MODUL 9 - LAPORAN 2 TUGAS BESAR - CONWAY'S GAME OF LIFE

Andhika Rahadian (13218030)
Sabrina Adeline (13218014)
Melia Fatimah (13218020)
Muhammad Adnan (13218019)
Asisten: Arief Hirmanto (13217076)
Kelompok/Rombongan : 3 / A
Tanggal Percobaan: 03/04/2020 - 17/04/2020
EL2208-Praktikum Pemecahan Masalah dengan C
Laboratorium Dasar Teknik Elektro - Sekolah Teknik Elektro dan Informatika ITB

Abstrak

Pada modul 9 ini adalah modul terakhir yang ada dalam mata kuliah EL2208 Praktikum Pemecahan Masalah dengan C. Berbeda dari modul modul sebelumnya, Modul 9 ini merupakan tugas besar yang dikerjakan secara berkelompok. Kami mendapatkan topik permasalahan 1 : Conway's Game Of Life dimana solusi permasalahannya akan diimplementasikan dalam bahasa C. Selain itu, praktikan juga diwajibkan untuk memakai Version Control System (VCS) untuk menyelesaikan masalah ini. Dan kami menggunakan platform GitHub sebagai VCS yang membantu kami menyelesaikan tugas besar ini.

Kata kunci: Tick, Animate, Seed, Cell.

1. PENDAHULUAN

Game yang akan dibuat pada Tugas Besar kali ini bernama Game of Life. Game of Life adalah permainan yang sudah ada sejak 1970 yang dikembangkan oleh John Horton Conway. Pada dasarnya game ini tidak memiliki pemain sehingga permainan akan berjalan bergantung dengan kondisi awal.

Dari aturan game secara real menurut sumber[1], "world" didefinisikan sebagai kotak kotak grid (orthogonal grid) yang merepresentasikan keadaan sel yang berbentuk kotak kotak berukuran tak hingga x tak hingga yang berisikan sel mati atau sel hidup. Namun untuk kasus tugas besar ini, world tersebut memiliki dimensi yang ditentukan sesuai dengan test case file seed yang ada, dan untuk isi dari kotak tersebut, character "-" menunjukkan (sel mati), character "X" menunjukkan sel hidup.

Setelah itu, akan ada yang namanya perubahan generasi yang biasa disebut dengan iterasi. Kondisi kondisi yang menyebabkan sel sel dari suatu world itu menjadi mati atau menjadi hidup atau tidak berubah adalah sebagai berikut :

1. Jika terdapat sebuah sel hidup yang memiliki kurang dari sama dengan 1 tetangga yang hidup, sel tersebut mati pada iterasi selanjutnya (underpopulation).
2. Jika terdapat sebuah sel hidup yang memiliki 2 tetangga yang hidup, sel tersebut tetap hidup pada iterasi selanjutnya (next generation).
3. Jika terdapat sebuah sel hidup yang memiliki lebih dari sama dengan 4 tetangga yang hidup, sel tersebut mati pada iterasi selanjutnya (overpopulation).
4. Jika terdapat sebuah sel mati yang memiliki 3 tetangga yang hidup, sel tersebut menjadi hidup pada iterasi selanjutnya (reproduction).

Game akan berjalan dengan meminta file eksternal berupa seedData dan dilanjutkan dengan pilihan menu tertentu yang akan dijelaskan lebih lanjut pada bagian selanjutnya dari laporan ini.

2. ALGORITMA YANG DIAJUKAN

Program akan diawali dengan menu yang berisi judul dari program dan deskripsi dari program. Setelah itu program akan menerima nama file eksternal dari input dan merubah format dari file menjadi array. Lalu program akan menjalankan menu dengan 3 pilihan input selanjutnya, yaitu:

1. Fungsi Animate : fungsi pada program yang berfungsi untuk mencetak array seed dengan parameter yang diinput sebagai jumlah iterasi
2. Fungsi Tick : Fungsi Tick merupakan fungsi pada program untuk meng iterasi satu generasi setelah array seed input. Fungsi ini dibuat dengan *return* array 2D baru yang berbentuk pointer lalu akan dicetak dengan pemanggilan displaySeed. Fungsi ini diimplementasikan dengan referensi ke [3] pada daftar pustaka.

