny algorytm gry

-dwie pętle przypisujące wartości ze struktury pom do main

-funkcja zapisująca cały plik w formacie .txt zawierający pierwszy wiersz oraz pole gry

4 Testy

Testy zostały przeprowadzone na kilku podstawowych dla Automatu komórkowego
strukturach takich jak pulsar, latarnia, żabka, szybowiec. Poniżej są przedstawione struktury tych plików, a pliki graficzne są dodane w paczce zip z powodu ich sporego rozmiaru
który psuje cały dokument:

4.1 Pulsar

Struktura pliku:
21 38 1 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

4.2 Latarnia

21 38 1 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

4.3 Żabka

21 38 1 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

4.4 Glider

21 38 1 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

Po takich testach jestem w stanie stwierdzić, że program będzie działał poprawnie dla każdych danych i każdej rozsądnej ilości iteracji oraz częstotliwości. Należy uważać z ilością iteracji, ponieważ zbyt duża ich ilość może w łatwy sposób pozbawić nas przestrzeni na dysku.

5 Podsumowanie

Na koniec chciałem podsumować całą pracę nad projektem. Myślę, że udało mi się zrealizować jego cel w stopniu dobrym. Algorytm jest całkiem szybki jak dla sposobu podawania danych. Oczywiście możnaby użyć przechowywania na jednym bicie informacji o danej komórce, ale moim zdaniem nie usprawni to wiele szybkości działania programu. Program działa stabilnie, jest napisany i sformatowany tak, żeby każda osoba wiedziała co od czego zależy i co jak działa. Wydaje mi sie, że sposób wczytywania przez mój program danych jest naturalny i zrozumiały. Moim głównym założeniem było, żeby program był jasny, bez żadnych często ułatwiających życie fajerwerków, tak aby każda osoba była wstanie samodzielnie zrozumieć kod. W paczce .zip, którą wrzucam na isod'a zawieram: kod źródłowy programu, kod do sprawozdania, samo sprawozdanie, folder obr z przykładowymi testami i ich graficznymi wynikami. Oczywiście cały kod jest wrzucony na zewnętrzne repozytorium:
https://github.com/nowaczq/projekt.git
Praca nad projektem bywała czasami frustrująca, a czasami ciekawa. Ogólnie jestem zadowolony z przebiegu pracy jak i z końcowego wyniku