Online Battleship Game

# Prerequisites

1. The project is to be completed **by teams of two**.
2. You can use any programming language you want. However, make sure your code is easily readable with sufficient explanations and comments. Also, use a proper and consistent naming convention. Do not copy/paste someone else’s code.
3. You can also discuss the project among yourselves or in the Google Classroom forum.
4. You must enclose instructions (a readme file) on how to compile, build and run your code (along with any system requirements).

# Submission

Create a single zip file that includes everything, name the file as your\_lastnames\_project1.zip and upload it in Google Classroom.

# Demo

You will show the demo and explain the code to the TA at a designated time.

# Problem to Solve

First of all, if you are not familiar with the Battleship game, read about it (<https://en.wikipedia.org/wiki/Battleship_(game)>). The game will be played among two players (two instances of your program) over the local area network. Your game will be all (colored) text based (no need for fancy graphics). There will be four ships to start with on each side.

1. Carrier (length of 5, denoted by C)
2. Battleship (length of 4, denoted by B)
3. Submarine (length of 3, denoted by S)
4. Destroyer (length of 2, denoted by D)

Each player will have two grids of 10x10 (A→J and 1→10) on the screen. The left grid is for the player’s own ships and the right grid will show the player’s attempts to sink his opponent’s ships. When the program is run, the four ships will be randomly placed over the left grid (horizontally or vertically).

The players will try shots in turns and whoever sinks all the ships on the other side first wins the game. Shots are denoted by a letter and number such as A1, C5, J2. If player 1 takes the shot and it is a hit, the shot will be designated by an “H” on a green background on the right grid for player 1, and it will be designated by red background on player 2’s left grid. If the shot is a miss, player 1’s right grid will show it with an “x” on a red background and player 2’s left grid will show it with an “x” on a green background.

**Player 1**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **You** | | | | | | | | | | |  | **Opponent** | | | | | | | | | | |
|  | **A** | **B** | **C** | **D** | **E** | **F** | **G** | **H** | **I** | **J** |  | **A** | **B** | **C** | **D** | **E** | **F** | **G** | **H** | **I** | **J** |
| **1** |  |  |  |  |  |  |  |  |  |  | **1** |  |  |  |  |  |  |  |  |  |  |
| **2** |  |  | **C** | **C** | **C** | **C** | **C** |  |  |  | **2** |  |  |  |  |  |  | **H** | **H** |  |  |
| **3** |  |  | **x** |  |  |  |  |  |  |  | **3** |  |  |  |  |  |  | **x** |  |  |  |
| **4** |  |  |  |  |  |  |  | **D** |  |  | **4** |  |  |  |  |  |  |  |  |  |  |
| **5** |  |  |  | **B** |  |  | **x** | **D** |  |  | **5** |  |  |  |  |  |  |  |  |  |  |
| **6** |  |  |  | **B** |  |  |  |  |  |  | **6** |  |  |  |  |  |  |  |  |  |  |
| **7** |  |  |  | **B** |  |  |  |  |  |  | **7** |  |  |  |  |  |  |  |  |  |  |
| **8** |  |  |  | **B** |  |  |  |  |  |  | **8** |  |  |  |  |  |  |  |  |  |  |
| **9** |  |  |  |  |  |  | **S** | **S** | **S** |  | **9** |  |  |  |  |  |  |  |  |  |  |
| **10** |  |  |  |  |  |  |  |  |  |  | **10** |  |  |  |  |  |  |  |  |  |  |

**Player 2**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **You** | | | | | | | | | | |  | **Opponent** | | | | | | | | | | |
|  | **A** | **B** | **C** | **D** | **E** | **F** | **G** | **H** | **I** | **J** |  | **A** | **B** | **C** | **D** | **E** | **F** | **G** | **H** | **I** | **J** |
| **1** |  |  |  |  |  |  |  |  |  |  | **1** |  |  |  |  |  |  |  |  |  |  |
| **2** |  |  |  |  |  |  | **B** | **B** | **B** | **B** | **2** |  |  |  |  |  |  |  |  |  |  |
| **3** |  |  | **C** |  |  |  | **x** |  |  |  | **3** |  |  | **x** |  |  |  |  |  |  |  |
| **4** |  |  | **C** |  |  |  |  |  |  |  | **4** |  |  |  |  |  |  |  |  |  |  |
| **5** |  |  | **C** |  |  |  |  |  |  |  | **5** |  |  |  |  |  |  | **x** |  |  |  |
| **6** |  |  | **C** |  |  |  |  |  | **D** | **D** | **6** |  |  |  |  |  |  |  |  |  |  |
| **7** |  |  | **C** |  |  |  |  |  |  |  | **7** |  |  |  |  |  |  |  |  |  |  |
| **8** |  |  |  |  |  | **S** |  |  |  |  | **8** |  |  |  |  |  |  |  |  |  |  |
| **9** |  |  |  |  |  | **S** |  |  |  |  | **9** |  |  |  |  |  |  | **H** |  |  |  |
| **10** |  |  |  |  |  | **S** |  |  |  |  | **10** |  |  |  |  |  |  |  |  |  |  |

Note that when a player takes a shot, he is only told whether the shot was a hit or miss. In other words, the player does not know which ship was hit.

# Things to Consider:

* Pick a port number (say 34000) and your program runs on that port (TCP). If the program is run without an IP address to connect to in the command line, it will start and wait for the other player. If the program is run with an IP address in the command line, it will start and try to connect to that IP address.
* The first player to which the second player is connected starts the game (takes the first shot).
* Shots are entered by the player like A2, D5. If invalid input is made, the player will be asked again.
* Your game is supposed to work only in a local area network (direct IP connection, no need to handle NAT issues).

# Bonus Points:

1. Do not allow players to take a shot that was taken before. If that happens, send an error message asking the player to try a new shot.
2. Implement a graceful exit when the game is over.
3. Implement sound effects for misses and hits.
4. Instead of random placement of the ships, ask the player to place them.
5. Ask players to enter a name, and use the names instead of “You” and “Opponent.”

**Some points on project:**

* You need to construct two nodes structure like peer to peer (P2P). For instance, if player #1 started the game on 192.168.1.110:3000, s/he can act like both server and client. And, the player #2 should connect the player #1 (192.168.1.110:3000 ). So, there is NO need for separate server eg 3 nodes: 1 server and 2 clients.
* Your solution needs to work on LAN.
* We will mostly grade your network solution, not the UI skills. As specified in the project doc, you can make a text based game (Entering shot locations). But, cool UI/UXs are all welcome.
* There was a couple of questions about making a web app, eg working solution on the browser. To do this, you will need another server to host the UI itself. Then, you need to connect your backend and frontend to send move info from backend to frontend. Most likely,  this will end up with 3(+) nodes structure that is not allowed. So, do not make your solution a web app. Use your language's UI frameworks if needed (eg swing in Java)

<https://github.com/stan1000/Battleships>

<https://www.geeksforgeeks.org/socket-programming-in-java/>

<https://www.geeksforgeeks.org/socket-programming-python/>

https://www.pubnub.com/blog/nodejs-websocket-programming-examples/