Project Proposal
Eva Zhong, Kaixing Wu, Megan Zhao

- **The title of the project.**

  2048

- **Description of the goal of the project and summary of the objectives and basic rules of the game.**

  Our project will implement the game *2048* and add one to two variations depending on the progress of the project. Users can choose among three levels. The first level is the original game. The second and the third levels are the variations. *2048* is a single-player sliding block puzzle game application. On a 4×4 grid, users can choose a direction and all numbered tiles will slide as far as possible in the direction until they are stopped by other tiles. Two colliding tiles that have the same number will merge to a new tile with the total value of the two tiles. The new tile cannot merge with another tile in the same move. Each time users choose a direction to slide, a new tile of 2 or 4 appears randomly in an empty tile spot. The objective of the game is to reach a tile with a value of 2048. Users can keep playing after they reach the 2048 tile. The game ends when there is no empty space in the grid and therefore users have no legal moves.

  The first level follows the same rules of the original application. The second level adds the function of gravity. Whenever users reach a certain number, for example, 64, the grid rotates 90 degrees to the left and all tiles fall down because of gravity. Like the original rule, the tiles slide as far as possible in the down direction until they are stopped by other tiles. Two colliding tiles that have the same number will merge to a new tile with the total value of the two tiles. We will determine the tile numbers by experimenting the game further so that the gravity function does not appear too often while adding certain difficulty to the game.

  The other variation we plan to add to the game is the subtraction feature. When the gravity function happens, two colliding tiles that have the same number subtract from each other and therefore both tiles disappear. For instance, if two 32 tiles collide with each other, they both disappear from the window. The subtraction would not happen if the gravity function is not called.

- **A list of the parts of the game we will implement. This list indicates the priority of each piece**

The first parts we need are the three fundamental components of this game: View, Model and Controller. The view is the game board GUI, the model is the game core algorithm and the controller is the functionality of each button.

Then we will combine them together to make a fully functional game and we will run many tests to examine this prototype.

Once we have a basic game (first level), we will add a second level to this game by adding the function of gravity, if time allows.

If there is still more time, we will add a new feature of subtraction in the game.

- **Design patterns of the game.**
- For the purpose of this project, we will use the MVC design pattern.
- The View:
    - Functionality:
        - The view displays an initial screen, which includes options of the three difficulty levels.
        - The view displays the 4*4 grid, and the value of the tiles. When the user presses the keyboard to change the position of the tiles, the view displays the new positions and numerical values of the tiles.
        - The view displays the user’s current scores.
        - When the user has no more moves to make, the view displays a message that the user has lost, and gives the option to start a new game.
    - Design pattern:
        - The view is an observer of the model and the controller. When there are changes in the model and in the controller, the view receives notifications.
        - The view is a composite of GUI components. The view consist of a combination of buttons, panels, leafs, windows and text. Some panels are composed of smaller panels and buttons. The different parts form the view together.
- The controller:
    - Functionality:
        - The controller takes in the input of the game’s difficulty level at the beginning of the game.
        - The controller takes in the user’s input from the keyboard, and passes the information to the model to make changes to the board.
    - Design Pattern:

        - The controller and view implement the strategy pattern. The view is responsible for the presentation of the game, while the controller provides different strategies (or algorithms). For example, when a user makes a move through the view, it is the controller that decides whether to merge the tiles or not, depending on the current values of the tiles. Another example would be when the user chooses a difficulty level via the view at the beginning, the controller provides strategies corresponding to that difficulty level.
    - The Model
        - Functionality:
            - The model stores the board, the tiles and the values associated with them; it also stores the user's current scores.
            - The model contains functions that takes in the user's move and modify the board.
            - The model contains functions that change the values of the tiles.
            - The model contains functions that modify the user's scores.
            - The model also implements the more advanced functionality, such as the gravity function and the subtraction feature.
        - Design Pattern
            - The model implements the Observer Pattern. It is observable for the view and the controller. When a change is made, it pushes its changes to its observers.

