Sprawozdanie końcowe - gra w życie „Elohim”

Michał Balas, Jan Dobrowolski

12 kwietnia 2019

Spis treści

Zmiany względem specyfikacji

Program Elohim został stworzony z pewnymi odstępstwami od opisu przedstawionego w specyfikacji. Najważniejsze zmiany obejmują:

1. **Brak podziału na biblioteki** – powodem jest zmniejszenie liczby plików oraz fakt, że niektóre moduły korzystają tylko z pojedynczych bibliotek.

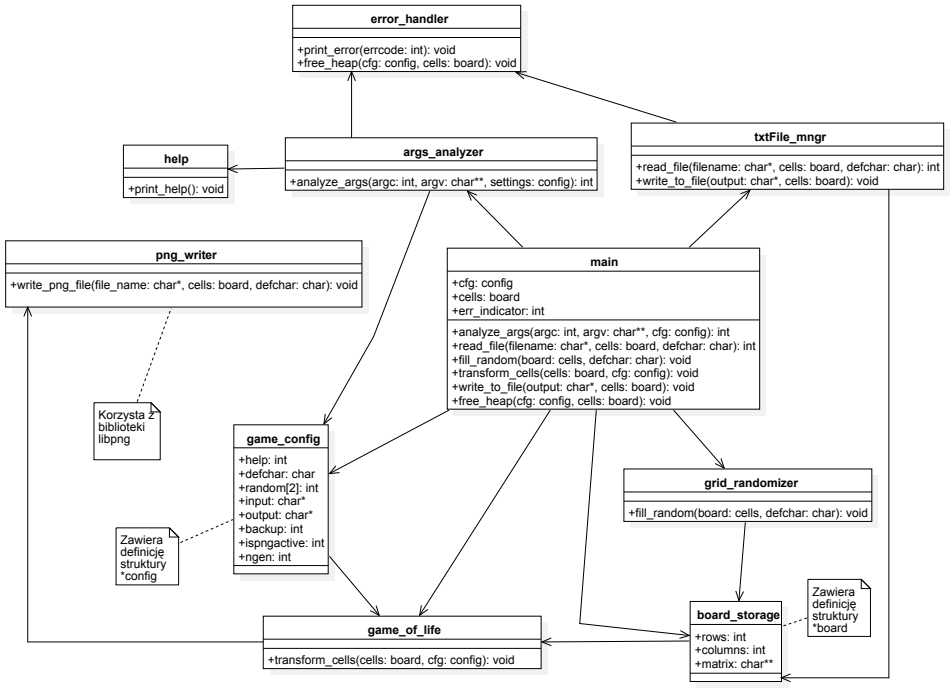
2. **Zmiana zawartości struktury board** – moduł board_storage. Zmieniony został typ tablicy matrix z int** na char** - umożliwiło to wydajniejszą kontrolę błędów (np. przy czytaniu z pliku tekstowego) oraz odczyt plików z dowolnymi znakami bez ich konwersji na 0 i 1.

3. **Scalenie funkcji process_file oraz write_png_file** – moduł png_writer. Dzięki temu można było uniknąć deklarowania zmiennych globalnych lub przekazywania zbyt dużej ilości argumentów między funkcjami. Umożliwiło to również pozbycie się wycieków pamięci.

4. **Zmiana typu danych zwracanych przez funkje: analyze_args oraz read_file** – dotyczy modułów args_analyzer oraz txtFile_mngr. Funkcje te od teraz zwracają wartość 0 jeśli nie napotkały błędów lub nie były to błędy krytyczne, albo 1, gdy wystąpił błąd, który uniemożliwia dalsze działanie programu.

5. **5. Dodanie funkcji free_heap do modułu error_handler** – funkcja ta pozwala w wydajny sposób zwalniać pamięć w przypadku zakończenia programu lub wystąpienia błędu.

Pozostałe zmiany dotyczą niektórych sygnatur funkcji i nie wpływają bezpośrednio na sposób działania programu. Zostały one ujęte wraz z powyższymi na diagramie, który przedstawia ostateczny kształt modułów:



Testy ogólne

Testy programu skompilowanego za pomocą pliku makefile – załączona biblioteka lpng oraz zmieniona nazwa programu (Elohim.exe).

