1. WAP to print the hostname, host address, port and protocol of www.google.com .

Source Code :

```
package lab;
import java.net.InetAddress;
import java.net.URL;
public class googleinfo_swosti {
        public static void main(String[] args) {
            try {
               URL url = new URL("https://www.google.com");
               System.out.println("Host Name: " + url.getHost());
               System.out.println("Protocol: " + url.getProtocol());
                InetAddress address = InetAddress.getByName(url.getHost());
               System.out.println("Host Address: " + address.getHostAddress());
               int port = url.getPort();
               if (port == -1) {
                  port = url.getDefaultPort();  }
               System.out.println("Port: " + port);
            } catch (Exception e) {
               System.out.println(e);
            }}}
```

Output :

```
<terminated> googleinfo_swosti [Java Application] C:\Program Files\Java\jdk-24\bin\javaw.exe  (Nov 16, 2025,
Host Name: www.google.com
Protocol: https
Host Address: 142.250.192.196
Port: 443
```

2. WAP to print all the network interfaces available in a machine .

Source Code :

```
package lab;
import java.net.InetAddress;
import java.net.NetworkInterface;
import java.util.Enumeration;
public class networkinterfacelist_swosti {
        public static void main(String[] args) {
    try {
      Enumeration<NetworkInterface> interfaces = NetworkInterface.getNetworkInterfaces();
      while (interfaces.hasMoreElements()) {
        NetworkInterface ni = interfaces.nextElement();
        System.out.println("Interface Name: " + ni.getName());
        System.out.println("Display Name : " + ni.getDisplayName());
        Enumeration<InetAddress> addresses = ni.getInetAddresses();
        while (addresses.hasMoreElements()) {
          InetAddress addr = addresses.nextElement();
          System.out.println("IP Address: " + addr.getHostAddress());
        }
        System.out.println("-------------------------------------");
      }
    } catch (Exception e) {
      System.out.println("Error: " + e); }}}
```

