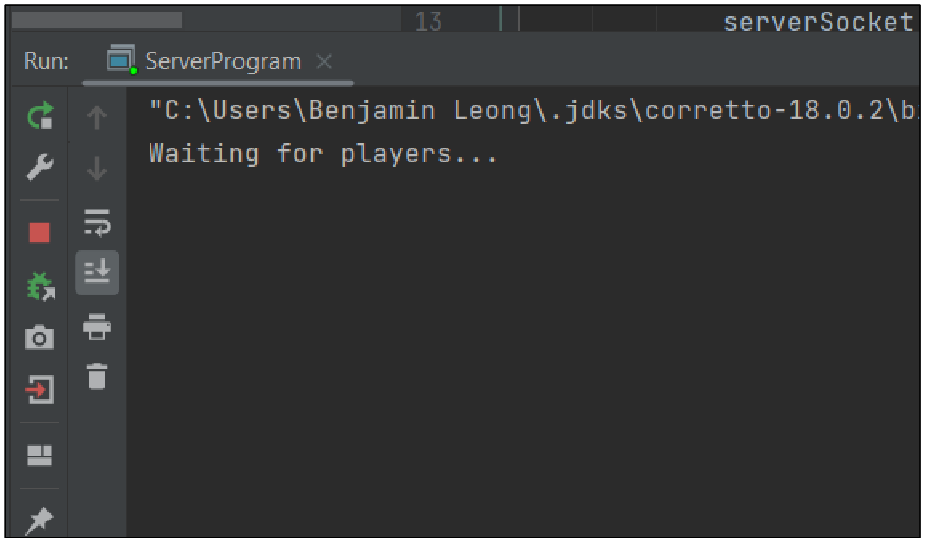
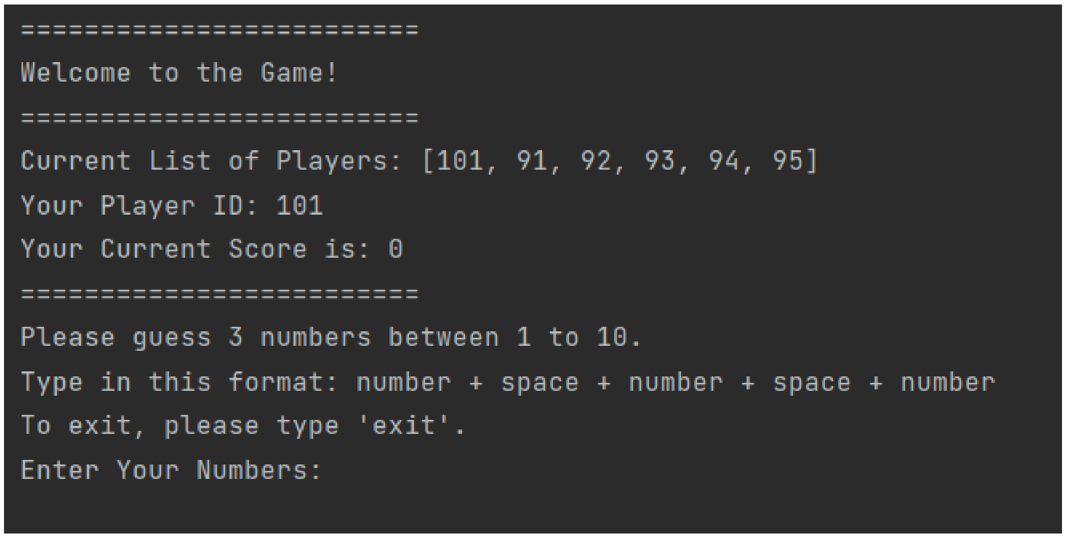
Screenshots

Run 'Server Program'



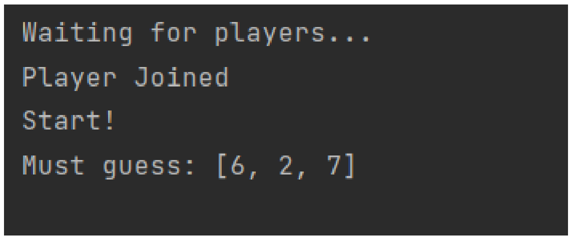
```
"C:\Users\Benjamin Leong\.jdks\corretto-18.0.2\b
Waiting for players...
```

Run "ClientProgram"



