Assignment 4 DESIGN.pdf

Vincent Liu

2/12/23

Purpose:

The purpose of this assignment is to replicate a simulation called “Conway’s game of life”. This assignment requires students to become familiar with memory allocation, pointers, dereferencing, as well as other fundamental concepts of C covered in previous assignments. Students will also be using ncurses to draw the simulation/game on a screen.

Files in Directory:

1. universe.h
    - specifies the interface to the Universe ADT. This file is provided and may not be modified.
2. universe.c
    - implements the Universe ADT.
3. life.c
    - contains main() and may contain any other functions necessary to complete your implementation of the Game of Life.
4. Makefile
    - Used to clean directory, generate associated executable file, and properly format .c files
5. README.md
    - Text file in Markdown format that describes how to build and run the program.
6. DESIGN.pdf
    - Describes the design for the program thoroughly with pseudocode
7. WRITEUP.pdf
    - Describes what the program does, and gives insight on the results found

---

Structure of Program

Main file: life.c

- ‘Update’ function to swap values+update the values between the two universes
- Main function
    - Getopt while loop

        - -t : Specify that the Game of Life is to be played on a toroidal universe.
        - -s : Silence ncurses. Enabling this option means that nothing should be displayed by ncurses.
        - -n generations : Specify the number of generations that the universe goes through. The default number of generations is 100.
        - -i input : Specify the input file to read in order to populate the universe. By default the input should be stdin
        - -o output : Specify the output file to print the final state of the universe to. By default the output should be stdout.
    - Check if using stdin/stdout
    - Create the two universes
    - Populate the universes
    - Check whether ncurses is silenced or not
    - For loops to print out onto either the ncurses screen or terminal
    - Close files if input/output files were specified
    - Free memory

Function files:

- **universe.c**

    - /*BUILDS STRUCTURE FOR UNIVERSE (psudocode provided in assignment 4 pdf*/
        - struct Universe {
            - uint32_t rows ;
            - uint32_t cols ;
            - bool ** grid ;
            - bool toroidal ;
        - };

    - Universe *uv_create(uint32_t rows, uint32_t cols, bool toroidal)
        - Allocates space for creation of universe
        - Allocate space for grid(using # of rows) and rows using num of columns
        - Add to struct: rows, cols and toroidal bool

    - void uv_delete(Universe *u)
        - Deletes the universe by freeing memory to prevent memory leaks
            - (frees pointers)

- uint32_t uv_rows(Universe *u)
  - Returns total num of rows using accessor function

- uint32_t uv_cols(Universe *u)
  - Returns columns using accessor function

- void uv_live_cell(Universe *u, uint32_t r, uint32_t c)
  - Sets a cell to be live using row & column coordinates

- void uv_dead_cell(Universe *u, uint32_t r, uint32_t c)
  - Sets a cell to be dead using row & column coordinates

- bool uv_get_cell(Universe *u, uint32_t r, uint32_t c)
  - Using coordinate arguments, find if the cell at said location is alive or dead. Returns a boolean depending on the circumstance.

- bool uv_populate(Universe *u, FILE *infile)
  - while loop (exit statement end of file)
    - Scanf text
      - If coordinates are out of bounds, return false
      - Plot coordinates of live cells
    - Return true

- uint32_t uv_census(Universe *u, uint32_t r, uint32_t c)
  - Check if universe is toroidal
    - If true
      - For row
        - For col
          - Use offsets to calculate location of “out of bound” cells
          - Return number of live neighbors
    - If false
      - For row
        - For cols
          - Use offsets to check surrounding tiles for live neighbors.
          - Return number of live neighbors

- void uv_print(Universe *u, FILE *outfile)
  - for loop (row)
    - For loop (cols)
      - If a coordinate has a live cell
        - Print ‘o’
      - Else
        - Print ‘.’

Makefile

Set compile for C language
Setflags -Wall -Wpedantic -Werror -Wextra -standard c17
Set universe and life ‘.o’ files
Generate ‘life’ executable
Compile .o files from .c files
Make ‘life’ to be main file with use of previously created object
Compile life.o from life.c

Clean:
Removes life, life.o as well as universe.o file

Format:
Formats all c files to c17