**CSCI 2312 - Design Document**

1. Title: James Battleship Game  
   Name: James Nguyen  
   ID:104785020  
   Date: 12/8/2020
2. Problem DescriptionThis program is going to allow the user to play a simplified Battleship Game against an AI. Since this game is based on luck, the AI will be programmed to generate completely random decisions. This will allow the user to feel as though they are playing against another legitimate player.
3. Input RequirementsA detailed **list of all external inputs** (from files or keyboard) including a description of the **data type** and **range of valid values** for each input. For input file format and interactive user input, you need to write what data type is used for every field and valid value and length.

External inputs: ship\_placement.csv, Char keyboard input, Int keyboard input.

ship\_placement.csv is a spreadsheet file with strings and characters. The format will be: “Type of Ship, Location, Horizontal or Vertical” Type of Ship will be a string data type that holds the name of the ship. Location will be two character data types in the form of a coordinate, and Horizontal or Vertical will also be another string data type. The range of values for Types of Ship will be: “Carrier”, “Battleship”, “Cruiser”, “Submarine”, “Destroyer”. The range of values for Location would be 1-10 for the coordinates. The Range of Values for Horizontal or Vertical is the strings “H” or “V”.

We will be using integer keyboard inputs from the user to use for the first coordinate for the user’s guess.

We will be using integer keyboard inputs from the user to use for the second coordinate for the user’s guess.

1. Output RequirementsThis program will output two user grids until the game is over. These two grids are GameBoard objects that are grids full of 100 chars. Strings will be outputted to display message prompts and the grids will be outputted to display the game.
2. Problem Solution Discussion  
   A summary description of the solution steps with algorithms analysis (1 paragraph, approximately 100 words). If any unusual techniques or algorithms are used that need further explanation, and additional paragraph may be used.

Three classes are created: GameBoard, Ship base class, and the ship type derived classes. The Board is an array of chars and each ship is an object of unique size and unique char. These ships are then placed on the grid like the board game would place ships on the board. The player will take turns with the CPU guessing coordinates until all ships are sunk on either side.

1. Overall Software Architecture

Major Functions: ReadShips(), isValid(), at(), setShip(), PrintGrid(), checkSunk(), checkHit(), FinishedGame()

ReadShips() is ship class function to read in ship objects from csv file into a vector.  
isValid() checks whether ship placement is valid on game board or not.  
at() is like VectorName.at() but for char pointers.  
setShip() is a game board function that uses ship as parameter to place on grid.  
PrintGrid() is a game board function to print the grid.  
checkSunk() is like isValid but instead of checking if free, checks for sunk ships.  
checkHit() is a function that uses at() on the gameboard to check if the coordinate guessed was a hit.  
FinishedGame() is a broken function that I attempted to create in order to use for the game loop.

Testing on csegrid was unable to be done because for some odd reason, CU Denver has blocked me from connecting to the CU Denver VPN, disabling me from being able to test on csegrid.