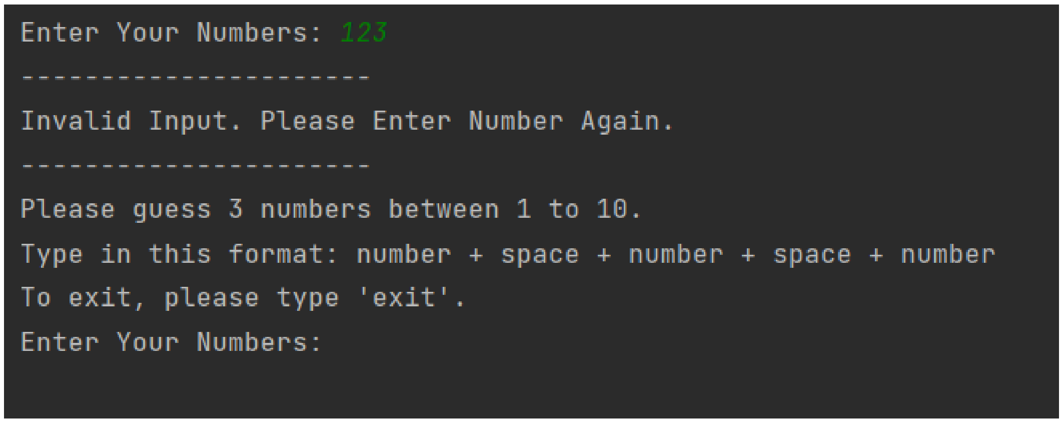
```
=========================
Welcome to the Game!
=========================
Current List of Players: [101, 91, 92, 93, 94, 95]
Your Player ID: 101
Your Current Score is: 0
=========================
Please guess 3 numbers between 1 to 10.
Type in this format: number + space + number + space + number
To exit, please type 'exit'.
Enter Your Numbers:
```

Server Side:

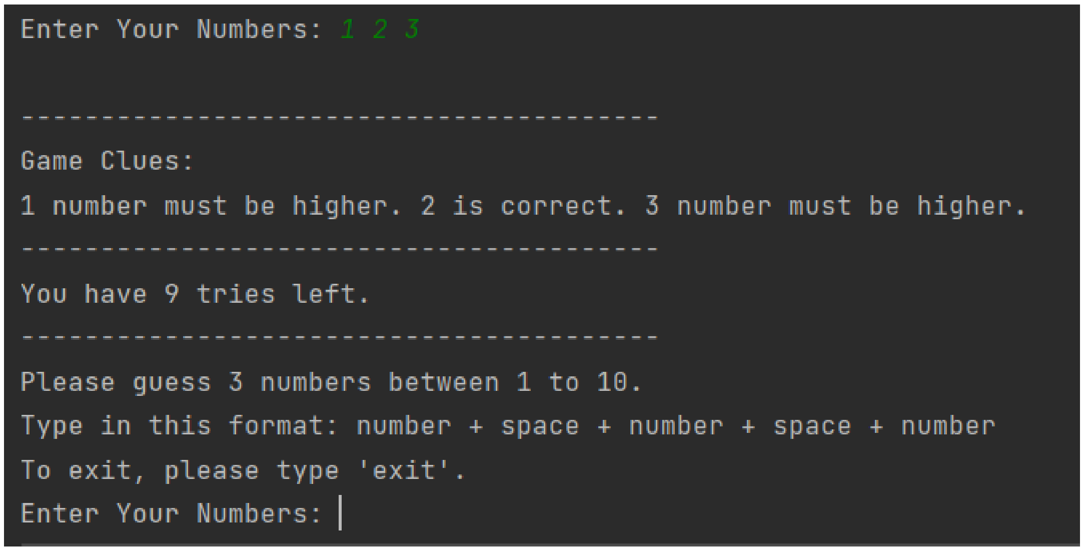


```
Waiting for players...
Player Joined
Start!
Must guess: [6, 2, 7]
```

Validation of Input

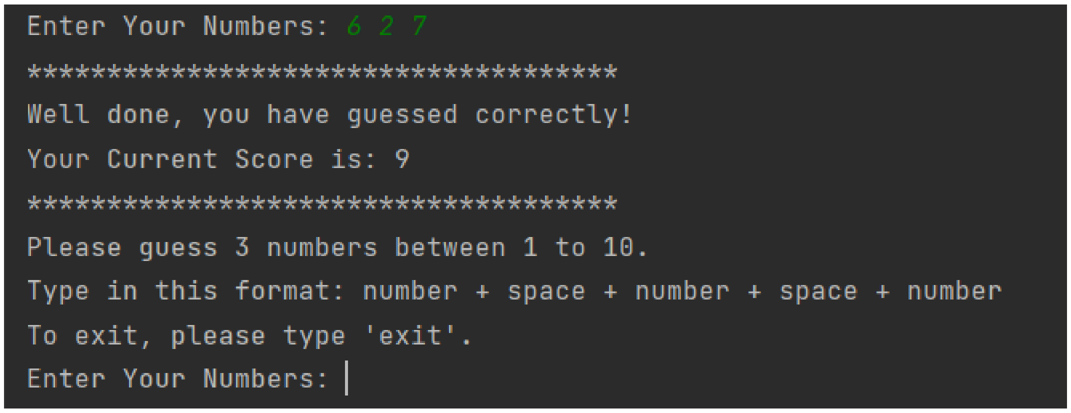

```
Enter Your Numbers: 123
----------------------
Invalid Input. Please Enter Number Again.
----------------------
Please guess 3 numbers between 1 to 10.
Type in this format: number + space + number + space + number
To exit, please type 'exit'.
Enter Your Numbers:
```