Output :

```
<terminated> networkinterfacelist_swosti [Java Application] C:\Program Files\Java\jdk-24\bin\javaw.exe  (Nov 16, 2025, 6:16:33 PM – 6:16:33 PM elapsed: 0:00:00.200) [pid: 13156]
Interface Name: ethernet_0
Display Name : Realtek PCIe GbE Family Controller-WFP Native MAC Layer LightWeight Filter-0000
-------------------------------------
Interface Name: ethernet_1
Display Name : Realtek PCIe GbE Family Controller-QoS Packet Scheduler-0000
-------------------------------------
Interface Name: ethernet_2
Display Name : Realtek PCIe GbE Family Controller-WFP 802.3 MAC Layer LightWeight Filter-0000
-------------------------------------
Interface Name: ethernet_3
Display Name : WAN Miniport (IP)-WFP Native MAC Layer LightWeight Filter-0000
-------------------------------------
Interface Name: ethernet_4
Display Name : WAN Miniport (IP)-QoS Packet Scheduler-0000
-------------------------------------
Interface Name: ethernet_5
Display Name : WAN Miniport (IPv6)-WFP Native MAC Layer LightWeight Filter-0000
-------------------------------------
Interface Name: ethernet_6
Display Name : WAN Miniport (IPv6)-QoS Packet Scheduler-0000
-------------------------------------
Interface Name: ethernet_7
Display Name : WAN Miniport (Network Monitor)-WFP Native MAC Layer LightWeight Filter-0000
-------------------------------------
Interface Name: ethernet_8
Display Name : WAN Miniport (Network Monitor)-QoS Packet Scheduler-0000
-------------------------------------
Interface Name: ethernet_32768
Display Name : Microsoft Kernel Debug Network Adapter
-------------------------------------
Interface Name: ethernet_32769
Display Name : Realtek PCIe GbE Family Controller
IP Address: fe80:0:0:0:c1ea:e687:d2fb:684c%ethernet_32769
-------------------------------------
```

```
Interface Name: ethernet_32770
Display Name : Bluetooth Device (Personal Area Network)
IP Address: fe80:0:0:0:b24f:4433:b60c:f0b8%ethernet_32770
---------------------------------------
Interface Name: ethernet_32771
Display Name : WAN Miniport (IP)
---------------------------------------
Interface Name: ethernet_32772
Display Name : WAN Miniport (IPv6)
---------------------------------------
Interface Name: ethernet_32773
Display Name : WAN Miniport (Network Monitor)
---------------------------------------
Interface Name: ppp_32768
Display Name : WAN Miniport (PPPOE)
---------------------------------------
Interface Name: loopback_0
Display Name : Software Loopback Interface 1
IP Address: 0:0:0:0:0:0:0:1
IP Address: 127.0.0.1
---------------------------------------
Interface Name: wireless_0
Display Name : Intel(R) Wireless-AC 9560 160MHz-WFP Native MAC Layer LightWeight Filter-0000
---------------------------------------
Interface Name: wireless_1
Display Name : Intel(R) Wireless-AC 9560 160MHz-Virtual WiFi Filter Driver-0000
---------------------------------------
Interface Name: wireless_2
Display Name : Intel(R) Wireless-AC 9560 160MHz-Native WiFi Filter Driver-0000
---------------------------------------
Interface Name: wireless_3
Display Name : Intel(R) Wireless-AC 9560 160MHz-QoS Packet Scheduler-0000
---------------------------------------
Interface Name: wireless_4
Display Name : Intel(R) Wireless-AC 9560 160MHz-WFP 802.3 MAC Layer LightWeight Filter-0000
---------------------------------------

Interface Name: wireless_5
Display Name : Microsoft Wi-Fi Direct Virtual Adapter-QoS Packet Scheduler-0000
---------------------------------------
Interface Name: wireless_6
Display Name : Microsoft Wi-Fi Direct Virtual Adapter-WFP Native MAC Layer LightWeight Filter-0000
---------------------------------------
Interface Name: wireless_7
Display Name : Microsoft Wi-Fi Direct Virtual Adapter-Native WiFi Filter Driver-0000
---------------------------------------
Interface Name: wireless_8
Display Name : Microsoft Wi-Fi Direct Virtual Adapter-WFP 802.3 MAC Layer LightWeight Filter-0000
---------------------------------------
Interface Name: wireless_9
Display Name : Microsoft Wi-Fi Direct Virtual Adapter #2-WFP Native MAC Layer LightWeight Filter-0000
---------------------------------------
Interface Name: wireless_10
Display Name : Microsoft Wi-Fi Direct Virtual Adapter #2-Native WiFi Filter Driver-0000
---------------------------------------
Interface Name: wireless_11
Display Name : Microsoft Wi-Fi Direct Virtual Adapter #2-QoS Packet Scheduler-0000
---------------------------------------
Interface Name: wireless_12
Display Name : Microsoft Wi-Fi Direct Virtual Adapter #2-WFP 802.3 MAC Layer LightWeight Filter-0000
---------------------------------------
Interface Name: wireless_32768
Display Name : Intel(R) Wireless-AC 9560 160MHz
IP Address: fe80:0:0:0:ba88:9173:8f70:23f2%wireless_32768
IP Address: 2405:acc0:1207:e52a:c014:7552:f9f1:93fe
IP Address: 2405:acc0:1207:e52a:8543:2eef:136a:f2ce
IP Address: 2405:acc0:1207:e52a:0:0:0:1
IP Address: 192.168.18.4
---------------------------------------
Interface Name: wireless_32769
Display Name : Microsoft Wi-Fi Direct Virtual Adapter
IP Address: fe80:0:0:0:a30c:2bca:80c3:c149%wireless_32769
---------------------------------------

Interface Name: wireless_32770
Display Name : Microsoft Wi-Fi Direct Virtual Adapter #2
IP Address: fe80:0:0:0:301f:9947:1194:4c58%wireless_32770
---------------------------------------
Interface Name: tunnel_32512
Display Name : Microsoft Teredo Tunneling Adapter
---------------------------------------
Interface Name: tunnel_32513
Display Name : Microsoft IP-HTTPS Platform Adapter
---------------------------------------
Interface Name: tunnel_32514
Display Name : Microsoft 6to4 Adapter
---------------------------------------
Interface Name: tunnel_32768
Display Name : WAN Miniport (SSTP)
---------------------------------------
Interface Name: tunnel_32769
Display Name : WAN Miniport (IKEv2)
---------------------------------------
Interface Name: tunnel_32770
Display Name : WAN Miniport (L2TP)
---------------------------------------
Interface Name: tunnel_32771
Display Name : WAN Miniport (PPTP)
---------------------------------------
```

3. WAP to print the network interface of “localhost”.

Source Code :

```
package lab;
import java.net.*;
public class localhost_swosti {
public static void main(String[] args) {
try {
InetAddress localHost = InetAddress.getLocalHost();
NetworkInterface ni = NetworkInterface.getByInetAddress(localHost);
if (ni != null) {
System.out.println("Interface Name: " + ni.getName());
System.out.println("Display Name: " + ni.getDisplayName());
byte[] mac = ni.getHardwareAddress();
if (mac != null) {
System.out.print("MAC Address: ");
for (int i = 0; i < mac.length; i++) {
System.out.format("%02X%s", mac[i], (i < mac.length - 1) ? "-" : "");}
System.out.println();}
System.out.println("IP Address: " + localHost.getHostAddress());
} else {
System.out.println("No network interface found for localhost.");}
} catch (Exception e) {
e.printStackTrace();
}}}
```

Output :

```
<terminated> localhost_swosti [Java Application] C:\Program Files\Java\jdk-24\bin\javaw.exe
Interface Name: wireless_32768
Display Name: Intel(R) Wireless-AC 9560 160MHz
MAC Address: A0-E7-0B-8F-C0-00
IP Address: 192.168.18.4
```

