Assignment 4 Writeup

Caitlin Smith

February 13, 2023

1 Structures

```
1 struct Universe {
2     uint32_t rows;
3     uint32_t cols;
4     bool **grid;
5     bool toroidal;
6 };
```

I learned a lot about structures in C in this assignment. We created the Universe structure, shown above, that contained different variables necessary for creating and manipulating a Universe grid. I learned what opaque meant in terms of where the struct was actually defined. Only having the declaration in universe.h didn't allow life.c to have direct access to the contents of the struct. Because of this, we had to create functions like uv_rows and uv_get_cell to get values from and change values in a Universe. When accessing the contents of a Universe in universe.c, which has the definition of Universe, I used u->variable.

2 Memory Allocation

This program called for multiple memory allocations. Allocating for a structure was simple when using sizeof(). By calling sizeof(Universe) I was able to easily allocate the exact memory needed for the struct and its contents. We use a double pointer for the grid variable so that was allocated additionally using a for loop. That grid memory had to be freed with loops as well. I had issues with seg faults at first, but I had forgotten to free both of the created Universes.

3 Double Pointers

In order for me to understand this concept more easily, I just thought of bool **grid as a matrix. In order to properly allocate memory for the double boolean pointer, I had to use a for loop. I first allocated memory for the number of rows, and then I allocated for the number of columns. This creates something similar to an array of arrays. Because of this, accessing each element in

the double pointer also needs to use for loops. For this program, I had a loop for the rows and then a loop for the columns inside that. To access an element I would use grid[row][column].

4 Ncurses

Short ncurses example.

```c
#include <ncurses.h>
#include <unistd.h> // For usleep().

#define ROW 0
#define DELAY 50000

int main(void) {
    initscr();                  // Initialize the screen.
    curs_set(FALSE);            // Hide the cursor.
    for (int col = 0; col < 40; col += 1) {
        clear();                // Clear the window.
        mvprintw(ROW, col, "o"); // Displays "o".
        refresh();              // Refresh the window.
        usleep(DELAY);          // Sleep for 50000 microseconds.
    }
    endwin();                   // Close the screen.
    return 0;
}
```

Before this assignment, I had never used ncurses. I referenced the above example code in order to write my own. I followed the structure/order of the example to then experiment and get mine working correctly.

5 Reading and Writing Files

I also learned about reading and writing files. I used FILE pointers originally set to stdin and stdout. In the command line option for in and outfiles I set the FILE pointers to the passed argument. It was in those that I also used fopen() with “r” and “w”. I was able to read the input using fscanf() and write to the output with fprintf(). I had trouble at first reading the input line by line but figured out how to correctly take in the two uint32_t’s and account for the newline.