Game Clues from Server

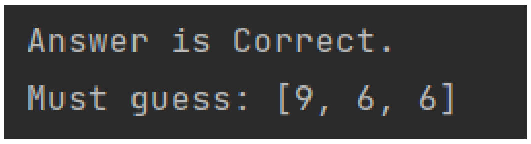

```
Enter Your Numbers: 1 2 3


----------------------------------------
Game Clues:
1 number must be higher. 2 is correct. 3 number must be higher.
----------------------------------------
You have 9 tries left.
----------------------------------------
Please guess 3 numbers between 1 to 10.
Type in this format: number + space + number + space + number
To exit, please type 'exit'.
Enter Your Numbers:
```

Correct Answer

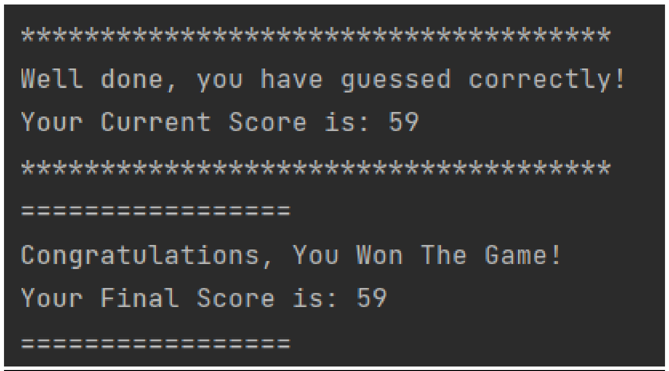

```
Enter Your Numbers: 6 2 7
*************************************
Well done, you have guessed correctly!
Your Current Score is: 9
*************************************
Please guess 3 numbers between 1 to 10.
Type in this format: number + space + number + space + number
To exit, please type 'exit'.
Enter Your Numbers:
```

Server Side – Generate New Numbers



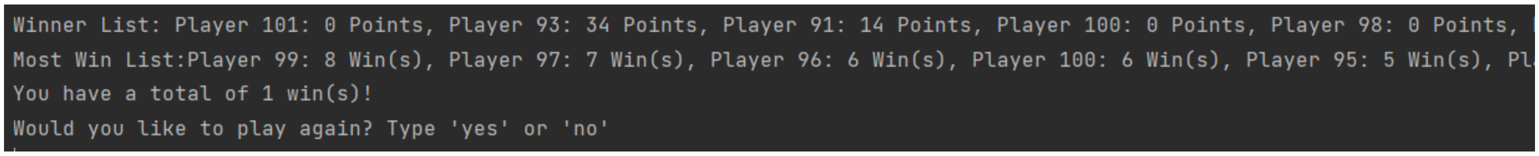
```
Answer is Correct.
Must guess: [9, 6, 6]
```

Game Won



```
*************************************
Well done, you have guessed correctly!
Your Current Score is: 59
*************************************
=================
Congratulations, You Won The Game!
Your Final Score is: 59
=================
```

“Winner” list and “Most Win” list is displayed & Player gets to decide to play again



```
Winner List: Player 101: 0 Points, Player 93: 34 Points, Player 91: 14 Points, Player 100: 0 Points, Player 98: 0 Points,
Most Win List:Player 99: 8 Win(s), Player 97: 7 Win(s), Player 96: 6 Win(s), Player 100: 6 Win(s), Player 95: 5 Win(s), Pl
You have a total of 1 win(s)!
Would you like to play again? Type 'yes' or 'no'
```

