

# Sprawozdanie końcowe Projektu „Gra W Życie”

Andrzej Czechowski, Bartosz Zakrzewski

Data utworzenia: 27.03.2020

Data ostatniej modyfikacji: 31.03.2020

## 1 Podsumowanie projektu

Udało nam się napisać program, który poprawnie tworzy kolejne generacje komórek według „Gry w życie” Johna Conwaya. Prezentuje je w postaci plików PNG - czarno-białej tablicy oraz plików TXT - tablicy zero-jedynkowej.

Nasz program obsługuje sąsiedztwo Moore’a (komórka ma 8 sąsiadów) oraz skrajne komórki nie mają sąsiadów po przeciwnej stronie tablicy („świat gry jest zamknięty”).

## 2 Aktualny stan projektu

## 2.1 Funkcjonalność

Użytkownik stosując określoną budowę argumentów wywołania może decydować o pliku wejściowym, folderu wyjściowym w którym zostaną umieszczone pliki TXT i PNG, liczbie generacji i może zadecydować czy zapisać tylko ostatnia, wybrane czy wszystkie generacje.

## 2.2 Rozwój programu

Nie udało nam się rozwinąć programu o rzeczy opisane w punkcie „4 Rozwój programu” w Specyfikacji Funkcjonalnej.

## 2.3 Ograniczenia

Podczas tworzenia plików PNG następują wycieki pamięci - ciąg dalszy w przykładzie uruchomienia. Nadal pozostaje narzucony format danych wejściowych oraz ich nazw plików wyjściowych. Pozostaliśmy przy maksymalnym rozmiarze planszy 100 x 100 oraz przy maksymalnej liczbie generacji 100.

Przedstawienie błędu tworzenia pliku gen101.png.

[illegible]

caption 1: Błędne tworzenie pliku gen101

## 3 Struktura projektu

### 3.1 Komunikaty błędów

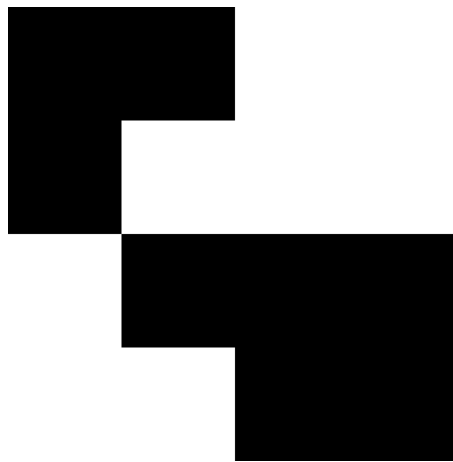
Pogram nie wypisuje nic na sdtin ani stdeer, zwraca jedynie kod błędu. Aby można było zauważyć zwracany kod błędu, zmieniliśmy wartości zwracane z ujemnych na dodatnie oraz uzupełniliśmy punkt „7 Komunikaty błędów / Sytuacje wyjątkowe” ze Specyfikacji Funkcjonalnej.

- brak argumentów wywołania - uruchomi się help:  
kod błędu: 1,
- nie wpisanie wymaganych argumentów do flag (np. nazwy pliku do -input):  
kod błędu: 2,
- podanie nie istniejącego pliku wejściowego:  
kod błędu: 3,
- podanie źle sformatowanego pliku wejściowego:  
kod błędu: 4,
- wpisanie złej liczby generacji:  
kod błędu: 5;
- nie wpisanie wymaganych flag:  
kod błędu: 6;
- przy wpisaniu argumentu wywołania -gen -n x z y, któryś z x z y będzie większy niż n (ilość generacji):  
kod błędu 7;
- nie utworzenie katalogu lub pliku wyjściowego:  
kod błędu 8;
- przepełnienie tablicy, błąd mallocowania:  
kod błędu 9;
- generacja w pliku wejściowym większa niż podana generacja:  
return 10;

### 3.2 Format plików

Format pliku wejściowego i wyjściowego pozostał ściśle określony:

```
4 4
1
1 1 0 0
1 0 0 0
0 1 1 1
0 0 1 1
```