3. Kondisi Quit : Quit adalah kondisi pada program yang akan berfungsi untuk keluar atau mereset program.

Lalu ada beberapa fungsi lainnya dari program yang dibuat, yaitu:

1. Fungsi getSeedData : fungsi untuk mengimport data seed pada file txt agar array seed dapat diproses pada main program. Fungsi ini me-*return* array 2D dalam bentuk pointer.
2. Fungsi displaySeed : mencetak gambar dari Seed yang telah disalin ke dalam program oleh fungsi getSeedData atau Tick.
3. Fungsi displayMenu : menampilkan menu pilihan untuk perintah yang ingin dijalankan yaitu Animate, Tick, dan Exit.
4. Fungsi displayInteface : mencetak tampilan awal dari program utama setelah dijalankan, sebelum meminta nama file untuk dibuka.
5. Fungsi destroyArray : fungsi yang digunakan untuk membebaskan memori yang sudah dialokasikan untuk membuat array 2D sebagai representasi world dari cell nya. Pada awalnya sedikit lebih panjang untuk implementasinya karena menggunakan for loop namun terdapat referensi yang implementasinya lebih sederhana sehingga kami gunakan yang lebih sederhana. Referensi dapat dilihat dari sumber [2] daftar pustaka.
6. Fungsi delay : berguna untuk memberikan jeda waktu antara tiap proses pencetakan seed ke array dengan inputan waktu dalam milisecond. [4]

Fungsi-fungsi tersebut akan diintegrasikan dalam satu program utama main.c yang akan meminta input user berupa filename/ file seed yang diinginkan untuk diproses sebagai kondisi awal. Lalu akan di iterasi dengan cara tick atau animate sesuai dengan keinginan user.

3. Struktur File Program dalam Repository

Repository tugas besar terdiri atas 5 buah branch yaitu branch master yang merupakan branch utama, branch animate, convertseedtoarray, display, dan function-tick. branch master terdiri dari file game_of_life_lib.h dan main.c. main.c merupakan algoritma kasar untuk program utama pada awal mulanya, namun dikembangkan menjadi program utama dari implementasi Conway's Game Of Life. game_of_life_lib.h merupakan file gabungan dari fungsi-fungsi yaitu getSeedData, tick, dan Animate, serta terdapat dua fungsi tambahan yaitu delay dan destroyArray. Terdapat juga file display.h, yang isinya fungsi fungsi khusus untuk mencetak tampilan user interface pada console nya, diimplementasikan dengan fungsi displayInterface, displayMenu dan displaySeed. Terakhir, dalam branch master, terdapat README.md sebagai penjelasan singkat tentang repository, dan LICENSE sebagai pernyataan lisensi GPL 3.0.

Pada dasarnya, setiap fitur untuk pendukung game ini, dibuat branch yang berbeda-beda yang dihandle oleh orang yang berbeda-beda pula. 1 orang pasti memegang 1 branch dan bertanggung jawab atas satu fitur tertentu (walaupun saat ada masalah tentu saja kami mengerjakan bersama sama). Setiap branch terdapat commit history yang dilakukan oleh masing masing dari kami. Masing masing branch memiliki struktur isi file yang berbeda pula, tergantung orang yang memegang branch tersebut. Ada yang banyak proses perubahannya sehingga banyak commit, ada yang sedikit proses perubahannya sehingga sedikit commit. Yang dapat dipastikan, setiap branch terdapat file berekstensi .c, yang berperan sebagai testfile untuk mencoba masing-masing fitur tersebut secara terpisah. Nama file yang berekstensi .c tersebut sesuai dengan fitur yang dikerjakan.

Kami menggunakan prinsip git flow agar memudahkan dalam debugging program. Setiap orang akan mengerjakan program bagiannya masing-masing dalam branch yang berbeda terlebih dahulu. Jika sudah selesai maka program akan di *merge* dari branch ke master. Ketika fungsi pada branch tidak bekerja dengan baik, hal tersebut akan terdeteksi lebih awal sehingga tidak akan mengganggu jalannya program yang sebelumnya telah berjalan dengan benar.

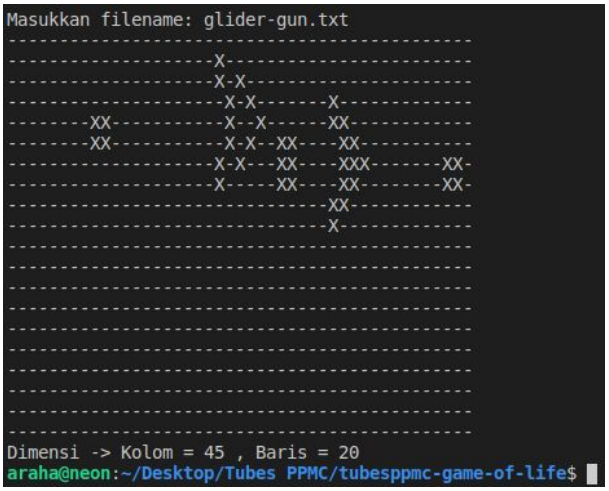
Untuk pembagian fitur dan branchnya dan siapa yang mengerjakan dapat dilihat dari README.md branch master repo kami.

