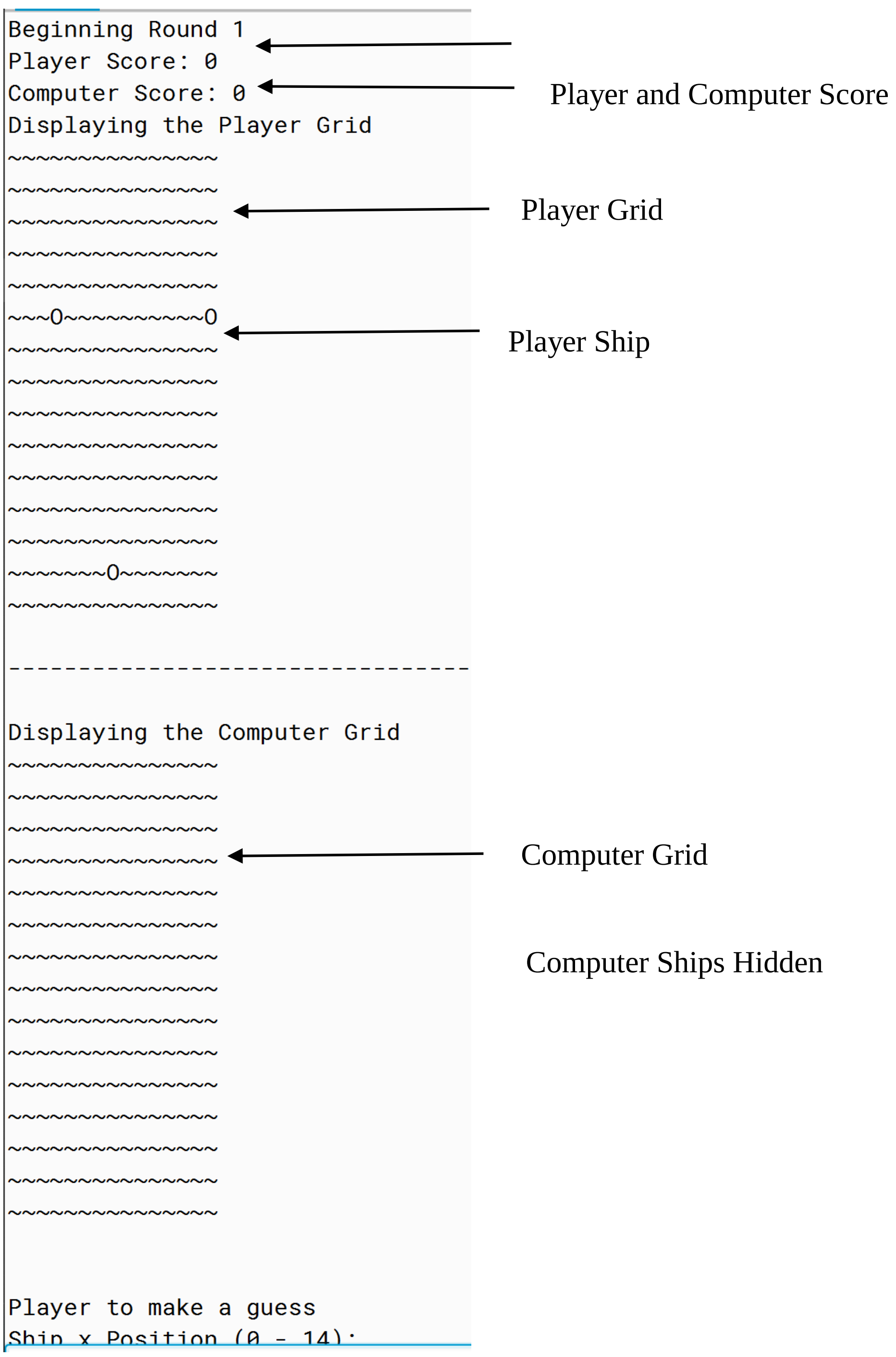
The game will allow the player to place a number of ships onto the playing grid based on x and y coordinates entered by the user. The computer opponent will also do the same randomly. Each ship will only occupy a single coordinate, as such, two ships cannot be placed at the same set of coordinates. To keep things interesting, each ship when places is given a random number between 1 – 5 which indicates its overall strength. This means that the same ship has to be hit that many times in order for it to be destroyed. Once a ship has been destroyed it is taken out of play on the grid. Each hit on a ship get points for the player or the computer. The game ends when all the ships are destroyed.

