Assignment 2

Programming 2048 Game in C - 8 points

Pedram Pasandide

Due Date: 2025, 2 March

Introduction

The 2048 game is a popular single-player puzzle game where the objective is to combine tiles with the same number to create higher value tiles, ultimately aiming to reach the 2048 tile. The game is played on a 4x4 grid, and each turn, a random tile (either 2 or 4) is added to an empty spot on the grid. The player can move the tiles in one of four directions (up, down, left, or right), and if two tiles of the same number collide, they merge to form a new tile with their sum. The game ends when the grid is full and no more valid moves are possible. You can play the game online on here.

Through this assignment, you will gain hands-on experience in implementing fundamental concepts of C programming, such as handling multi-dimensional arrays, random number generation, input/output handling, etc. You will also develop problem-solving skills related to designing efficient game logic, including handling tile merging, user input, and detecting win/lose conditions. By structuring the game, students will learn how to design modular code, manage state, and simulate a real-world system using programming.

The game logic for 2048 requires receiving user input to control the movement of the tiles. To handle this input, we have already provided you with a function called `getch_unix()`, which reads a single character from the keyboard without waiting for the Enter key to be pressed. This function allows for immediate feedback based on the player's key press.

Before moving forward with implementing the game logic, you can test and familiarize yourself with how the `getch_unix()` function works. **You don't need to know how exactly it works. You just need to know how to use it.** To do so, you can work with the provided `testKeyInput.c` file, which demonstrates how to capture key presses such as `'a'`, `'d'`, `'w'`, `'s'`, and `'q'`. These keys correspond to the movement directions in the game (left, right, up, down) and the 'q' key to quit the game. Once you understand how to capture input from the user, you'll be able to proceed with implementing the actual game logic.

Programming the Game Logic (6.5 points)

You are provided with the following files to implement the 2048 game:

- `game2048.c`: This file contains the core logic of the 2048 game, where the game loop will be implemented.
- `utility.h`: This header file contains function declarations and necessary macro definitions that will be used across different source files. It also defines any constants that are required for the game logic (such as the board size). You will reference this file when implementing game-related logic in `game2048.c`.
- `utility.c`: This source file implements utility functions declared in `utility.h` as well as local functions only used in `utility.c`. It includes helper functions for board operations like placing random tiles, shifting rows/columns, and merging tiles. The goal here is to implement these utility functions based on the provided comments and ensure that they integrate correctly with the core game logic in `game2048.c`.

Follow the comments in each of these files to implement the game logic. You'll need to handle user input, move tiles in the correct direction, merge tiles when needed, update the game board, and check for game-over conditions. Be sure to implement the game mechanics as outlined, ensuring proper interaction between these files to create a fully functional game.

I should be able to play your game using `./game2048` command. For example, my game works like:

```
2048 Game
.       .       .       2
.       .       .       .
.       .       .       .
.       4       .       .
```

After pressing `'a'` key:

```
2048 Game
2       .       .       .
.       .       .       .
.       2       .       .
4       .       .       .
```

After pressing `'w'` key:

```
2048 Game
2       2       .       .
4       .       .       .
2       .       .       .
.       .       .       .
```

After pressing `'d'` key:

```
2048 Game
.       .       .       4
.       2       .       4
.       .       .       2
.       .       .       .
```

After pressing `'s'` key:

```
2048 Game
.       .       .       .
.       .       .       .
.       .       .       8
.       2       2       2
```

and after playing many rounds at one point:

```
2048 Game
2       8       2       4
16      128     64      16
4       16      32      8
2       8       4       2
Game Over!
```

Report and Makefile (1.5 points)

Create a PDF file named `Report.pdf` that includes the documentation for your program. The report should cover essential information, such as how to compile and run the program, how to play the 2048 game, and any relevant details about your implementation. **Create a section called Appendix in your report and include all the source code files as text (not screenshots)**.

Additionally, write a `Makefile` to compile your program. The `Makefile` should define the rules for compiling the various source files (`game2048.c`, `utility.c`) and linking them into an executable. Make sure to include the **necessary flags** for the compiler and ensure that your `Makefile` works correctly to produce a compiled binary when the make command is run. Consider adding common options like `-Wall` for warnings and `-o` to specify the output file name.

Graphical User Interface (+1 Bonus)

For this section, you are encouraged to go beyond the lecture notes and explore additional resources to develop a graphical user interface (GUI) for the 2048 game. **This is a bonus task**, meaning it is not required, but it can enhance your project and demonstrate your ability to integrate graphical elements into a C program.

A GUI (Graphical User Interface) provides a visual way for users to interact with software applications through graphical elements like windows, buttons, menus, and icons, rather than using text-based commands. GUIs make applications more user-friendly, intuitive, and visually appealing by providing an interactive environment for users to navigate.

One of the most commonly used libraries for GUI programming in C is GTK (GIMP Toolkit). GTK is an open-source toolkit designed for creating graphical applications. It offers a variety of widgets and tools to build interactive interfaces and is widely used in Linux desktop applications. Although GTK is written in C, it provides bindings for many other programming languages, making it versatile.

If you choose to implement a GUI for your 2048 game, your design should be visually pleasant, competing with online versions of the game that you may find. While the exact design is up to you, consider adding features such as:

- Buttons like Play, Restart, Exit, etc.
- A scoring system to keep track of points during gameplay.
- A graphical board representation instead of a text-based version.

For an additional challenge, you can implement saving the highest record to a file on disk, but this is optional and not required for this assignment.

Submission On Avenue to Learn

Submit your assignment as:

1. `game2048.c` (or `game2048_GUI.c` if you did GUI version),
2. `utility.h` (or `utility_GUI.h` if you did GUI version),
3. `utility.c` (or `utility_GUI.c` if you did GUI version),
4. `Makefile`,
5. a report PDF file **named** `Report.pdf`

Please do **NOT** submit any zip files.