Link Repository :
https://github.com/dhkar/tubesppmc-game-of-life

4. Pengujian Program

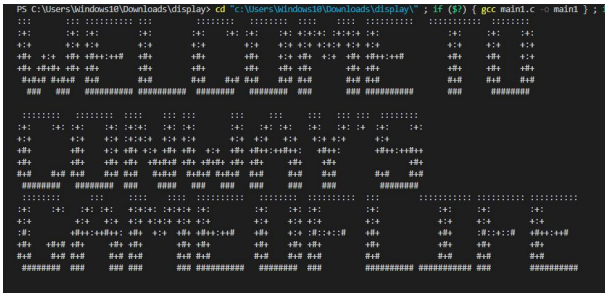
Pada bagian ini akan ditampilkan pengujian dari masing-masing fungsi yang terdapat pada program, diantaranya adalah fungsi getSeedData, displayInterface, displayMenu, Tick, Animate, serta program utama..

1. Fungsi getSeedData



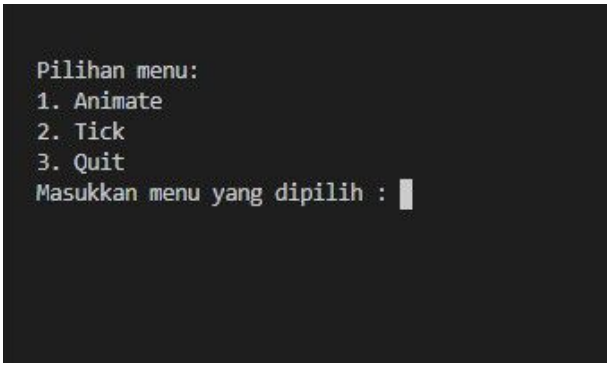
Hasil pengujian subprogram fungsi getSeedData() yang berada pada file getSeedData.c pada branch convertseedtoarray. Program berjalan dengan baik dan mengeluarkan data seed sesuai dengan file yang diassign ke program. Begitu juga dengan dimensi seednya juga sudah sesuai.

2. Fungsi displayInterface



Fungsi ini merupakan fungsi yang pertama kali dipanggil sekaligus interface saat game dimulai.

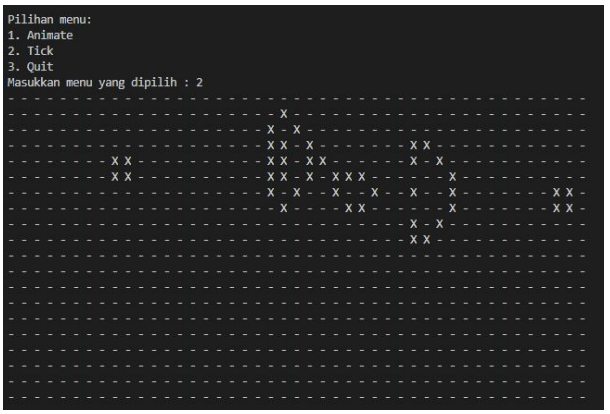
3. Fungsi displayMenu



Menu akan ditampilkan lalu user dapat memilih antara 3 kondisi tersebut. Tick untuk mencetak 1 generasi selanjutnya, animate untuk mencetak tick berulang-ulang, dan quit untuk keluar dari program.

4. Fungsi Tick

a. Kondisi Awal

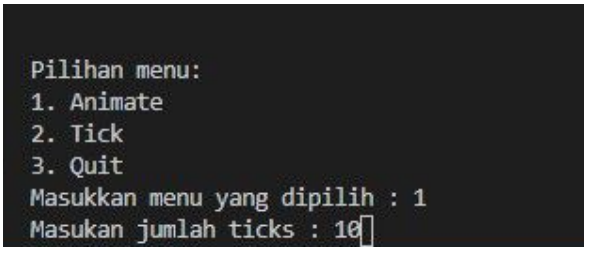


b. Kondisi Akhir

Fungsi tick menghasilkan generasi selanjutnya dari array seedData yang menjadi parameter fungsi tersebut. Jika suatu sel memiliki tepat 2 atau 3 tetangga yang hidup maka sel tersebut akan hidup pada generasi selanjutnya selain kondisi tersebut maka sel akan mati.