- **A series of sketches the GUI.**

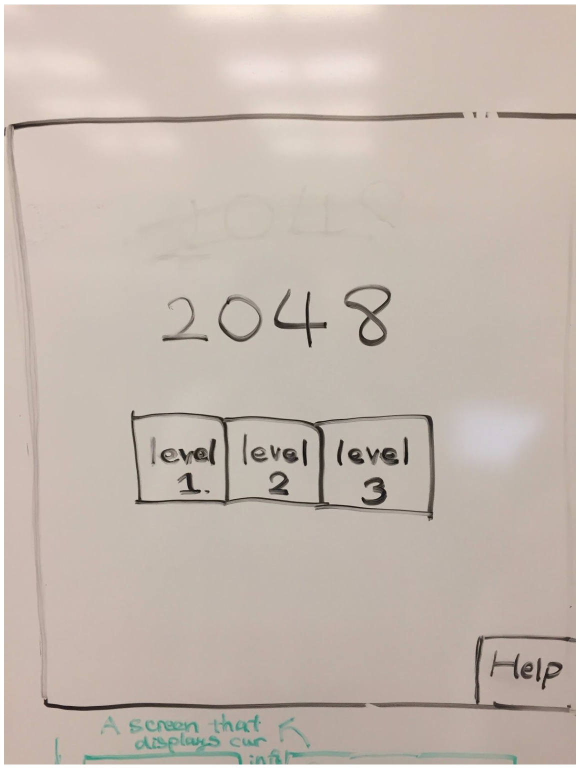


Figure 1. Home page

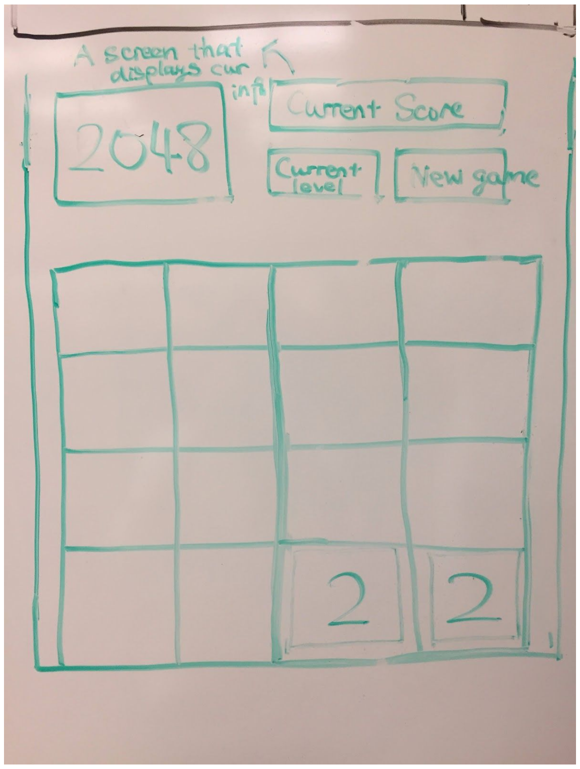
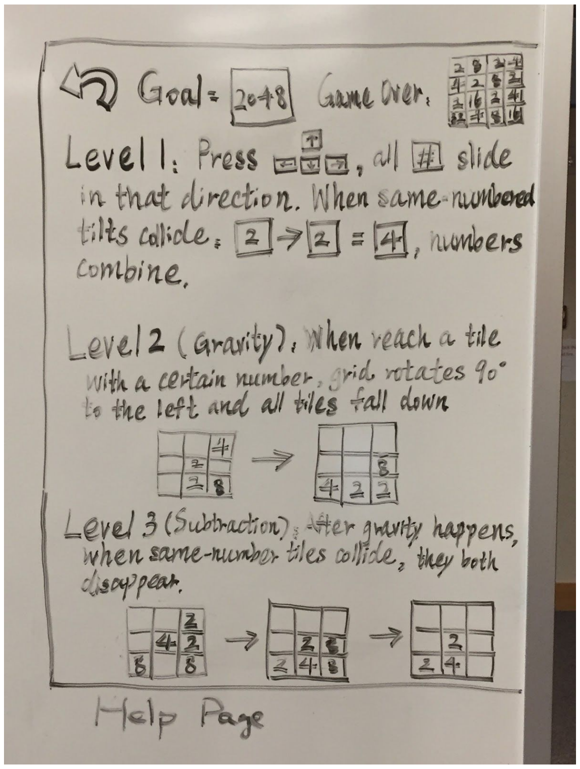


Figure 2. Game page



Goal: 2048 Game Over:

Level 1: Press ←↑↓→, all # slide in that direction. When same-numbered tilts collide: 2 → 2 = 4, numbers combine.

