

**Department of Computer Science and Engineering**

**21st Batch**

**Lab Report 1**

|  |  |
| --- | --- |
| Course title | : |
| Course Code | : |

|  |  |  |  |
| --- | --- | --- | --- |
| **Submitted By** | | **Submitted To** | |
|  |  |  |  |
| Name | : | Name | : |
| ID | : | Designation | : Lecturer,  Varendra University,  Rajshahi. |
| Section | : |  |
| Semester | : |  |
| Batch | : | Name | : |
|  |  | Designation | : Lecturer,  Varendra University,  Rajshahi. |
|  |  |  |

-----------

|  |  |
| --- | --- |
| Signature | Signature |
|  |  |

**Question:** Implementing server-client interaction.

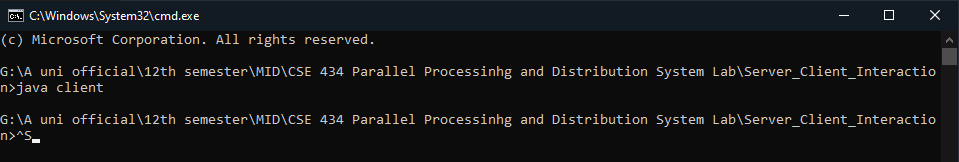
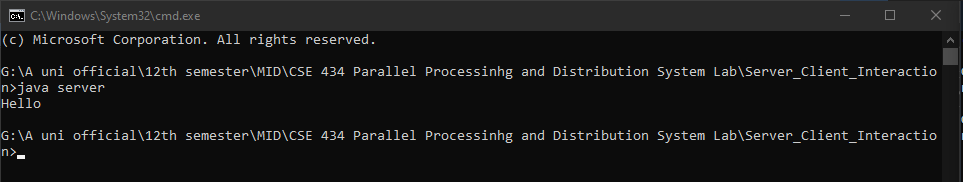
**Solution (Server):**

|  |
| --- |
| import java.net.\*;  import java.io.\*;  public class Server {      public static void main(String[] args) throws IOException {          ServerSocket serverSocket = new ServerSocket(1234);          Socket clientSocket = serverSocket.accept();          System.out.println("Client connected.");          PrintWriter out = new PrintWriter(clientSocket.getOutputStream(), true);          BufferedReader in = new BufferedReader(new InputStreamReader(clientSocket.getInputStream()));          String inputLine;          while ((inputLine = in.readLine()) != null) {              System.out.println("Client: " + inputLine);              out.println("Server received: " + inputLine);              if (inputLine.equals("the end")) {                  break;              }          }          out.close();          in.close();          clientSocket.close();          serverSocket.close();      }  } |

**Solution (Client):**

|  |
| --- |
| import java.net.\*;  import java.io.\*;  public class Client {      public static void main(String[] args) throws IOException {          Socket socket = new Socket("localhost", 1234);          System.out.println("Connected to server.");          PrintWriter out = new PrintWriter(socket.getOutputStream(), true);          BufferedReader in = new BufferedReader(new InputStreamReader(socket.getInputStream()));          BufferedReader stdin = new BufferedReader(new InputStreamReader(System.in));          String userInput;          while ((userInput = stdin.readLine()) != null) {              out.println(userInput);              String serverResponse = in.readLine();              System.out.println("Server: " + serverResponse);              if (userInput.equals("the end")) {                  break;              }          }          out.close();          in.close();          stdin.close();          socket.close();      }  } |

**Output:**

****

**Discussion:**

Implementing a basic server and client application is a fundamental task in computer networking. This involves creating two separate programs -one for the server and one for the client -that communicate with each other over a network connection. The server program is responsible for listening for incoming connections and processing requests from clients. It typically runs continuously in the background, waiting for requests to arrive. The client program, on the other hand, initiates the connection with the server and sends requests for information or services.