Stella Dong

Eva Farroha

MS2 Report

Vision

Our current plan is to implement the existing game 2048 with level challenges. We would like to implement a GUI so our project is visually appealing, unlike the current 2048 GUI. This plan is similar to our original plan of 2048, however, implementing levels of difficulty increases the complexity of our code and makes it more fun for the user. The implementation of levels also allows our program to differ from the original 2048 game because if we just recreated the game, our code would not be intuitive.

Summary of Progress

On the backend side, we have implemented the move left and move right functions. Our gameboard is a 4x4 grid, and its values are represented as lists of integer lists so we can use the plethora of list OCaml functions. The 0's on our gameboard represent placeholders and are ignored in shifts. The first challenge was implementing code so that a single row can be moved left or right. Once we got the values to shift from left to right, we implemented a function to ensure there are always 4 values in each list since we essentially delete and repopulate our board with 0s as needed after each shift. Our last challenge was to introduce a case where 2 values are added together. Now, by using the controls implemented in the UI, the individual rows move either left or right, adding adjacent numbers if their values match. For example, the row [2; 0; 2; 2] shifted left would produce [4; 2; 0; 0]. The same row shifted right would produce [0; 0; 2; 4].

We also worked on the UI, where the user interacts with the game. Using ASCII characters, there's currently a 4x4 table displayed in the powershell. Each cell has the capacity for 6 characters, so after 6 digit numbers, we plan to make the game end, or maybe have the board adjust for the large numbers. In the powershell, a user can type in w, a, s, d to interact with the board. Since we only have the left and right functions, the user is only able to use a and d to interact with the game without an error. There's also the option to type q to end the game.

Activity Breakdown

Eva Farroha

- Responsibilities: Build functioning movement controls for a gameboard moving left and right
- Delivered Features: when running the game in the terminal, as the user presses a, the gameboard shifts to the left, repositioning empty spaces and numbers. As the user presses d, the gameboard shifts to the right. If adjacent numbers are identical prior to the shift, they will be added together.
- Hours: 10

Stella Dong

- Responsibilities: Build a system that could take in user input to interact with the board
- Delivered Features: when running the game in the terminal, a user is prompted to use w, a, s, d, or q to either move the board or quit. If something is not implemented, the game will fail with "not implemented" and if another key is pressed, the game fails with

"incorrect key". The current board is made up of ASCII characters and prints a new, updated board after a user enters their key.

- Hours: 10

Productivity Analysis

Before we started working, we hoped to have a version of the game where one number would appear and that number could move around the board. Initially, we definitely underestimated the difficulty of starting the project and understanding the logic of the game, so it took a long time to get anything that could be tested. Setting up the files also took a very long time, but it definitely got easier as we got deeper into the project. Given the code line requirement and scope of the project, we definitely think we could achieve more than we initially planned to include, like possibly working on a GUI using OCaml Graphics instead of only having the ASCII interface. Doing the user interaction is more challenging than we initially expected because it was not reviewed in class so it was hard to approach, and it was difficult to figure out how to connect the user input to the game logic. The game logic was also challenging to understand initially.

Deliverable 2: Gallery Entry

```
---

# Members of your group.
group:
  - name: Stella Dong
    netid: ssd74
  - name: Eva Farroha
    netid: edf55

# Your PM.
pm:
  name: Jessica Wu
  netid: jw877
# Set to false if you don't want your gallery entry to be public.
publish: true

# Pithy title
title: "2048 Game Simulator"

# OK if this is a Cornell Github link, but public gallery viewers
won't be able to see it.
git-repo: "https://github.coecis.cornell.edu/ssd74/3110-final"
# If you have no demo screencast, replace the url string with an
empty string ""
demo-video-url: "https://www.youtube.com/watch?v=LeI-4Zp4Kjo"

# Write a short, attention-grabbing description of your project.
desc: >
  Our project is 2048 but with a twist. 2048 gets boring when you
play it over time, so we decided to change it up. You can play the
game as normal, but we also added levels for increased difficulty.
The user can use their keyboard to induce movement of the gameboard.
```