Life

Jan Salwowski

5 kwietnia 2014

1 Moduły

main.c
file.c
file.h
functions.c
functions.h
life.c
life.h
png.c
png.h

W pliku file.c są zawarte funkcje do czytania/zapisywania konfiguracji z/do pliku, a w pliku file.h ich deklaracje, w functions.c są zawarte funkcje usprawniające obsługę programu, a w functions.h ich deklaracje, w life.c są funkcje dot. działań na komórkach, a w life.h ich deklaracje, natomiast w pliku png.c są funkcje do obsługi plików PNG, a w png.h ich deklaracje.

2 Funkcje i argumenty

2.1 life.c

2.1.1 int** createLife(*int m*, *int n*)

Funkcja służy do tworzenia tablicy o wielkości **m**x**n**, w której będą przechowywane stany komórek. Początkowo będzie wypełniona komórkami martwymi (zerami). Funkcja zwraca tablicę typu *int***.

2.1.2 void setAlive(*int **tab,int x, int y*)

Funkcja ustawia komórkę w pozycji **x**, **y**, jako komórkę żywą (jeden) w tablicy ****tab**.

2.1.3 int** copyLife(*int **tab, int x, int y*)

Funkcja kopiuje stan tablicy ***tab* o wielkości *x*, *y* i zwraca nową talicę.

2.1.4 int checkNeighbour(*int **t, int x, int y, int xdim, int ydim*)

Funkcja sprawdza ilość sąsiadów danej komórki, jeśli ilość jest równa 3, a komórka jest martwa to zwraca 1, jeśli 2 lub 3, a komórka jest żywa to również zwraca 1, jeśli ich ilość jest inna to zwraca 0. Wszystkie operacje są na tablicy **t. Argumenty **int x**, **int y** to rozmiar tablicy.

2.1.5 void newGeneration(*int **tab, int **tmp, int xdim, int ydim*)

Funkcja służy do tworzenia nowej generacji. Pobiera tablicę główną ****tab** oraz tablicę tymczasową ****tmp** oraz ich wymiary *xdim*, *ydim*.Wewnątrz używa funkcji **copyLife** by skopiować tablicę główną ****tab** do tablicy tymczasowej ****tmp**, a następnie wywołuje funkcję **checkNeighbour** dla każdej komórki.

2.1.6 void showLife(*int** tab, int xdim, int ydim*)

Funkcja służy do wyświetlania stanu generacji zapisanej w tablicy ****tab**, o wymiarach **xdim**, **ydim**, w formie 0 i 1 (0 - komórka martwa, 1 - komórka żywa).

2.1.7 struct life makeLife()

Funkcja losuje wymiary tablicy, oraz loso wypełnia ją komórkami żywymi. Funckja zwraca strukturę life, o następującej budowie:

```
struct life{
  int **tab;
  int xdim;
  int ydim;
};
```

gdzie ****tab** to tablica z zapisaną losową generacją, **xdim** i **ydim** to wymiary wygenerowanej tablicy.

2.1.8 void simulateLife(*int **life_tab, int **life_tmp, int x, int y, int nopt, int Nopt, int topt, int s*)

Jest to główna funkcja programu, jej argumentami są: **int **life_tab** - tablica z początkową generacją, *int **life_tmp* - tablica tymczasowa służąca do tworzenia kolejnej generacji, *int x* oraz *int y* - wymiary tablicy, *int nopt* - liczba generacji, *int Nopt* - liczba, co ile generacji ma być tworzone PNG, *int topt* - czas w sekundach ile ma się wyświetlać kolejna generacja na ekranie, *int s* - czy stan generacji ma być wyświetlany na ekranie(0 lub 1).
Funkcja zawiera m.in. pętlę for, w której są wywoływane odpowiednie funkcje to zmieniania, wyświetlania i zapisywania stanu generacji.

2.2 file.c

2.2.1 void loadLife(*char *name, int **t*)

Wczytuje stan generacji z pliku o nazwie **generation_YYYY-MM-DD_HH:mm**.