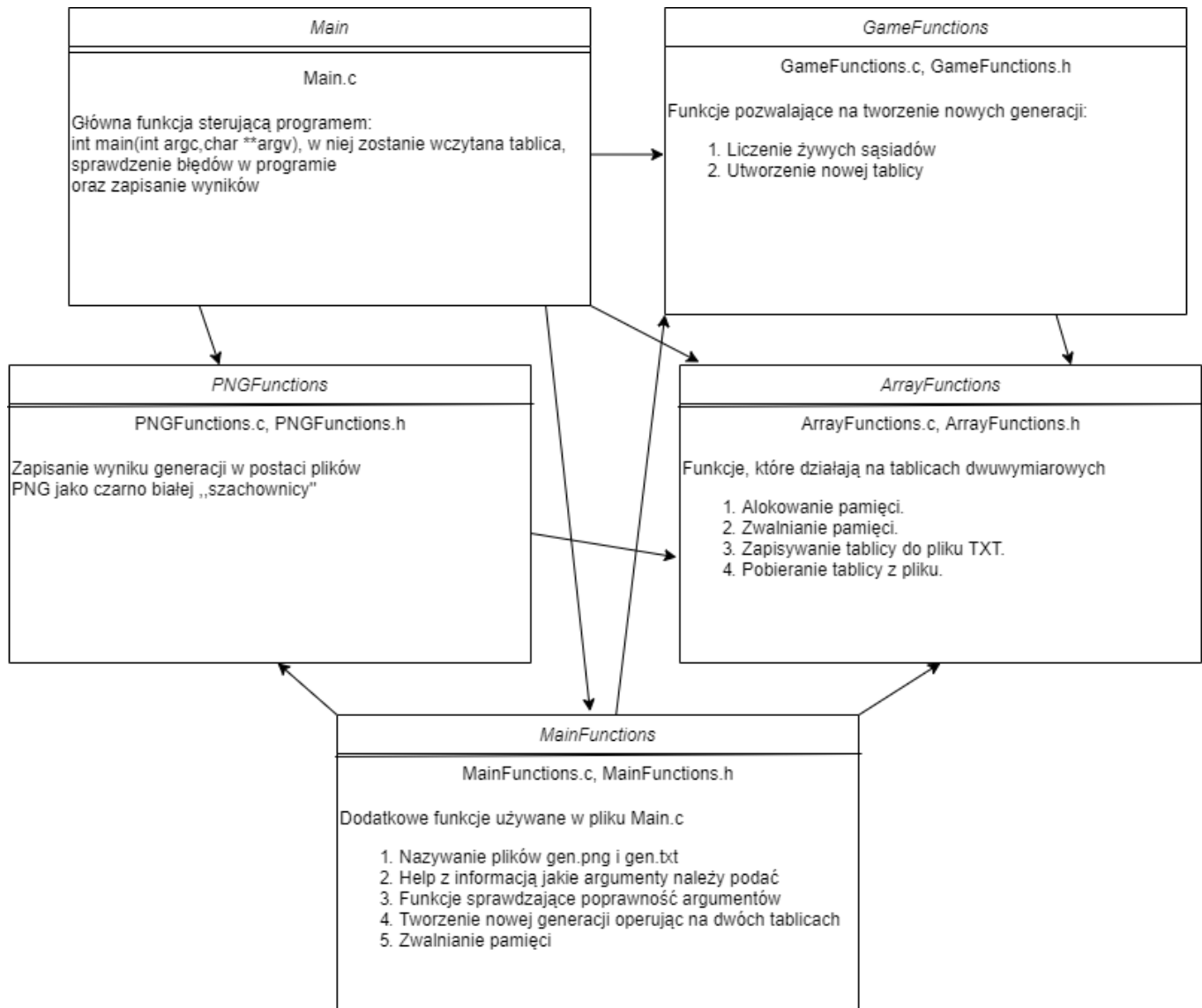


caption 2: Przykładowe przedstawienie generacji jako plik PNG

### 3.3 Użyte biblioteki

- `#include <stdlib.h>`
- `#include <stdio.h>`
- `#include <string.h>`
- `#include <sys/types.h>` // stworzenie nowego folderu
- `#include <sys/stat.h>` // stworzenie nowego folderu
- `#include <unistd.h>` // stworzenie nowego folderu
- `#include <stdarg.h>` // biblioteka z pliku PNGFunctions.c
- `#include <png.h>` // biblioteka z pliku PNGFunctions.c

### 3.4 Diagram plików i funkcji



caption 3: Zaktualizowany diagramu plików i funkcji

## 4 Posumowanie współpracy

### 4.1 Środowisko pracy

Nie zmienialiśmy środowiska pracy - pozostały takie jak opisane w punkcie 2 Specyfikacji Implementacyjnej.