Game play

The following section outlines the game play for each of the involved participants.

Round Number



```
Beginning Round 1
Player Score: 0
Computer Score: 0
Displaying the Player Grid
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~0~~~~~~~~~~0
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~0~~~~~~~
~~~~~~~~~~~~~~~

---------------------------------

Displaying the Computer Grid
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~


Player to make a guess
Ship x Position (0 - 14):
```

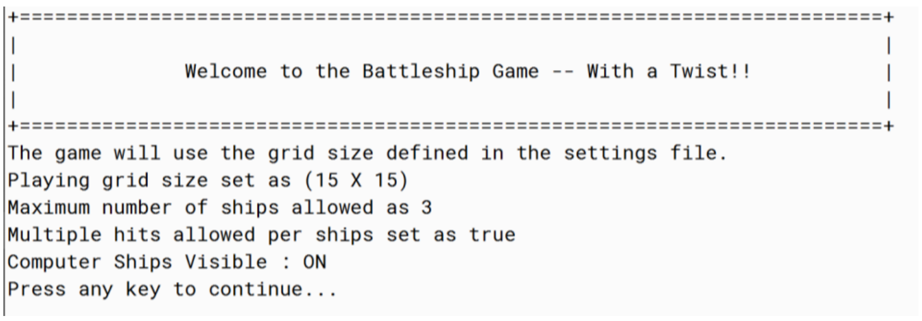
The Playing Grid

The playing grid will be of a fixed size for the duration of the game. On starting the game, the game will load the following preferences from a file called “gamesettings.txt”.

Grid Width and Height: 5
Multiple Hits Allowed: true
Computer Ships Visible: false
No of Ships : 3

The file will represent this default data as follows: 5,true,false,3

Your program must work based on the parameters specified in this file. (Note: default values not used)



