

Chapter 2: InetAddress

Lab 1

Objective: To check whether address is IPv4 or IPv6.

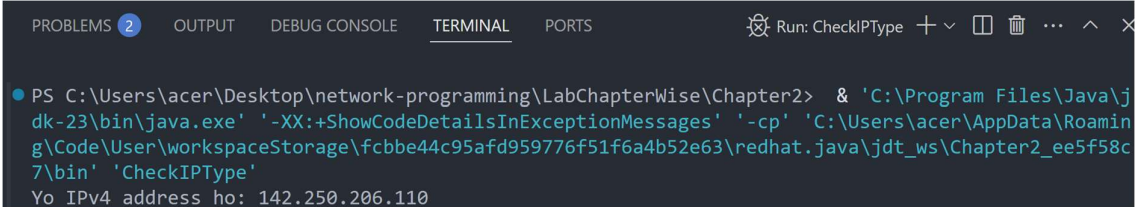
Source Code:

```
import java.net.*;

public class CheckIPType {
    public static void main(String[] args) {
        try {
            // InetAddress object banaune (example: google.com ko lagi)
            InetAddress address = InetAddress.getByName("google.com");

            // Address ko type check garne
            if (address instanceof Inet4Address) {
                System.out.println("Yo IPv4 address ho: " + address.getHostAddress());
            } else if (address instanceof Inet6Address) {
                System.out.println("Yo IPv6 address ho: " + address.getHostAddress());
            } else {
                System.out.println("Unknown address type: " + address.getHostAddress());
            }
        } catch (Exception e) {
            System.out.println("Error aayo: " + e.getMessage());
        }
    }
}
```

Output:



```
PS C:\Users\acer\Desktop\network-programming\LabChapterWise\Chapter2> & 'C:\Program Files\Java\jdk-23\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\acer\AppData\Roaming\Code\User\workspaceStorage\fcbbe44c95afd959776f51f6a4b52e63\redhat.java\jdt_ws\Chapter2_ee5f58c7\bin' 'CheckIPType'
Yo IPv4 address ho: 142.250.206.110
```

Lab 2

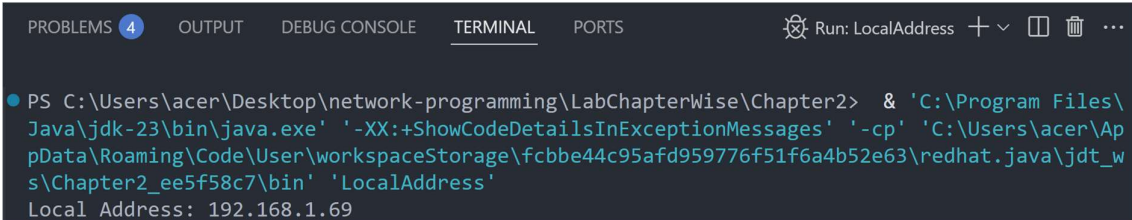
Objective: To find the address of local machine.

Source Code:

```
import java.net.*;

public class LocalAddress {
    public static void main(String[] args) {
        try {
            // Local machine ko address lina
            InetAddress local = InetAddress.getLocalHost();
            System.out.println("Local Address: " + local.getHostAddress());
        } catch (Exception e) {
            System.out.println("Error aayo: " + e.getMessage());
        }
    }
}
```

Output:



```
PROBLEMS 4 OUTPUT DEBUG CONSOLE TERMINAL PORTS Run: LocalAddress + - ...
PS C:\Users\acer\Desktop\network-programming\LabChapterWise\Chapter2> & 'C:\Program Files\Java\jdk-23\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\acer\AppData\Roaming\Code\User\workspaceStorage\fcbb44c95afd959776f51f6a4b52e63\redhat.java\jdt_ws\Chapter2_ee5f58c7\bin' 'LocalAddress'
Local Address: 192.168.1.69
```

Lab 3

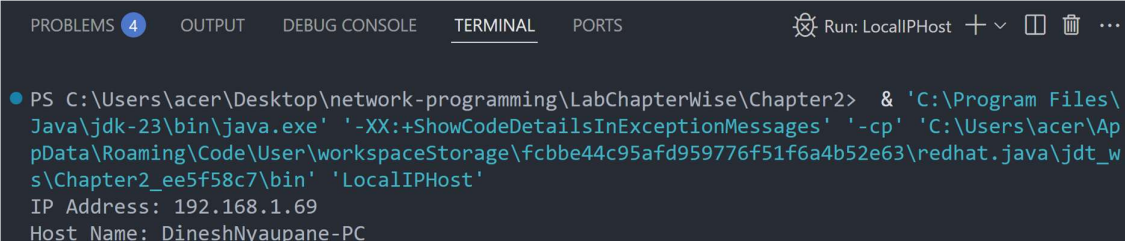
Objective: To find the IP Address and host name of local machine.

Source Code:

```
import java.net.*;

public class LocalIPHost {
    public static void main(String[] args) {
        try {
            // Local machine ko IP ra Hostname print garne
            InetAddress local = InetAddress.getLocalHost();
            System.out.println("IP Address: " + local.getHostAddress());
            System.out.println("Host Name: " + local.getHostName());
        } catch (Exception e) {
            System.out.println("Error aayo: " + e.getMessage());
        }
    }
}
```

Output:



```
PROBLEMS 4 OUTPUT DEBUG CONSOLE TERMINAL PORTS Run: LocalIPHost + - [ ] ...
● PS C:\Users\acer\Desktop\network-programming\LabChapterWise\Chapter2> & 'C:\Program Files\
Java\jdk-23\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\acer\AppData\Roaming\Code\User\workspaceStorage\fcbb44c95afd959776f51f6a4b52e63\redhat.java\jdt_ws\Chapter2_ee5f58c7\bin' 'LocalIPHost'
IP Address: 192.168.1.69
Host Name: DineshNyaupane-PC
```

Lab 4

Objective: To demonstrate the SpamCheck.

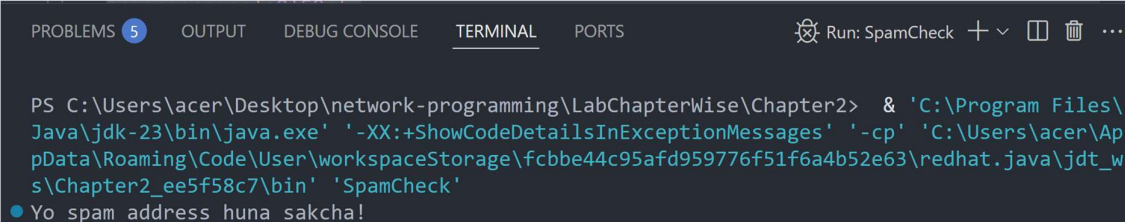
Source Code:

```
import java.net.*;

public class SpamCheck {
    public static void main(String[] args) {
        try {
            // Domain name ko address khojne
            InetAddress address = InetAddress.getByName("www.google.com");

            // Hostname spam chaina bhane reply garne
            if (!address.isAnyLocalAddress()) {
                System.out.println("Yo spam address huna sakcha!");
            } else {
                System.out.println("Yo safe address ho.");
            }
        } catch (Exception e) {
            System.out.println("Error aayo: " + e.getMessage());
        }
    }
}
```

Output:



The screenshot shows an IDE interface with a terminal window. The terminal title bar says "Run: SpamCheck". The command prompt shows the execution of the Java program with various flags. The output of the program is displayed in the terminal.

```
PS C:\Users\acer\Desktop\network-programming\LabChapterWise\Chapter2> & 'C:\Program Files\Java\jdk-23\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\acer\AppData\Roaming\Code\User\workspaceStorage\fcbb44c95afd959776f51f6a4b52e63\redhat.java\jdt_ws\Chapter2_ee5f58c7\bin' 'SpamCheck'
● Yo spam address huna sakcha!
```

Lab 5

Objective: To compare the domain names “*www.ibiblio.org*” and “*helios.ibiblio.org*”.

Source Code:

```
import java.net.*;

public class CompareDomain {

    public static void main(String[] args) {

        try {

            // Duita domain ko IP address khojne

            InetAddress addr1 = InetAddress.getByName("www.ibiblio.org");

            InetAddress addr2 = InetAddress.getByName("helios.ibiblio.org");

            // Address compare garne

            if (addr1.equals(addr2)) {

                System.out.println("Domain haru same address ma point garcha.");

            } else {

                System.out.println("Domain haru different address ma chha.");

            }

        } catch (Exception e) {

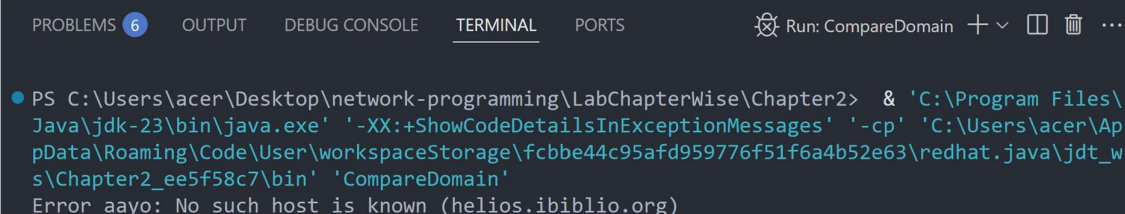
            System.out.println("Error aayo: " + e.getMessage());

        }

    }

}
```

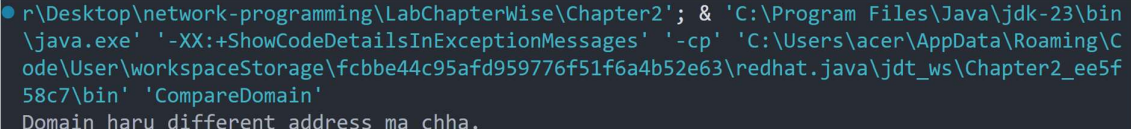
Output:



```
PROBLEMS 6 OUTPUT DEBUG CONSOLE TERMINAL PORTS Run: CompareDomain + - □ × ...

● PS C:\Users\acer\Desktop\network-programming\LabChapterWise\Chapter2> & 'C:\Program Files\Java\jdk-23\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\acer\AppData\Roaming\Code\User\workspaceStorage\fcbb44c95afd959776f51f6a4b52e63\redhat.java\jdt_ws\Chapter2_ee5f58c7\bin' 'CompareDomain'
Error aayo: No such host is known (helios.ibiblio.org)
```