4. WAP to test a URL is reachable within 2 seconds.

Source Code :

```
package lab;
import java.net.*;
public class urlreachability_swosti {
public static void main(String[] args) {
String link = "https://www.google.com"; // URL to test
try {  URL url = new URL(link);
HttpURLConnection connection = (HttpURLConnection) url.openConnection();
connection.setConnectTimeout(2000); // 2 seconds
connection.setReadTimeout(2000); // 2 seconds
connection.setRequestMethod("GET");
int responseCode = connection.getResponseCode();
if (responseCode == 200) {
System.out.println(link + " is reachable.");
} else {
System.out.println(link + " is NOT reachable. Response code: " +
responseCode);}
} catch (Exception e) {
System.out.println(link + " is NOT reachable. Error: " + e.getMessage());}}}
```

Output :

```
<terminated> urlreachability_swosti [Java Application] C:\Program Files\Java\jdk-24\bin\javaw.exe
https://www.google.com is reachable.
```

5. You are given an IP address as “255.234.180.23”, you are asked to check its address type.

Source Code :

```
package lab;
import java.net.*;
public class addresstype_swosti {
public static void main(String[] args) {
try {
InetAddress ip = InetAddress.getByName("255.234.180.23");
System.out.println("IP Address: " + ip.getHostAddress());
if (ip.isAnyLocalAddress())
System.out.println("Any Local Address");
else if (ip.isLoopbackAddress())
System.out.println("Loopback Address");
else if (ip.isMulticastAddress())
System.out.println("Multicast Address");
else if (ip.isLinkLocalAddress())
System.out.println("Link-Local Address");
else if (ip.isSiteLocalAddress())
System.out.println("Site-Local Address");
else
System.out.println("Global Address");
} catch (Exception e) {
System.out.println(e);}}}
```

Output :

```
<terminated> addresstype_swosti [Java Application] C:\Program Files\Java\jdk-24\bin\javaw.exe
IP Address: 255.234.180.23
Global Address
```

6. WAP to check if a given IP address is of v4 or v6.

Source Code :

```
package lab;
import java.net.*;
public class checkIPv_swosti {
public static void main(String[] args) {
try {
InetAddress ip = InetAddress.getByName("255.234.180.23"); // change IP here
if (ip instanceof Inet4Address) {
System.out.println("The IP address is IPv4");
} else if (ip instanceof Inet6Address) {
System.out.println("The IP address is IPv6");
}
} catch (Exception e) {
System.out.println(e);
}}}
```

Output :

```
<terminated> checkIPv_swosti [Java Application] C:\Program Files\Java\jdk-24\bin\javaw.exe
The IP address is IPv4
```

7. WAP to print the IP address and MAC address of any host.

Source Code :

```
package lab;
import java.net.*;
public class IPandMAC_swosti {
public static void main(String[] args) {
try {
// You can change this to any host (example: "www.google.com")
String host = "localhost";
InetAddress address = InetAddress.getByName(host);
System.out.println("Host: " + host);
System.out.println("IP Address: " + address.getHostAddress());
NetworkInterface ni = NetworkInterface.getByInetAddress(address);
if (ni != null) {
byte[] mac = ni.getHardwareAddress();
if (mac != null) {
StringBuilder sb = new StringBuilder();
for (byte b : mac) {
sb.append(String.format("%02X-", b));}
System.out.println("MAC Address: " + sb.substring(0, sb.length() - 1));
} else {
System.out.println("MAC Address: Not available");}
} else {
System.out.println("Network Interface not found.");}
} catch (Exception e) {
System.out.println("Error: " + e.getMessage());}}}
```

Output :

```
<terminated> IPandMAC_swosti [Java Application] C:\Program Files\Java\jdk-24\bin\javaw.exe
Host: localhost
IP Address: 127.0.0.1
MAC Address: Not available
```

8. WAP to print scheme, authority, path, section and query string(alsoknown as parts of URL) of

https://example.com/en/4.2/querysets/#queryset-api.

```
package lab;
import java.net.URL;
public class URLparts_swosti {
public static void main(String[] args) {
try {
URL url = new URL("https://example.com/en/4.2/querysets/#queryset-api");
System.out.println("Scheme : " + url.getProtocol());
System.out.println("Authority : " + url.getAuthority());
System.out.println("Path : " + url.getPath());
System.out.println("Query : " + url.getQuery());
System.out.println("Fragment : " + url.getRef()); // section part
} catch (Exception e) {
System.out.println(e);}}}
```

Output :

```
<terminated> URLparts_swosti [Java Application] C:\Program Files\Java\jdk-24\bin\javaw.exe
Scheme     : https
Authority  : example.com
Path       : /en/4.2/querysets/
Query      : null
Fragment   : queryset-api
```

9. You are given a URI “http://example.com” and a path “colleges/SWASTIK#admission”, now resolve them to form a complete URI.

