EXPERIMENT 2

CLIENT PROGRAM

```
import java.net.*;
import java.io.*;
import java.awt.image.*;
import javax.imageio.*;

public class Client {
   public static void main(String[] args) {
      Socket soc;
      BufferedImage img = null;

      try {

         img = ImageIO.read(new File("C:/Users/HP/Documents/digital_image_processing.jpg"));

         soc = new Socket("localhost", 4000);
         System.out.println("Client is running.");

         ByteArrayOutputStream baos = new ByteArrayOutputStream();
         ImageIO.write(img, "jpg", baos);
         baos.flush();
         byte[] bytes = baos.toByteArray();
         baos.close();

         OutputStream out = soc.getOutputStream();
         DataOutputStream dos = new DataOutputStream(out);
         dos.writeInt(bytes.length);
         dos.write(bytes, 0, bytes.length);

         System.out.println("Image sent to server.");
         dos.close();
         out.close();
         soc.close();
      } catch (IOException e) {
         System.out.println("Exception: " + e.getMessage());
      }
   }
}
```

SERVER PROGRAM

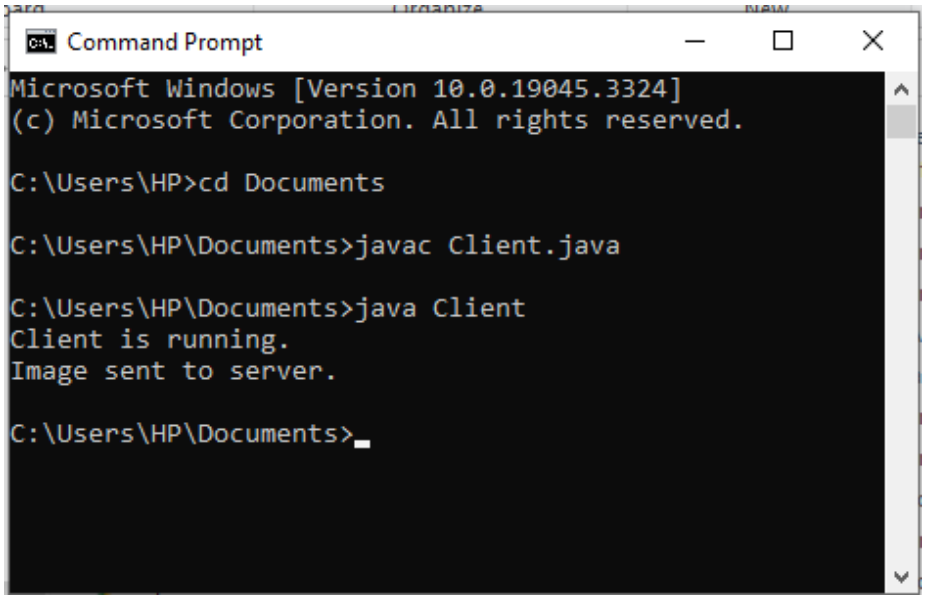
```
import java.net.*;
import java.io.*;
import java.awt.image.*;
import javax.imageio.*;
import javax.swing.*;
```

```
class Server {
    public static void main(String args[]) throws Exception {
        ServerSocket server = null;
        Socket socket = null;

        try {
            // Create a server socket listening on port 4000
            server = new ServerSocket(4000);
            System.out.println("Server waiting for image...");

            // Accept the connection from the client
            socket = server.accept();
            System.out.println("Client connected.");

            // Create input streams to receive the image
            InputStream in = socket.getInputStream();
            DataInputStream dis = new DataInputStream(in);

            // Read the size of the incoming image data
            int len = dis.readInt();
            System.out.println("Image Size: " + len / 1024 + "KB");

            // Create a byte array to hold the image data and read it from the input stream
            byte[] data = new byte[len];
            dis.readFully(data);

            // Convert the byte array back into a BufferedImage
            InputStream ian = new ByteArrayInputStream(data);
            BufferedImage bImage = ImageIO.read(ian);

            // Display the received image in a JFrame
            JFrame f = new JFrame("Server");
            ImageIcon icon = new ImageIcon(bImage);
            JLabel l = new JLabel();
            l.setIcon(icon);
            f.add(l);
            f.pack();
            f.setVisible(true);

            // Close the input streams
            dis.close();
            in.close();
        } catch (Exception e) {
            System.out.println("Exception: " + e.getMessage());
        } finally {
            // Close the socket and server socket
            if (socket != null) {
                socket.close();

            }
```