### 4.2 Praca z Gitem

Przestaliśmy pracować na gałęziach `czechoa1`, `zakrzewb`.

Aby rozwinąć program tworzyliśmy gałęzie o nazwie związanej z planowanym rozwinięciem funkcjonalności.

Na gałąź `master` umieszczaliśmy kod przetestowany i dający się skompilować oraz wskazujący na postępy w projekcie.

Trzymaliśmy się zasad commitów opisanych w 3 punkcie Specyfikacji Implementacyjnej.

Postanowiliśmy kod pisać do angielsku, komentarze w kodzie oraz testowe wyświetlanie komunikatów też było po angielsku.

Na `masterze` są dwa tagi: `PoC_C_1` - wersja przedstawiająca Proof of Concept oraz `FINAL` - czyli tag określający wersję do oddania projektu do dnia 1.04.2020

### 4.3 Zasady komunikacji

Ustaliśmy punkt komunikacyjny, którym była nowo stworzona grupa na messengerze.

Pisaliśmy w kodzie komentarze o postępach i rzeczach do zrobienia.

Nie dzieliśmy się zadaniami oraz nie wyznaczaliśmy, ani nie śledziliśmy czasu wykonywania konkretnego zadania.

Oboje wykonaliśmy około tyle samo pracy.

## 5 Uruchomienie programu

Wymagane argumenty wywołania:

- `-input <filename>`
- `-outgen <numer generacji>`

Opcjonalne argumenty wywołania:

- `-output <dirname>`
- `-outgen <n> -SBS`
- `-outgen <n> x y z`

## 5.1 Przykładowe uruchomienie

Tworzenie plików PNG wraz z wyciekami pamięci:

```
bartosz@ubuntu:~/2020L_JIMP2_git_proj_gr12/PROJEKT GRA W ZYCIE/kod$ valgrind ./out -input dane20 -output test -outgen 99 -SBS; echo $?
==11266== Memcheck, a memory error detector
==11266== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==11266== Using Valgrind-3.13.0 and LibVEX; rerun with -h for copyright info
==11266== Command: ./out -input dane20 -output test -outgen 99 -SBS
==11266==
==11266== HEAP SUMMARY:
==11266==    in use at exit: 424,876,476 bytes in 102,367 blocks
==11266==   total heap usage: 202,105 allocs, 99,738 frees, 526,056,960 bytes allocated
==11266==
==11266== LEAK SUMMARY:
==11266==    definitely lost: 965,368 bytes in 390 blocks
==11266==    indirectly lost: 423,539,104 bytes in 101,957 blocks
==11266==    possibly lost: 89,536 bytes in 7 blocks
==11266==    still reachable: 282,468 bytes in 13 blocks
==11266==         suppressed: 0 bytes in 0 blocks
==11266== Rerun with --leak-check=full to see details of leaked memory
==11266==
==11266== For counts of detected and suppressed errors, rerun with: -v
==11266== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
0
```