2.2.2 void saveLife(*char *file, int **tab, int Xdim, int Ydim*)

Funkcja zapisuje stan ostatniej generacji zawartej w tablicy **int **tab** do pliku **char *file**, a **int Xdim**, **int Ydim** to wymiary tablicy.

2.2.3 char* fileName()

Funkcja generuje datę w formacie **YYYY-MM-DD_HH:mm** i zwraca ją.

2.2.4 int fileExists(*const char * filename*)

Funkcja sprawdzająca czy plik o nazwie ***filename** istnieje, a następnei zwraca 1 jeśli tak, a 0 jeśli nie.

2.3 png.c

2.3.1 void encodeOneStep(const char* filename, const unsigned char* image, unsigned width, unsigned height)

Ta funkcja jest pobrana z przykładu (example_1) ze strony biblioteki **lodepng.h**, jest używana to wytworzenia pliku PNG.

2.3.2 void savePNG(*const char* filename*, *int **tab*, *int width*, *int height*)

Zapisuje obraz stanu generacji do pliku o nazwie z argumentu **filename**, ****tab** to tablica z generacją do zapisu, a **int width**, **int height** to wymiary pliku PNG.

2.4 functions.c

2.4.1 void setColor()

Funkcja ustawiająca kolor tekstu na czerwony.

2.4.2 void resetColor()

Funkcja przywracająca podstawowy kolor tekstu.

2.4.3 int isNumeric (*const char * s*)

Funkcja sprawdzająca czy dany argument ***s** składa się ze znaków, które są liczbami. Jeśli tak to zwraca 1, jeśli nie to swraca 0.

2.4.4 struct options getOptions(*struct options opt*, *int argc*, *char **argv*)

Funkcja służąca do wykrywania i ustawiania opcji podanych jako argumenty wywołania programu. Jako argumenty przyjmuje strukturę **options opt** do której ma zapisać dane, **argc** - ilość argumentów wywołania programu, oraz **argv** - tablica elementów wywołania programu. Wykrywa argumenty takie jak: n, N, t, s, a następnie zapisuje je następującej strukturze:

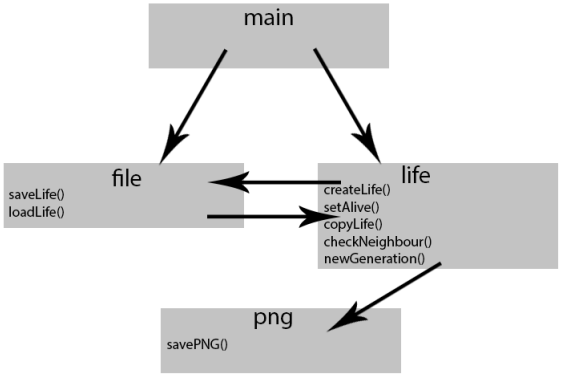
```
struct options{
  int nopt;
  int topt;
  int Nopt;
  int s;
  char *fopt;
};
```

i zwraca ją.

3 Działanie programu

Program poczatkowo alokuje pamięć za pomocą funkcji *createLife*, a następnie wypełnia wczytanymi liczbami z pliku za pomocą funkcji *loadLife*. Następnie za pomocą pętli **for** są tworzone kolejne generacje, przy użyciu funkcji *newGeneration*, następnie w utworzonej kopii generacji po kolei sprawdza komórki i ilość ich sąsiadów za pomocą funkcji *checkNeighbour* i jej wynik zapisuje jako stan danej komórki. Potem wywoływana jest, zgodnie z użytymi opcjami programu, funkcja *savePNG*. Po przejściu całej pętli jest wywoływana funkcja *saveFile* i generacja jest zapisywana do pliku zgodnie z działaniem funkcji *saveFile*.

4 Schemat blokowy



5 Język

Użyty język do pisania projektu to C(89).

6 Biblioteki

Do zapisu plików PNG zostanie użyta biblioteka lodepng.h, a do funkcji getopt w środowisku Windows zostanie użyta biblioteka wingetopt.h.