5. Fungsi Animate

a. Memasukan jumlah ticks

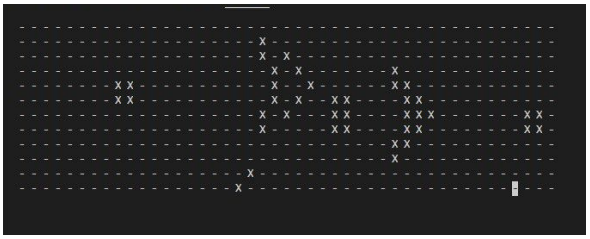


Jumlah ticks akan menentukan banyaknya iterasi animate yang akan menentukan sejauh mana generasi yang dituju dan selama apa program berjalan.

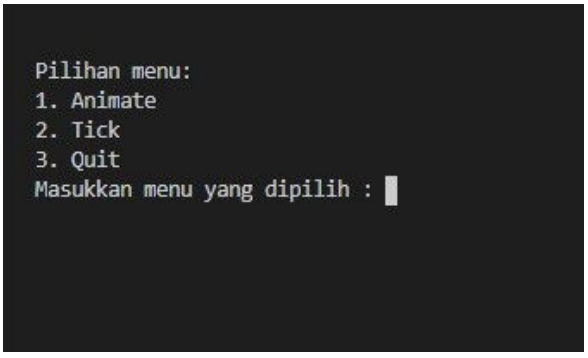
b. Saat Animate berjalan (1)

Animate akan menampilkan hasil tiap generasi yang hanya menampilkan generasi pada saat itu saja pada satu waktu.

c. Saat Animate berjalan (2)



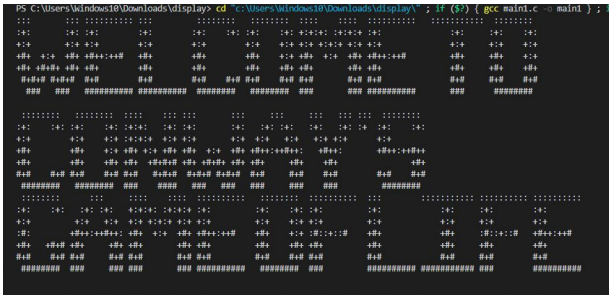
d. Akhir dari Animate



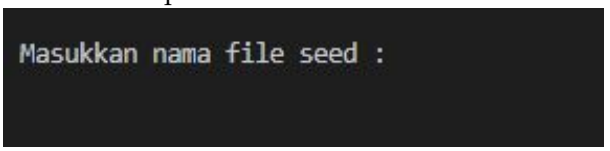
Pengguna dapat memilih tick, quit, ataupun animate kembali

6. Program Utama

a. Interface awal saat program dijalankan



b. Meminta input user



Pengguna memasukan nama file seed.

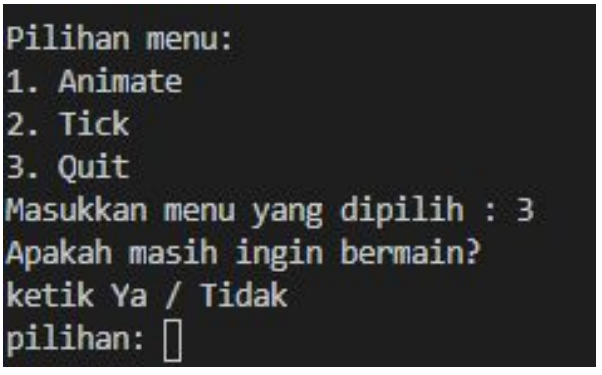
c. Kasus file tidak ditemukan

Program akan meminta input nama file kembali apabila file yang dituju tidak ditemukan di folder yang sama.

d. Tampilan saat file ditemukan

Tampilan program utama saat melakukan fungsi Tick dan Animate seperti pada bagian sebelumnya. program akan menampilkan tampilan awal seed sebelum pengguna memilih menu Tick atau Animate.

e. Keluar dari program (1)



Tampilan awal saat pengguna memilih untuk keluar dari program, program akan bertanya ke pengguna apakah ingin bermain lagi atau tidak.