If we check www.google.com and www.youtube.com :



```
● r\Desktop\network-programming\LabChapterWise\Chapter2'; & 'C:\Program Files\Java\jdk-23\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\acer\AppData\Roaming\Code\User\workspaceStorage\fcbb44c95afd959776f51f6a4b52e63\redhat.java\jdt_ws\Chapter2_ee5f58c7\bin' 'CompareDomain'
Domain haru different address ma chha.
```

Chapter 3: URLs and URIs

Lab 6

Objective: To write a program that splits the parts of a URL [Splitting URL into pieces information]

Source Code:

```
import java.net.URL;

public class SplitURL {
    public static void main(String[] args) {
        try {
            URL url = new
            URL("https://www.example.com:8080/path/to/resource?query=nepal#section");

            System.out.println("Protocol: " + url.getProtocol());

            System.out.println("Host: " + url.getHost());

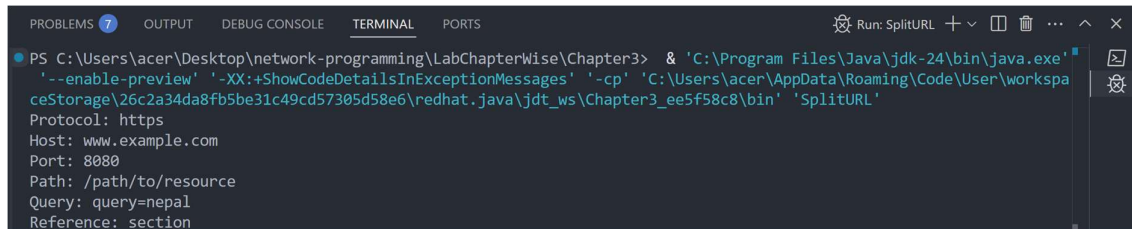
            System.out.println("Port: " + url.getPort());

            System.out.println("Path: " + url.getPath());

            System.out.println("Query: " + url.getQuery());

            System.out.println("Reference: " + url.getRef());
        } catch (Exception e) {
            System.out.println("URL ko parts split garna error: " + e);
        }
    }
}
```

Output:

A screenshot of a terminal window with a dark background. The terminal shows the command prompt 'PS C:\Users\acer\Desktop\network-programming\LabChapterWise\Chapter3>' followed by the command to run the Java program. The output of the program is displayed line by line: 'Protocol: https', 'Host: www.example.com', 'Port: 8080', 'Path: /path/to/resource', 'Query: query=nepal', and 'Reference: section'. The terminal window has tabs for 'PROBLEMS', 'OUTPUT', 'DEBUG CONSOLE', 'TERMINAL', and 'PORTS'. The 'TERMINAL' tab is active. There are also icons for running, stopping, and refreshing the terminal output.

Lab 7

Objective: To write a program that checks the which protocols does a virtual machine support or not?

Source Code:

```
import java.net.URLConnection;

import java.util.Arrays;

import java.util.List;

public class CheckProtocols {

    public static void main(String[] args) {

        List<String> protocols = Arrays.asList("http", "https", "ftp", "file", "mailto");

        for (String protocol : protocols) {

            try {

                URLConnection connection = new java.net.URL(protocol +

                "://test.com").openConnection();

                System.out.println(protocol + " is supported.");

            } catch (Exception e) {

                System.out.println(protocol + " is NOT supported.");

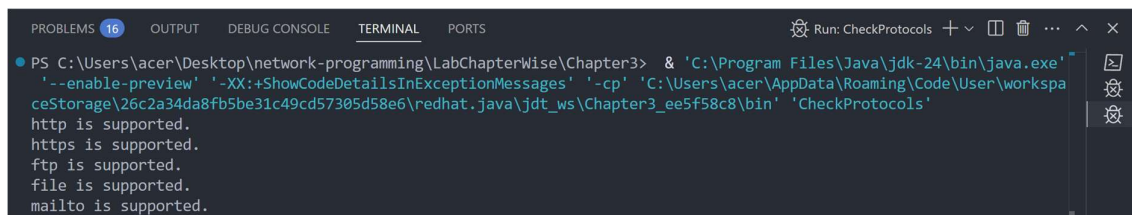
            }

        }

    }

}
```

Output:



```
PS C:\Users\acer\Desktop\network-programming\LabChapterWise\Chapter3> & 'C:\Program Files\Java\jdk-24\bin\java.exe'
'--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\acer\AppData\Roaming\Code\User\workspa
ceStorage\26c2a34da8fb5be31c49cd57305d58e6\redhat.java\jdt_ws\Chapter3_ee5f58c8\bin' 'CheckProtocols'
http is supported.
https is supported.
ftp is supported.
file is supported.
mailto is supported.
```


Lab 8

Objective: To write a program to download a web page of a given web address.

Source Code:

```
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.net.URL;
public class DownloadWebPage {
    public static void main(String[] args) {
        try {
            URL url = new URL("https://example.com");
            BufferedReader reader = new BufferedReader(new
InputStreamReader(url.openStream()));
            String line;
            while ((line = reader.readLine()) != null) {
                System.out.println(line);
            }
            reader.close();
        } catch (Exception e) {
            System.out.println("Webpage download garna error: " + e);
        }
    }
}
```

Output:



```
PS C:\Users\acer\Desktop\network-programming\LabChapterWise\Chapter3> & 'C:\Program Files\Java\jdk-24\bin\java.exe'
'--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\acer\AppData\Roaming\Code\User\workspa
ceStorage\26c2a34da8fb5be31c49cd57305d58e6\redhat.java\jdt_ws\Chapter3_ee5f58c8\bin' 'DownloadWebPage'
<!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">
    body {
      background-color: #f0f0f2;
      margin: 0;
      padding: 0;
      font-family: -apple-system, system-ui, BlinkMacSystemFont, "Segoe UI", "Open Sans", "Helvetica Neue", Helvet
ica, Arial, sans-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>
```


Lab 9

Objective: To write a program for resolving relatives URI

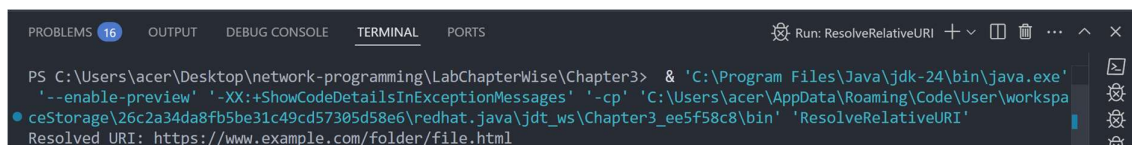
Source Code:

```
import java.net.URI;

public class ResolveRelativeURI {
    public static void main(String[] args) {
        try {
            URI base = new URI("https://www.example.com/folder/");
            URI relative = new URI("file.html");
            URI resolved = base.resolve(relative);

            System.out.println("Resolved URI: " + resolved.toString());
        } catch (Exception e) {
            System.out.println("Relative URI resolve garna error: " + e);
        }
    }
}
```

Output:



```
PS C:\Users\acer\Desktop\network-programming\LabChapterWise\Chapter3> & 'C:\Program Files\Java\jdk-24\bin\java.exe'
'--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\acer\AppData\Roaming\Code\User\workspa
ceStorage\26c2a34da8fb5be31c49cd57305d58e6\redhat.java\jdt_ws\Chapter3_ee5f58c8\bin' 'ResolveRelativeURI'
Resolved URI: https://www.example.com/folder/file.html
```

Lab 10

Objective: To write a program to download an object.

Source Code:

```
import java.io.FileOutputStream;
import java.io.InputStream;
import java.net.URL;

public class DownloadObject {
    public static void main(String[] args) {
        try {
            URL url = new URL("https://www.example.com");
            InputStream input = url.openStream();
            FileOutputStream output = new FileOutputStream("downloaded_sample.pdf");

            byte[] buffer = new byte[2048];
            int bytesRead;

            while ((bytesRead = input.read(buffer)) != -1) {
                output.write(buffer, 0, bytesRead);
            }

            input.close();
            output.close();
            System.out.println("File download successfully.");
        } catch (Exception e) {
            System.out.println("File download garna error: " + e);
        }
    }
}
```

Output:



Lab 11

Objective: To write a program to demonstrate the x-www-form-URL encoded strings.

Source Code:

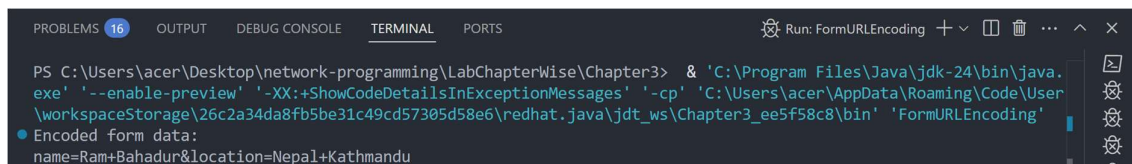
```
import java.net.URLEncoder;
import java.nio.charset.StandardCharsets;

public class FormURLEncoding {
    public static void main(String[] args) {
        try {
            String name = "Ram Bahadur";
            String location = "Nepal Kathmandu";

            String encodedName = URLEncoder.encode(name, StandardCharsets.UTF_8);
            String encodedLocation = URLEncoder.encode(location, StandardCharsets.UTF_8);

            System.out.println("Encoded form data:");
            System.out.println("name=" + encodedName + "&location=" + encodedLocation);
        } catch (Exception e) {
            System.out.println("Form URL encoding garna error: " + e);
        }
    }
}
```

Output:

A screenshot of an IDE terminal window. The terminal title bar shows 'Run: FormURLEncoding'. The command prompt shows the execution of the Java program. The output displayed in the terminal is 'Encoded form data:' followed by a new line and 'name=Ram+Bahadur&location=Nepal+Kathmandu'.

```
PS C:\Users\acer\Desktop\network-programming\LabChapterwise\Chapter3> & 'C:\Program Files\Java\jdk-24\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\acer\AppData\Roaming\Code\User\workspaceStorage\26c2a34da8fb5be31c49cd57305d58e6\redhat.java\jdt_ws\Chapter3_ee5f58c8\bin' 'FormURLEncoding'
Encoded form data:
name=Ram+Bahadur&location=Nepal+Kathmandu
```

Lab 12

Objective: To write a program that communicating with Server-Side Programs Through GET.

