

# Assignment 2

## Operating System

Remote Method Invocation  
Client-Server

By:

**MHD Maher Azkoul**  
**438017578**  
**[s438017578@st.uqu.edu.sa](mailto:s438017578@st.uqu.edu.sa)**  
**course: operating systems, class: 3, no: 24**

Professor:

**pr. AbdulBaset Gaddah**

**Umm Al-Qura University**  
**College of Computers and Information Systems**  
**Operating Systems Course - 14012203 - 1441**

Nov 15, 2019

[Github Link](#)

Content:

[Part-1: Java RMI Lottery App](#)

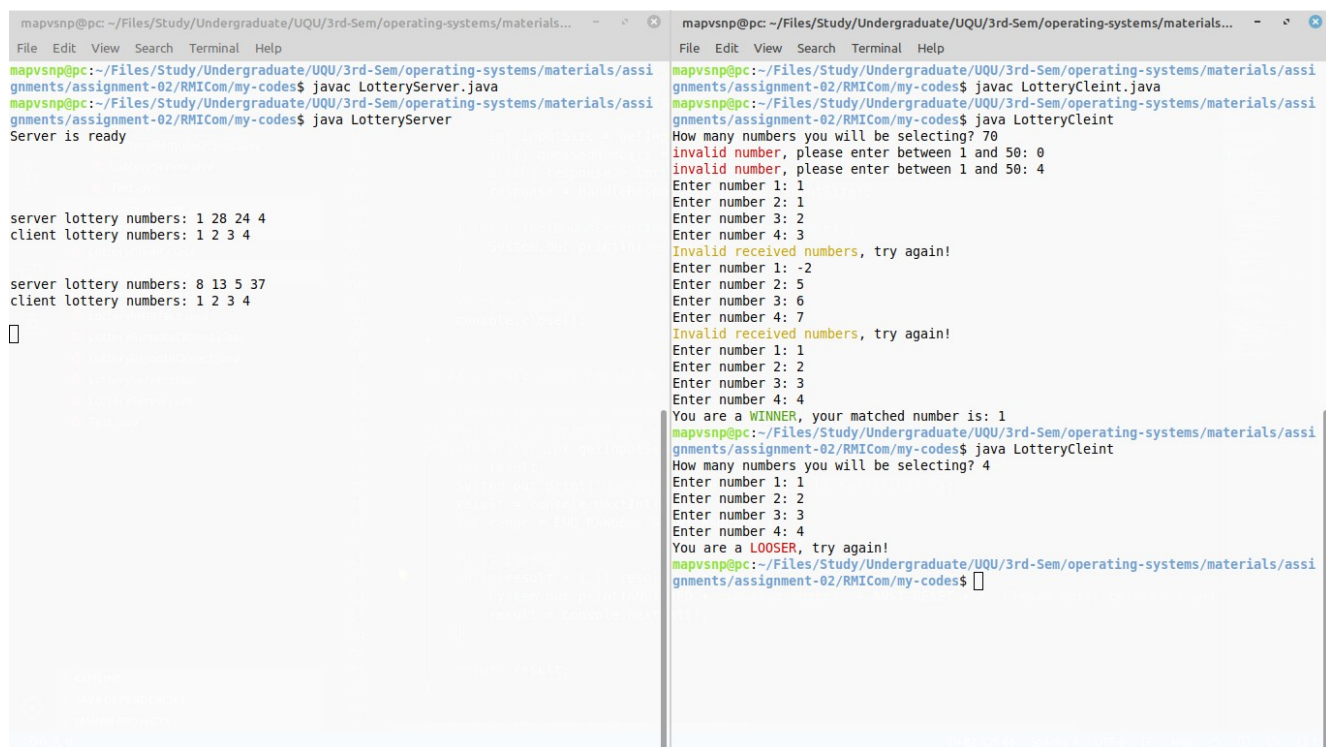
[Part-2: Client-Server Count Chars App](#)

## Part I - Java RMI Lottery Application:

Use Java RMI to write a simple client-server application to play a Lottery:

The goal of this program is a lottery game, it allows a client to guess numbers and enter a lottery to win or loose. It is developed with remote method invocation approach.