Source Code :

```
package lab;
import java.net.URI;
import java.net.URISyntaxException;
public class URItest_swosti {
public static void main(String[] args) {
try {
URI baseURI = new URI("http://example.com");
String relativePath = "colleges/SWASTIK#admission";
URI completeURI = baseURI.resolve(relativePath);
System.out.println("Complete URI: " + completeURI.toString());
} catch (URISyntaxException e) {
System.out.println("Error: Invalid URI syntax.");
e.printStackTrace();
}}}
```

Output :

```
<terminated> URItest_swosti [Java Application] C:\Program Files\Java\jdk-24\bin\javaw.exe
Complete URI: http://example.com/colleges/SWASTIK#admission
```

10. WAP to download a web page in Java.

Source Code :

```
package lab;
import java.net.*;
import java.io.*;
public class DownloadWebPage_swosti {
public static void main(String[] args) throws Exception {
URL url = new URL("https://example.com");
URLConnection connection = url.openConnection();
InputStream raw = connection.getInputStream();
InputStream buffer = new BufferedInputStream(raw);
Reader reader = new InputStreamReader(buffer);
int c;
//The read() method reads a single character from the stream and returns
its Unicode value as an int (range: 0 to 65535)
while ((c = reader.read()) != -1) {
System.out.print((char) c);
}
reader.close();
}
}
```

Output :

```
<terminated> DownloadWebPage_swosti [Java Application] C:\Program Files\Java\jdk-24\bin\javaw.exe (Dec 28, 2025, 7:35:48 PM
<!doctype html><html lang="en"><head><title>Example Domain</title><meta name="viewport" content=
```

11. WAP to encode “This string has space” and then decode it to the original string.

Source Code :

```
package lab;
import java.net.URLDecoder;
import java.net.URLEncoder;
public class EncodeDecode_swosti {
public static void main(String[] args) {
try {
String url = "This string has space";
String encodedUrl = URLEncoder.encode(url, "UTF-8");
System.out.println(encodedUrl);
String decodedUrl = URLDecoder.decode(encodedUrl, "UTF-8");
System.out.println(decodedUrl);
} catch (Exception e) {
e.printStackTrace();
}
}
}
```

Output :

```
<terminated> EncodeDecode_swosti [Java Application] C:\Program Files\Java\jdk-24\bin\javaw.exe
This+string+has+space
This string has space
```

12. WAP to retrieve the cookie information and store them in the system.

Source Code :

```
package lab;
import java.net.*;
import java.util.*;
public class CookieInfo_swosti {
public static void main(String[] args) {
try {
CookieManager manager = new CookieManager();
CookieHandler.setDefault(manager);
URL url = new URL("https://www.google.com");
URLConnection connection = url.openConnection();
connection.getContent();
CookieStore store = manager.getCookieStore();
List<HttpCookie> cookies = store.getCookies();
System.out.println("Cookies Retrieved:");
for (HttpCookie cookie : cookies) {
System.out.println("Name : " + cookie.getName());
System.out.println("Value : " + cookie.getValue());
System.out.println("Domain: " + cookie.getDomain());
System.out.println("----------------------");
}
} catch (Exception e) {
System.out.println(e);}}}
```

Output :

```
<terminated> CookieInfo_swosti [Java Application] C:\Program Files\Java\jdk-24\bin\jav
Domain: .google.com
----------------------
Name   : __Secure-BUCKET
Value : CLkG
Domain: .google.com
----------------------
```

13. WAP to demonstrate the ProxySelector class in Java.

Source Code :

```
package lab;
import java.net.*;
import java.io.*;
import java.util.List;
public class ProxySelector_swosti {
public static void main(String[] args) {
try {
ProxySelector.setDefault(new ProxySelector() {
@Override
public List<Proxy> select(URI uri) {
System.out.println("Selecting proxy for URI: " + uri);
return List.of(Proxy.NO_PROXY);
}
@Override
public void connectFailed(URI uri, SocketAddress sa, IOException ioe) {
System.out.println("Connection failed to " + uri + " via " + sa);
}
});
URI uri = new URI("http://www.google.com");
ProxySelector selector = ProxySelector.getDefault();
List<Proxy> proxies = selector.select(uri);
System.out.println("Proxies returned for " + uri + ":");
for (Proxy proxy : proxies) {
System.out.println(proxy);
}
} catch (Exception e) {
e.printStackTrace();
}
}
}
```

Output :

```
<terminated> ProxySelector_swosti [Java Application] C:\Program Files\Java\jdk-24\bin\javaw.exe
Selecting proxy for URI: http://www.google.com
Proxies returned for http://www.google.com:
DIRECT
```

