THE HANGMAN GAME REPORT

CPCS 371 Project

NOVEMBER 15, 2020

FIRAS MAHMOUD | 1845407
ABDULAZIZ AJAJ | 1854387

Contents

Group Project Rubric

Computer Networks-I (CPCS-371)

Computer Science Department, FCIT
King Abdulaziz University, Jeddah, KSA

Semester: ***Fall***

Member No.	***Student ID***	***Student Name***	***Section***
1	***1854387***	***Abdulaziz Ajaj***	***DT***
2	***1845407***	***Firas Mahmoud***	

					Student # 1 Marks	***Student # 2 Marks***
Program Solution [SO-C]	**Unacceptable (1):** An incomplete solution is implemented on the required platform. It does not compile and/or run.	**Poor (2):** A completed solution is implemented on the required platform and uses the compiler specified. It runs but has logical errors.	**Good (3):** A completed solution is tested and runs but does not meet all the specifications and/or work for all test data.	**Excellent (4):** A completed solution runs without errors. It meets all the specifications and works for all test data.	----------- / 4	----------- / 4
Program Design [SO-C]	**Unacceptable (1):** Few of the selected structures are appropriate. Program elements are not well designed.	**Poor (2):** Not all of the selected structures are appropriate. Some of the program elements are appropriately designed.	**Good (3):** The program design generally uses appropriate structures. Program elements exhibit good design.	**Excellent (4):** The program design uses appropriate structures. The overall program design is appropriate.	----------- / 4	----------- / 4
Team work / Participation	**Unacceptable (0):** Student did not participate in group work.	**Poor (0.25):** Student did not participated fully in group work. Student needed reminders from team member.	**Good (0.5):** Student mostly participated in group work. Student did what was required.	**Excellent (1):** Student fully participate in group work. Went above and beyond to help group members succeed.	----------- / 1	----------- / 1
Code Readability, Format and Style of report	**Unacceptable (0):** Insufficient program documentation, incorrect indentation, and/or poor identifier selection.	**Poor (0.25):** Program is minimally documented, some identifiers are inappropriate or inconsistent indentation.	**Good (0.5):** Some required documentation is missing, or identifiers are inappropriate, or statements are not indented correctly.	**Excellent (1):** All required documentation is present, the program is correctly indented, and appropriate identifiers are selected	----------- / 1	----------- / 1
				Total marks	----------- / 10	----------- / 10

Instructor Name: ___________________________ *Signature*: _____________________ Date: _____________________________

Introduction:

Firas and I created a hangman game over a TCP connection that allows a client to connect to a server and play hangman game. This report will detail the work of each class and utility we used in the program to accomplish this feat.

Classes and Utilities:

- Server:
  - In order for the server to run, we need to create a server socket via importing (java.net.ServerSocket) and, to teach it how to behave with this server socket, we also imported (java.net.Socket) to create a client socket. This way, we can connect these two sockets and create a TCP connection.
  - We then created a server socket and initiated it to port (5056), which it will accept client sockets' requests over.
  - We then run a loop that will keep the server listening on port (5056) for new clients' requests.
  - For multithreading, and handling many clients at once, we created and started a thread for every new client, and the server control will be passed to a class called "clientHandler".
  - The server class must run before clients can request to connect to it.

- Client:
  - The client must run after the server was started.
  - Every time we run the client class, a new client socket will be created, and it will have a local host IP address, and will try to connect to the server on port (5056).