```
+======================================================================+
|                                                                      |
|              Welcome to the Battleship Game -- With a Twist!!        |
|                                                                      |
+======================================================================+
The game will use the grid size defined in the settings file.
Playing grid size set as (15 X 15)
Maximum number of ships allowed as 3
Multiple hits allowed per ships set as true
Computer Ships Visible : ON
Press any key to continue...
```

Manoeuvring the Grid:

To move the various elements along the grid, the grid is to be represented as coordinates. Each position on the grid is given a coordinate in the format x,y.

No two ships can occupy the SAME coordinate at any given time.

This is only applicable to each player. For e.g. the player can place a ship at 2,3 and the computer can place its ship at 2,3. But both the player and computer cannot place another ship at the same coordinate.

The Grid

The grid will use the following notation:

~ – to indicate water and an unoccupied grid. The computer ships until hit will also display this symbol.
O – to indicate a player ship to the player if one is placed at those coordiantes
D – a ship which has been hit at least once to indicate a damaged ship.
X – a destroyed ship.

The computer’s ships MUST be made visible or be hidden from the player based on the flag set in the settings file above. When the computer ships are hidden, the D and X status of each ship MUST be shown on screen when hit or destroyed. However undamaged ships are shown as ~ symbol.

Game Setup:

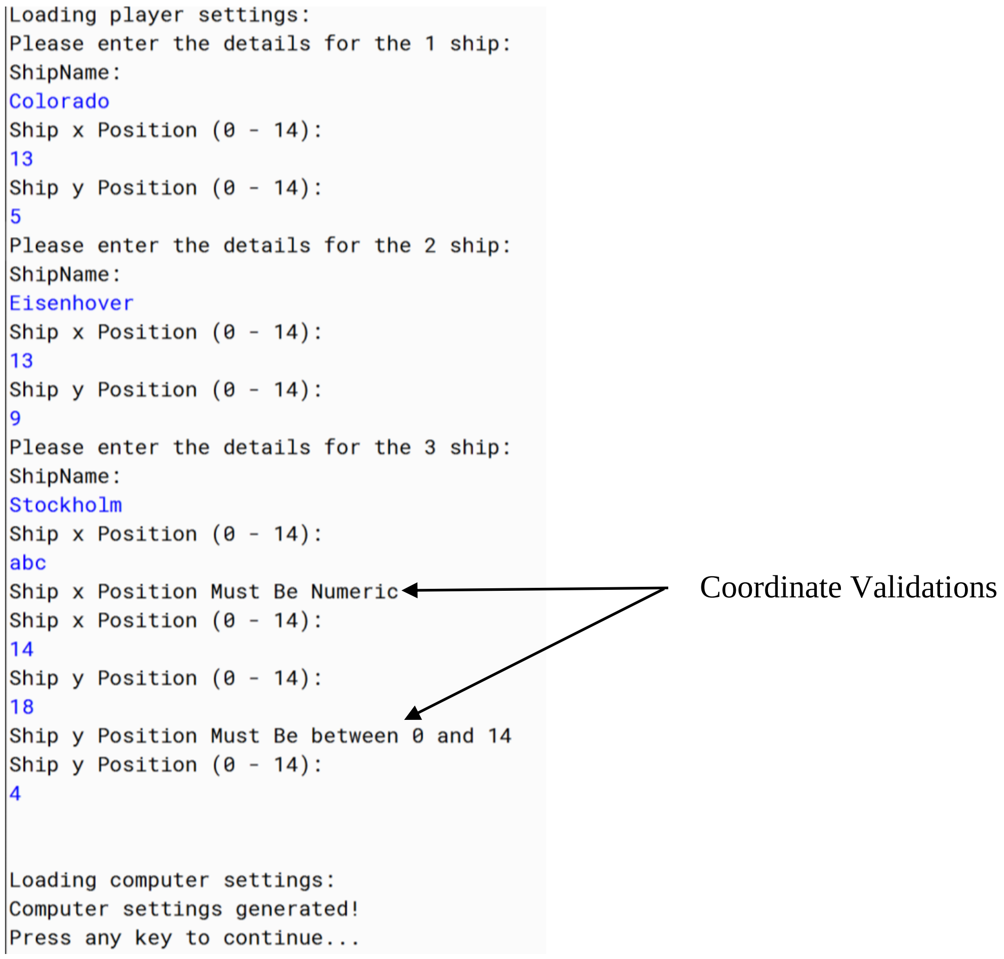
After loading the game settings from the file, the system will then ask the user to enter the following information for each ship:

1. Ship Name – String between 3 and 15 characters long.

2. x Coordinate – X coordinate position between 0 and the maximum grid size from the settings file.

3. y Coordinate – y coordinate position between 0 and the maximum grid size from the settings file

4. If any coordinate is incorrect (e.g. alphabetic or outside range of grid) an error message should be displayed.



```
Loading player settings:
Please enter the details for the 1 ship:
ShipName:
Colorado
Ship x Position (0 - 14):
13
Ship y Position (0 - 14):
5
Please enter the details for the 2 ship:
ShipName:
Eisenhover
Ship x Position (0 - 14):
13
Ship y Position (0 - 14):
9
Please enter the details for the 3 ship:
ShipName:
Stockholm
Ship x Position (0 - 14):
abc
Ship x Position Must Be Numeric
Ship x Position (0 - 14):
14
Ship y Position (0 - 14):
18
Ship y Position Must Be between 0 and 14
Ship y Position (0 - 14):
4