14. WAP to demonstrate the Authenticator class in Java.

Source Code :

```
package lab;
import java.net.*;
import java.io.*;
public class Authenticator_swosti extends Authenticator {
@Override
protected PasswordAuthentication getPasswordAuthentication() {
// Prompt the user for a username and password
String username = "03268048";
String password = "wefef";
// Return the authentication credentials
return new PasswordAuthentication(username, password.toCharArray());
}

public static void main(String[] args) {
// Set the default authenticator to your custom authenticator
Authenticator.setDefault(new Authenticator_swosti());

try {
// Create a URL object
URL url = new URL("https://web.facebook.com/#/login");

// Open a connection to the URL
URLConnection connection = url.openConnection();

// Read the response from the server
BufferedReader reader = new BufferedReader(new
InputStreamReader(connection.getInputStream()));
String line;
while ((line = reader.readLine()) != null) {
System.out.println(line);
}
reader.close();
} catch (IOException e) {
System.err.println("Error: " + e.getMessage());
}
}
}
```

Output :

```
<terminated> Authenticator_swosti [Java Application] C:\Program Files\Java\jdk-24\bin\javaw.exe  (Dec 31, 2025, 3:21:18 PM – 3:21:20 PM elapsed: 0:00:01.427) [pid: 15272]
<!DOCTYPE html>
<html lang="en" id="facebook" class="no_js">
<head><meta charset="utf-8" /><meta name="referrer" content="default" id="meta_referrer" /><script nonce="FJpWmuOw">function envFlush(e){function t(t){for
<link type="text/css" rel="stylesheet" href="https://static.xx.fbcdn.net/rsrc.php/v5/yq/l/0,cross/GqMCJfpI75k.css" data-bootloader-hash="OOmeQkK" crossori
<link type="text/css" rel="stylesheet" href="https://static.xx.fbcdn.net/rsrc.php/v5/y4/l/0,cross/ILQtRPO-q6Y.css" data-bootloader-hash="pXv0a7+" crossori
<script src="https://static.xx.fbcdn.net/rsrc.php/v4/yN/r/QO7_WOZoD9p.js" data-bootloader-hash="dThAkGE" crossorigin="anonymous"></script>
<script nonce="FJpWmuOw">requireLazy(["HasteSupportData"],function(m){m.handle({"clpData":{"6476":{"r":1000,"s":1},"1838142":{"r":1,"s":1},"1814852":{"r":
<script nonce="FJpWmuOw">requireLazy(["HasteSupportData"],function(m){m.handle({"hxData":{"875231":{"uri":"https:\/\/static.xx.fbcdn.net\/rsrc.php\/v1\/r\
```

15. WAP to print the last modified date, content length, and content type of a URL of your choice.

Source Code :

```
package lab;
import java.net.*;
import java.io.*;
import java.text.SimpleDateFormat;
import java.util.Date;
public class URLInfo_swosti {
public static void main(String[] args) {
try {
// URL of your choice
URL url = new URL("https://www.example.com");

// Open a connection to the URL
URLConnection connection = url.openConnection();

// Get last modified date
long lastModified = connection.getLastModified();
String lastModifiedStr = (lastModified == 0) ? "Not available"
: new SimpleDateFormat("dd/MM/yyyy HH:mm:ss").format(new
Date(lastModified));

// Get content length
int contentLength = connection.getContentLength();

// Get content type
String contentType = connection.getContentType();

// Print the information
System.out.println("URL: " + url);
System.out.println("Last Modified: " + lastModifiedStr);
System.out.println("Content Length: " + ((contentLength == -1) ? "Not
available" : contentLength + " bytes"));
System.out.println("Content Type: " + contentType);

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

Output :

```
<terminated> URLInfo_swosti [Java Application] C:\Program Files\Java\jdk-24\bin\javaw.exe
URL: https://www.example.com
Last Modified: 20/12/2025 11:27:21
Content Length: Not available
Content Type: text/html
```

16. WAP to print all the headers and its corresponding values present in any HTTP response.

Source Code :

```
package lab;
import java.net.*;
import java.io.*;
import java.util.*;

public class HTTPHeader_swosti {


public static void main(String[] args) {
try {
// URL of your choice
URL url = new URL("https://www.example.com");

// Open connection
HttpURLConnection connection = (HttpURLConnection) url.openConnection();

// Optional: Set request method (GET is default)
connection.setRequestMethod("GET");

// Connect to the URL
connection.connect();

// Get all headers
System.out.println("HTTP Response Headers:");
Map<String, List<String>> headers = connection.getHeaderFields();
for (Map.Entry<String, List<String>> entry : headers.entrySet()) {
String headerName = entry.getKey();
List<String> headerValues = entry.getValue();
System.out.println((headerName != null ? headerName : "Status") + ": " +
String.join(", ", headerValues));
}

// Disconnect
connection.disconnect();

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

Output :

```
<terminated> HTTPHeader_swosti [Java Application] C:\Program Files\Java\jdk-24\bin\javaw.exe
CF-RAY: 9b51770a8b7786dd-KTM
Server: cloudflare
Connection: keep-alive
Date: Sun, 28 Dec 2025 13:39:39 GMT
Age: 2784
Content-Type: text/html
```