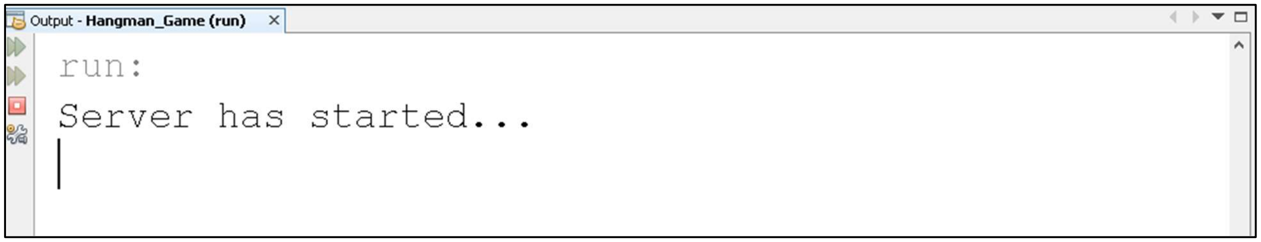
  - Every client is assigned a unique ID.
  - The client can handle receiving and sending messages to the server.

- Client Handler:
  - The client handler is the core of this multithreaded connection.
  - It will take a client's socket and will be able to send and receive messages from and to the client it is handling.
  - As a result, this client handler will listen to clients' requests. If this client sends "exit" at any point during its connection, the client handler will close this client's connection.
  - Otherwise, the client will play a hangman game through the client handler.

- Hangman Game:
  - This class is similar in every way to the client handler.
  - It will take a client's socket and will be able to send and receive messages from and to the client it is handling.
  - It will run a new hangman game per client's request.
  - Every client will be assigned a different hidden word, and a score that is independent of other clients' scores.
  - This class will keep running hangman game for every client that is playing and will receive guess letters and updates every client's hidden word until the client decides to exit.

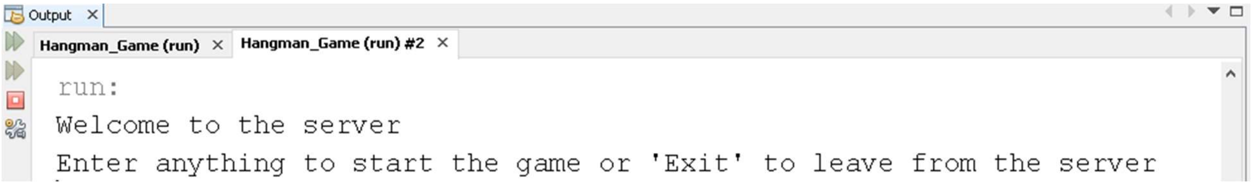
Sample Runs:

- First run the server



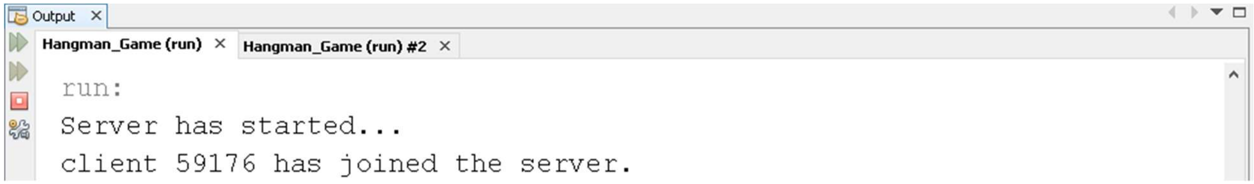
```
run:
Server has started...
```

- Then run the client to receive messages from the server



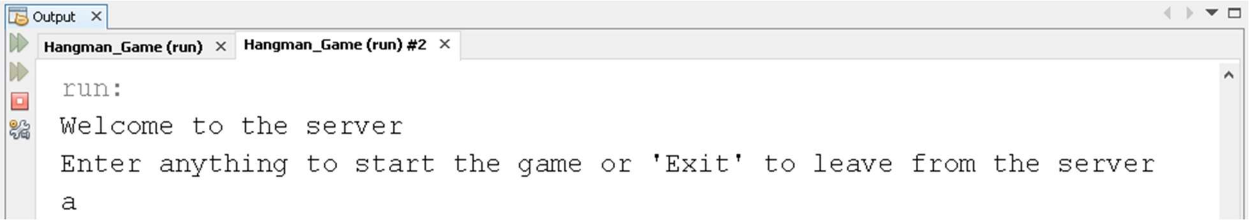
```
run:
Welcome to the server
Enter anything to start the game or 'Exit' to leave from the server
```