### Output Screenshot:



```
mapvsnp@pc: ~/Files/Study/Undergraduate/UQU/3rd-Sem/operating-systems/materials...
File Edit View Search Terminal Help
mapvsnp@pc:~/Files/Study/Undergraduate/UQU/3rd-Sem/operating-systems/materials/assig
nments/assignment-02/RMCom/my-codes$ javac LotteryServer.java
mapvsnp@pc:~/Files/Study/Undergraduate/UQU/3rd-Sem/operating-systems/materials/assig
nments/assignment-02/RMCom/my-codes$ java LotteryServer
Server is ready

server lottery numbers: 1 28 24 4
client lottery numbers: 1 2 3 4

server lottery numbers: 8 13 5 37
client lottery numbers: 1 2 3 4
□

mapvsnp@pc: ~/Files/Study/Undergraduate/UQU/3rd-Sem/operating-systems/materials...
File Edit View Search Terminal Help
mapvsnp@pc:~/Files/Study/Undergraduate/UQU/3rd-Sem/operating-systems/materials/assig
nments/assignment-02/RMCom/my-codes$ javac LotteryCleint.java
mapvsnp@pc:~/Files/Study/Undergraduate/UQU/3rd-Sem/operating-systems/materials/assig
nments/assignment-02/RMCom/my-codes$ java LotteryCleint
How many numbers you will be selecting? 70
invalid number, please enter between 1 and 50: 0
invalid number, please enter between 1 and 50: 4
Enter number 1: 1
Enter number 2: 1
Enter number 3: 2
Enter number 4: 3
Invalid received numbers, try again!
Enter number 1: -2
Enter number 2: 5
Enter number 3: 6
Enter number 4: 7
Invalid received numbers, try again!
Enter number 1: 1
Enter number 2: 2
Enter number 3: 3
Enter number 4: 4
You are a WINNER, your matched number is: 1
mapvsnp@pc:~/Files/Study/Undergraduate/UQU/3rd-Sem/operating-systems/materials/assig
nments/assignment-02/RMCom/my-codes$ java LotteryCleint
How many numbers you will be selecting? 4
Enter number 1: 1
Enter number 2: 2
Enter number 3: 3
Enter number 4: 4
You are a LOOSER, try again!
mapvsnp@pc:~/Files/Study/Undergraduate/UQU/3rd-Sem/operating-systems/materials/assig
nments/assignment-02/RMCom/my-codes$ □
```

## Explanation:

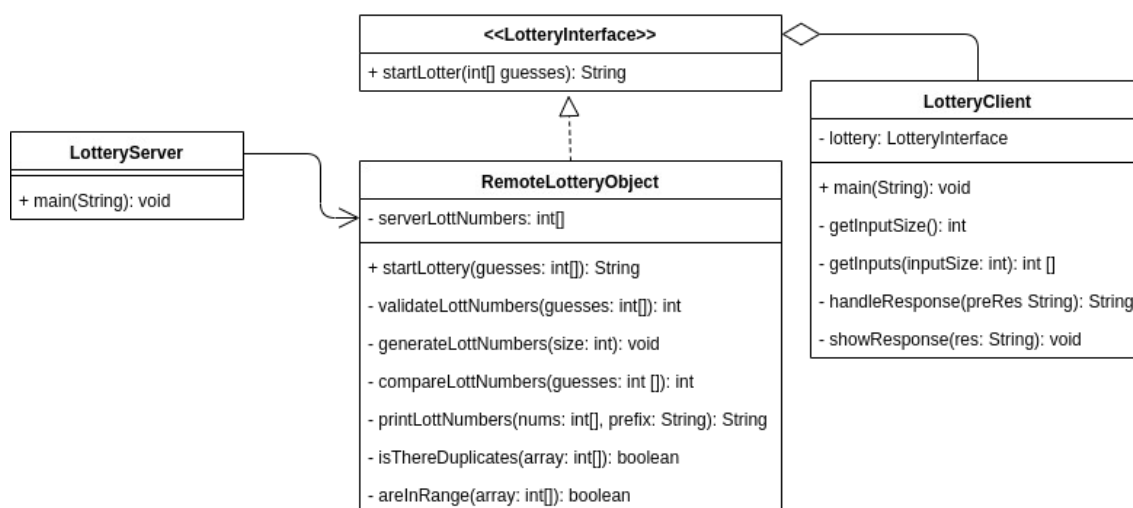
The whole program is divided to 3 parts:

