ECE 175: Computer Programming for Engineering Applications

Homework Assignment 7

Conventions: Name your C programs as *hwx_py.c* where *x* corresponds to the homework number and *y* corresponds to the problem number. As an example the C program for hw1 problem 1 should be named as *hw1_p1.c*.

Submission Instructions: Submit Hw on D2L.

1 Interactive Tic Tac Toe (25 points)

Implement an interactive tic tac toe game between two players. The game starts by printing an empty board. Each player takes turns and adds an 'X' or an 'O'. The game ends when either of the user wins, or the board is full and no winner has been declared. Your code should employ *at least* the following two functions.

int check_board(int x[ ][3]); // returns 1 if player 1 wins, 2 if player 2 wins, or 0 if tie. You can use any additional arguments you deem necessary.

void print_board(int x[ ][3]); // prints the board on screen. // returns 1 if player 1 wins, 2 if player 2 wins, or 0 if tie. You can use any additional arguments you deem necessary

Optional: You can automate Player 2 to be played by the computer!

Sample execution

```
Let's play Tic Tac Toe:

   |    |
-- | -- | --
   |    |
-- | -- | --
   |    |

Player 1: 1 1


Player 1 entered an 'X' at (1, 1)

   |    |
-- | -- | --
   | X  |
-- | -- | --
   |    |

Player 2: 2 2
```

```
    |    |
 -- | -- | --
    | X  |
 -- | -- | --
    |    | O
```

Player 2 entered an 'O' at (2, 2)

...

```
 O  |    |
 -- | -- | --
 X  | X  | X
 -- | -- | --
    |    | O
```

Player 1 wins the game

2 The Game of Life (45 points)

Submit your pseudo-code for this problem

Imagine a world of organisms living in a two-dimensional cell grid of size $n \times m$. Each organism can only occupy a single cell. Each cell, except those at the boundaries (the edge of the world), has exactly eight neighboring cells (up, down, left, right, and four diagonals). The cells at the boundaries have less than eight neighboring cells. The world evolves from one generation to the other using the following rules:

1. Any organism with fewer than two neighbors (a neighbor is an organism that lives in a neighboring cell) dies (out of loneliness).
2. Any organism with more than three live neighbors dies (overcrowding).
3. Any organism with two or three live neighbors lives on to the next generation
4. Any vacant cell with exactly three live neighbors becomes occupied by a new organism (birth).

Write a C program that plays the game of life. Your program should:

1. Read an initial world configuration from a file world.txt. The world is of size 10×10.
2. Evolve the world to the next generation.
3. Display the old and new world generation on screen.
4. Ask the user if he/she wants to continue evolution to the next generation.
5. Display a message if the entire world is extinct.

You should ignore the cells at the boundary of the world since the do not have enough neighbors to play the game. For a modular design, you should use the following functions:

```
void read_world(FILE *inp, int x[ ][SIZE]); // reads a world from a file to array x
void print_world(int x[ ][SIZE]); // prints world stored in array x on screen
int evolve(int x[ ][SIZE], int row, int col); // returns the evolved cell value for cell x[row][col]
void copy_world(int x[ ][SIZE], int y[ ][SIZE]); // copies world in x to y
int extinct(int x[][SIZE]); // returns a zero if all organisms are extinct
```

Sample initial worlds are provided.

Hints:

- Play the game by hand on a small array to understand it.
- You need 2 two-dimensional arrays: one that stores the current generation and one that stores the future one.
- Start your cell procession from cell (1,1) up to cell (1,m-2) and cell (m-2, 1) to cell (m-2, m-2).
- For better understanding of your code use #define ALIVE 1 and #define DEAD 0, to denote the organism state.
- Print "*" for an alive mechanism and a "0" for a dead one.

Sample evolution.

Old World:

```
0   0   0   0   0   0   0   0   0   0
0   0   0   0   0   0   *   0   0   0
0   *   *   *   0   0   0   0   0   0
0   0   0   0   0   0   0   0   0   0
0   0   0   0   *   *   *   *   *   0
0   0   0   0   0   0   0   0   0   0
0   0   *   *   0   0   0   0   0   0
0   0   *   *   0   0   0   0   0   0
0   0   *   *   0   0   0   0   0   0
0   0   0   0   0   0   0   0   0   0
```

New World:

```
0   0   0   0   0   0   0   0   0   0
0   0   *   0   0   0   0   0   0   0
0   0   *   0   0   0   0   0   0   0
0   0   *   *   *   *   *   *   0   0
0   0   0   0   0   *   *   *   0   0
0   0   0   *   *   *   *   *   0   0
0   0   *   *   0   0   0   0   0   0
0   *   0   0   *   0   0   0   0   0
0   0   *   *   0   0   0   0   0   0
0   0   0   0   0   0   0   0   0   0
```

Submit your .c files named hw7_p1.c hw7_p2.c and pseudo-code via D2L dropbox