caption 4: Tworzenie plików PNG - valgrind

```
bartosz@ubuntu:~/2020L_JIMP2_git_proj_gr12/PROJEKT GRA W ZYCIE/kod$ make
cc -c -o Main.o Main.c
cc -c -o ArrayFunctions.o ArrayFunctions.c
cc -c -o GameFunctions.o GameFunctions.c
cc -c -o MainFunctions.o MainFunctions.c
cc -c -o PNGFunctions.o PNGFunctions.c
cc -o out Main.o ArrayFunctions.o GameFunctions.o MainFunctions.o PNGFunctions.o -lpng
bartosz@ubuntu:~/2020L_JIMP2_git_proj_gr12/PROJEKT GRA W ZYCIE/kod$ ./out -input dane20 -output test -outgen 99 -SBS
bartosz@ubuntu:~/2020L_JIMP2_git_proj_gr12/PROJEKT GRA W ZYCIE/kod$ ./out -input dane20 -output test -outgen 99 -SBS; echo $?
0
bartosz@ubuntu:~/2020L_JIMP2_git_proj_gr12/PROJEKT GRA W ZYCIE/kod$ ls
ArrayFunctions.c  ArrayFunctions.o  GameFunctions.c  GameFunctions.o  MainFunctions.c  MainFunctions.o  makefile  PNGFunctions.c  PNGFunctions.o
ArrayFunctions.h  dane20             GameFunctions.h  Main.c           MainFunctions.h  Main.o           out        PNGFunctions.h  test
bartosz@ubuntu:~/2020L_JIMP2_git_proj_gr12/PROJEKT GRA W ZYCIE/kod/test$ ls
gen10.png  gen18.png  gen25.png  gen32.png  gen3.png  gen47.png  gen54.png  gen61.png  gen69.png  gen76.png  gen83.png  gen90.png  gen98.png
gen10.txt  gen18.txt  gen25.txt  gen32.txt  gen3.txt  gen47.txt  gen54.txt  gen61.txt  gen69.txt  gen76.txt  gen83.txt  gen90.txt  gen98.txt
gen11.png  gen19.png  gen26.png  gen33.png  gen40.png  gen48.png  gen55.png  gen62.png  gen6.png  gen77.png  gen84.png  gen91.png  gen99.png
gen11.txt  gen19.txt  gen26.txt  gen33.txt  gen40.txt  gen48.txt  gen55.txt  gen62.txt  gen6.txt  gen77.txt  gen84.txt  gen91.txt  gen99.txt
gen12.png  gen1.png  gen27.png  gen34.png  gen41.png  gen49.png  gen56.png  gen63.png  gen70.png  gen78.png  gen85.png  gen92.png  gen9.png
gen12.txt  gen1.txt  gen27.txt  gen34.txt  gen41.txt  gen49.txt  gen56.txt  gen63.txt  gen70.txt  gen78.txt  gen85.txt  gen92.txt  gen9.txt
gen13.png  gen20.png  gen28.png  gen35.png  gen42.png  gen4.png  gen57.png  gen64.png  gen71.png  gen79.png  gen86.png  gen93.png
gen13.txt  gen20.txt  gen28.txt  gen35.txt  gen42.txt  gen4.txt  gen57.txt  gen64.txt  gen71.txt  gen79.txt  gen86.txt  gen93.txt
gen14.png  gen21.png  gen29.png  gen36.png  gen43.png  gen50.png  gen58.png  gen65.png  gen72.png  gen7.png  gen87.png  gen94.png
gen14.txt  gen21.txt  gen29.txt  gen36.txt  gen43.txt  gen50.txt  gen58.txt  gen65.txt  gen72.txt  gen7.txt  gen87.txt  gen94.txt
gen15.png  gen22.png  gen2.png  gen37.png  gen44.png  gen51.png  gen59.png  gen66.png  gen73.png  gen80.png  gen88.png  gen95.png
gen15.txt  gen22.txt  gen2.txt  gen37.txt  gen44.txt  gen51.txt  gen59.txt  gen66.txt  gen73.txt  gen80.txt  gen88.txt  gen95.txt
gen16.png  gen23.png  gen30.png  gen38.png  gen45.png  gen52.png  gen5.png  gen67.png  gen74.png  gen81.png  gen89.png  gen96.png
gen16.txt  gen23.txt  gen30.txt  gen38.txt  gen45.txt  gen52.txt  gen5.txt  gen67.txt  gen74.txt  gen81.txt  gen89.txt  gen96.txt
gen17.png  gen24.png  gen31.png  gen39.png  gen46.png  gen53.png  gen60.png  gen68.png  gen75.png  gen82.png  gen8.png  gen97.png
gen17.txt  gen24.txt  gen31.txt  gen39.txt  gen46.txt  gen53.txt  gen60.txt  gen68.txt  gen75.txt  gen82.txt  gen8.txt  gen97.txt
bartosz@ubuntu:~/2020L_JIMP2_git_proj_gr12/PROJEKT GRA W ZYCIE/kod/test$
```