Source Code:

```
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.net.HttpURLConnection;
import java.net.URL;

public class HTTPGetRequest {
    public static void main(String[] args) {
        try {
            URL url = new URL("https://jsonplaceholder.typicode.com/posts/1");
            HttpURLConnection conn = (HttpURLConnection) url.openConnection();

            conn.setRequestMethod("GET");
            int responseCode = conn.getResponseCode();

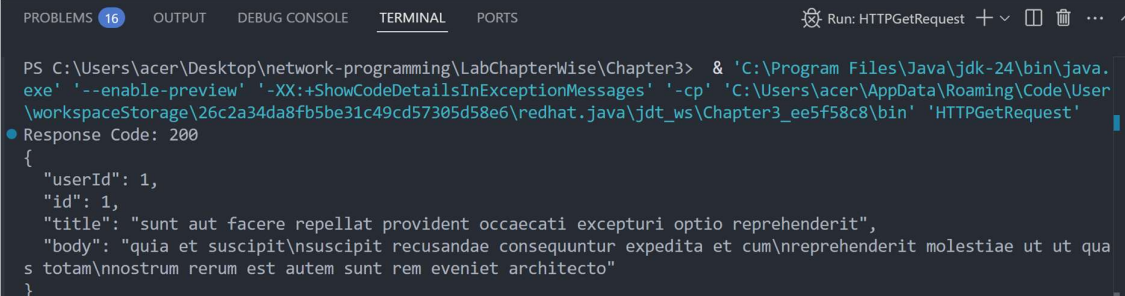
            System.out.println("Response Code: " + responseCode);

            BufferedReader reader = new BufferedReader(new
InputStreamReader(conn.getInputStream()));
            String line;

            while ((line = reader.readLine()) != null) {
                System.out.println(line);
            }

            reader.close();
        } catch (Exception e) {
            System.out.println("Server sanga GET communication garna error: " + e);
        }
    }
}
```

Output:



```
PS C:\Users\acer\Desktop\network-programming\LabChapterWise\Chapter3> & 'C:\Program Files\Java\jdk-24\bin\java.exe' '-enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\acer\AppData\Roaming\Code\User\workspaceStorage\26c2a34da8fb5be31c49cd57305d58e6\redhat.java\jdt_ws\Chapter3_ee5f58c8\bin' 'HTTPGetRequest'
● Response Code: 200
{
  "userId": 1,
  "id": 1,
  "title": "sunt aut facere repellat provident occaecati excepturi optio reprehenderit",
  "body": "quia et suscipit\nsuscipit recusandae consequuntur expedita et cum\nreprehenderit molestiae ut ut quas totam\nnostrum rerum est autem sunt rem eveniet architecto"
}
```

Chapter 4: HTTP

Lab 13

Objective: To write a program that shows a simple CookiePolicy that blocks cookies from .gov domains, but allows others.

Source Code:

```
import java.net.*;

public class CookiePolicyExample {
    public static void main(String[] args) throws Exception {
        CookiePolicy policy = new CookiePolicy() {
            @Override
            public boolean shouldAccept(Uri uri, HttpCookie cookie) {
                String host = uri.getHost();
                return !host.endsWith(".gov.np");
            }
        };

        CookieManager manager = new CookieManager();
        manager.setCookiePolicy(policy);
        CookieHandler.setDefault(manager);

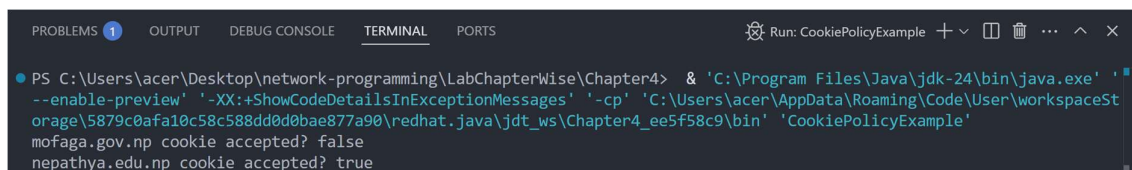
        Uri govUri = new Uri("http://mofaga.gov.np"); // blocked
        Uri eduUri = new Uri("http://nepathya.edu.np"); // allowed

        HttpCookie govCookie = new HttpCookie("session", "gov_cookie");
        HttpCookie eduCookie = new HttpCookie("session", "edu_cookie");

        boolean isGovAccepted = policy.shouldAccept(govUri, govCookie);
        boolean isEduAccepted = policy.shouldAccept(eduUri, eduCookie);

        System.out.println("mofaga.gov.np cookie accepted? " + isGovAccepted);
        System.out.println("nepathya.edu.np cookie accepted? " + isEduAccepted);
    }
}
```

Output:



```
PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS Run: CookiePolicyExample + - □ □ ... ^ ×
● PS C:\Users\acer\Desktop\network-programming\LabChapterWise\Chapter4> & 'C:\Program Files\Java\jdk-24\bin\java.exe' '-
--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\acer\AppData\Roaming\Code\User\workspaceSt
orage\5879c0afa10c58c588dd0d0bae877a90\redhat.java\jdt_ws\Chapter4_ee5f58c9\bin' 'CookiePolicyExample'
mofaga.gov.np cookie accepted? false
nepathya.edu.np cookie accepted? true
```

Lab 14

Objective: To implement the CookieStore Methods (add, read, delete) cookies.

Source Code:

```
import java.net.*;
import java.util.*;

public class CookieStoreExample {
    public static void main(String[] args) throws Exception {
        CookieManager manager = new CookieManager();
        CookieHandler.setDefault(manager);
        CookieStore store = manager.getCookieStore();

        URI nepUri = new URI("http://nepathya.edu.np");

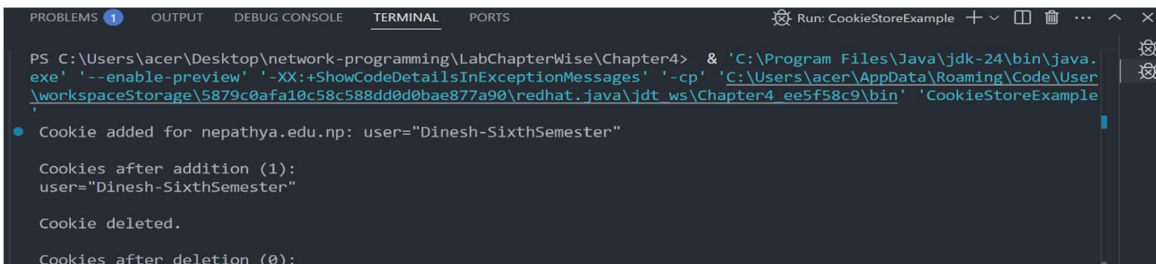
        // Add a cookie
        HttpCookie cookie = new HttpCookie("user", "Dinesh-SixthSemester");
        store.add(nepUri, cookie);

        System.out.println(" Cookie added for nepathya.edu.np: " + cookie);
        System.out.println("\n Cookies after addition (" + store.getCookies().size() + "):");
        for (HttpCookie c : store.getCookies()) {
            System.out.println(" " + c);
        }

        // Delete cookie
        store.remove(nepUri, cookie);
        System.out.println("\n Cookie deleted.");

        // Verify deletion
        List<HttpCookie> cookies = store.getCookies();
        System.out.println("\n Cookies after deletion (" + cookies.size() + "):");
        for (HttpCookie c : cookies) {
            System.out.println("➡ " + c);
        }
    }
}
```

Output:



```
PS C:\Users\acer\Desktop\network-programming\LabChapterWise\Chapter4> & 'C:\Program Files\Java\jdk-24\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\acer\AppData\Roaming\Code\User\workspaceStorage\5879c0afa10c58c588dd0d0bae877a90\redhat.java\jdt_ws\Chapter4_ee5f58c9\bin' 'CookieStoreExample'

• Cookie added for nepathya.edu.np: user="Dinesh-SixthSemester"

Cookies after addition (1):
user="Dinesh-SixthSemester"

Cookie deleted.

Cookies after deletion (0):
```

Chapter 5: URL Connections

Lab 15

Objective: To write a program to download a web page using URLConnection.