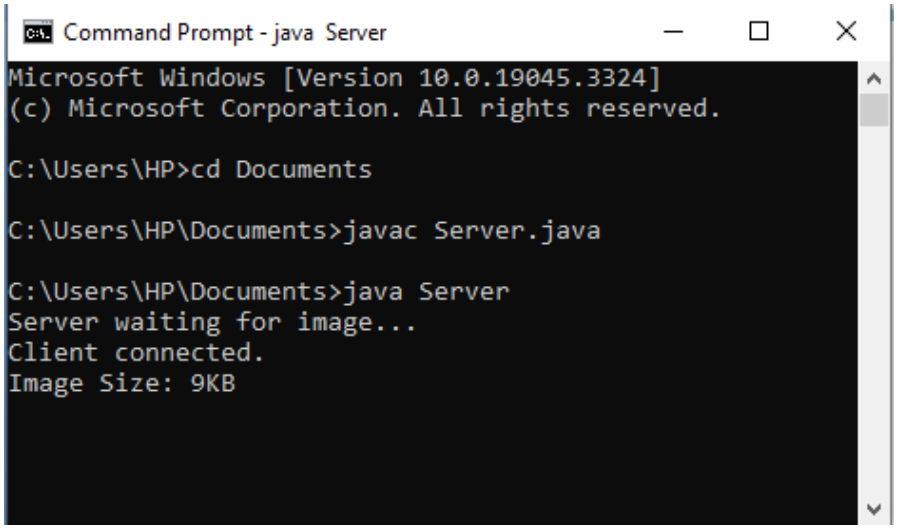
```
        if (server != null) {
            server.close();
        }
    }
  }
}
```

OUTPUT

CLIENT



SERVER



Client program:

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

public class SimpleHttpClient {
   public static void main(String[] args) {
      String host = "example.com"; // The host you're connecting to
      int port = 80; // HTTP port

      try (Socket socket = new Socket(host, port)) {
         // Send HTTP GET request
         PrintWriter out = new PrintWriter(socket.getOutputStream(),
true);
         out.println("GET / HTTP/1.1");
         out.println("Host: " + host);
         out.println("Connection: Close");
         out.println(); // Blank line to indicate end of headers

         // Read the response
         BufferedReader in = new BufferedReader(new
InputStreamReader(socket.getInputStream()));
         String responseLine;
         while ((responseLine = in.readLine()) != null) {
            System.out.println(responseLine);
```

```
        }
    } catch (IOException e) {
        e.printStackTrace();
    }
  }
}
```

Output:

Command Prompt

```
Microsoft Windows [Version 10.0.19045.3324]
(c) Microsoft Corporation. All rights reserved.

C:\Users\HP>cd Documents

C:\Users\HP\Documents>javac SimpleHttpClient.java

C:\Users\HP\Documents>java SimpleHttpClient
HTTP/1.1 200 OK
Accept-Ranges: bytes
Age: 414246
Cache-Control: max-age=604800
Content-Type: text/html; charset=UTF-8
Date: Tue, 20 Aug 2024 13:47:12 GMT
Etag: "3147526947"
Expires: Tue, 27 Aug 2024 13:47:12 GMT
Last-Modified: Thu, 17 Oct 2019 07:18:26 GMT
Server: ECAcc (lac/55DF)
Vary: Accept-Encoding
X-Cache: HIT
Content-Length: 1256
Connection: close

<!doctype html>
<html>
<head>
    <title>Example Domain</title>

    <meta charset="utf-8" />
    <meta http-equiv="Content-type" content="text/html; charset=utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1" />
    <style type="text/css">
```

Command Prompt

