CS 232 Introduction to C and Unix

HW3 (Due on February 3, 2022, 11:59pm)

The goal of this homework assignment is to test your ability to implement functions in C. After this homework assignment, you should feel comfortable writing programs in C that use functions. You should also be familiar with reading someone else's code and working with a larger program. This homework will also introduce you to Makefile. Please read the entire homework description carefully before beginning the coding. Please start this homework assignment early.

Collaboration policy:

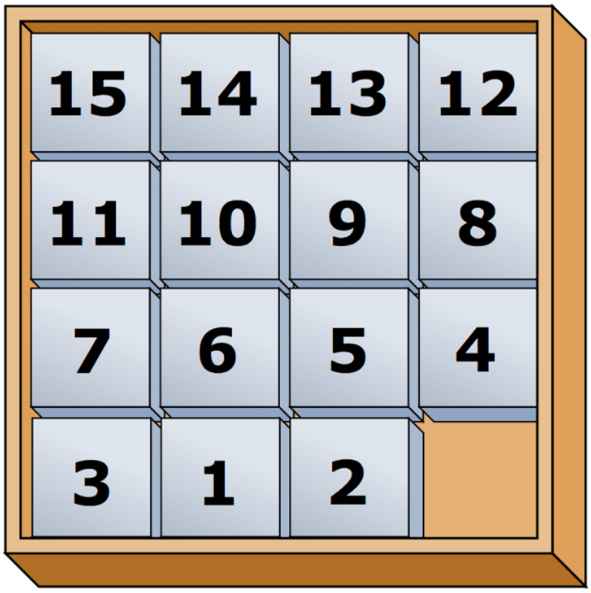
Similar to homework 2, you are to work alone. If you get stuck on part of the homework, you are encouraged to come by office hours. If you cannot make office hours, just send me an email, and we can find a time to meet. I'm here to help you, but I do not know how to help if you don't ask me.

Homework description (Game of Fifteen):

The Game of Fifteen is a puzzle played on a square, two-dimensional board with numbered tiles that slide. The goal of this puzzle is to arrange the board's tiles from smallest to largest, left to right, top to bottom, with an empty space in board's bottom-right corner, as in the below.

Sliding any tile that borders the board's empty space in that space constitutes a "move." Although the configuration above depicts a game already won, notice how the tile numbered 12 or the tile numbered 15 could be slid into the empty space. Tiles may not be moved diagonally, though, or forcibly removed from the board.

Although other configurations are possible, we shall assume that this game begins with the board's tiles in reverse order, from largest to smallest, left to right, top to bottom, with an empty space in the board's bottom-right corner. **If, however, and only if the board contains an odd number of tiles (i.e., the height and width of the board are even), the positions of tiles numbered 1 and 2 must be swapped, as in the below.** The puzzle is solvable from this configuration.



In VS Code homework workspace, under homework directory, create a new subdirectory "hw3". In the Brightspace, you can find a file "hw3.zip". Put this file to your VS Code homework workspace in hw3 directory. Unzip the file by running the command:

$ unzip hw3.zip

You can find five files:

- fifteen.c (with part of a program already written)
- fifteen_expected (the executable file created in Linux showing the expected behavior of the program)
- Makefile (used to compile the C file)
- 3x3.txt and 4x4.txt (used for testing the program)

You should only modify the file “fifteen.c” in the places that are marked “**TODO**”. That is, please implement four functions: init, draw, move, and won.

Read over the code and comments in fifteen.c and then attempt to answer the questions below.

1. Besides 4 × 4 (which are Game of Fifteen’s dimensions), what other dimensions does the framework allow?
2. With what sort of data structure is the game’s board represented?
3. What function is called to greet the player at game’s start?
4. What functions do you apparently need to implement?

No worries if you’re not quite sure how fprintf or fflush work; we’re simply using those to automate some testing.

When you implement the code, remember to take "baby steps." Don’t try to bite off the entire game at once. Instead, implement one function at a time and be sure that it works before forging ahead. In particular, we suggest that you implement the framework’s functions in this order: init, draw, move, won. Any design decisions not explicitly prescribed herein (e.g., how much space you should leave between numbers when printing the board) are intentionally left to you. Presumably the board, when printed, should look something like the below, but we leave it to you to implement your own vision.

$ ./fifteen_expected 4

```
15 14 13 12
11 10  9  8
 7  6  5  4
 3  1  2  _
```

Incidentally, recall that the positions of tiles numbered 1 and 2 should only start off swapped (as they are in the 4 × 4 example above) if the board has an odd number of tiles (as does the 4 × 4 example above). If the board has an even number of tiles, those positions should not start off swapped. And so they do not in the 3 × 3 example below:

$ ./fifteen_expected 3

```
 8  7  6
 5  4  3
 2  1  _
```

Compiling and Testing:

To compile the code, please use the Makefile. Specifically, for the Makefile, run the command:

$ mingw32-make.exe

in Windows or

$ make

in Linux, under "hw3" directory to compile your code "fifteen.c". Indeed, "make" command looks into the Makefile to run:

gcc -g -std=c11 -Wall -o fifteen fifteen.c -lm

The Makefile is a very convenient way to compile multiple files, which we will discuss later.

To test your implementation of fifteen, you can certainly try playing it. (Know that you can force your program to quit by hitting ctrl-c.) Be sure that you (and we) cannot crash your program, as by providing bogus tile numbers. Moreover, you can automate execution of this game. In fact, files 3x3.txt and 4x4.txt provide the winning sequences of moves for a 3 × 3 board and a 4 × 4 board, respectively. To test your program with those inputs, execute the below.

$ ./fifteen 3 < 3x3.txt
$ ./fifteen 4 < 4x4.txt

If you'd like to play with the instructor's own implementation of fifteen, you may execute the "fifteen_expected" program in Linux server "diamond.pfw.edu".

Submission

After testing the program, push the code fifteen.c to your GitHub "Homework_CS232" repository under the directory "hw3".

If you have any questions, please let the instructor know.

Grading rubric:

- Program fifteen.c can be compiled – 10pt
- Function init is correctly implemented – 20pt
- Function draw is correctly implemented – 20pt
- Function move is correctly implemented – 30pt
- Function won is correctly implemented – 20pt

Total: 100 points.

Note: If the code cannot be compiled, you cannot get any point for this homework assignment.

Acknowledgement:

This homework assignment comes from CS50 by David Malan.