Source Code:

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

public class DownloadWebPage {
    public static void main(String[] args) {
        String website = "http://nepathyacollege.edu.np";

        try {
            URL url = new URL(website);
            URLConnection connection = url.openConnection();

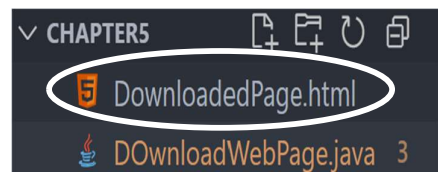
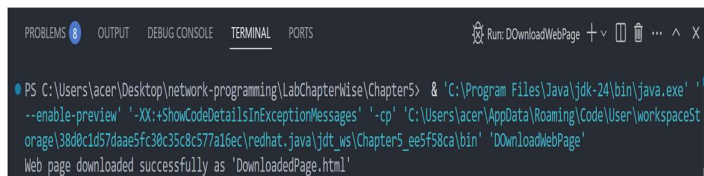
            BufferedReader reader = new BufferedReader(
                new InputStreamReader(connection.getInputStream()));
            BufferedWriter writer = new BufferedWriter(
                new FileWriter("DownloadedPage.html"));

            String line;
            while ((line = reader.readLine()) != null) {
                writer.write(line);
                writer.newLine();
            }

            reader.close();
            writer.close();

            System.out.println("Web page downloaded successfully as 'DownloadedPage.html'");
        } catch (Exception e) {
            System.out.println("Error: " + e.getMessage());
        }
    }
}
```

Output:



Lab 16

Objective: To write a program to read value of HTTP Header fields.

Source Code:

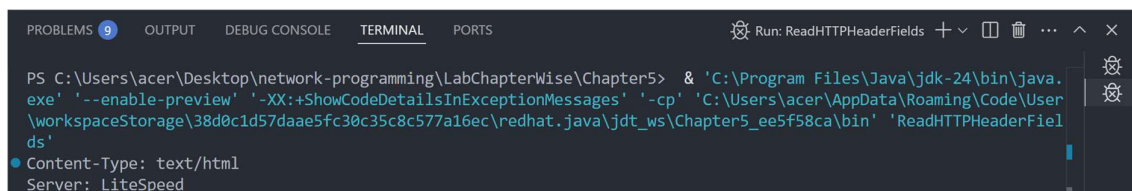
```
import java.net.*;

public class ReadHTTPHeaderFields {
    public static void main(String[] args) {
        try {
            URL url = new URL("http://nepathycollege.edu.np");
            URLConnection connection = url.openConnection();

            String contentType = connection.getHeaderField("Content-Type");
            String server = connection.getHeaderField("Server");

            System.out.println("Content-Type: " + contentType);
            System.out.println("Server: " + server);
        } catch (Exception e) {
            System.out.println("Error: " + e.getMessage());
        }
    }
}
```

Output:



```
PS C:\Users\acer\Desktop\network-programming\LabChapterWise\Chapter5> & 'C:\Program Files\Java\jdk-24\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\acer\AppData\Roaming\Code\User\workspaceStorage\38d0c1d57daae5fc30c35c8c577a16ec\redhat.java\jdt_ws\Chapter5_ee5f58ca\bin' 'ReadHTTPHeaderFields'
Content-Type: text/html
Server: LiteSpeed
```


Lab 17

Objective: To write a program to print the entire HTTP header

Source Code:

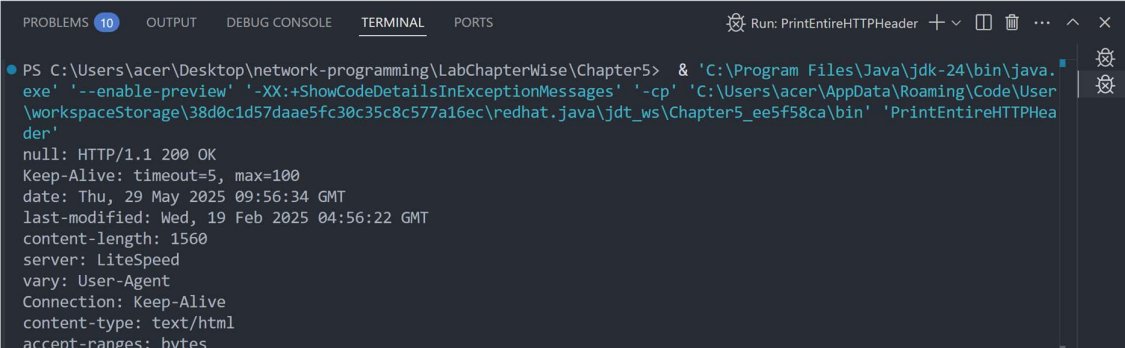
```
import java.net.*;
import java.util.*;

public class PrintEntireHTTPHeader {
    public static void main(String[] args) {
        try {
            URL url = new URL("http://nepathyacollege.edu.np");
            URLConnection connection = url.openConnection();

            Map<String, List<String>> headers = connection.getHeaderFields();

            for (Map.Entry<String, List<String>> entry : headers.entrySet()) {
                String key = entry.getKey();
                List<String> values = entry.getValue();
                System.out.println(key + ": " + String.join(" ", values));
            }
        } catch (Exception e) {
            System.out.println("Error: " + e.getMessage());
        }
    }
}
```

Output:



```
PS C:\Users\acer\Desktop\network-programming\LabChapterWise\Chapter5> & 'C:\Program Files\Java\jdk-24\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\acer\AppData\Roaming\Code\User\workspaceStorage\38d0c1d57daae5fc30c35c8c577a16ec\redhat.java\jdt_ws\Chapter5_ee5f58ca\bin' 'PrintEntireHTTPHeader'
null: HTTP/1.1 200 OK
Keep-Alive: timeout=5, max=100
date: Thu, 29 May 2025 09:56:34 GMT
last-modified: Wed, 19 Feb 2025 04:56:22 GMT
content-length: 1560
server: LiteSpeed
vary: User-Agent
Connection: Keep-Alive
content-type: text/html
accept-ranges: bytes
```

Lab 18

Objective: To write a program for HTTP Request Method.

Source Code:

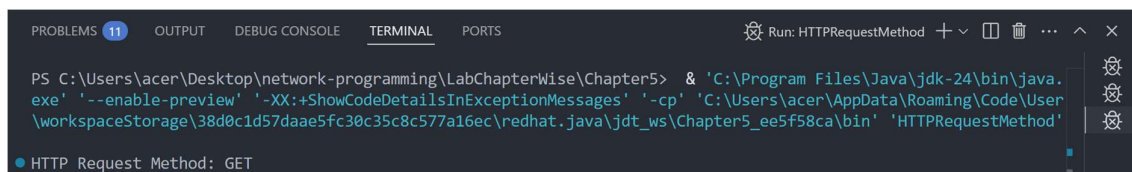
```
import java.net.*;

public class HTTPRequestMethod {
    public static void main(String[] args) {
        try {
            URL url = new URL("http://nepathyacollege.edu.np");
            HttpURLConnection connection = (HttpURLConnection) url.openConnection();

            connection.setRequestMethod("GET"); // You can try "POST", "HEAD" etc.

            System.out.println("HTTP Request Method: " + connection.getRequestMethod());
        } catch (Exception e) {
            System.out.println("Error: " + e.getMessage());
        }
    }
}
```

Output:



```
PS C:\Users\acer\Desktop\network-programming\LabChapterWise\Chapter5> & 'C:\Program Files\Java\jdk-24\bin\java.exe' '-enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\acer\AppData\Roaming\Code\User\workspaceStorage\38d0c1d57daae5fc30c35c8c577a16ec\redhat.java\jdt_ws\Chapter5_ee5f58ca\bin' 'HTTPRequestMethod'
```

● HTTP Request Method: GET

Lab 19

Objective: To write a program to print the URL of a URL Connection to “npathyacollege.edu.np”

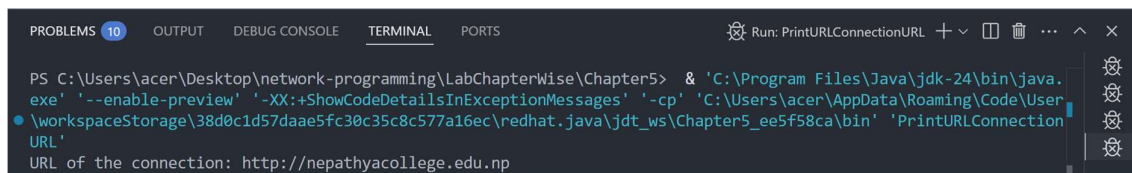
Source Code:

```
import java.net.*;

public class PrintURLConnectionURL {
    public static void main(String[] args) {
        try {
            URL url = new URL("http://npathyacollege.edu.np");
            URLConnection connection = url.openConnection();

            System.out.println("URL of the connection: " + connection.getURL());
        } catch (Exception e) {
            System.out.println("Error: " + e.getMessage());
        }
    }
}
```

Output:



```
PROBLEMS 10 OUTPUT DEBUG CONSOLE TERMINAL PORTS
Run: PrintURLConnectionURL + - [ ] ... ^ x
PS C:\Users\acer\Desktop\network-programming\LabChapterWise\Chapter5> & 'C:\Program Files\Java\jdk-24\bin\java.
exe' '-enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\acer\AppData\Roaming\Code\User
\workspaceStorage\38d0c1d57daae5fc30c35c8c577a16ec\redhat.java\jdt_ws\Chapter5_ee5f58ca\bin' 'PrintURLConnection
URL'
URL of the connection: http://npathyacollege.edu.np
```

Lab 20

Objective: To write a program to get the time when a URL was last changed.

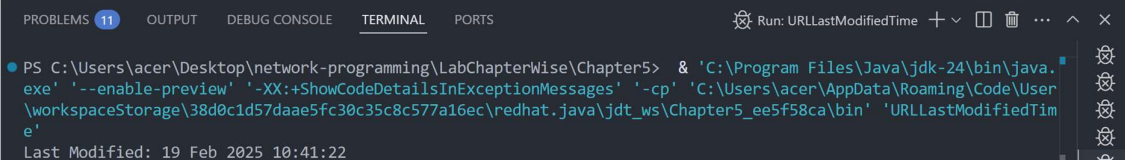
Source Code:

```
import java.net.*;
import java.text.SimpleDateFormat;
import java.util.Date;

public class URLLastModifiedTime {
    public static void main(String[] args) {
        try {
            URL url = new URL("http://nepathyacollege.edu.np");
            URLConnection connection = url.openConnection();

            long lastModified = connection.getLastModified();
            if (lastModified == 0) {
                System.out.println("No Last-Modified information.");
            } else {
                Date date = new Date(lastModified);
                SimpleDateFormat sdf = new SimpleDateFormat("dd MMM yyyy HH:mm:ss");
                System.out.println("Last Modified: " + sdf.format(date));
            }
        } catch (Exception e) {
            System.out.println("Error: " + e.getMessage());
        }
    }
}
```

Output:



```
PS C:\Users\acer\Desktop\network-programming\LabChapterWise\Chapter5> & 'C:\Program Files\Java\jdk-24\bin\java.exe' '-enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\acer\AppData\Roaming\Code\User\workspaceStorage\38d0c1d57daae5fc30c35c8c577a16ec\redhat.java\jdt_ws\Chapter5_ee5f58ca\bin' 'URLLastModifiedTime'
Last Modified: 19 Feb 2025 10:41:22
```

Chapter 6: Socket for Clients

Lab 21

Objective: To write a program socket to client.

Source Code:

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

public class SimpleSocketClient {
    public static void main(String[] args) {
        String serverAddress = "localhost"; // or IP address of the server
        int port = 5000; // port number the server is listening on

        try (Socket socket = new Socket(serverAddress, port);
            PrintWriter out = new PrintWriter(socket.getOutputStream(), true);
            BufferedReader in = new BufferedReader(new
InputStreamReader(socket.getInputStream()));
            BufferedReader userInput = new BufferedReader(new
InputStreamReader(System.in))) {

            System.out.println("Connected to server at " + serverAddress + ":" + port);

            // Read user input and send to server
            System.out.print("Enter message to send: ");
            String message = userInput.readLine();
            out.println(message);

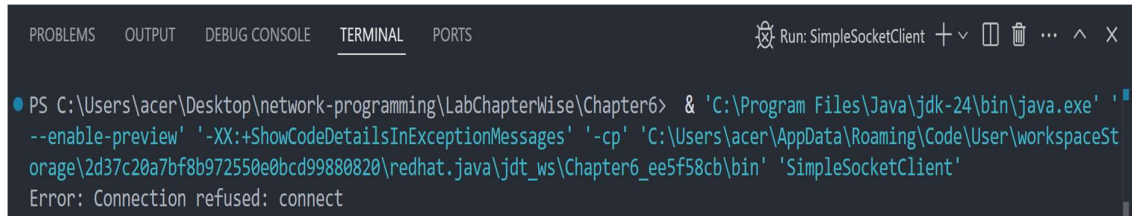
            // Read and print server response
            String response = in.readLine();
            System.out.println("Server response: " + response);

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

```
}  
}
```

Output:

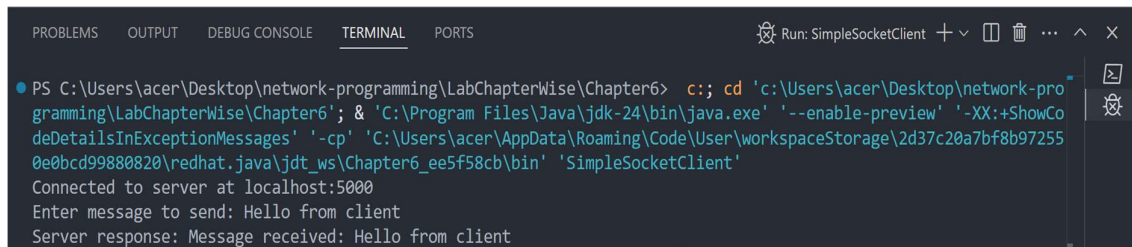
(before connecting to server)



The screenshot shows an IDE terminal window with the 'TERMINAL' tab selected. The title bar indicates 'Run: SimpleSocketClient'. The terminal output shows a PowerShell command being executed, which results in an 'Error: Connection refused: connect'.

```
PS C:\Users\acer\Desktop\network-programming\LabChapterWise\Chapter6> & 'C:\Program Files\Java\jdk-24\bin\java.exe' '-  
--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\acer\AppData\Roaming\Code\User\workspaceSt  
orage\2d37c20a7bf8b972550e0bcd99880820\redhat.java\jdt_ws\Chapter6_ee5f58cb\bin' 'SimpleSocketClient'  
Error: Connection refused: connect
```

(after connecting to server)



The screenshot shows the same IDE terminal window after a successful connection. The terminal output shows the user running 'c;' and 'cd' commands, followed by the connection message 'Connected to server at localhost:5000'. The user then enters 'Enter message to send: Hello from client', and the server responds with 'Server response: Message received: Hello from client'.

```
PS C:\Users\acer\Desktop\network-programming\LabChapterWise\Chapter6> c;; cd 'c:\Users\acer\Desktop\network-pro  
gramming\LabChapterWise\Chapter6'; & 'C:\Program Files\Java\jdk-24\bin\java.exe' '-enable-preview' '-XX:+ShowCo  
deDetailsInExceptionMessages' '-cp' 'C:\Users\acer\AppData\Roaming\Code\User\workspaceStorage\2d37c20a7bf8b97255  
0e0bcd99880820\redhat.java\jdt_ws\Chapter6_ee5f58cb\bin' 'SimpleSocketClient'  
Connected to server at localhost:5000  
Enter message to send: Hello from client  
Server response: Message received: Hello from client
```

Chapter 7: Socket for Server

Lab 22

Objective: To write a program socket to server.

Source Code:

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

public class SimpleSocketServer {
    public static void main(String[] args) {
        int port = 5000; // Server will listen on this port

        try (ServerSocket serverSocket = new ServerSocket(port)) {
            System.out.println("Server started. Waiting for client connection on port " + port);

            Socket clientSocket = serverSocket.accept(); // Accept client connection
            System.out.println("Client connected: " + clientSocket.getInetAddress());

            BufferedReader in = new BufferedReader(
                new InputStreamReader(clientSocket.getInputStream()));
            PrintWriter out = new PrintWriter(clientSocket.getOutputStream(), true);

            // Read message from client
            String message = in.readLine();
            System.out.println("Received from client: " + message);

            // Send response to client
            out.println("Message received: " + message);

            // Close client connection
            clientSocket.close();
            System.out.println("Client connection closed.");
        } catch (IOException e) {
            System.err.println("Error: " + e.getMessage());
        }
    }
}
```

```
}  
  
}  
  
}
```

Output:

(before any client connects)



```
PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS Run: SimpleSocketServer + - [ ] ... ^ X  
PS C:\Users\acer\Desktop\network-programming\LabChapterWise\Chapter7> & 'C:\Program Files\Java\jdk-24\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\acer\AppData\Roaming\Code\User\workspaceStorage\8b2fb401b2d2894d511aa44b510cc133\redhat.java\jdt_ws\Chapter7_ee5f58cc\bin' 'SimpleSocketServer'  
Server started. Waiting for client connection on port 5000
```

(after client connects)



```
PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS Run: SimpleSocketServer + - [ ] ... ^ X  
PS C:\Users\acer\Desktop\network-programming\LabChapterWise\Chapter7> c:: cd 'c:\Users\acer\Desktop\network-programming\LabChapterWise\Chapter7'; & 'C:\Program Files\Java\jdk-24\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\acer\AppData\Roaming\Code\User\workspaceStorage\8b2fb401b2d2894d511aa44b510cc133\redhat.java\jdt_ws\Chapter7_ee5f58cc\bin' 'SimpleSocketServer'  
Server started. Waiting for client connection on port 5000  
Client connected: /127.0.0.1  
Received from client: Hello from client  
Client connection closed.
```


Chapter 8: Secure Sockets

Lab 23

Objective: To write a program for Creating Secure Sockets with tufohss.edu.np.

Source Code:

```
import java.io.*;
import javax.net.ssl.SSLSocket;
import javax.net.ssl.SSLSocketFactory;

public class SecureSocketClient {
    public static void main(String[] args) {
        String host = "tufohss.edu.np";
        int port = 443; // HTTPS default port

        try {
            // Create SSL socket factory and socket
            SSLSocketFactory factory = (SSLSocketFactory) SSLSocketFactory.getDefault();
            SSLSocket sslSocket = (SSLSocket) factory.createSocket(host, port);

            // Start handshake (optional, but recommended)
            sslSocket.startHandshake();

            // Output stream to send data to server
            PrintWriter out = new PrintWriter(new BufferedWriter(
                new OutputStreamWriter(sslSocket.getOutputStream())));

            // Input stream to read server response
            BufferedReader in = new BufferedReader(
                new InputStreamReader(sslSocket.getInputStream()));

            // Send HTTP GET request
            out.println("GET / HTTP/1.1");
            out.println("Host: " + host);
            out.println("Connection: Close");
```

```

        out.println(); // blank line to end request headers
        out.flush();

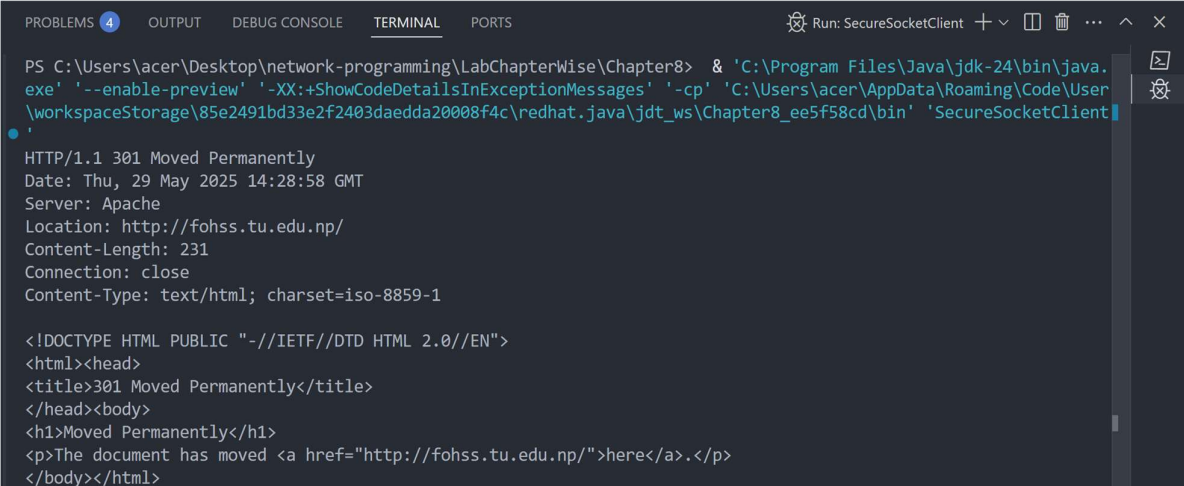
        // Read and print the response line by line
        String line;
        while ((line = in.readLine()) != null) {
            System.out.println(line);
        }

        // Close streams and socket
        in.close();
        out.close();
        sslSocket.close();

    } catch (Exception e) {
        System.err.println("Error: " + e.getMessage());
        e.printStackTrace();
    }
}
}

```

Output:



```

PS C:\Users\acer\Desktop\network-programming\LabChapterWise\Chapter8> & 'C:\Program Files\Java\jdk-24\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\acer\AppData\Roaming\Code\User\workspaceStorage\85e2491bd33e2f2403daedda20008f4c\redhat.java\jdt_ws\Chapter8_ee5f58cd\bin' 'SecureSocketClient'
HTTP/1.1 301 Moved Permanently
Date: Thu, 29 May 2025 14:28:58 GMT
Server: Apache
Location: http://fohss.tu.edu.np/
Content-Length: 231
Connection: close
Content-Type: text/html; charset=iso-8859-1

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>301 Moved Permanently</title>
</head><body>
<h1>Moved Permanently</h1>
<p>The document has moved <a href="http://fohss.tu.edu.np/">here</a>.</p>
</body></html>

```

Lab 24

Objective: To write a program for Creating Secure Server Sockets and Client Sockets.