```
    <meta charset="utf-8" />
    <meta http-equiv="Content-type" content="text/html; charset=utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1" />
    <style type="text/css">
    body {
        background-color: #f0f0f2;
        margin: 0;
        padding: 0;
        font-family: -apple-system, system-ui, BlinkMacSystemFont, "Segoe UI", "Open Sans", "Helvetica Neue", Helvetica, Arial, sa
ns-serif;

    }
    div {
        width: 600px;
        margin: 5em auto;
        padding: 2em;
        background-color: #fdfdff;
        border-radius: 0.5em;
        box-shadow: 2px 3px 7px 2px rgba(0,0,0,0.02);
    }
    a:link, a:visited {
        color: #38488f;
        text-decoration: none;
    }
    @media (max-width: 700px) {
        div {
            margin: 0 auto;
            width: auto;
        }
    }
    </style>
</head>

<body>
<div>
    <h1>Example Domain</h1>
    <p>This domain is for use in illustrative examples in documents. You may use this
    domain in literature without prior coordination or asking for permission.</p>
    <p><a href="https://www.iana.org/domains/example">More information...</a></p>
</div>
</body>
</html>
```

EXPERIMENT 3(a) echo

CLIENT PROGRAM

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

public class EClient {
   public static void main(String[] args) {
      try {
         Socket socket = new Socket("localhost", 9000);
         BufferedReader userInput = new BufferedReader(new
InputStreamReader(System.in));
         PrintWriter out = new PrintWriter(socket.getOutputStream(), true);
         BufferedReader in = new BufferedReader(new
InputStreamReader(socket.getInputStream()));

         String line;
```

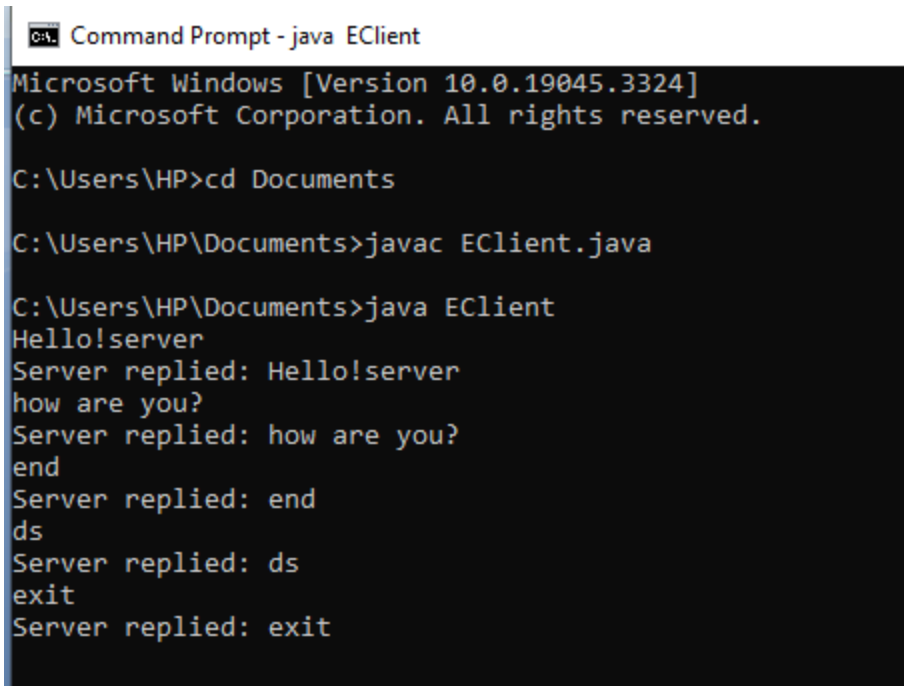
```
            while ((line = userInput.readLine()) != null) {
                out.println(line);
                System.out.println("Server replied: " + in.readLine());
            }

            socket.close();
        } catch (IOException e) {
            System.err.println("IOException: " + e.getMessage());
        }
    }
}
```

SERVER PROGRAM

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

public class EServer {
    public static void main(String[] args) {
        ServerSocket serverSocket = null;
        Socket clientSocket = null;
        DataInputStream inputStream = null;
        PrintStream outputStream = null;

        try {
            // Create a ServerSocket to listen on port 9000
            serverSocket = new ServerSocket(9000);
            System.out.println("Server is listening on port 9000...");