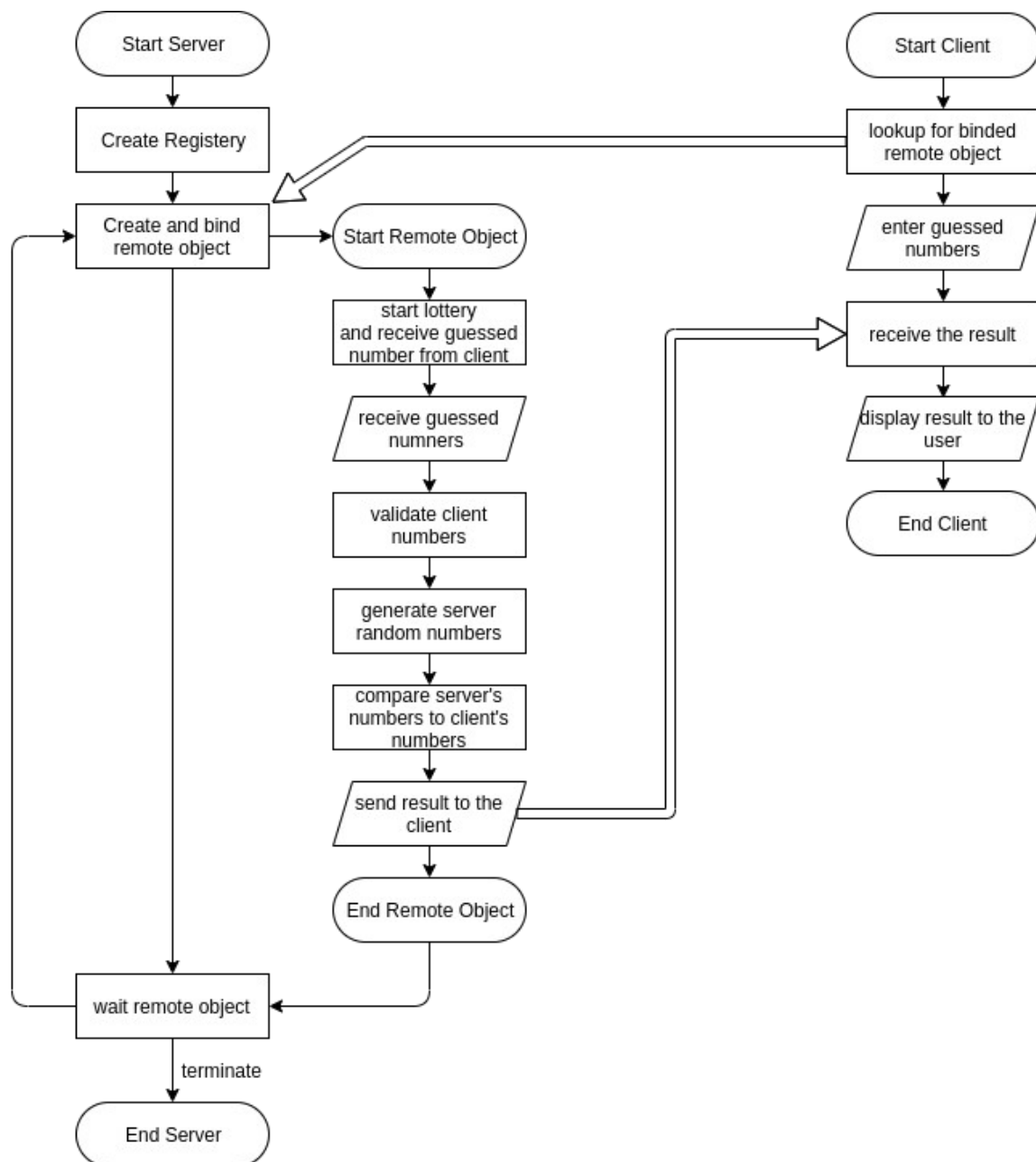
- **remote object**: contains the remote method and handle all the logic.
- **server**: create a registry and listen for clients, and bind the object with a key.
- **client**: interact with the user and communicate with the server.

First of all, the server create a registry start listening on a port, then it binds a key to the remote object that contains remote method.

Then the client lookup for the bonded remote object by the key specified by the server, and let the user send the guessed numbers.