f. Keluar dari program (2) - lanjut bermain

```
Pilihan menu:
1. Animate
2. Tick
3. Quit
Masukkan menu yang dipilih : 3
Apakah masih ingin bermain?
ketik Ya / Tidak
pilihan: Ya

Masukkan nama file seed :
```

Program akan meminta nama file seed yang baru apabila user ingin main kembali.

g. Keluar dari program (2) - selesai bermain

```
Pilihan menu:
1. Animate
2. Tick
3. Quit
Masukkan menu yang dipilih : 3
Apakah masih ingin bermain?
ketik Ya / Tidak
pilihan: Tidak

Terima kasih telah bermain.
PS C:\Users\Windows10\Downloads\display>
```

Program berhenti dan pengguna tidak dapat memasukan input kembali.

5. Kesulitan yang Dihadapi

Andhika :

Dari segi proses implementasi fungsi getSeedData() yang mengonversi file yang berisi seed, ukuran world seednya menjadi suatu array 2D berukuran MXN, dapat berjalan dengan hanya dengan sekali commit, namun memiliki kendala dalam pembuatannya. Yaitu saya sering dikecohkan dengan ukuran seednya yang terkait dengan indexing dari array 2D yang berisi seed. beberapa kali sering dijumpai adalah permasalah segmentation fault yang membuat keluaran console tidak signifikan.

Ternyata hal ini disebabkan karena saat saya ingin mengetes kesesuaian antara isi file dengan console, saya melakukan traversal untuk indeks array (sebutlah i dan j sebagai index), namun saya mengisinya dengan yang terbalik. Sebagai contoh : ada suatu seed berukuran 5x9, dan saya mengalokasikan memory sebanyak 5x9, nilai ini disimpan dalam x_dimension -> i = 5 dan y_dimension -> j = 9 dan urutannya adalah seed[i][j]. Namun karena melakukannya terbalik, akan ada saatnya program mencari nilai yang terdapat pada seed[9][5] yang ini tidak didefinisikan dalam malloc sehingga muncul segmentation fault. Meskipun demikian, masalah dapat diselesaikan dengan menukar traverse i dan j agar tetap berbentuk seed[i][j] bukan lagi seed[j][i].

Melia :

Fungsi tick yang dibuat pada awalnya tidak bersifat toroidal sehingga terjadi *error* di awal ketika mencetak seed generasi selanjutnya dengan pemanggilan fungsi tick. Ujung-ujung dari seed generasi selanjutnya tidak dapat mendeteksi tetangga dengan benar karena itu pada generasi selanjutnya setiap sisi paling kanan, kiri, atas, maupun bawah tidak tercetak ketika ditampilkan hasilnya di layar

Fungsi ini pada akhirnya dibuat toroida dengan batasan ketika kondisi-kondisi di sisi paling ujung dari setiap array saja. Perhitungan *neighbor* pada sisi-sisi tersebut akan dialihkan ke sisi yang berlawanan sehingga fungsi akan bersifat toroidal dan dapat mengembalikan nilai dari sisi-sisi yang sebelumnya tidak dapat mendeteksi *neighbor-neighbor* karena berada di ujung-ujung array.

Kendala juga muncul karena fungsi tick dibuat dengan memperhatikan dimensi dari array seed yang menjadi parameter fungsi. Ketika salah meletakan dimensi menjadi tertukar maka loop tidak dapat dijalankan dan fungsi tidak menghasilkan apapun. Maka dari itu, peletakan variabel dari panjang dan lebar array menjadi krusial pada fungsi ini.

Adnan :

Kendala saya adalah saat membuat fungsi tick Saya sendiri. Saya membagi menjadi beberapa sub fungsi dan mencoba tick Saya sendiri. Namun, masalahnya terdapat di sub-fungsi yang berguna untuk menghitung jumlah *neighbors*. Saat masuk ke sub-fungsi tersebut, Saya dapat mencetak lokasi seed ke i,j namun saat berusaha di masukan ke *conditional statement* fungsi selalu crash, padahal seed ke i,j sendiri bisa Saya cetak ke layar. Sehingga pada akhirnya Saya menggunakan fungsi tick yang dibuat oleh programmer lain. Masalah lainnya yang Saya temukan namun sudah teratasi adalah program mencetak terlalu lama, karena fungsi delay menerima input dalam satuan detik yang kemudian saya ubah menjadi mili detik, serta seed yang tidak berubah karena Saya belum *assign* ke seed baru hasil tick.