Implemented Functionality

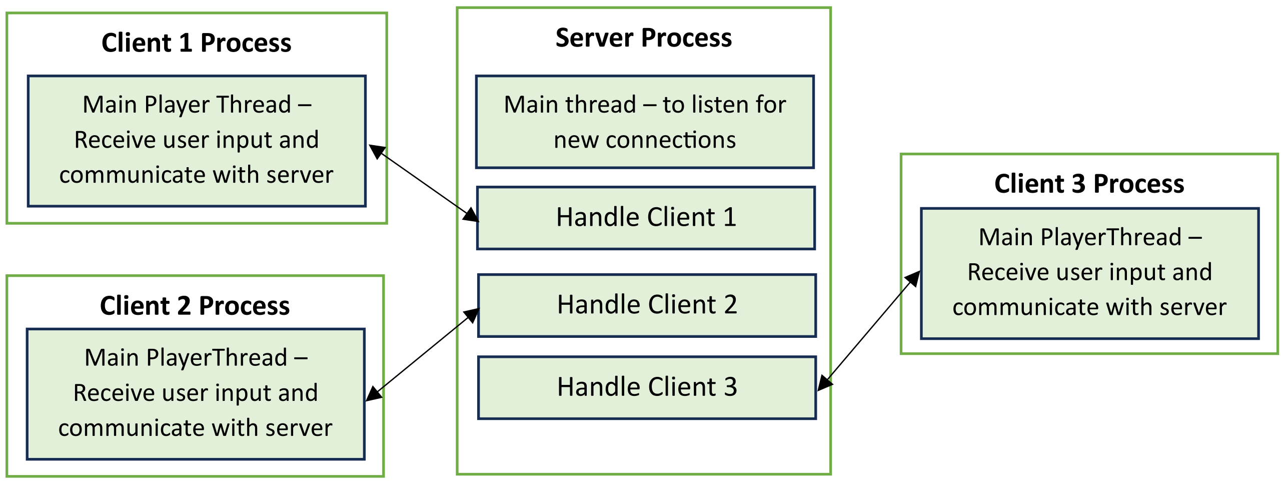
Functionality	Completed
Each client application is a player.	Yes
Once a player connects, the server will generate 3 numbers.	Yes
The clients can connect and disconnect from the server at any time.	Yes
Every time the client connects again, a new set of 3 numbers is generated to the client.	Yes
Each person has 10 guesses.	Yes
The number of points scored depends on the number of guesses left.	Yes
After a successful guess, the points are awarded, and the number of guesses is reset to 10 and a new game start.	Yes
Every unsuccessful guess would trigger the system to drop hints to the player on how many numbers are guessed correctly and how many are in the correct position, as well as the number of guesses used for this game.	Yes
New players can join the game at any time.	Yes
The number of players in the system is unlimited.	Yes
Every player joining the system must be assigned a unique ID that will not change until they leave the game and will not be reused after they leave the game.	Yes
The game will be continued to be played until a player accumulates > 50 points.	Yes
Once the first player reaches 50 points and above, all the points will be reset to zero and the server will record the player’s id.	Yes
Each client must display their own ID, information about all the players currently playing and the IDs of the players, and their current scores	Yes
The server will display the list of players who have previously won would be displayed.	Yes
The server will display a most wins list which show the top 10 winners with the most wins in descending order.	Yes
The server will display the top 3 winners with most frequencies of guessing the numbers in the correct order in a single guess.	No

Protocol

This is a client-server application using a text-based protocol. The client's can send a single text line, while the server sends a single text line or multiple lines.

Client response	**Server response**
	Creates socket and listen for clients `Socket socket = serverSocket.accept();`
Creates connection `Socket socket = new Socket("localhost", port);`	Sends client information: • List of current players • Assigned player ID • Player's current score
	Server generates 3 numbers for client.
<String> Input 3 numbers	Validate input format with regex Compare input with generated numbers
	If correct, send "true" If not correct, send over "feedback".
--- Client continue to input until guess is correct ---	
	<Integer> Sends player's score over and total number of wins
Client displays player's score and total number of wins	
	<String> Send list of players who won <String> Send list of top 10 winners
Client displays player's score, total number of wins and top 10 winners	
Client displays option to play again	
<String> Restart	
	Server loops back to the start

Client and Server Threads


Client 1 Process
Main Player Thread – Receive user input and communicate with server
Server Process
Main thread – to listen for new connections
Handle Client 1
Handle Client 2
Handle Client 3
Client 3 Process
Main PlayerThread – Receive user input and communicate with server
Client 2 Process
Main PlayerThread – Receive user input and communicate with server

Project review

This project was quite challenging as it involves communication (via text protocol) between two processes. I am happy that the game can generally function smoothly and meet almost all of the requirements.

The easy part of this project was developing the high-level structure of the program. This involves setting up the sockets, client handler and basic classes for the game, players and clients. One difficulty was having to constantly monitor messages sent across the server and client. If a message is sent wrongly, the program might stop or display an error. Sometimes, the messages might be sent wrongly to the client or server, which might result in a wrong calculation or wrong message displayed. Another difficulty that was faced was trying to allow the scanner.readline() to read completely multiple lines that was sent over. It was always able to read the lines, however it stops at the final line and hangs there. This difficulty was not resolved, and multiple lines were still connected into one single line.

The project management segment could have been improved. There was little version control conducted, which means that backtracking was not possible, and past work could not be retrieved (if needed).

Areas of improvement:

- Version control
- Complete delegation of communication to the Client class
- Implementing a graphical user interface