The server receive a request to invoke the remote method, so it creates an instance of remote object, and the remote object will receive the numbers form the client, will validate them, generate lottery numbers, then compare generated numbers to client numbers, and finally send the result to the client.



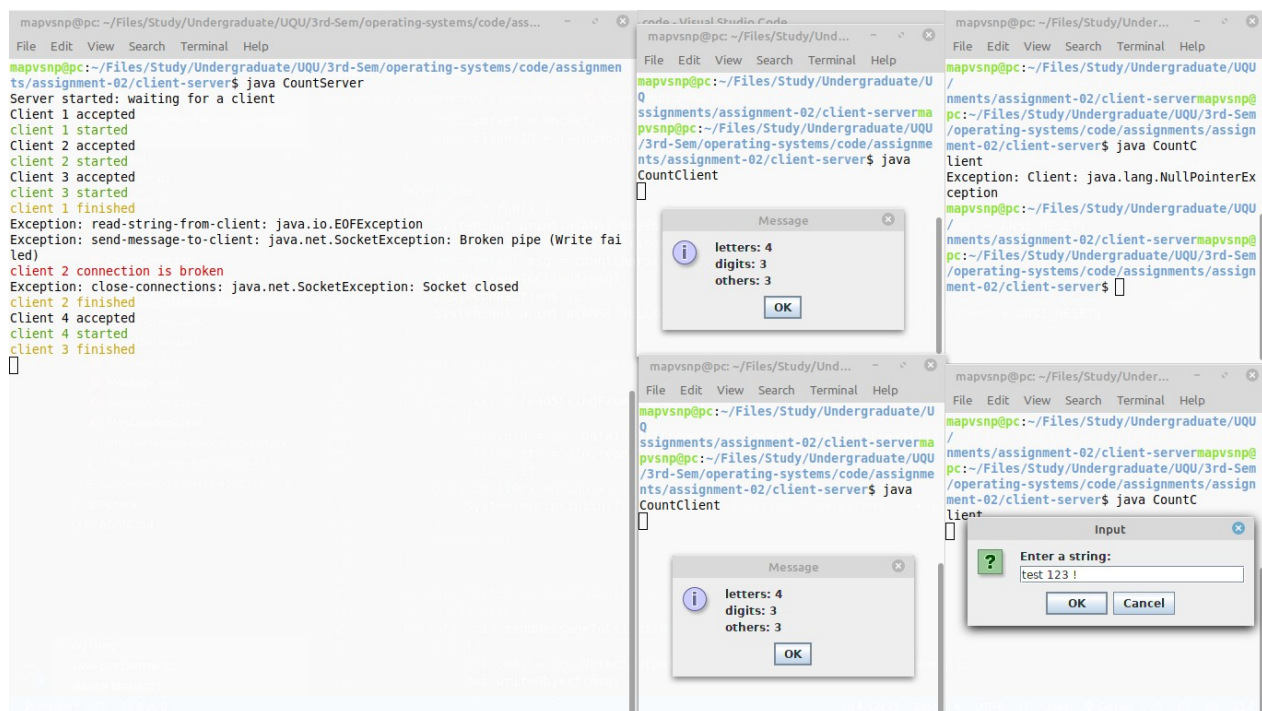


## Part II - Client-Server Count Characters Application:

The Write a client-server application using Java sockets that allows a client to write a message (as a String) to a socket. A server will read this message, count the number of letters, digits, and any other characters than letters or digits in the message, and then send these three counts back to the client:

The goal of this program allow client user to enter a text and analyze it, count the number of digits, number of letters and number of other chars. The program is divided into server and client. The server can handle multiple clients at the same time using threads.

### Output Screenshot:



## Explanation:

The whole program is divided to 3 parts:

- **count-client-handler**: a thread that handles all the logic of receiving text form the client, counting the characters, and sending the result back to the client.
- **server**: listens to a clients, and create a **count-client-handler** thread for each client.
- **client**: interact with the user and communicate with the server.

First of all, the server start listening for a connections with client. After that, a client can connect to the server. The server creates a thread to handle that connected client. The client asks the user to enter some text. The client sends that text to the server. The server (the client handler thread) receives the text, count the letters, digits and other characters, and send the result back to the client. The client displays the results to the user.