Loading computer settings:
Computer settings generated!
Press any key to continue...
```

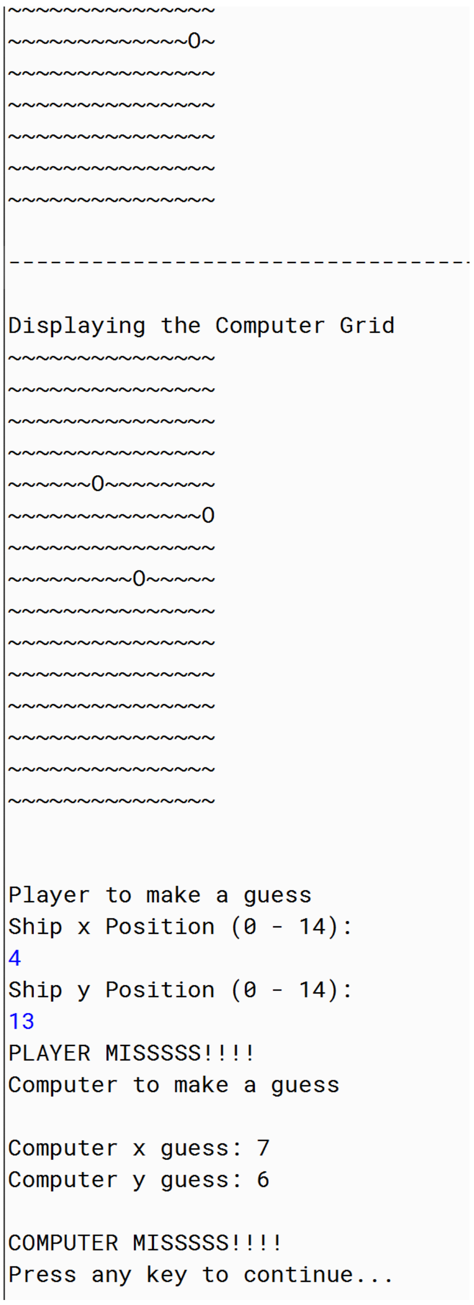
The number of ships set is determined from the game settings file.

After the player ships are set, the computer ships are automatically generated.

<u>Turn Based Game Play:</u>

Note: For demo, default grid size is not used, and enemy ships are shown on the screen.

Turn 1: The player is asked to make a guess, the system will then check if the player hits a computer ships or misses completely. The system will then display the appropriate message to the user.

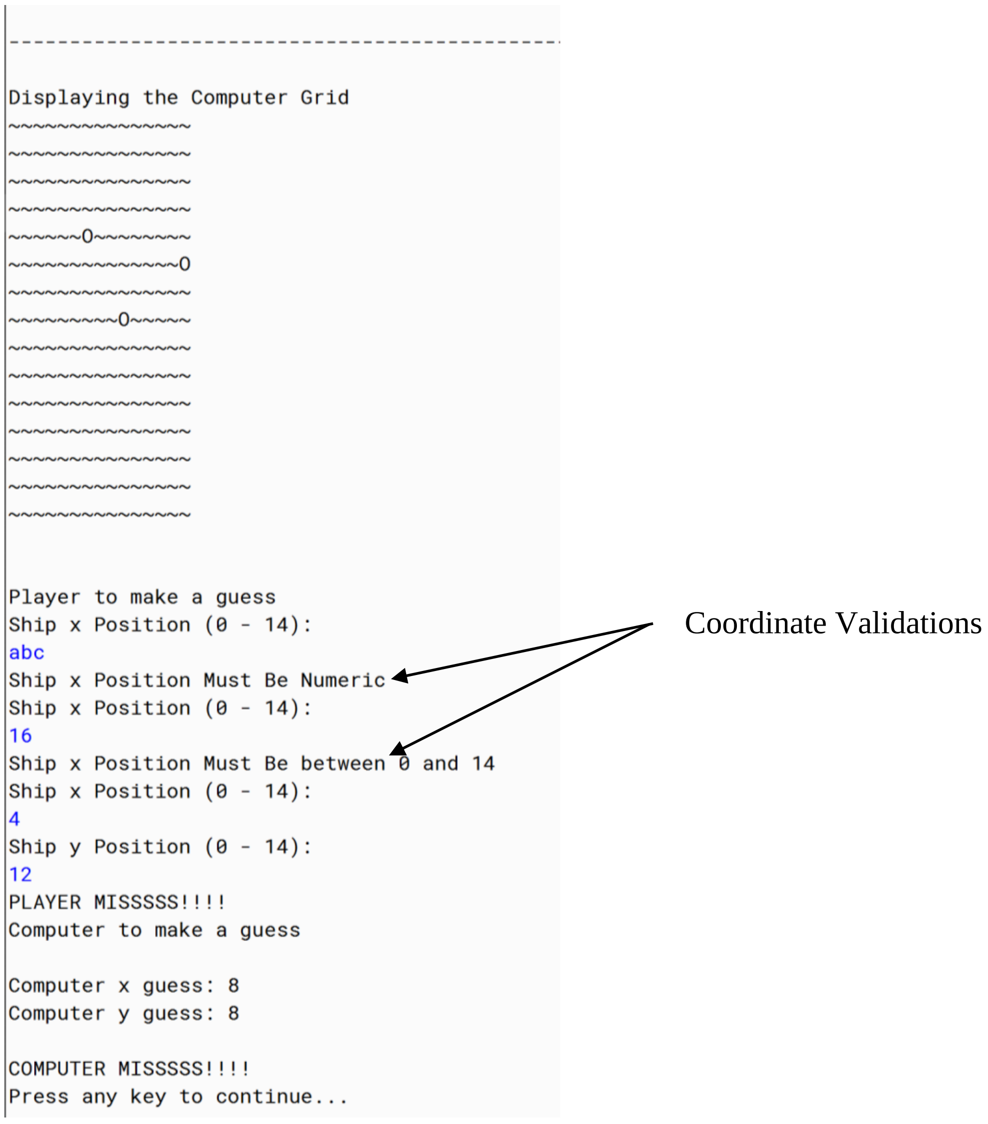