- Upon connection to the server, the server will confirm this client's connection:



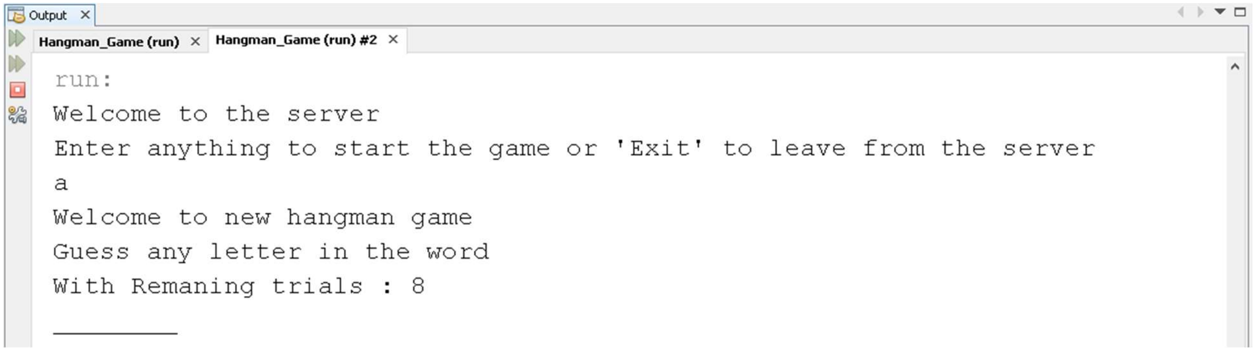
```
run:
Server has started...
client 59176 has joined the server.
```

- The client for example will send the letter “a” to start the game:



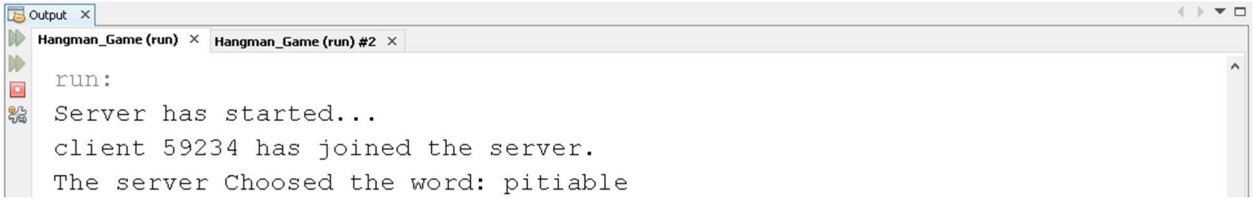
```
run:
Welcome to the server
Enter anything to start the game or 'Exit' to leave from the server
a
```

- The client now will be assigned a new word, score and number of trials:



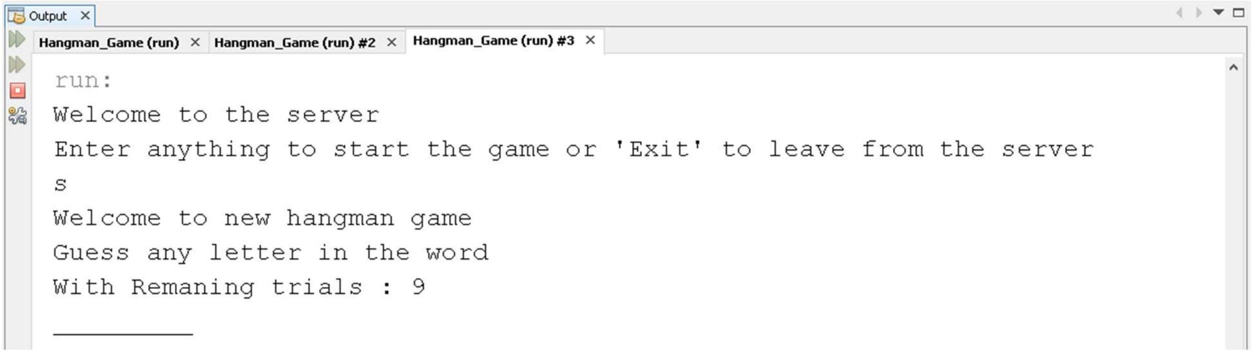
```
run:
Welcome to the server
Enter anything to start the game or 'Exit' to leave from the server
a
Welcome to new hangman game
Guess any letter in the word
With Remaning trials : 8
________
```