- Losowa plansza 10x10, 5 generacji, tworzenie pliku tekstowego z ostatnią generacją.

```
balasm@jimp:~/testy/gameoflife/final$ ls
args_analyzer.c  error_handler.c  game_of_life.h     help.h    png_writer.c   1
args_analyzer.h  error_handler.h  grid_randomizer.c  main.c    png_writer.h
board_storage.h  game_config.h    grid_randomizer.h  makefile  txtFile_mngr.c
Elohim.exe       game_of_life.c   help.c             plik.txt  txtFile_mngr.h
balasm@jimp:~/testy/gameoflife/final$ ./Elohim.exe --random 10 10 --ngen 5 --backup   2
Output files created in directory output
balasm@jimp:~/testy/gameoflife/final$ ls
args_analyzer.c  error_handler.c  game_of_life.h     help.h    plik.txt       txtFile_mngr.h
args_analyzer.h  error_handler.h  grid_randomizer.c  main.c    png_writer.c
board_storage.h  game_config.h    grid_randomizer.h  makefile  png_writer.h
Elohim.exe       game_of_life.c   help.c             output    txtFile_mngr.c
balasm@jimp:~/testy/gameoflife/final$ cd output                                  3
balasm@jimp:~/testy/gameoflife/final/output$ ls
backup.txt
balasm@jimp:~/testy/gameoflife/final/output$
```

- Plansza czytana z pliku zawierającego znaki „0” i „#”, 10 generacji, zapis png.

```
balasm@jimp:~/testy/gameoflife/final$ ls
args_analyzer.c  error_handler.c  game_of_life.h     help.h    png_writer.c    txtFile_mngr.h
args_analyzer.h  error_handler.h  grid_randomizer.c  main.c    png_writer.h
board_storage.h  game_config.h    grid_randomizer.h  makefile  test.txt
Elohim.exe       game_of_life.c   help.c             plik.txt  txtFile_mngr.c
balasm@jimp:~/testy/gameoflife/final$ ./Elohim.exe --ngen 10 --input test.txt --png --defchar '#'
Input read successfully
10-th generation:
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
Output files created in directory output
balasm@jimp:~/testy/gameoflife/final$ cd output/
balasm@jimp:~/testy/gameoflife/final/output$ ls
000.png  001.png  002.png  003.png  004.png  005.png  006.png  007.png  008.png  009.png  010.png
balasm@jimp:~/testy/gameoflife/final/output$
```

Plik wejściowy test.txt:

```
0 # # # # # 0 0 # 0
# 0 0 # 0 # # 0 0 0
0 # # # # 0 0 # # 0
0 # 0 0 0 # # # 0 #
# # # 0 0 # # 0 0 0
# # 0 0 # # 0 0 # #
0 # # 0 0 # # 0 # #
~
~
~
"test.txt" 7L, 140C zapisano
```

- Losowa plansza 20x20, 10 generacji, zapis png do folderu o nazwie *png*.

```
balasm@jimp:~/testy/gameoflife/final$ ls
args_analyzer.c  error_handler.c  game_of_life.h     help.h    png_writer.c
args_analyzer.h  error_handler.h  grid_randomizer.c  main.c    png_writer.h
board_storage.h  game_config.h    grid_randomizer.h  makefile  txtFile_mngr.c
Elohim.exe       game_of_life.c   help.c             plik.txt  txtFile_mngr.h
balasm@jimp:~/testy/gameoflife/final$ ./Elohim.exe --ngen 10 --random 20 20 --output png --png
Output files created in directory png
balasm@jimp:~/testy/gameoflife/final$ cd png
balasm@jimp:~/testy/gameoflife/final/png$ ls
000.png  001.png  002.png  003.png  004.png  005.png  006.png  007.png  008.png  009.png  010.png
balasm@jimp:~/testy/gameoflife/final/png$
```

- Plansza losowa 100x100, tworzenie pliku backup z symbolem „X” zamiast „1”, 100 generacji.