```
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~0~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~


----------------------------------


Displaying the Computer Grid
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~0~~~~~~~~
~~~~~~~~~~~~~~0
~~~~~~~~~~~~~~~
~~~~~~~~~0~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~


Player to make a guess
Ship x Position (0 - 14):
4
Ship y Position (0 - 14):
13
PLAYER MISSSSS!!!!
Computer to make a guess

Computer x guess: 7
Computer y guess: 6

COMPUTER MISSSSS!!!!
Press any key to continue...
```

After the players guess, the computer is asked to make a guess, and the system checks to see if the computer makes a hit or misses the player ships.

Turn 2:

The player is again asked to continue making a guess at the coordinates. Appropriate validations must be done as before for the entered coordinates.

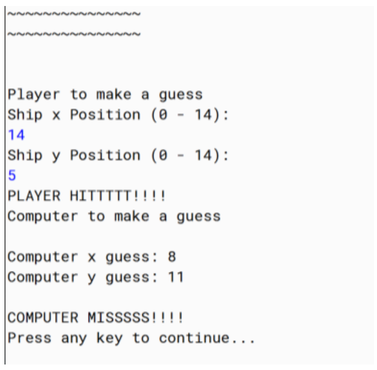


```
---------------------------------------------


Displaying the Computer Grid
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~O~~~~~~~~
~~~~~~~~~~~~~~O
~~~~~~~~~~~~~~~
~~~~~~~~~O~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~


Player to make a guess
Ship x Position (0 - 14):
abc
Ship x Position Must Be Numeric
Ship x Position (0 - 14):
16
Ship x Position Must Be between 0 and 14
Ship x Position (0 - 14):
4
Ship y Position (0 - 14):
12
PLAYER MISSSSS!!!!
Computer to make a guess

Computer x guess: 8
Computer y guess: 8

COMPUTER MISSSSS!!!!
Press any key to continue...
```

After the player guess, the computer is asked to make a guess and the system validates the guess to check for a hit or a miss.

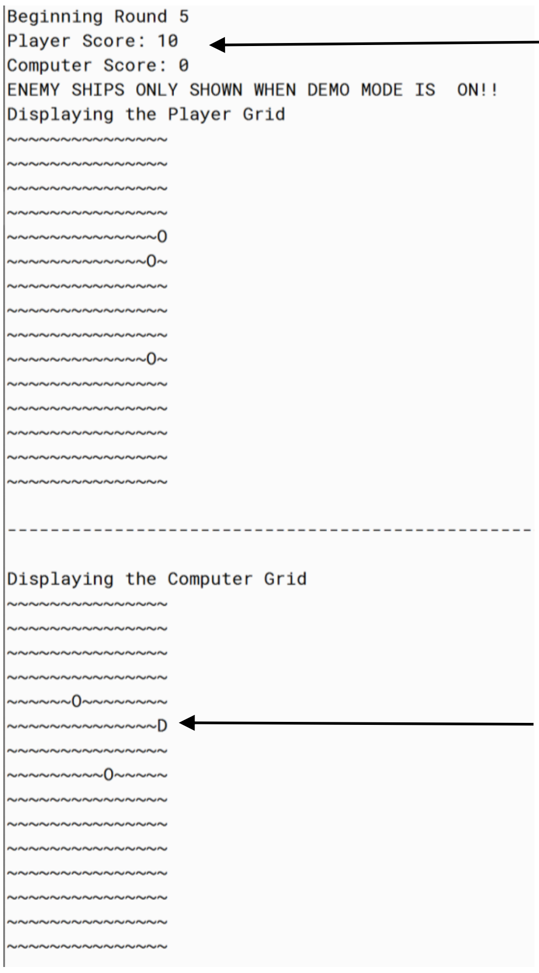
Turn 3:

The player makes a hit.