            // Accept a connection from the client
            clientSocket = serverSocket.accept();
            System.out.println("Client connected.");

            // Create input and output streams
            inputStream = new DataInputStream(clientSocket.getInputStream());
            outputStream = new PrintStream(clientSocket.getOutputStream());

            // Read lines from the client and echo them back
```

```
            String line;
            while ((line = inputStream.readLine()) != null) {
                outputStream.println(line);
                System.out.println("Received and sent back: " + line);
            }
        } catch (IOException e) {
            System.err.println("IOException: " + e.getMessage());
        } finally {
            // Close all resources
            try {
                if (inputStream != null) inputStream.close();
                if (outputStream != null) outputStream.close();
                if (clientSocket != null) clientSocket.close();
                if (serverSocket != null) serverSocket.close();
            } catch (IOException e) {
                System.err.println("Error closing resources: " + e.getMessage());
            }
        }
    }
}
```

OUTPUT



```
Command Prompt - java EServer
Microsoft Windows [Version 10.0.19045.3324]
(c) Microsoft Corporation. All rights reserved.

C:\Users\HP>cd Documents

C:\Users\HP\Documents>javac EServer.java
Note: EServer.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.

C:\Users\HP\Documents>java EServer
Server is listening on port 9000...
Client connected.
Received and sent back: Hello!server
Received and sent back: how are you?
Received and sent back: end
Received and sent back: ds
Received and sent back: exit
```

Experiment 3(b)-chat

Client program

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

class UDPclient {
   public static DatagramSocket ds;
   public static int clientport = 789, serverport = 790;

   public static void main(String args[]) throws Exception {
      byte buffer[] = new byte[1024];
      ds = new DatagramSocket(serverport);
      BufferedReader dis = new BufferedReader(new
InputStreamReader(System.in));
      System.out.println("Server waiting");
      InetAddress ia = InetAddress.getLocalHost();

      while (true) {
         System.out.print("Client: ");
         String str = dis.readLine();
```

```
        if (str.equals("end"))
            break;
        buffer = str.getBytes();
        ds.send(new DatagramPacket(buffer, str.length(), ia, clientport));
        DatagramPacket p = new DatagramPacket(buffer, buffer.length);
        ds.receive(p);
        String psx = new String(p.getData(), 0, p.getLength());
        System.out.println("Server: " + psx);
    }
    ds.close();
  }
}
```

Server program

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

class UDPserver {
  public static DatagramSocket ds;
  public static byte buffer[] = new byte[1024];
  public static int clientport = 789, serverport = 790;

  public static void main(String args[]) throws Exception {
    ds = new DatagramSocket(clientport);
    System.out.println("Press Ctrl+C to quit the program");
    BufferedReader dis = new BufferedReader(new
InputStreamReader(System.in));
    InetAddress ia = InetAddress.getLocalHost();

    while (true) {
        DatagramPacket p = new DatagramPacket(buffer, buffer.length);
        ds.receive(p);
        String psx = new String(p.getData(), 0, p.getLength());
        System.out.println("Client: " + psx);
        System.out.print("Server: ");
        String str = dis.readLine();
```

```
        if (str.equals("end"))
            break;
        buffer = str.getBytes();
        ds.send(new DatagramPacket(buffer, str.length(), ia, serverport));
    }
    ds.close();
  }
}
```

output

Command Prompt

```
^C
C:\Users\HP\Documents>javac UDPclient.java

C:\Users\HP\Documents>java UDPclient
Server waiting
Client: hello
Server: HI,ho
Client: fine
Server: grea
Client: what are you doing?
Server: am watching movie
Client: end

C:\Users\HP\Documents>_
```

Select Command Prompt - java UDPserver

```
Server: en^C
C:\Users\HP\Documents>javac UDPserver.java

C:\Users\HP\Documents>java UDPserver
Press Ctrl+C to quit the program
Client: hello
Server: HI,how are you?
Client: fine
Server: great
Client: what
Server: am watching movie
```