caption 5: Tworzenie plików PNG - wyniki



caption 6: Tworzenie plików PNG - wyniki cd



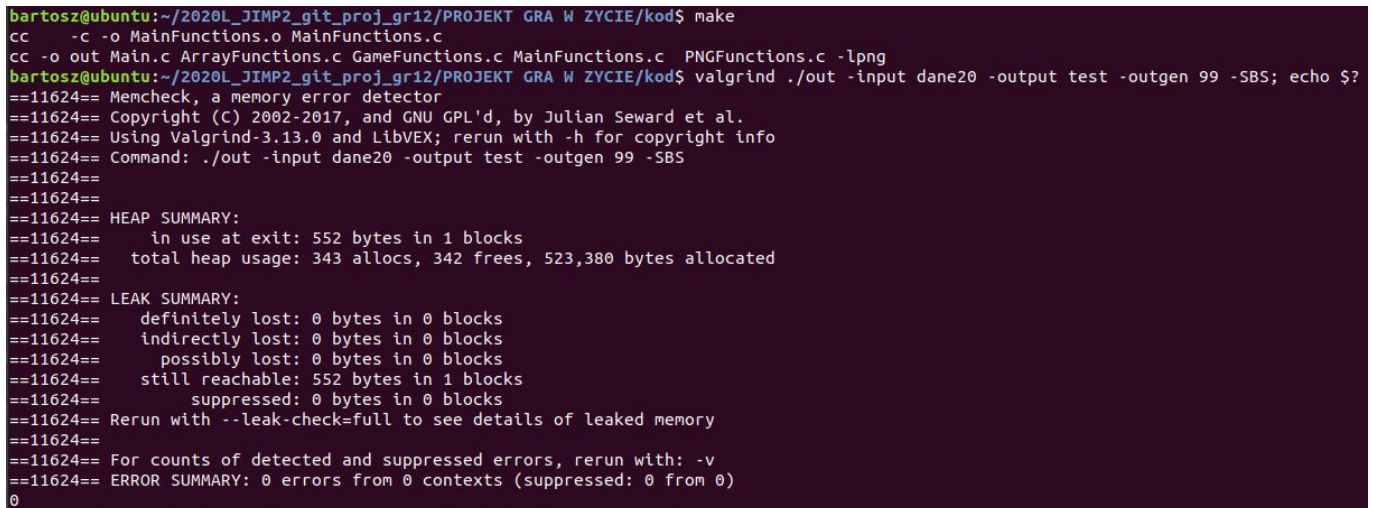
Brak wycieków pamięci przy usunięciu opcji tworzenia plików PNG:



```
Open MainFunctions.c ~/2020L_JIMP2_git_proj_gr12/PROJEKT GRA W ZYCIE/kod Save
help();
return 1;
} else if (inputFlagUsed) {
    printf("No required argument -input!\n");
    help();
    return 1;
}
return 0;
}

int makeNewGeneration(int **arr, int **newArr, int r, int c, int genI, char *outGenFile, int length,
FILE *out, int *makeGen)
{
    makeNextGeneration2DArray(arr, newArr, r, c);
    if (makeGen[genI])
    {
        if ((out = fopen(writeOutFile(outGenFile, length, genI + 1), "w")) == NULL)
        {
            return -8;
        }
        print2DArrayToFile(newArr, r, c, genI + 1, out);
        fclose(out);
        if ((out = fopen(writeOutFile(outGenFile, length, genI + 1), "r")) == NULL)
        {
            return -8;
        }
        //makePNGFile(writeOutFilePNG(outGenFile, length, genI + 1), out);
        fclose(out);
    }
    return 0;
}
```