```
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~


Player to make a guess
Ship x Position (0 - 14):
14
Ship y Position (0 - 14):
5
PLAYER HITTTTT!!!!
Computer to make a guess

Computer x guess: 8
Computer y guess: 11

COMPUTER MISSSSS!!!!
Press any key to continue...
```

The system displays the appropriate message on the screen and at the start of the next turn, the grid layout is changed as follows:



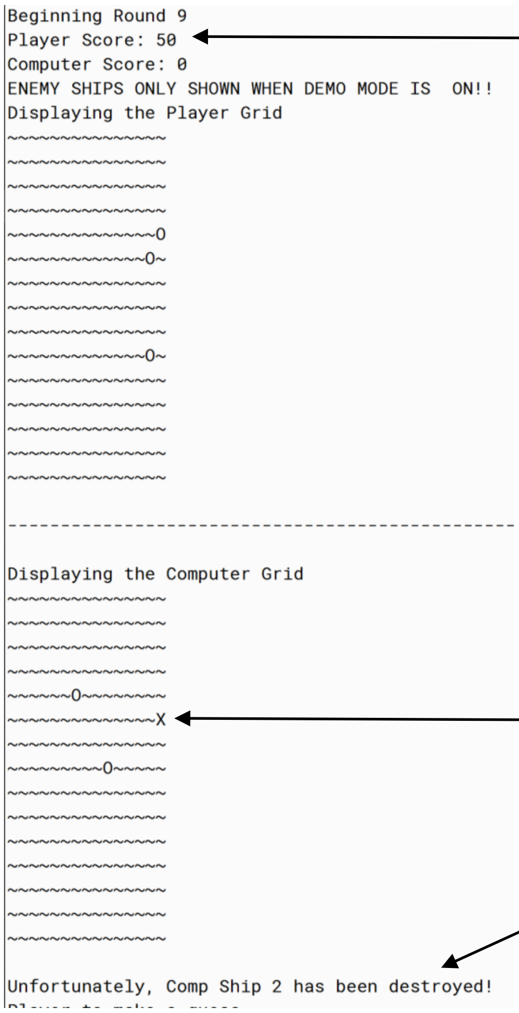
```
Beginning Round 5
Player Score: 10
Computer Score: 0
ENEMY SHIPS ONLY SHOWN WHEN DEMO MODE IS  ON!!
Displaying the Player Grid
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~O
~~~~~~~~~~~~~O~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~O~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~

-------------------------------------------------

Displaying the Computer Grid
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~O~~~~~~~~
~~~~~~~~~~~~~~D
~~~~~~~~~~~~~~~
~~~~~~~~~O~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
```

Scores are updated
Each hit is 10 points

Ship status changed to D

Turn x:

The player finally destroys an enemy ship.



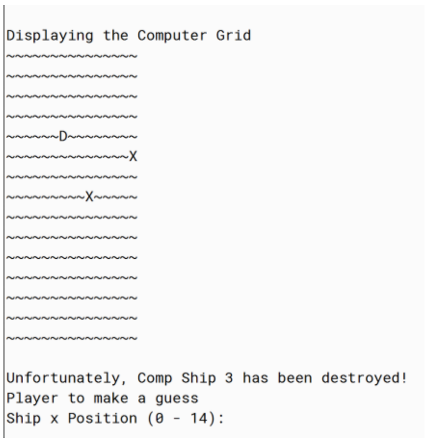
```
Beginning Round 9
Player Score: 50
Computer Score: 0
ENEMY SHIPS ONLY SHOWN WHEN DEMO MODE IS  ON!!
Displaying the Player Grid
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~O
~~~~~~~~~~~~~O~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~O~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~

------------------------------------------------

Displaying the Computer Grid
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~O~~~~~~~~
~~~~~~~~~~~~~~X
~~~~~~~~~~~~~~~
~~~~~~~~~O~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~

