Challenges on Implementing Game of Life

Hsiang Yun Lu

February 13, 2023

1 Opaque struct

The first challenge of this assignment was definitely to understand and implement the concept of an opaque struct. Though I thought I got the idea of indirectly interacting with the *Universe* struct, I still got errors when compiling. One reason was that I didn't figure out uv_create() function returned a pointer to a *Universe*, since in *life.c* we could only access an *Universe* through a pointer. The other reason was that I made some helper functions in *life.c* but tried to directly access the *Universe* struct and functions in *universe.c*. I fixed these errors by creating pointers to *Universe* and wrote my helper functions in *universe.c*. Also, I declare my added functions at the beginning of *life.c* because they were not included in the header file *universe.h*.

I definitely gained more understanding of the concept of an opaque struct, and am pretty confident that I can apply it to other projects.

2 File Input/ File Output

This assignment provides an opportunity for me to get familiar with the usage of input/output file stream, especially on *fscanf()* and *fprintf()*. After reading the *C* textbook, I was still confused about how *fscanf()* read in files. Should it be line by line or character by character? After researching, I can not distinguish the usage of *fscanf()*, *scanf()* and *sscanf()*. I also got more comfortable with handling STDIN and STDOUT and manipulating them and files.

3 Where A Variable Should Be Declared

I was stuck on errors like "unused variable 'uni_A'" and "use of undeclared identifier 'uni_A'" for a while. Then I figured out that it was a very basic but easily ignored (or only by me) mistake. I declared the variables in the block of IF statement so that the assignment won't stay after the IF block ends. I believe I will always remember this after getting stuck for so long.

4 Dynamic Memory Allocation for A Struct and Matrix

The implementation of function *uv_create()* gave the chance to figure out using *calloc()* to dynamically allocate memory for the *Universe* stuct and the grid in an *Universe*. At first I just blindly followed the instruction to build matrix for grid in *uv_create()* without even noticing that I need to allocate memory for a struct and return a pointer. I didn't realized something's wrong until I got the error saying, "variable has incomplete type 'Universe'". Because my *Universe* instance in *life.c* was not a pointer to an *Universe*. I was glad that I finally fixed it after some hints from the TAs and doing some research.

5 Use of *ncurses* Library

The *ncurses* library was a totally new thing to me. Although for this assignment we could just follow the example code provided in the assignment specifics, I still got the chance to play around with some features that I can do on the screen, like changing the time for sleep, *printw()*, *mvprintw()*, *attron()*, and playing with colors and patterns.

6 Modular Arithmetic

I was stuck for modular arithmetic for a while after I got "Segmentaion Fault" when running *life* in toroidal mode. The mistake resulted from me mixing the concept of mathematical modulo (%) and modulo in *C*. For negative numbers, the calculation of these two concepts are so different. $a\%b$ in *C* is a remainder operation and not a modulo operation. According to the C99 Specification: $a == (a/b) * b + a\%b$, which make the operation of negative numbers different from simply $a\%b$.