Team 5

Professor: Russell Hunter

Class: CSE 210

Meeting Date: 11/24/2021

Attendees: Alirio, Jeremy, Ryder, Andrew

Link to our repository: [cse210-team-final-project/minesweeper/game at master · Jeremy-Diamond/cse210-team-final-project (github.com)](https://github.com/Jeremy-Diamond/cse210-team-final-project)

Grading Category: 5

I put grading level 5 because we hit all of the requirements for our Beta release and all #1 features have been completed. I have also updated the features list and added a list of classes that we are working on. To this document which I will have uploaded in our rep documents folder. I have also added it to the readme file. I am also starting to go through the code and add in more notes like you suggested last week.

**Final Project Name - Mine Sweeper**

**Game Play**

* This is a game where using logic and a bit of luck you can clear the board of all of the mines. Hopefully without getting blown up first!

Left click to dig out a square and expose what is beneath the surface, but be careful because if you dig up a mine it will blow up and you will lose. We have also buried a lot of clues for you to find so if you dig up a number it will let you know exactly how many mines are touching it.

If you are sure, you know where a mine is right click on it once to mark it with a flag, if you are not 100% sure right click twice to make it a question mark, and a third right click will return it to its original value so you can act like you were not even there.

That is pretty much it. Once you have dug up every hole that does not have a min in it YOU WIN

**Getting Started**

---

Make sure you have Python 3.8.0 or newer and Arcade 2.1 or newer installed

and running on your machine. You can install Arcade by opening a terminal

and running the following command.

```

$ python -m pip install arcade

```

After you've installed the required libraries, open a terminal and browse too the

project's root folder. Start the program by running the following command.

```

python3 minesweeper

```

You can also run the program from an IDE like Visual Studio Code. Start your IDE

and open the project folder. Select the main module inside the minesweeper folder and

click the "run" icon.

**Project Structure**

---

The project files and folders are organized as follows:

```

root (project root folder)

+-- minesweeper (source code for game)

+-- game (specific game classes)

+-- \_\_init\_\_.py (python package file)

+-- \_\_main\_\_.py (entry point for program)

+-- README.md (general info)

```

**Required Technologies**

\* Python 3.8.0

\* Arcade 2.1

**Features to Include**

#1 = must have

#2 = nice to have

#3 = just for fun if there is time

Start Screen #1

* Select custom grid size and number of mines #3
* select grid size by difficulty #2 (DONE)
* start button #1 (DONE integrated into difficulty selection button)
* game instructions #1 (DONE)

Grid Screen (Game Board) #1

* grid #1 (DONE)
* mines #1(DONE)
* clues #1(DONE)
  + these are the numbers that will surround the mines
* Show multiple boxes if a grid box is checked that is not touching any bombs. #3
* Right Click features #2
  + Right click to plant flag
  + Second Right Click adds “? “
  + Third right click returns to default

Scoreboard #1

* count of flags remaining #2
* Add a timer that shows how long the game has been going. #3
* ~~count of mines remaining #2~~  (REMOVED because it would tell the user if they guessed right or not.)

Reset button #1

* make button interactive when clicked #3
  + on click #3
  + after click still alive #3
  + after click dead #3

Win Screen #1

* make it a fun screen with a guy that digs up all of the mines #3

Lose Screen #1

* make it a fun scene #3

**Classes**

* MenuView
  + Controls the initial menu and lets the user select the difficulty level for the game.
* SetupBoard
  + Used to update constants that will be used for the game and create the game board screen
* Constants
  + Not a class but is used to store the information used during gameplay.
* GameBoard (this acts as our “Director” Class)
  + This is the main grid where the game is played
* ExplosionCheck
  + Checks to see if the grid box that was checked is a bomb
  + If it is not a bomb it will also check to see how many bombs are touching the square and return that number
* MovePiece
  + Moves the sprites that are on the grid according to what the explosion check returns
* GameOver
  + This will either show a you win screen or a you lose screen depending on weather you win or lose.

**Timeline**

* We will complete this project over the next 4 weeks
  + Week 1. Initial brainstorm and planning, create readme file.
  + Week 2. Identify all classes and modules build frame of game.
  + Week 3. Write functions and get game functioning.
  + Week 4. Workout any bugs and make game enhancements if there is time.
  + Week 5. Finalize enhancements and rollout game.

**Authors**

Jeremy Diamond socjeremyd@gmail.com

Alirio Mieres mie21001@byui.edu

Ryder Garache gryderjose@gmail.com

Pierre Phil Bangay pierrephild@gmail.com