Unfortunately, Comp Ship 2 has been destroyed!
```

Score indicates that ship had a strength of 5.

Ship status changed to X

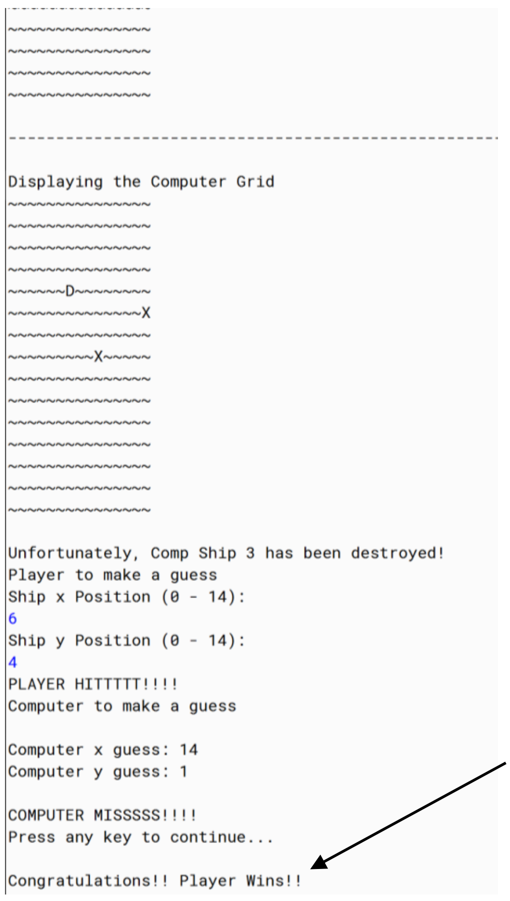
Message to state the ship has been destroyed

Two Ships Destroyed:



```
Displaying the Computer Grid
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~D~~~~~~~~
~~~~~~~~~~~~~~X
~~~~~~~~~~~~~~~
~~~~~~~~~X~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~

Unfortunately, Comp Ship 3 has been destroyed!
Player to make a guess
Ship x Position (0 - 14):
```

All Ships Destoryed:



```
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~

----------------------------------------------------

Displaying the Computer Grid
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~D~~~~~~~~
~~~~~~~~~~~~~~X
~~~~~~~~~~~~~~~
~~~~~~~~~X~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~

Unfortunately, Comp Ship 3 has been destroyed!
Player to make a guess
Ship x Position (0 - 14):
6
Ship y Position (0 - 14):
4
PLAYER HITTTTT!!!!
Computer to make a guess

Computer x guess: 14
Computer y guess: 1

COMPUTER MISSSSS!!!!
Press any key to continue...