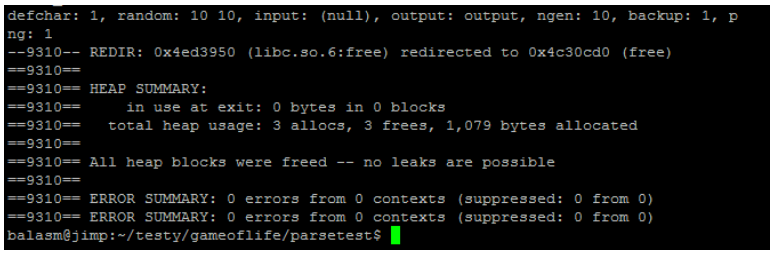
Testy jednostkowe modułów

1. **Test analizatora argumentów**

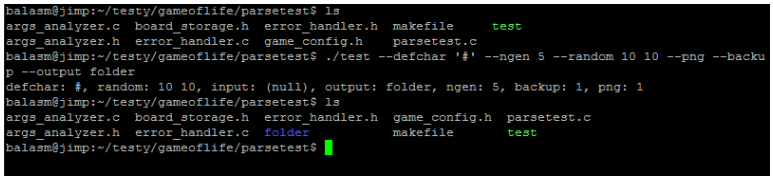
   Nazwa: parsetest.c

   Moduł testujący przyjmuje argumenty wpisywane tak, jak do właściwego programu, po czym zapisuje je w strukturze. Następnie wypisuje elementy struktury do stdout. Dodatkowo podłączony został moduł *error_handler*, który ma na celu jedynie wypisanie komunikatu o nieprawidłowościach i ewentualną korektę błędów.

   Wynik: Sukces – zarejestrowano poprawne działanie oraz brak wycieków pamięci.



   Przykład wywołania:



2. **Test tworzenia png z macierzą**

   Nazwa: pngtest.c

   Argumenty modułu testującego to wymiary tworzonej macierzy (obydwa paramatry typu int). Program tworzy plik png o wymiarach 640x640 px przestawiający wylosowaną macierz o zadanych wymiarach. Plik „out.png" tworzony jest w folderze, gdzie znajduje się plik wykonywalny.

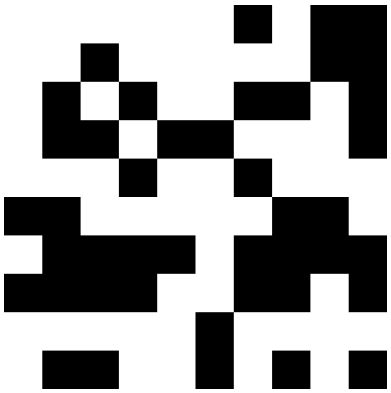
   Wynik: Został utworzony plik out.png w katalogu roboczym. Brak wycieków pamięci.

```
==9455== HEAP SUMMARY:
==9455==     in use at exit: 0 bytes in 0 blocks
==9455==   total heap usage: 1,309 allocs, 1,309 frees, 1,118,904 bytes allocated
==9455==
==9455== All heap blocks were freed -- no leaks are possible
==9455==
==9455== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
==9455== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
balasm@jimp:~/testy/gameoflife/pngtest$
```

   Przykład wywołania:

```
balasm@jimp:~/testy/gameoflife/pngtest$ ls
board_storage.h  game_config.h  makefile  png_writer.c  png_writer.h  test
balasm@jimp:~/testy/gameoflife/pngtest$ ./test 10 10
balasm@jimp:~/testy/gameoflife/pngtest$ ls
board_storage.h  game_config.h  makefile  out.png  png_writer.c  png_writer.h  test
balasm@jimp:~/testy/gameoflife/pngtest$
```

   plik out.png (rzeczywiste wymiary 640x640px):



3. **Test obsługi plików tekstowych**

Nazwa: file_test.c

Celem testu było sprawdzenie poprawności odczytu z pliku txt oraz zapisu danych do pliku txt. Jako argumenty wywołania programu podawana była ścieżka do pliku tekstowego zawierającego macierz przedstawiającą układ komórek, oraz znak stosowany jako oznaczenie żywej komórki. Program czytał plik tekstowy oraz tworzył folder output w bieżącej lokalizacji – tam zapisywany był plik backup.txt z macierzą komórek. Test sprawdzał także reakcje funkcji read_file na błędy – błędny znak w czytanym pliku, zła ścieżka, pusty/nieistniejący plik, zły format macierzy.