Source Code:

SecureClient.java

```
import java.io.*;
import java.security.KeyStore;
import javax.net.ssl.*;

public class SecureClient {
    public static void main(String[] args) throws Exception {
        char[] password = "password".toCharArray();

        // Set system properties to point to the truststore file containing server cert or CA cert
        System.setProperty("javax.net.ssl.trustStore", "clienttruststore.jks");
        System.setProperty("javax.net.ssl.trustStorePassword", "password");

        // Load the truststore file (this step is optional if you want to create SSLContext manually)
        KeyStore ts = KeyStore.getInstance("JKS");
        try (FileInputStream fis = new FileInputStream("clienttruststore.jks")) {
            ts.load(fis, password);
        }

        // Initialize TrustManagerFactory with the truststore
        TrustManagerFactory tmf = TrustManagerFactory.getInstance("SunX509");
        tmf.init(ts);

        // Set up SSL context using the trust managers from the truststore
        SSLContext sc = SSLContext.getInstance("TLS");
        sc.init(null, tmf.getTrustManagers(), null);

        // Create SSLSocketFactory from SSL context
```

```

SSLSocketFactory ssf = sc.getSocketFactory();

// Create an SSL socket connected to the server
try (SSLSocket socket = (SSLSocket) ssf.createSocket("localhost", 12345);
    BufferedReader in = new BufferedReader(new
InputStreamReader(socket.getInputStream()))) {

    // Read response from server
    String response = in.readLine();
    System.out.println("Message from Server: " + response);
}
}
}

```

SecureServer.java

```

import java.io.*;
import java.security.KeyStore;
import javax.net.ssl.*;

public class SecureServer {
    public static void main(String[] args) throws Exception {
        char[] password = "password".toCharArray();

        // Load keystore
        KeyStore ks = KeyStore.getInstance("JKS");
        ks.load(new FileInputStream("serverkeystore.jks"), password);

        // Initialize KeyManagerFactory
        KeyManagerFactory kmf = KeyManagerFactory.getInstance("SunX509");
        kmf.init(ks, password);
    }
}

```

```

// Set up SSL context
SSLContext sc = SSLContext.getInstance("TLS");
sc.init(kmf.getKeyManagers(), null, null);

SSLServerSocketFactory ssf = sc.getServerSocketFactory();
SSLServerSocket serverSocket = (SSLServerSocket) ssf.createServerSocket(12345);

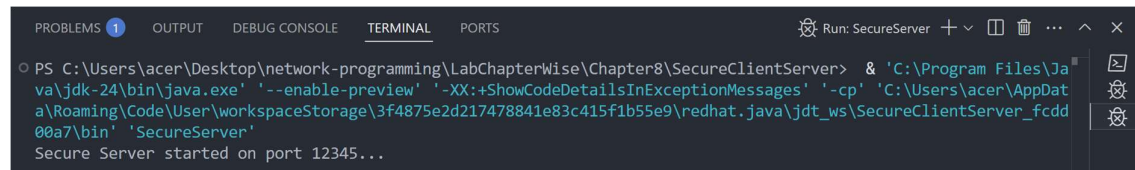
System.out.println("Secure Server started on port 12345...");

while (true) {
    SSLSocket socket = (SSLSocket) serverSocket.accept();
    PrintWriter out = new PrintWriter(socket.getOutputStream(), true);
    out.println("Hello from secure server!");
    socket.close();
}
}
}

```

Output:

(SecureServer.java)

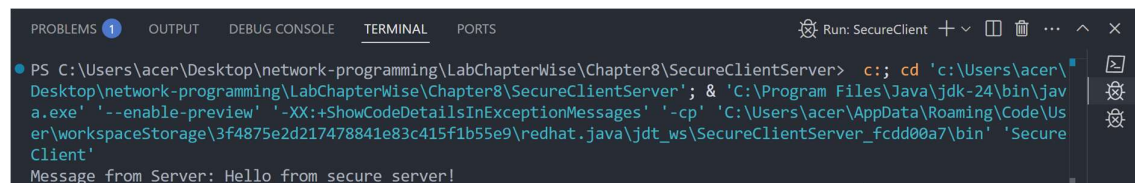


```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS Run: SecureServer
PS C:\Users\acer\Desktop\network-programming\LabChapterWise\Chapter8\SecureClientServer> & 'C:\Program Files\Java\jdk-24\bin\java.exe' '-enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\acer\AppData\Roaming\Code\User\workspaceStorage\3f4875e2d217478841e83c415f1b55e9\redhat.java\jdt_ws\SecureClientServer_fcdd00a7\bin' 'SecureServer'
Secure Server started on port 12345...

```

(SecureClient.java)



```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS Run: SecureClient
PS C:\Users\acer\Desktop\network-programming\LabChapterWise\Chapter8\SecureClientServer> c;; cd 'c:\Users\acer\Desktop\network-programming\LabChapterWise\Chapter8\SecureClientServer'; & 'C:\Program Files\Java\jdk-24\bin\java.exe' '-enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\acer\AppData\Roaming\Code\User\workspaceStorage\3f4875e2d217478841e83c415f1b55e9\redhat.java\jdt_ws\SecureClientServer_fcdd00a7\bin' 'SecureClient'
Message from Server: Hello from secure server!

```

Chapter 9: Non Blocking I/O

Lab 25

Objective: To write program to list all supported socket options for the different types of network channels

Source Code:

```
import java.net.*;

import java.nio.channels.*;

import java.util.Set;

public class ListSocketOptions {

    public static void main(String[] args) throws Exception {

        System.out.println("SocketChannel options:");

        SocketChannel socketChannel = SocketChannel.open();

        Set<SocketOption<?>> socketOptions = socketChannel.supportedOptions();

        for (SocketOption<?> option : socketOptions) {

            System.out.println(" - " + option.name());

        }

        System.out.println("\nServerSocketChannel options:");

        ServerSocketChannel serverSocketChannel = ServerSocketChannel.open();

        Set<SocketOption<?>> serverSocketOptions =

serverSocketChannel.supportedOptions();

        for (SocketOption<?> option : serverSocketOptions) {

            System.out.println(" - " + option.name());

        }

    }

}
```

```

        System.out.println("\nDatagramChannel options:");

        DatagramChannel datagramChannel = DatagramChannel.open();

        Set<SocketOption<?>> datagramOptions = datagramChannel.supportedOptions();

        for (SocketOption<?> option : datagramOptions) {

            System.out.println(" - " + option.name());

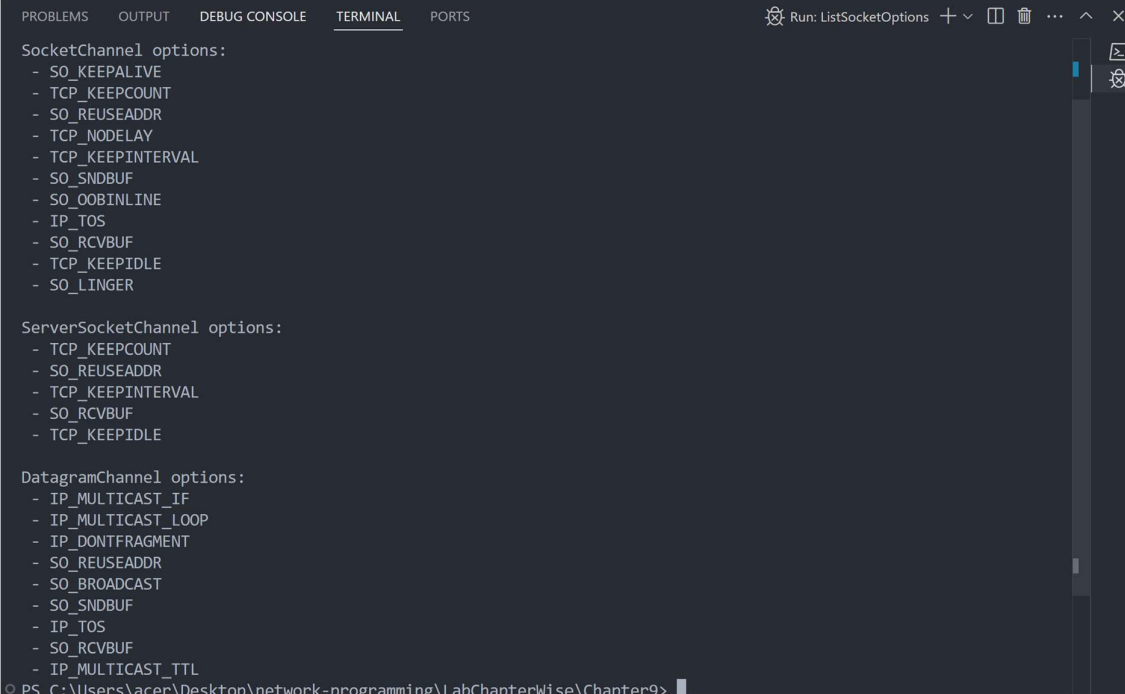
        }

    }

}

```

Output:



```

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
Run: ListSocketOptions + - [ ] ... ^ x

SocketChannel options:
- SO_KEEPALIVE
- TCP_KEEPCOUNT
- SO_REUSEADDR
- TCP_NODELAY
- TCP_KEEPINTERVAL
- SO_SNDBUF
- SO_OOBINLINE
- IP_TOS
- SO_RCVBUF
- TCP_KEEPIDLE
- SO_LINGER

ServerSocketChannel options:
- TCP_KEEPCOUNT
- SO_REUSEADDR
- TCP_KEEPINTERVAL
- SO_RCVBUF
- TCP_KEEPIDLE

DatagramChannel options:
- IP_MULTICAST_IF
- IP_MULTICAST_LOOP
- IP_DONTFRAGMENT
- SO_REUSEADDR
- SO_BROADCAST
- SO_SNDBUF
- IP_TOS
- SO_RCVBUF
- IP_MULTICAST_TTL

```

Lab 26

Objective: To write program to implement the concept on Filling and Draining buffer, duplicating buffer, Slicing buffer, Compact buffer.