17. WAP to read from a socket.

Source Code :

```
package lab;

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

public class ReadFromSocket_swosti {
public static void main(String[] args) {
try (Socket socket = new Socket("time.nist.gov", 13)) {
socket.setSoTimeout(15000);
InputStream in = socket.getInputStream();

InputStreamReader reader = new InputStreamReader(in);
BufferedReader br = new BufferedReader(reader);
String line;
while((line=br.readLine()) != null){
System.out.println(line);
}
} catch (IOException e) {
System.out.println("something went wrong");
}
}
}
```

Output :

```
<terminated> ReadFromSocket_swosti [Java Application] C:\Program Files\Java\jdk-24\bin\javaw.exe

61040 25-12-31 10:27:52 00 0 0  33.7 UTC(NIST) *
```

18. WAP to create a server socket and print the log message, message on the console whenever a new client is connected, or if any error occurs.

Source Code :

```
package lab;

import java.net.ServerSocket;
import java.net.Socket;

public class ServerLog_swosti {
public static void main(String[] args) {
try {
// Create server socket on port 5000
ServerSocket serverSocket = new ServerSocket(5000);
System.out.println("Server started. Waiting for clients...");

while (true) {
// Wait for client connection
Socket client = serverSocket.accept();
System.out.println("New client connected: "
+ client.getInetAddress().getHostAddress());
}

} catch (Exception e) {
System.out.println("Error occurred: " + e.getMessage());
}
}
}
```

Output :

```
ServerLog_swosti [Java Application] C:\Program Files\Java\jdk-24\bin\javaw.exe
Server started. Waiting for clients...
```

19. Write a client-server socket program such that the client program will read an integer value from the user as standard input and send it to a server which will return the factorial of that number.

Source Code :

Server_swosti:

```
package lab19;
import java.net.*;
import java.io.*;
public class Server_swosti {
       public static void main(String[] args) {
        try {
            ServerSocket serverSocket = new ServerSocket(5000);
            System.out.println("Server started. Waiting for client...");

            Socket socket = serverSocket.accept();
            System.out.println("Client connected.");

            DataInputStream in = new
DataInputStream(socket.getInputStream());
            DataOutputStream out = new
DataOutputStream(socket.getOutputStream());

            int number = in.readInt();

            long fact = 1;
            for (int i = 1; i <= number; i++) {
                fact = fact * i;
            }

            out.writeLong(fact);

            socket.close();
            serverSocket.close();

        } catch (Exception e) {
            System.out.println(e);
        }
    }
}
```

Client_swosti:

```
package lab19;
import java.net.*;
import java.io.*;
import java.util.*;
public class Client_swosti {
        public static void main(String[] args) {
               try {
                   Socket socket = new Socket("localhost", 5000);

                   DataInputStream in = new
DataInputStream(socket.getInputStream());
                   DataOutputStream out = new
DataOutputStream(socket.getOutputStream());

                   Scanner sc = new Scanner(System.in);
                   System.out.print("Enter a number: ");
                   int num = sc.nextInt();

                   out.writeInt(num);

                   long result = in.readLong();
                   System.out.println("Factorial: " + result);

                   socket.close();

               } catch (Exception e) {
                   System.out.println(e);
               }
           }
}
```

```
Server_swosti [Java Application] C:\Program Files\Java\jdk-24\bin\javaw.exe
Server started. Waiting for client...
```

```
<terminated> Server_swosti [Java Application] C:\Program Files\Java\jdk-24\bin\javaw.exe
Server started. Waiting for client...
Client connected.
```

```
<terminated> Client_swosti [Java Application] C:\Program Files\Java\jdk-24\bin\javaw.exe
Enter a number: 6
Factorial: 720
```

20. Write a multithreaded client server program for daytime service.

Source Code :

DaytimeServer_swosti:

```
package lab_multithreaded;
import java.net.*;
import java.io.*;
import java.util.Date;
public class DaytimeServer_swosti {
public static void main(String[] args) {
int port=13;
try (ServerSocket server = new ServerSocket(port)) {
System.out.println("Daytime Server Started on port"+port);
while (true) {
try (Socket connection = server.accept()) {
Writer out = new OutputStreamWriter(connection.getOutputStream());
Date now = new Date();
out.write(now.toString() + "\n");
out.flush();
connection.close();
}
catch (IOException ex) {
}
}
} catch (IOException ex) {
System.err.println(ex);
}
}
}
```

DaytimeClient_swosti:

```
package lab_multithreaded;
import java.net.*;
import java.io.*;
public class DaytimeClient_swosti {
public static void main(String[] args) {
String serverAddress = "localhost"; int port = 13;
try (Socket socket = new Socket(serverAddress, port)) {
BufferedReader in = new BufferedReader(new
InputStreamReader(socket.getInputStream()));
String serverResponse = in.readLine();
System.out.println("Server Date and Time: " + serverResponse);
} catch (IOException ex) {
System.err.println(ex);
}
}
}
```