Sabrina :

Awalnya salah membuat fungsi displaySeed karena mengira displaySeed adalah fungsi untuk mengeprint seed dari file tetapi hanya mengeprint file. Lalu kendala lainnya ada pada pembuatan main menu. Pada main menu, kami mau membuat agar saat kondisi quit program tidak langsung berhenti melainkan terdapat pilihan apakah ingin bermain lagi atau tidak. Permasalahan pertama adalah ketika user telah memasukkan input menu 1 untuk animate(), atau 2 untuk tick(), program langsung keluar sehingga ketika dianalisis ternyata kesalahannya ada pada looping input file dan main menu yang terpisah. Kemudian setelah disatukan menjadi satu loop besar, setelah animate atau tick, program langsung meminta file baru. kemudian ditambahkan looping untuk menu didalam looping input file seed, dan akhirnya program untuk selain quit bisa dijalankan. Setelah itu pada kondisi quit terdapat kendala bagaimana untuk fclose dan keluar loop.

permasalahan kedua adalah ketika menggabungkan fungsi pada library game_of_life_lib.h dengan main1.c. setelah digabungkan dan mencoba dirun, setelah user memilih animate atau tick, program tidak mengeluarkan hasil animate dan tick, melainkan langsung keluar program. sehingga beberapa parameter fungsi diubah dan parameter untuk implementasi juga diubah.

6. KONTRIBUSI ANGGOTA

Fungsi	Berkas/Library Terkait	Programmer	Tester
getSeedData()	game_of_life_lib.h getSeedData.c	Andhika	Adnan
displaySeed()	display.h display.c	Sabrina	Melia
displayMenu()	display.h display.c	Sabrina	Sabrina
displayInterface()	display.h display.c	Sabrina	Adnan
Animate()	game_of_life_lib.h animate.c	Adnan	Melia
tick()	game_of_life_lib.h tick.c	Melia	Andhika
destroyArray()	game_of_life_lib.h getSeedData.c	Andhika	Sabrina
delay()	game_of_life_lib.h animate.c	Adnan	Andhika
Integrasi ke Program Utama	main.c	Andhika, Adnan, Melia, Sabrina	Andhika, Adnan, Melia, Sabrina

Jika dibandingkan dengan laporan 1, terdapat fungsi tambahan yang ternyata dibutuhkan saat proses implementasi game. yaitu fungsi destroyArray() dan delay().

7. Kesimpulan dan Saran

Kesimpulan dari Implementasi Game of Life menggunakan bahasa C adalah:

1. Bahasa C memungkinkan pengguna untuk mengakses memori yang digunakan oleh program dengan mudah, sehingga pengguna mampu mengalokasikan memori yang digunakan untuk program sesuai kebutuhan memaksimalkan penggunaan memori dalam suatu program. Hal tersebut sangat berguna pada kasus dimana pengguna membutuhkan memori untuk menyimpan array yang belum diketahui ukurannya pada saat kompilasi
2. Bentuk dari array toroidal berguna pada beberapa kasus yang membutuhkan bentuk array tersebut pada pengaplikasian program, seperti Game of Life yang telah diimplementasikan pada laporan ini
3. Penggunaan dari pointer dan array pada bahasa C sangatlah dibutuhkan karena dengan memahami penggunaan pointer di bahasa C, pengguna dapat memahami cara kerja array sehingga mampu mengoperasikan atau mengolah suatu array dengan mudah

Implementasi game Conway's Game Of Life yang kami buat dalam bahasa C (dengan konsep konsep yang dijelaskan di atas), dapat dikatakan berhasil sesuai dengan spesifikasi yang ada. Hal ini dapat ditunjukkan bahwa program dapat:

1. Mengonversi file berisi seed ke seed array
2. Seed array tersebut diproses sesuai rules dari Conway's Game Of Life (sehingga bisa melakukan iterasi sekali dan n kali/animate)
3. Dapat mencetaknya dalam stdout, yang berperan sebagai console game. Bahkan ada fitur tambahan seperti user interfacenya

Daftar Pustaka

[1] https://en.wikipedia.org/wiki/Conway%27s_Game_of_Life 5 April 2020, 19:43

[2] https://stackoverflow.com/questions/5201708/how-to-return-a-2d-array-to-a-function-in-c, 7 April 2020, 21:24

[3] https://codereview.stackexchange.com/questions/208973/conways-game-of-life-in-c-3-mode, 9 April 2020, 18.02.

[4] https://www.geeksforgeeks.org/time-delay-c/, 10 April 2020, 13.56