Source Code:

```
import java.nio.ByteBuffer;

public class BufferOperations {

    public static void main(String[] args) {

        // Fill buffer
        ByteBuffer buffer = ByteBuffer.allocate(10);
        for (byte i = 0; i < 10; i++) {
            buffer.put(i);
        }

        // Flip (prepare for reading)
        buffer.flip();

        // Drain buffer
        System.out.print("Draining buffer: ");
        while (buffer.hasRemaining()) {
            System.out.print(buffer.get() + " ");
        }

        // Refill
        buffer.clear();
        for (byte i = 10; i < 20; i++) {
            buffer.put(i);
        }

        // Duplicate
        ByteBuffer duplicate = buffer.duplicate();
        duplicate.flip();
        System.out.print("\nDuplicate buffer: ");
        while (duplicate.hasRemaining()) {
            System.out.print(duplicate.get() + " ");
        }
    }
}
```



```

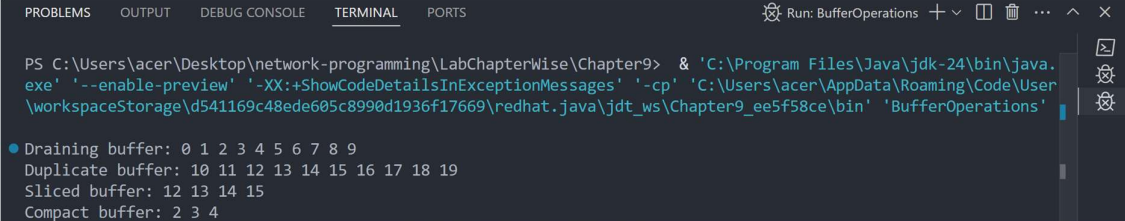
    }

    // Slice buffer
    buffer.position(2);
    buffer.limit(6);
    ByteBuffer slice = buffer.slice();
    System.out.print("\nSliced buffer: ");
    for (int i = 0; i < slice.capacity(); i++) {
        System.out.print(slice.get(i) + " ");
    }

    //compact
    buffer.clear();
    buffer.put((byte) 1);
    buffer.put((byte) 2);
    buffer.put((byte) 3);
    buffer.flip();
    buffer.get(); // read one byte
    buffer.compact(); // move remaining bytes to beginning
    buffer.put((byte) 4); // add more
    buffer.flip();
    System.out.print("\nCompact buffer: ");
    while (buffer.hasRemaining()) {
        System.out.print(buffer.get() + " ");
    }
}
}
}

```

Output:



```

PS C:\Users\acer\Desktop\network-programming\LabChapterWise\Chapter9> & 'C:\Program Files\Java\jdk-24\bin\java.exe' '-enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\acer\AppData\Roaming\Code\User\workspaceStorage\d541169c48ede605c8990d1936f17669\redhat.java\jdt_ws\Chapter9_ee5f58ce\bin' 'BufferOperations'

Draining buffer: 0 1 2 3 4 5 6 7 8 9
Duplicate buffer: 10 11 12 13 14 15 16 17 18 19
Sliced buffer: 12 13 14 15
Compact buffer: 2 3 4

```

Lab 27

Objective: To write a program to implement the concept on Data Conversion

Source Code:

```
import java.nio.ByteBuffer;
import java.nio.CharBuffer;
import java.nio.charset.Charset;
import java.nio.charset.StandardCharsets;

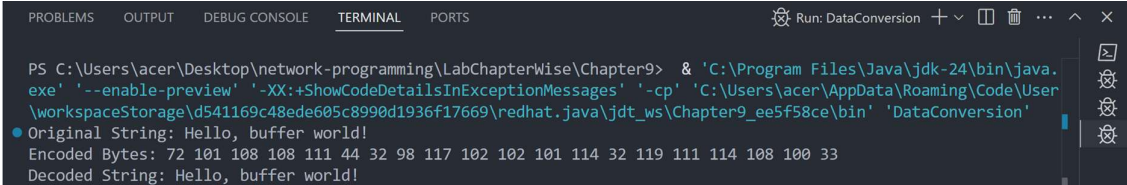
public class DataConversion {
    public static void main(String[] args) {
        String input = "Hello, buffer world!";
        System.out.println("Original String: " + input);

        Charset charset = StandardCharsets.UTF_8;
        ByteBuffer byteBuffer = charset.encode(input);

        System.out.print("Encoded Bytes: ");
        while (byteBuffer.hasRemaining()) {
            System.out.print(byteBuffer.get() + " ");
        }

        byteBuffer.rewind();
        CharBuffer decoded = charset.decode(byteBuffer);
        System.out.println("\nDecoded String: " + decoded.toString());
    }
}
```

Output:



```
PS C:\Users\acer\Desktop\network-programming\LabChapterWise\Chapter9> & 'C:\Program Files\Java\jdk-24\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\acer\AppData\Roaming\Code\User\workspaceStorage\d541169c48ede605c8990d1936f17669\redhat.java\jdt_ws\Chapter9_ee5f58ce\bin' 'DataConversion'
Original String: Hello, buffer world!
Encoded Bytes: 72 101 108 108 111 44 32 98 117 102 102 101 114 32 119 111 114 108 100 33
Decoded String: Hello, buffer world!
```

Chapter 10: UDP

Lab 28

Objective: To write a program for UDP Client

Source Code:

```
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;
import java.util.Scanner;

public class UDPClient {
    public static void main(String[] args) {
        try {
            DatagramSocket clientSocket = new DatagramSocket();
            InetAddress serverAddress = InetAddress.getByName("localhost");
            Scanner scanner = new Scanner(System.in);

            System.out.print("Enter message to send: ");
            String message = scanner.nextLine();
            byte[] sendBuffer = message.getBytes();

            DatagramPacket sendPacket = new DatagramPacket(sendBuffer, sendBuffer.length,
serverAddress, 9876);
            clientSocket.send(sendPacket);

            // Receive response
            byte[] receiveBuffer = new byte[1024];
            DatagramPacket receivePacket = new DatagramPacket(receiveBuffer,
receiveBuffer.length);
            clientSocket.receive(receivePacket);

            String response = new String(receivePacket.getData(), 0, receivePacket.getLength());
```

```

        System.out.println("Server replied: " + response);

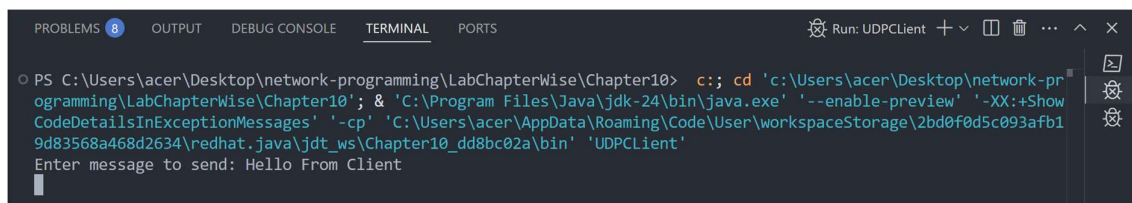
        clientSocket.close();

        scanner.close();
    } catch (Exception e) {
        e.printStackTrace();
    }
}
}

```

Output:

(before connecting to server)

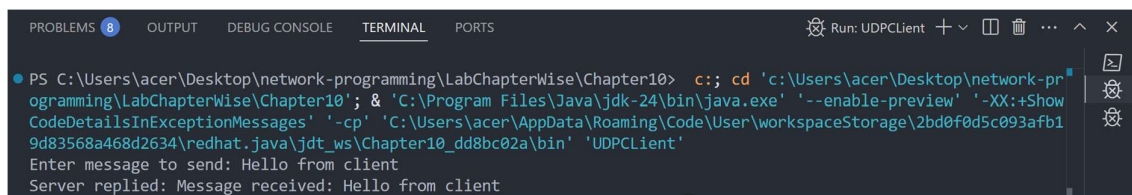


```

PS C:\Users\acer\Desktop\network-programming\LabChapterWise\Chapter10> c:; cd 'c:\Users\acer\Desktop\network-programming\LabChapterWise\Chapter10'; & 'C:\Program Files\Java\jdk-24\bin\java.exe' '-enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\acer\AppData\Roaming\Code\User\workspaceStorage\2bd0f0d5c093afb19d83568a468d2634\redhat.java\jdt_ws\Chapter10_dd8bc02a\bin' 'UDPClient'
Enter message to send: Hello From Client

```

(after connecting to server)



```

PS C:\Users\acer\Desktop\network-programming\LabChapterWise\Chapter10> c:; cd 'c:\Users\acer\Desktop\network-programming\LabChapterWise\Chapter10'; & 'C:\Program Files\Java\jdk-24\bin\java.exe' '-enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\acer\AppData\Roaming\Code\User\workspaceStorage\2bd0f0d5c093afb19d83568a468d2634\redhat.java\jdt_ws\Chapter10_dd8bc02a\bin' 'UDPClient'
Enter message to send: Hello from client
Server replied: Message received: Hello from client

```

Lab 29

Objective: To write a program for UDP Server.