- Behind the scenes, the server will know which word this client was assigned:



```
run:
Server has started...
client 59234 has joined the server.
The server Choosed the word: pitiable
```

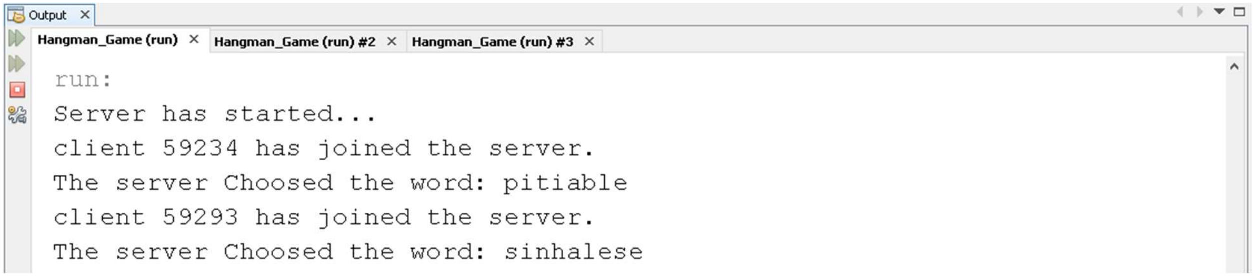
- Suppose another client connected to the server and wanted to play a game.



```
run:
Welcome to the server
Enter anything to start the game or 'Exit' to leave from the server
s
Welcome to new hangman game
Guess any letter in the word
With Remaning trials : 9

_________
```

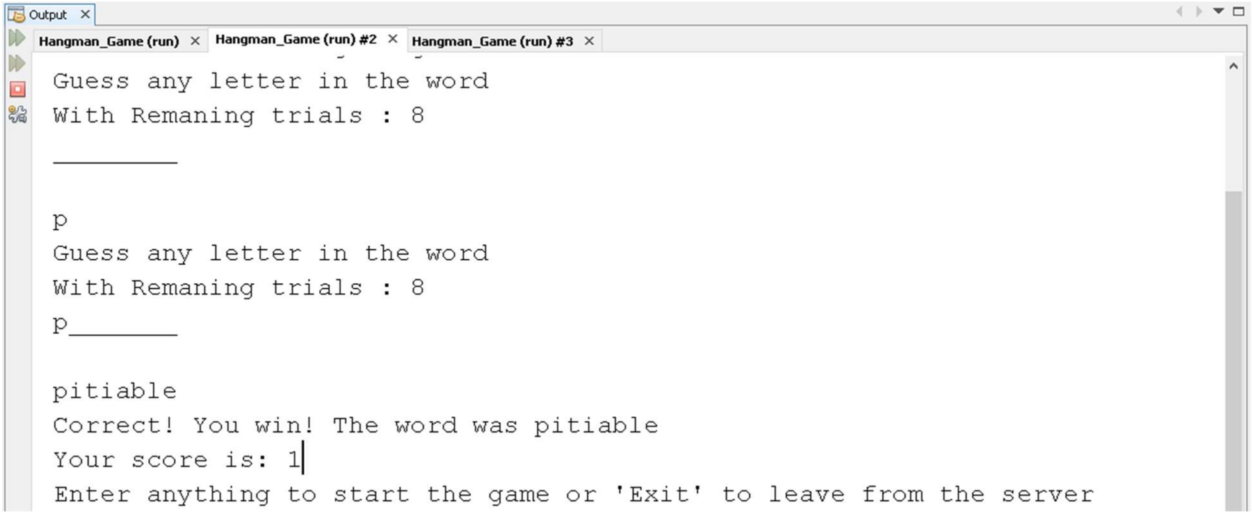
- The server will confirm its connection and assign it a new word with an independent number of attempts and score to the prior client that was connected:



```
run:
Server has started...
client 59234 has joined the server.
The server Choosed the word: pitiable
client 59293 has joined the server.
The server Choosed the word: sinhalese
```