caption 7: Usunięcie opcji tworzenia plików PNG



```
bartosz@ubuntu:~/2020L_JIMP2_git_proj_gr12/PROJEKT GRA W ZYCIE/kod$ make
cc -c -o MainFunctions.o MainFunctions.c
cc -o out Main.c ArrayFunctions.c GameFunctions.c MainFunctions.c PNGFunctions.c -lpng
bartosz@ubuntu:~/2020L_JIMP2_git_proj_gr12/PROJEKT GRA W ZYCIE/kod$ valgrind ./out -input dane20 -output test -outgen 99 -SBS; echo $?
==11624== Memcheck, a memory error detector
==11624== Copyright (c) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==11624== Using Valgrind-3.13.0 and LibVEX; rerun with -h for copyright info
==11624== Command: ./out -input dane20 -output test -outgen 99 -SBS
==11624==
==11624== HEAP SUMMARY:
==11624==   in use at exit: 552 bytes in 1 blocks
==11624==   total heap usage: 343 allocs, 342 frees, 523,380 bytes allocated
==11624==
==11624== LEAK SUMMARY:
==11624==   definitely lost: 0 bytes in 0 blocks
==11624==   indirectly lost: 0 bytes in 0 blocks
==11624==   possibly lost: 0 bytes in 0 blocks
==11624==   still reachable: 552 bytes in 1 blocks
==11624==   suppressed: 0 bytes in 0 blocks
==11624== Rerun with --leak-check=full to see details of leaked memory
==11624==
==11624== For counts of detected and suppressed errors, rerun with: -v
==11624== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
0
```

caption 8: Usunięcie opcji tworzenia plików PNG - brak wycieków

## 5.2 Przedstawienie reakcji na błędy

Testowanie odbywało się na serwerze jimp.iem.pw.edu.pl, na gałęzi „Testowanie”, gdzie usunęliśmy tworzenie plików PNG i zmieniliśmy kody błędów na dodatnie. Następnie master został z nią zmergowany, dodaliśmy opcję PNG i usunęliśmy foldery testowe.

```
bartosz@ubuntu:~$ ssh zakrzewb@jimp.iem.pw.edu.pl
```

caption 9: Połączenie się z serwerem jimp.iem.pw.edu.pl

```
zakrzewb@jimp:~/2020L_JIMP2_git_proj_gr12/PROJEKT GRA W ZYCIE/kod$ ls
ArrayFunctions.c dane GameFunctions.h MainFunctions.c Main.o PNGFunctions.c test
ArrayFunctions.h dane20 GameFunctions.o MainFunctions.h makefile PNGFunctions.h test1
ArrayFunctions.o GameFunctions.c Main.c MainFunctions.o out PNGFunctions.o testdane
zakrzewb@jimp:~/2020L_JIMP2_git_proj_gr12/PROJEKT GRA W ZYCIE/kod$ cat dane20
20 20
0
0 0 0 0 0 0 0 0 0 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 1 1 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0
0 1 1 0 0 0 1 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 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 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 1 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 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 0 0 0 0 0 0 0 0 0 0 0 0 0

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

zakrzewb@jimp:~/2020L_JIMP2_git_proj_gr12/PROJEKT GRA W ZYCIE/kod$
```

caption 10: Przykładowe dane oraz zawartość folderu kod

Przedstawienie zwracanych kodów błędów:

```
zakrzewb@jimp:~/2020L_JIMP2_git_proj_gr12/PROJEKT GRA W ZYCIE/kod$ ./out -input dane20 -output test3 -outgen 10; echo $?
0
```

caption 11: Kod błędu 0 - poprawnie wykonany program

Błędy przy argumentcie input:

```
zakrzewb@jimp:~/2020L_JIMP2_git_proj_gr12/PROJEKT GRA W ZYCIE/kod$ ./out -input -output test3 -outgen 10; echo $?
2
zakrzewb@jimp:~/2020L_JIMP2_git_proj_gr12/PROJEKT GRA W ZYCIE/kod$ ./out -output test3 -outgen 10; echo $?
No required argument -input!
Required arguments are:
-input <name of input file> and -outgen <number of generations>
Additional argument:
-output <name of directory with PNG and TXT files>
Additional options for -outgen argument:
-outgen <n> -SBS or -outgen <n> <x> <y> <z>
-SBS will save all generations to n. generation
<x> <y> <z> will save generation number x, y and z
6
zakrzewb@jimp:~/2020L_JIMP2_git_proj_gr12/PROJEKT GRA W ZYCIE/kod$ ./out -input dan_e -output test3 -outgen 10; echo $?
3
zakrzewb@jimp:~/2020L_JIMP2_git_proj_gr12/PROJEKT GRA W ZYCIE/kod$ ./out -inputt dan_e -output test3 -outgen 10; echo $?
2
zakrzewb@jimp:~/2020L_JIMP2_git_proj_gr12/PROJEKT GRA W ZYCIE/kod$ ./out -inputt dane20 -output test3 -outgen 10; echo $?
2
```