Source Code:

```
import java.net.DatagramPacket;
import java.net.DatagramSocket;

public class UDPServer {
    public static void main(String[] args) {
        try {
            DatagramSocket serverSocket = new DatagramSocket(9876);
            byte[] receiveBuffer = new byte[1024];

            System.out.println("Server is running and waiting for data...");

            // Receive packet
            DatagramPacket receivePacket = new DatagramPacket(receiveBuffer,
            receiveBuffer.length);

            serverSocket.receive(receivePacket);

            String clientMessage = new String(receivePacket.getData(), 0,
            receivePacket.getLength());

            System.out.println("Client says: " + clientMessage);

            // Send response
            String reply = "Message received: " + clientMessage;
            byte[] sendBuffer = reply.getBytes();

            DatagramPacket sendPacket = new DatagramPacket(
                sendBuffer,
                sendBuffer.length,
                receivePacket.getAddress(),
                receivePacket.getPort()
            );
        }
    }
}
```

```

    );

    serverSocket.send(sendPacket);

    serverSocket.close();
} catch (Exception e) {
    e.printStackTrace();
}
}
}
}

```

Output:

(before any client connects)

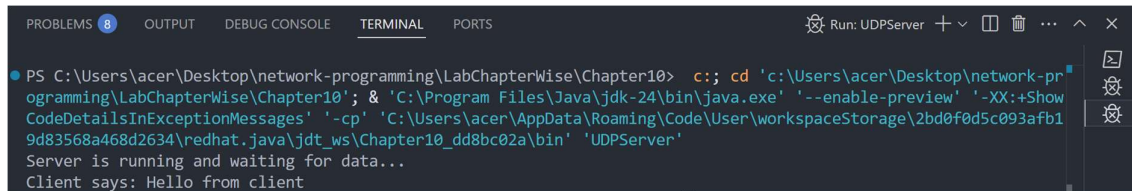


```

PROBLEMS 8 OUTPUT DEBUG CONSOLE TERMINAL PORTS
Run: UDPServer + v [] ... ^ x
PS C:\Users\acer\Desktop\network-programming\LabChapterWise\Chapter10> c:: cd 'c:\Users\acer\Desktop\network-pr
ogramming\LabChapterWise\Chapter10'; & 'C:\Program Files\Java\jdk-24\bin\java.exe' '--enable-preview' '-XX:+Show
CodeDetailsInExceptionMessages' '-cp' 'C:\Users\acer\AppData\Roaming\Code\User\workspaceStorage\2bd0f0d5c093afb1
9d83568a468d2634\redhat.java\jdt_ws\Chapter10_dd8bc02a\bin' 'UDPServer'
Server is running and waiting for data...

```

(after client connects)



```

PROBLEMS 8 OUTPUT DEBUG CONSOLE TERMINAL PORTS
Run: UDPServer + v [] ... ^ x
PS C:\Users\acer\Desktop\network-programming\LabChapterWise\Chapter10> c:: cd 'c:\Users\acer\Desktop\network-pr
ogramming\LabChapterWise\Chapter10'; & 'C:\Program Files\Java\jdk-24\bin\java.exe' '--enable-preview' '-XX:+Show
CodeDetailsInExceptionMessages' '-cp' 'C:\Users\acer\AppData\Roaming\Code\User\workspaceStorage\2bd0f0d5c093afb1
9d83568a468d2634\redhat.java\jdt_ws\Chapter10_dd8bc02a\bin' 'UDPServer'
Server is running and waiting for data...
Client says: Hello from client

```

Chapter 11: IP Multicast

Lab 30

Objective: To verify that you are receiving multicast data at a particular host.

Source Code:

```
import java.io.IOException;

import java.net.DatagramPacket;

import java.net.InetAddress;

import java.net.MulticastSocket;


public class MulticastReceiver {

    public static void main(String[] args) {

        final int PORT = 4446; // Use the same port as sender

        final String MULTICAST_GROUP = "230.0.0.0"; // A valid multicast IP


        try {

            MulticastSocket socket = new MulticastSocket(PORT);

            InetAddress group = InetAddress.getByName(MULTICAST_GROUP);

            socket.joinGroup(group);

            System.out.println("Joined multicast group. Listening for messages...");


            byte[] buffer = new byte[1024];

            DatagramPacket packet = new DatagramPacket(buffer, buffer.length);

            socket.receive(packet);

            String received = new String(packet.getData(), 0, packet.getLength());
```

```
System.out.println("Received multicast message: " + received);

socket.leaveGroup(group);

socket.close();

} catch (IOException e) {


    e.printStackTrace();

}

}

}
```

Output:



```
PROBLEMS 5 OUTPUT DEBUG CONSOLE TERMINAL PORTS Run: MulticastReceiver + - [ ] ... ^ x
o PS C:\Users\acer\Desktop\network-programming\LabChapterWise\Chapter11> & 'C:\Program Files\Java\jdk-24\bin\java
.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\acer\AppData\Roaming\Code\Use
r\workspaceStorage\0b96ba4409508c1d080dcf6c89cec34\redhat.java\jdt_ws\Chapter11_dd8bc02b\bin' 'MulticastReceiv
e'
Joined multicast group. Listening for messages...
|
```


Chapter 12: RMI

Lab 31

Objective: To add two numbers using RMI

Source Code:

Calculator.java

```
import java.rmi.Remote;  
import java.rmi.RemoteException;  
  
public interface Calculator extends Remote{  
    int add(int a, int b) throws RemoteException;  
}
```

CalculatorRemote.java

```
import java.rmi.RemoteException;  
import java.rmi.server.UnicastRemoteObject;  
  
public class CalculatorRemote extends UnicastRemoteObject implements Calculator{  
    //constructor  
    public CalculatorRemote() throws RemoteException {  
        super();  
    }  
  
    @Override  
    public int add(int a, int b){  
        return a + b;  
    }  
}
```

Server.java

```
import java.rmi.registry.LocateRegistry;
import java.rmi.registry.Registry;

public class Server {
    public static void main(String[] args) {
        try{
            CalculatorRemote objCalculatorRemote = new CalculatorRemote();
            Registry registry = LocateRegistry.createRegistry(9000);
            registry.bind("Multiply", objCalculatorRemote);
            System.out.println("Server is ready to accept requests...");
        } catch (Exception e) {
            System.out.println("Server exception: " + e.getMessage());
            e.printStackTrace();
        }
    }
}
```

Client.java

```
import java.rmi.registry.LocateRegistry;
import java.rmi.registry.Registry;

public class Client {
    public static void main(String[] args) {
        try {
            Registry registry = LocateRegistry.getRegistry("localhost", 9000);
            Calculator calculator = (Calculator) registry.lookup("Multiply");
            int value = calculator.add(60, 10);
            System.out.println("Result is: "+value);
        }
    }
}
```

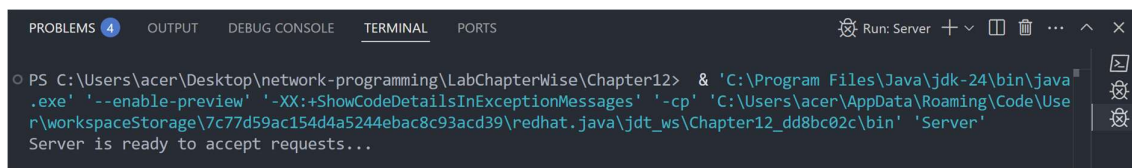
```

    } catch (Exception e) {
        System.out.println("Client exception: " + e.getMessage());
        e.printStackTrace();
    }
}
}
}

```

Output:

(server)



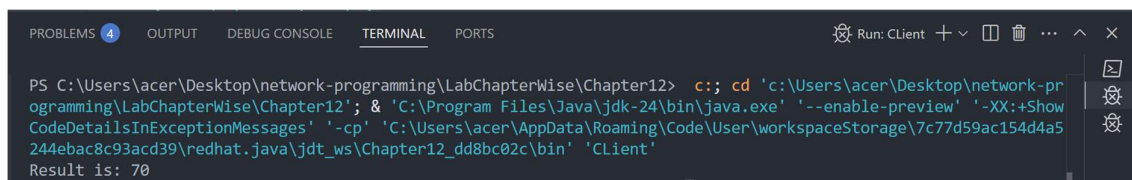
```

PROBLEMS 4 OUTPUT DEBUG CONSOLE TERMINAL PORTS
Run: Server + - [ ] [ ] ... ^ x

PS C:\Users\acer\Desktop\network-programming\LabChapterWise\Chapter12> & 'C:\Program Files\Java\jdk-24\bin\java
.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\acer\AppData\Roaming\Code\User
\workspaceStorage\7c77d59ac154d4a5244ebac8c93acd39\redhat.java\jdt_ws\Chapter12_dd8bc02c\bin' 'Server'
Server is ready to accept requests...

```

(client)



```

PROBLEMS 4 OUTPUT DEBUG CONSOLE TERMINAL PORTS
Run: Client + - [ ] [ ] ... ^ x

PS C:\Users\acer\Desktop\network-programming\LabChapterWise\Chapter12> c:: cd 'c:\Users\acer\Desktop\network-pr
ogramming\LabChapterWise\Chapter12'; & 'C:\Program Files\Java\jdk-24\bin\java.exe' '--enable-preview' '-XX:+Show
CodeDetailsInExceptionMessages' '-cp' 'C:\Users\acer\AppData\Roaming\Code\User\workspaceStorage\7c77d59ac154d4a5
244ebac8c93acd39\redhat.java\jdt_ws\Chapter12_dd8bc02c\bin' 'Client'
Result is: 70

```