Output :

```
DaytimeServer_swosti [Java Application] C:\Program Files\Java\jdk-24\bin\javaw.exe
Daytime Server Started on port13

<terminated> DaytimeClient_swosti (1) [Java Application] C:\Program Files\Java\jdk-24\bin\javaw.exe
Server Date and Time: Sun Jan 04 17:30:16 GMT+05:45 2026
```

21. Write a multithreaded client server program for daytime service, use a threadpool of 50.

Source Code :

DaytimeServer_swosti:

```
package lab_threadpool;
import java.net.*;
import java.io.*;
import java.util.Date;
import java.util.concurrent.*;
public class DaytimeServer_swosti {
public static void main(String[] args) {
int port = 1313;
int poolSize = 50;
ExecutorService pool = Executors.newFixedThreadPool(poolSize);
try (ServerSocket server = new ServerSocket(port)) {
System.out.println("Multithreaded Daytime Server Started on port " + port);
while (true) {
Socket connection = server.accept();
pool.submit(() -> {
try {
System.out.println("Client connected: " +
connection.getInetAddress().getHostAddress());
Writer out = new OutputStreamWriter(connection.getOutputStream());
Date now = new Date();
out.write(now.toString() + "\n");
out.flush();
connection.close();
System.out.println("Connection closed.");
} catch (IOException e) {
System.err.println("Error: " + e.getMessage());
}
});
}
} catch (IOException ex) {
System.err.println(ex);}}}
```

DaytimeClient_swosti:

```
package lab_threadpool;
import java.net.*;
import java.io.*;
public class DaytimeClient_swosti {
public static void main(String[] args) {
String serverAddress = "localhost";
int port = 1313;
try (Socket socket = new Socket(serverAddress, port)) {
BufferedReader in = new BufferedReader(new
InputStreamReader(socket.getInputStream()));
String serverResponse = in.readLine();
System.out.println("Server Date and Time: " + serverResponse);
} catch (IOException ex) {
System.err.println(ex);}}}
```

Output :

```
DaytimeServer_swosti (1) [Java Application] C:\Program Files\Java\jdk-24\bin\javaw.exe
Multithreaded Daytime Server Started on port 1313

<terminated> DaytimeClient_swosti (2) [Java Application] C:\Program Files\Java\jdk-24\bin\javaw.exe
Server Date and Time: Sun Jan 04 17:36:44 GMT+05:45 2026
```

22.Create a secure client socket to read from a secure server socket.

Source Code :

SecureServer_swosti:

```
package lab_securesocket;
import java.io.*;
import javax.net.ssl.*;
public class SecureServer_swosti {
public static void main(String[] args) {
int port = 8443; try {
SSLServerSocketFactory sslServerFactory =
(SSLServerSocketFactory) SSLServerSocketFactory.getDefault();
SSLServerSocket sslServerSocket =
(SSLServerSocket) sslServerFactory.createServerSocket(port);
System.out.println("Secure Server started on port " + port);
SSLSocket sslSocket = (SSLSocket) sslServerSocket.accept();
System.out.println("Client connected securely!");
BufferedReader in = new BufferedReader(
new InputStreamReader(sslSocket.getInputStream()));
PrintWriter out = new PrintWriter(
sslSocket.getOutputStream(), true);
String received = in.readLine();
System.out.println("Client says: " + received);
out.println("Secure Server response: received \"" + received + "\"");
out.close();
in.close();
sslSocket.close();
sslServerSocket.close();
} catch (IOException ex) {
System.err.println("Server error: " + ex.getMessage());
}
}
}
```

SecureClient_swosti:

```
package lab_securesocket;
import java.io.*;
import javax.net.ssl.*;
public class SecureClient_swosti {
public static void main(String[] args) {
String host = "localhost";
int port = 8443;
try {
SSLSocketFactory sslSocketFactory =
(SSLSocketFactory) SSLSocketFactory.getDefault();
SSLSocket sslSocket =
(SSLSocket) sslSocketFactory.createSocket(host, port);
System.out.println("Connected securely to server!");
PrintWriter out = new PrintWriter(
sslSocket.getOutputStream(), true);
BufferedReader in = new BufferedReader(
new InputStreamReader(sslSocket.getInputStream()));
out.println("Hello Secure Server!");
System.out.println("Server says: " + in.readLine());
in.close();
out.close();
sslSocket.close();
```

```
} catch (IOException ex) {
System.err.println("Client error: " + ex.getMessage());
}
}
}
```

Output :

```
SecureServer_swosti (1) [Java Application] C:\Program Files\Java\jdk-24\bin\javaw.exe
Secure Server started on port 8443
<terminated> SecureServer_swosti (1) [Java Application] C:\Program Files\Java\jdk-24\bin\javaw.exe
Secure Server started on port 8443
Client connected securely!
Server error: (handshake_failure) No available authentication scheme
```

23.WAP to create a UDP client-socket echo program.