Congratulations!! Player Wins!!
```

Winning Message

On completing the game, the system will write the outcome to a file called "gameoutcome.txt".

The outcome file will have the following output:

Player wins. Final Score Player (110) and Computer (0)

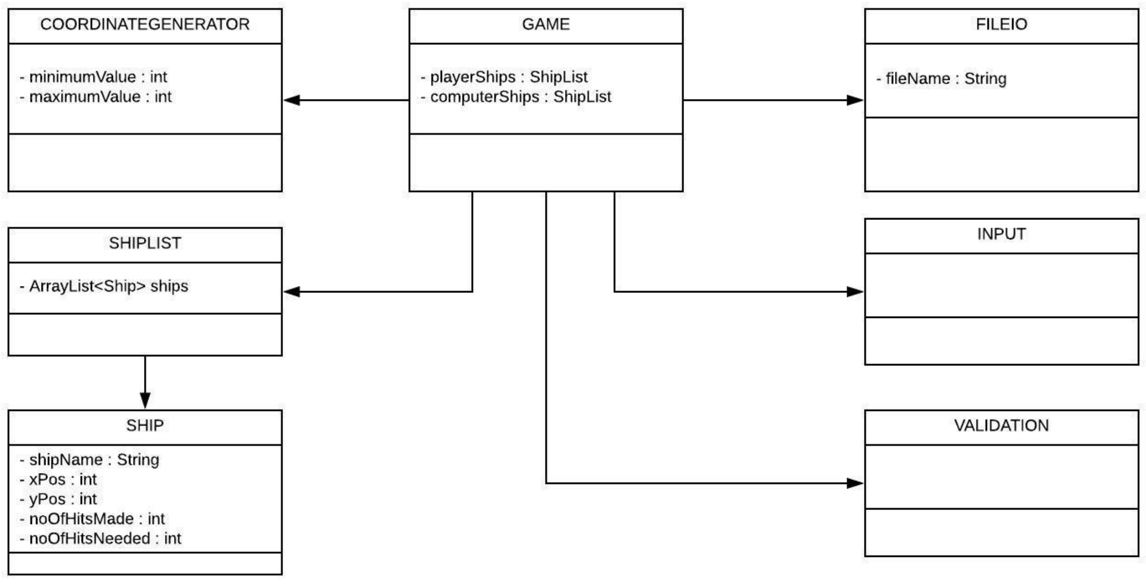
Game Rules

The following game rules MUST be adhered to when playing the game:

- No two ships can occupy the same X coordinate and Y coordinate for each player
- Guesses must be within the specified range
- Each hit gets a score of 10 points
- Each player gets one guess per turn
- The hull strength of each ship is determined randomly between 1 – 5. This indicates the number of hits required for the ship to be destroyed.

Program design

The objective of this assignment is for students to understand coding, as well as to get the basics of design. The following class diagram is **proposed** for this assignment.



Your program **MUST** consist of the following classes:

1. Game
2. ShipList
3. CoordinateGenerator
4. Ship

The following classes are desirable and will score marks if implemented. But if object interaction is still challenging, these classes can be integrated into the other 4 classes mentioned above.

5. FileIO
6. Input
7. Validation

You MUST follow good programming practices and use loops where required to ensure good program design.

Design changes to the above must be discussed with your tutor prior to proceeding.

Game class

The Game class will specify the attributes and behaviours of the game. An object of the Game class will have the following fields only:

- *playerShips* – (ShipList) an object of the ShipList class.
- *computerShips* – (ShipList) an object of the ShipList class.

This class is responsible for initiating the game, reading the file, loading the settings, interacting with the other classes, and writing to the file when the game ends. **Reading and Writing to the file must only occur ONCE.**

ShipList class

The ShipList class will specify the attributes and behaviours of all the ships within the game. An object of the ShipList class will have the following fields only:

- *ships* – (ArrayList<Ship>) – stores objects of the class Ship within an ArrayList collection.

This class is responsible for creating an arraylist which stores each ship within the player grid for the respective player or computer.

CoordinateGenerator Class

The CoordinateGenerator class will specify the attributes and behaviours of generating random coordinates which will be used by the Game class. An object of the CoordinateGenerator class will have the following fields only:

- *minimumValue* – (int) stores the lower limit of the number to be generated.
- *maximumValue* – (int) stores the upper limit of the number to be generated.

This class is responsible for generating a random number which can be used for deciding the X and Y coordinates for each ship and even for deciding the random hull strength of each ship within the game.

Ship Class

The Ship class will specify the attributes and behaviours of all ships within the game. An object of the Ship class will have the following fields only:

- *shipName* – (String) stores the name of the ship between 3 – 15 characters.
- *xPos* – (int) stores the x position of the ship on the grid.
- *yPos* – (int) stores the y position of the ship on the grid.

- *noOfHitsMade* – (int) stores the number of times the ship has been hit.

- *noOfHitsNeeded* – (int) stores the number of times the ship needs to be hit to be destroyed. This number if randomly generated by the game for each ship for both the player and the computer.

This class is responsible for recording a new ship added to the grid.

Additional Classes:

FileIO Class

The FileIO class will specify the attributes and behaviours for reading and writing to a file. An object of the FileIO class will have the following fields only:

- *filename* – (String) the name of the file to be read or written to.

This class is responsible for reading and writing to a file only. The data read should be passed to the calling class to handle.

Input Class

The Input class will specify the attributes and behaviours for reading input from the user via the keyboard. An object of the Input class will have no fields. However, all methods included can be made class methods.

Validation Class

The Validation class will allow the system to validate all user inputs accepted via the keyboard from the user. An object of the Validation class will have no fields.

Hints and Suggestions

The following hints and suggestions may be useful in designing your program:

- To facilitate a clean layout of your game, consider looking at the tutorial notes in the homework exercises on how to clear the terminal screen in BlueJ (Week 4).

- When reading and writing to a file with multiple values, consider looking up the String class for the split method to understand how it can be used to separate multiple values.

- Your program MUST validate string and numeric inputs for this assignment. Consider using exception handling well in order to ensure your program does not crash.

- **YOUR PROGRAM MUST NOT CRASH NOT MATTER WHAT THE USER DOES.**

- File input will always be treated as correct and the input file will not be modified directly.