- Each client will now be able to play the game independently and will have their score and attempts and words updated as they play.

- Client 1:

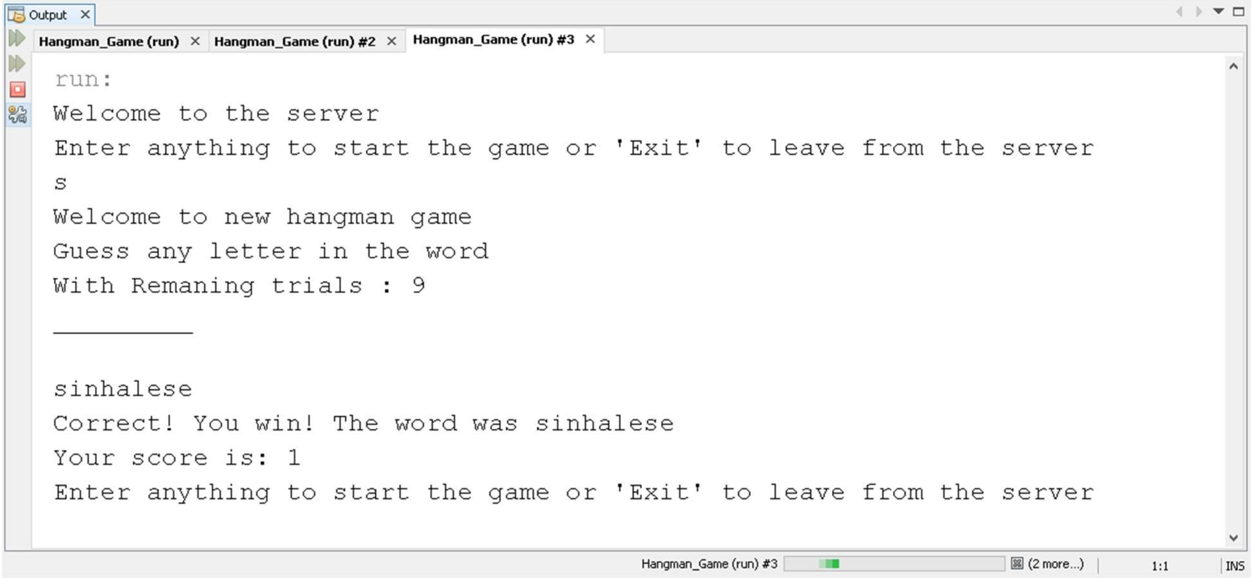


```
Guess any letter in the word
With Remaning trials : 8
________

p
Guess any letter in the word
With Remaning trials : 8
p_______

pitiable
Correct! You win! The word was pitiable
Your score is: 1
Enter anything to start the game or 'Exit' to leave from the server
```

- Client 2:



```
run:
Welcome to the server
Enter anything to start the game or 'Exit' to leave from the server
s
Welcome to new hangman game
Guess any letter in the word
With Remaning trials : 9
_________

sinhalese
Correct! You win! The word was sinhalese
Your score is: 1
Enter anything to start the game or 'Exit' to leave from the server
```