Jeśli funkcja read_file napotkała błędy – zwracała wartość jeden, co powodowało wypisanie stosownego komunikatu przez funkcję main.

Poniżej fragment kodu:

```
int n = read_file(input, test, defchar);
if (n==0) {
    for (int i=0; i<test->rows; i++) {
        for (int j=0; j<test->columns; j++)
            printf (``\%c'', test->matrix[i][j]);
        printf(``\n'');
    }
write_to_file(output, test);
} else {
    printf (``Errors encountered! \n'');
    return EXIT_FAILURE;
}
```

Wynik: : Sukces – poprawny odczyt i zapis danych, a takzę reakcja na błędy. Brak wycieków pamięci.

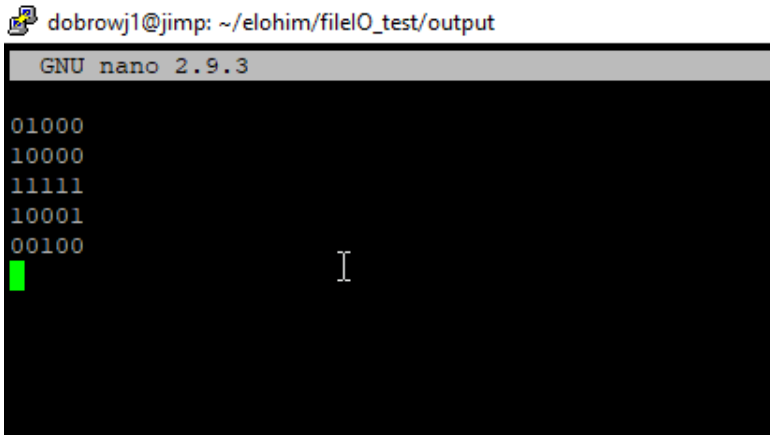
```
Input read successfully
01000
10000
11111
10001
00100
--13350-- REDIR: 0x4ef2540 (libc.so.6:__strcpy_sse2_unaligned) redirected to 0x4c32dd0 (strcpy)
==13350==
==13350== HEAP SUMMARY:
==13350==     in use at exit: 0 bytes in 0 blocks
==13350==   total heap usage: 13 allocs, 13 frees, 18,611 bytes allocated
==13350==
==13350== All heap blocks were freed -- no leaks are possible
==13350==
==13350== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
==13350== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
dobrowjl@jimp:~/elohim/fileIO_test$
```

```
Errors encountered!
==13333==
==13333== HEAP SUMMARY:
==13333==     in use at exit: 0 bytes in 0 blocks
==13333==   total heap usage: 3 allocs, 3 frees, 1,592 bytes allocated
==13333==
==13333== All heap blocks were freed -- no leaks are possible
==13333==
==13333== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
==13333== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
dobrowjl@jimp:~/elohim/fileIO_test$
```

Przykład wywołania:

```
dobrowjl@jimp:~/elohim/fileIO_test$ ./test ~/elohim/txt/test.txt 1
Input read successfully
01000
10000
11111
10001
00100
dobrowjl@jimp:~/elohim/fileIO_test$ ls
board_storage.h  make.txt  test            txtFile_mngr.h
file_test.c      output    txtFile_mngr.c
dobrowjl@jimp:~/elohim/fileIO_test$ cd output
dobrowjl@jimp:~/elohim/fileIO_test/output$ ls
backup.txt
dobrowjl@jimp:~/elohim/fileIO_test/output$
```

Podano ścieżkę do istniejącego pliku. Został on poprawnie odczytany, a następnie jego zawartość zapisana została w folderze output, w pliku backup.txt.



4. **Test funkcji przejścia**

   Nazwa: lifetest.c

   Moduł testujący przyjmuje 3 argumenty: pierwsze dwa to wymiary macierzy do losowego wypełnienia, a trzeci jest liczbą generacji do przeprowadzenia (w celu weryfikacji poprawności działania modułu dobrym zabiegiem jest wywołanie go z parametrami jednocyfrowymi). N-ta generacja macierzy wypisywana jest do stdout.

