CMPUT 355 A4

Name: Bosco Chan

Group: Argo

My Contribution: I did all the planning and programming for this project.

Used Code: I did not use any code to start this assignment. There are sources for functions I have used in the code from online.
GitHub: https://github.com/bosco4/CMPUT355-A4.git
Video: See here

Chosen Game: BattleShip
Description: 2 players place 5 battle ships on a grid (with each ship consuming a different number of spaces on the grid), and then attacking each other's battle ships by alternating turns to guess a coordinate on the grid (simulating missles attacks) to where their opponent's battle ships could be to sink them. The winner is determined when a player has destroyed all of their opponent's battleships. Best Explanation: See here

Original Goals: Implement Player vs Player (PvP), Player vs Environment/Computer (PvE), and different versions of battle ship such as Classic, Salvo, RealTime (where you could move your ships during battle). For PvP the goal is to just have a running version of the game where each player shares a computer, places their ships in secret, and attacks each other through the grid system. For placing of the ships, and attacking, each player should be able to see two grids; a grid containing their own battle ships (showing ONLY the condition of the ships, i.e. which have parts haven been attacked), and a grid containing launched attacks to their opponents (showing only "successful attacks" and "missed attacks"). For PvE the goal is to have the same gaming system implemented in PvP, but with the goal of having three difficulties (easy, normal, hard). The computer difficulty should be dictated by having the computer increasingly be better at guessing ship coordinates (getting closer and closer to human guessing). Finished Goals: Complete both Classic and Salvo mode, implemented both PvP and PvE (PvE is a random player). Missed Goals: PvE increasing in difficulty (better at guessing) through probability (easy, normal, hard), and realtime mode. I really loved making this game from scratch, it made me think on how difficult it is to implement probabilistic choices in computer decisions given past data. If I could work on this more, I would love to try and understand this topic further.

Performance data: I compared my player with this game (http://en.battleship-game.org/). In this player they had markers for where neighboring battleships could be for the attacker, and also had allowed for multiple attacks if the previous one had been successful (which was unorthodox to the rules I followed). I also noticed that they had a UI for placing ships, where a mouse-drag could change ship placements. My player only had command line inputs for coordinates to place ships and had two game modes while theirs only had one.

Yes, I am just satisfied with the quality of my project. I think that given all the UI design I had implemented with the two game modes given (PvP, PvE), it was okay but I would have loved to continue working on increasing single player

difficulty and making the game more refined (even more user friendly by using better UI, and use machine learning somehow).

Game Choice: I chose to make this game as a Player because I thought it was a fun and simple way to understand how games using coordinate systems can be programmed in terms of user interface and data storage (e.g. how grid position change when player vs player), and how computer prediction can be implemented at a basic level. Given the different modes the game's form can take, I was also curious how this would affect computer prediction, I was interested in the different versions of computer prediction needed at each game mode.

November 9, 2020:

2 hours - Deciding what game to program, what game modes to include, what kind of players to include in the game (PvP, PvE), what kind of user interface (UI) to have at the menu and at each round. Then, researching what all these areas of the game would need to include in their design prior to programming (i.e. what they would need to have to be able to accurately portray the game, and their corresponding modes).

November 16, 2020:

13 hours: Implemented user input for game mode, environment type for PvP, grid positions, board-size, showGrid() function to display a player's size-determined-grid on screen given a player's ship placement coordinates on their own grid. Finished the getShipCoords() function where it handles boat placements that are only either horizontal or vertical, but dis-allows diagonal placements or placements that are not equal to the boat's length. Handled overlapping boat placements as well. Implemented player turn based attackCoord() function in classic mode and its subsequent handling of valid inputs for a attack coordinate, tweaked getShipCoords() to record a player's ship placement coordinates on their own position grid for future game modes.

November 23, 2020:

7.5 hour: Finished classic mode for PvP, ensuring that players can place their ships, attack each other based on turn based inputs, and lastly finished the win condition for both human players. Implemented generateShipCoords() and generateAttackCoords() for the computer generated placements and attacks, completed PvE for a random player and completed salvo mode. Completed lots of valid input fixes.