caption 12: Błędy argumentu input

```
zakrzewb@jimp:~/2020L_JIMP2_git_proj_gr12/PROJEKT GRA W ZYCIE/kod$ ./out -input dane -output test3 -outgen 10; echo $?
10
```

caption 13: Błędy argumentu input cd

Komunikat „No required argument” jest jedynym komunikatem wyświetlającym się użytkownikowi, po to aby później mógł pojawić się help, który podaje wymagane argumenty do uruchomienia programu.

Przykładowy zły format pliku wejściowego:

```
czechoa1@jimp: ~/2020L_JIMP2_git_proj_gr12/PROJEKT GRA W ZYCIE/kod
File Edit View Search Terminal Help

czechoa1@jimp:~/2020L_JIMP2_git_proj_gr12/PROJEKT GRA W ZYCIE/kod$ cat wrongdane
4 4
0
1 1 0 0
1 0 0 0
0 0 1 0
0 0 1 1
3
czechoa1@jimp:~/2020L_JIMP2_git_proj_gr12/PROJEKT GRA W ZYCIE/kod$ ./out -input wrongdane -outgen 10; echo $?
4
czechoa1@jimp:~/2020L_JIMP2_git_proj_gr12/PROJEKT GRA W ZYCIE/kod$
```

caption 14: Zły format pliku wejściowego

Błędy przy argumentcie outgen:

```
zakrzewb@jimp:~/2020L_JIMP2_git_proj_gr12/PROJEKT GRA W ZYCIE/kod$ ./out -input dane20 -output test3 -outgen 10; echo $?
0
zakrzewb@jimp:~/2020L_JIMP2_git_proj_gr12/PROJEKT GRA W ZYCIE/kod$ ./out -input dane20 -output test3 -outgen; echo $?
2
zakrzewb@jimp:~/2020L_JIMP2_git_proj_gr12/PROJEKT GRA W ZYCIE/kod$ ./out -input dane20 -output test3 -outgen 1.5; echo $?
5
zakrzewb@jimp:~/2020L_JIMP2_git_proj_gr12/PROJEKT GRA W ZYCIE/kod$ ./out -input dane20 -output test3 -outgen -7; echo $?
5
zakrzewb@jimp:~/2020L_JIMP2_git_proj_gr12/PROJEKT GRA W ZYCIE/kod$ ./out -input dane20 -output test3 -outgena 1; echo $?
2
zakrzewb@jimp:~/2020L_JIMP2_git_proj_gr12/PROJEKT GRA W ZYCIE/kod$ ./out -input dane20 -output test3; echo $?
No required argument -outgen!
Required arguments are:
-input <name of input file> and -outgen <number of generations>
Additional argument:
-output <name of directory with PNG and TXT files>
Additional options for -outgen argument:
-outgen <n> -SBS or -outgen <n> <x> <y> <z>
-SBS will save all generations to n. generation
<x> <y> <z> will save generation number x, y and z
6
```

caption 15: Błędy argumentu outgen

```
zakrzewb@jimp:~/2020L_JIMP2_git_proj_gr12/PROJEKT GRA W ZYCIE/kod$ ./out -input dane -output test3 -outgen 10; echo $?
10
```

caption 16: Błędy argumentu outgen cd

Błędy przy argumentcie output:

```
zakrzewb@jimp:~/2020L_JIMP2_git_proj_gr12/PROJEKT GRA W ZYCIE/kod$ ./out -input dane20 -output -outgen 10; echo $?
2
zakrzewb@jimp:~/2020L_JIMP2_git_proj_gr12/PROJEKT GRA W ZYCIE/kod$ ./out -input dane20 -outputt test3 -outgen 10; echo $?
2
zakrzewb@jimp:~/2020L_JIMP2_git_proj_gr12/PROJEKT GRA W ZYCIE/kod$
```