   Wynik: Wypisano poprawną macierz. Brak wycieków pamięci.

```
Matrix 10x10, 10-th generation
0 0 0 1 0 1 0 0 0 0
0 0 1 0 0 0 0 0 0 0
0 1 0 0 0 1 1 0 0 0
0 1 0 0 0 0 0 0 0 0
0 1 0 0 1 0 0 0 0 0
0 1 0 0 1 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
==9485==
==9485== HEAP SUMMARY:
==9485==     in use at exit: 0 bytes in 0 blocks
==9485==   total heap usage: 25 allocs, 25 frees, 1,448 bytes allocated
==9485==
==9485== All heap blocks were freed -- no leaks are possible
==9485==
==9485== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
==9485== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
balasm@jimp:~/testy/gameoflife/lifetest$
```

   Przykład wywołania:

```
balasm@jimp:~/testy/gameoflife/lifetest$ ls
board_storage.h  game_config.h  game_of_life.c  game_of_life.h  lifetest.c  makefile  test
balasm@jimp:~/testy/gameoflife/lifetest$ ./test 5 5 5
Matrix 5x5, 5-th generation
0 0 1 0 0
0 1 0 1 0
1 1 0 1 1
1 0 0 1 0
1 1 1 0 0
balasm@jimp:~/testy/gameoflife/lifetest$ ./test a b c
Matrix 0x0, 0-th generation
balasm@jimp:~/testy/gameoflife/lifetest$ ./test 10 10 a
Matrix 10x10, 0-th generation
1 1 0 0 1 0 1 0 0 0
0 0 0 1 1 0 1 0 1 0
0 1 0 0 1 0 1 0 1 1
1 0 1 1 1 0 1 0 0 0
0 0 0 1 1 1 1 0 1 0
0 1 0 0 1 1 1 1 0 0
0 1 1 1 1 0 1 0 0 1
0 1 0 0 0 1 1 1 0 0
0 0 0 0 1 1 1 0 0 1
0 1 1 1 0 0 1 0 0 0
balasm@jimp:~/testy/gameoflife/lifetest$
```

5. **Test generatora losowej planszy**

   Nazwa: randomizer_test.c

   Celem testu było wygenerowanie macierzy o zadanych parametrach (ilość rzędów, ilość kolumn, znak do użycia zamiast '1' – 3 kolejne argumenty wywołania programu). Następnie macierz ta wypisywana była na stdout. Test ten nie sprawdza reakcji modułu na błędne parametry – w rzeczywistym programie argumenty podawane do funkcji fill_random są sprawdzane przed jej wywołaniem.

   Wynik: Sukces – nie zarejestrowano błędów ani wycieków pamięci.

```
--12451-- REDIR: 0x4ed3950 (libc.so.6:free) redirected to 0x4c30cd0 (free)
==12451==
==12451== HEAP SUMMARY:
==12451==     in use at exit: 0 bytes in 0 blocks
==12451==   total heap usage: 23 allocs, 23 frees, 1,600 bytes allocated
==12451==
==12451== All heap blocks were freed -- no leaks are possible
==12451==
==12451== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
==12451== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
dobrowjl@jimp:~/elohim/randomizer_test$
```

   Przykład wywołania:

```
dobrowjl@jimp:~/elohim/randomizer_test$ ./test 20 20 X
00XXX0X00XXXXX00X0X0
00XXXX0X00XX000XXXXX
0000X0000XX0000XX0X0
00X0XXX000X00X0X0X00
0XX0XX00XX00XX000000
0XXXXX0X00XXX0X0XXXX
0XX0XX0XXX000XXX0X0X
000X0X00XXXX00XXX0XX
XXXX00X0XXXXXXX0XX00
0XX0XX00XXX000X00000
00XXX0X0XX0X000XX000
XX0X0XX00X00X0XX0XX0
00XX0X0XX0X00XX00000
0X0XX00XX0XX0X000XX0
X00XX0XX0000X00000XX
0XX00X000X00XXX0XX0X
X000000X0X0X0XX0X0XX
XXXX0XXX0XXXXXX0XXXX
0X0XX0X000XX0X000000
XXXX00XXX00XXX00X00X
```