Level 2 (Gravity): When reach a tile with a certain number, grid rotates 90° to the left and all tiles fall down

Level 3 (Subtraction): After gravity happens, when same-number tiles collide, they both disappear.

Help Page

Figure 3. Help page

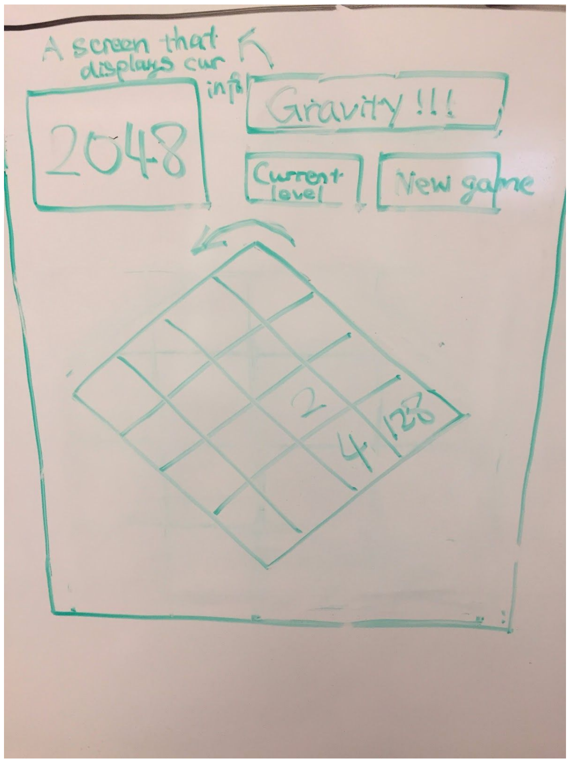


Figure 4. Page of gravity function

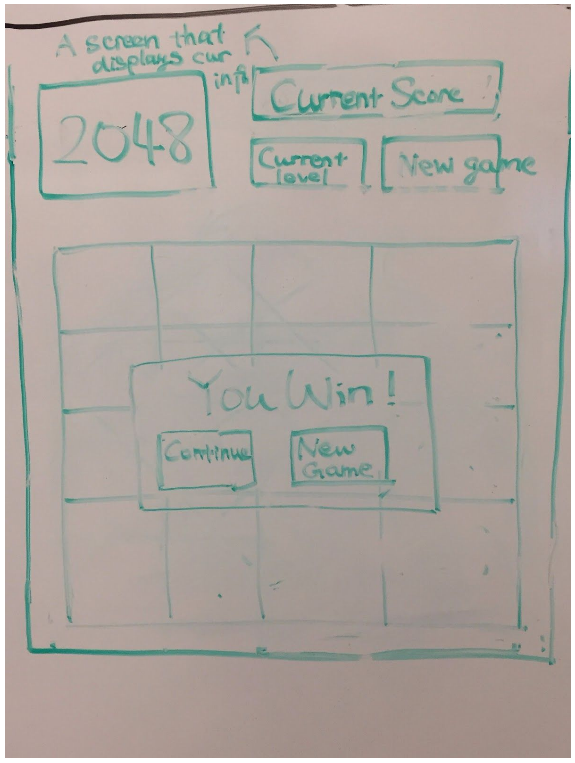


Figure 5. Page when winning the game

- **A list of the roles/tasks that each of the team members will undertake.**

  This project can be divided into three parts: View(GUI), Model and Controller.

  We are planning to develop our 2048 game GUI using JavaFX. Eva and Megan will mainly work on this part, including the design of the game board, the animation of each move and the display of scores.

  The model is made using Java to get inputs from the controller and process the data as the game goes on. Kaixing will work on this part, including creating a java class for each block, managing the data in each block class and writing the core algorithm for the game.

  The controller will be made after we finished implementing both view and model, because we can make sure the controller connects to the view and the model only when we have the basic structure of the view and the model. We will work on this part together as a team.

Once we have finished all three parts, we will run some tests on the basic version of our 2048 game to make sure the game works as we expected. Then, if time allows, we can implement more advanced features to increase the difficulty and creativeness.