VEERMATA JIJABAI TECHNOLOGICAL INSTITUTE

Matunga, Mumbai-400 019

Autonomous Institute affiliated to University of Mumbai

EXAMINATION	MST - March 2021	DATE OF EXAM	17th March 2021
SEMESTER & PROGRAM	Sem-I, First Year M Tech (Software Engineering)	TIME	12.00 pm to 01:00 pm
TIME ALLOWED	**1.0 HRS.**	MARKS	**20**
COURSE NAME – (CODE)	TCP/IP and Network Programming – (CO5063T)		

Instructions
1. All questions are compulsory.
2. Figures to the right indicate full marks.

Q.1 (a) Mr. GM owns a small medical laboratory at the ground floor of a building where there are 5 PCs in use - one at the reception, one in the supervisor's cabin, and 3 in the lab - forming one single network. (06M) [CO1]

Now Mr. GM wants to set up a new lab at the ground floor of the adjacent building with the same network setup having 5 PCs and the two networks should be connected with each other.

As a network engineer, you want to suggest to Mr. GM that the PCs in these two networks with similar usage should form one group.

So write your suggestion to Mr. GM by explaining the technique to be used for this purpose in brief.
Visualize the overall network and describe 2 benefits of it.

(b) Consider a LAN with the default gateway router 20.219.0.129 / 25. (04M) [CO2]
You will have to create either 16 subnets (if your roll number is odd) or 8 subnets (if your roll number is even).
Write IP address range of the first and the sixth subnet with slash notation.

Now consider one IP address from one of these subnets created by you.
The last byte of this IP address is calculated by adding 146 to the last two digits in your roll number. (e.g. if your roll number is 202190020 then the last byte of the IP address will be 20+146=166).

So write the IP address range of the subnet in which this IP address belongs to.

Q.2 Consider following programs: (GMServer.java and SEClient.java) (10M) [CO4]

(i) In the client program,
- write statement(s) to print the IP address of given server at the start of the try block.
- identify one bug and rewrite that statement.

(ii) Assuming that both the programs are present on the same machine,
- write the java command to run them and display their outputs.

(iii) In the server program, if the given port is not available then check for next 5 consecutive ports,
- rewrite the entire program to include this scenario.
(underline the statement(s) you add/modify).

```
import java.net.*;
import java.io.*;

public class GMServer {

  public static void main(String [] args) {
    int port = Integer.parseInt(args[0]);
    ServerSocket serverSocket;

    try {
      System.out.println("Opening connection on port -> " + port);
      serverSocket = new ServerSocket(port);
      System.out.println("Listening on port -> " +  serverSocket.getLocalPort() + "...");

      Socket s = serverSocket.accept();
      System.out.println("Connected to: " + s.getRemoteSocketAddress());

      DataInputStream inFromClient = new DataInputStream(s.getInputStream());
      System.out.println(inFromClient.readUTF());

      DataOutputStream outToClient = new DataOutputStream(s.getOutputStream());
      outToClient.writeUTF("Reply from server: " + s.getLocalSocketAddress());
      System.out.println("Reply sent to the client.");
      s.close();
    }
    catch (SocketException se) {
      System.out.println("Port <" + port + "> already in use...");
    }
    catch (IOException e) {
      e.printStackTrace();
    }
  }
}
```

```
import java.net.*;
import java.io.*;

public class SEClient {

  public static void main(String [] args) {
    String serverName = args[0];
    int serverPort = Integer.parseInt(args[1]);

    try {
      // To do => print IP address of the given server //
      System.out.println("Connecting to " + serverName + " on port -> " + serverPort);
      Socket clientSocket = new Socket(serverName);
      System.out.println("Connected to: " + clientSocket.getRemoteSocketAddress());

      OutputStream outToServer = clientSocket.getOutputStream();
      DataOutputStream out = new DataOutputStream(outToServer);
      out.writeUTF("Message from client: " + clientSocket.getLocalSocketAddress());
      System.out.println("Message sent to the server.");

      InputStream inFromServer = clientSocket.getInputStream();
      DataInputStream in = new DataInputStream(inFromServer);
      System.out.println(in.readUTF());
      clientSocket.close();
    }
    catch (IOException e) {
      e.printStackTrace();
    }
  }
}
```