caption 17: Błędy argumentu output

```
czechoa1@jimp: ~/2020L_JIMP2_git_proj_gr12/PROJEKT GRA W ZYCIE/kod
File Edit View Search Terminal Help
czechoa1@jimp:~/2020L_JIMP2_git_proj_gr12/PROJEKT GRA W ZYCIE/kod$ ./out -input dane -output test2/test6 -outgen 10 -SBS; echo $?
8
czechoa1@jimp:~/2020L_JIMP2_git_proj_gr12/PROJEKT GRA W ZYCIE/kod$
```

caption 18: Błąd tworzenia folderu



Uruchomienie help:

```
zakrzewb@jimp:~/2020L_JIMP2_git_proj_gr12/PROJEKT GRA W ZYCIE/kod$ ./out
Required arguments are:
-input <name of input file> and -outgen <number of generations>
Additional argument:
-output <name of directory with PNG and TXT files>
Additional options for -outgen argument:
-outgen <n> -SBS or -outgen <n> <x> <y> <z>
-SBS will save all generations to n. generation
<x> <y> <z> will save generation number x, y and z
zakrzewb@jimp:~/2020L_JIMP2_git_proj_gr12/PROJEKT GRA W ZYCIE/kod$ ./out -h
Required arguments are:
-input <name of input file> and -outgen <number of generations>
Additional argument:
-output <name of directory with PNG and TXT files>
Additional options for -outgen argument:
-outgen <n> -SBS or -outgen <n> <x> <y> <z>
-SBS will save all generations to n. generation
<x> <y> <z> will save generation number x, y and z
zakrzewb@jimp:~/2020L_JIMP2_git_proj_gr12/PROJEKT GRA W ZYCIE/kod$
```

caption 19: Help

## 6 Wnioski po wykonaniu projektu

### 6.1 Wnioski z Proof of Concept

Jednym zauważonych problemu w Proof of Concept było tworzenie pustego folderu przy podaniu błędnych argumentów wywołania. Rozwiązaniem była zmiana organizacji kodu. Najpierw wczytujemy argumenty wywołania i pliki wejściowe, aby zapobiec możliwym błędom, a następnie wykonuje się program.

## 6.2 Co można by usprawnić w działaniu programu

- Dodanie jednej funkcji `Error()`, która czyściła by zamallocowane tablice oraz zwracała kod błędu z opisem na `stderr`.
- Naprawienie wycieków pamięci oraz zwiększenie maksymalnej liczby generacji.
- W podstawowych wymaganiach programu nie ma sąsiedztwa von Neumanna czy „świata otwartego”, jednak jest to funkcjonalność, którą można by było mieć w programie.

## 6.3 Organizacja

- Zaplanowanie kolejności dodawania modułów.
- Pisanie w kodzie o postępach pracy oraz rzeczy do zrobienia lepiej się sprawdza niż w na grupie w messengerze, jednak nie tak prosto jest przekazać jakie zmiany się dokonało.
- Postępy w projekcie powinny być lepiej opisywane - np. podczas codziennych/cotygodniowych spotkań
- Zadania powinny być rozdzielone między uczestników projektu.
- Powinniśmy śledzić czas pracy, jeżeli chcemy dokładnie określić, że wkład jest taki sam (np. jeżeli byłoby jakieś wynagrodzenie projektu).
- Pomocne w projekcie byłoby narzędzie do wspólnego przechowywania screenów ekranu (np. wspólny Dysk Google).
- W Specyfikacji Implementacyjnej nie było określonego nazewnictwa funkcji i plików ani języka komentarzy w kodzie czy wyświetlania komunikatów.

## 7 Źródła

- Ten dokument został utworzony w LaTeX’ie za pomocą strony <https://www.overleaf.com>
- Jest to czwarty z kolei dokument dotyczący projektu „Gra w Życie”.
- Dokument ten nawiązuje do Specyfikacji Funkcjonalnej, Specyfikacji Implementacyjnej, Proof of concept projektu „Gra w Życie” w języku programowania C