Source Code :

UDPClient_swosti:

```
package lab_UDP;
import java.net.*;
import java.util.Scanner;
public class UDPClient_swosti {

	public static void main(String args[]) throws Exception {
				byte[] sendData = new byte[1024];
		byte[] receiveData = new byte[1024];
		System.out.println("Say Something: ");
		Scanner sc = new Scanner(System.in);
		String inFromUser = sc.nextLine();
		DatagramSocket clientSocket = new DatagramSocket(0);
		InetAddress IPAddress = InetAddress.getByName("localhost");

		sendData = inFromUser.getBytes();
		DatagramPacket sendPacket = new DatagramPacket(sendData,
sendData.length, IPAddress, 9876);
		clientSocket.send(sendPacket);
		DatagramPacket receivePacket = new DatagramPacket(receiveData,
receiveData.length);
		clientSocket.receive(receivePacket);
		String modifiedSentence = new String(receivePacket.getData());
		System.out.println("FROM SERVER:" + modifiedSentence);
		clientSocket.close();
	}
}
```

UDPSocket_swosti:

```
package lab_UDP;
import java.net.*;
public class UDPSocket_swosti {
public static void main(String args[]) throws Exception {
byte[] receiveData = new byte[1024];
byte[] sendData = new byte[1024];
DatagramSocket serverSocket = new DatagramSocket(9876);
while(true) {
DatagramPacket receivePacket = new DatagramPacket(receiveData,
receiveData.length);
serverSocket.receive(receivePacket);
String sentence = new String(receivePacket.getData());
System.out.println("Client says: "+ sentence);
InetAddress IPAddress = receivePacket.getAddress();
```

```
int port = receivePacket.getPort();
sendData = sentence.getBytes();
DatagramPacket sendPacket = new DatagramPacket(sendData, sendData.length,
IPAddress, port);
serverSocket.send(sendPacket);
}
}
}
```

Output:

```
UDPClient_swosti [Java Application] C:\Program Files\Java\jdk-24\bin\javaw.exe
Say Something:
This is NP lab.
```

```
<terminated> UDPClient_swosti [Java Application] C:\Program Files\Java\jdk-24\bin\javaw.exe
Say Something:
This is NP lab.
FROM SERVER:This is NP lab.
```

24.WAP to join a computer system in a multicast group.

Source Code :

multicastjoiner1_swosti :

```
package lab_multicast;
import java.net.DatagramPacket;
import java.net.InetAddress;
import java.net.MulticastSocket;
public class multicastjoiner1_swosti {
@SuppressWarnings("deprecation")
public static void main(String[] args) {
try {
// Define the multicast group address and port
InetAddress multicastAddress = InetAddress.getByName("224.0.0.1");
int multicastPort = 8888;

// Create a multicast socket and join the group
MulticastSocket socket = new MulticastSocket(multicastPort);
socket.joinGroup(multicastAddress);

System.out.println("Joined multicast group: " +
multicastAddress.getHostAddress());

// Create a buffer to store received data
byte[] buffer = new byte[1024];

// Continuously receive and process incoming multicast packets
while (true) {
DatagramPacket packet = new DatagramPacket(buffer, buffer.length);
socket.receive(packet);

// Extract the received data from the packet
byte[] receivedData = packet.getData();
int receivedLength = packet.getLength();
String message = new String(receivedData, 0, receivedLength);

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

// Add your logic to process the received message here

// Check if you want to exit the loop and leave the multicast group
// For example, if a certain condition is meet
if (message.equals("exit")) {
break;
}
}

// Leave the multicast group and close the socket
socket.leaveGroup(multicastAddress);
socket.close();

System.out.println("Left multicast group");
} catch (Exception e) {
e.printStackTrace();
}
}
}
```

multicastjoiner2_swosti :

```
package lab_multicast;
import java.net.DatagramPacket;
import java.net.InetAddress;
import java.net.MulticastSocket;
public class multicastjoiner2_swosti {
@SuppressWarnings("deprecation") //@SuppressWarnings("deprecation")
annotation in Java is used to tell the compiler to ignore warnings related
to the use of deprecated code.
public static void main(String[] args) {
try {
// Define the multicast group address and port
InetAddress multicastAddress = InetAddress.getByName("224.0.0.1");
int multicastPort = 8888;

// Create a multicast socket and join the group
MulticastSocket socket = new MulticastSocket(multicastPort);
socket.joinGroup(multicastAddress);

System.out.println("Joined multicast group: " +
multicastAddress.getHostAddress());

// Create a buffer to store received data
byte[] buffer = new byte[1024];

// Continuously receive and process incoming multicast packets
while (true) {
DatagramPacket packet = new DatagramPacket(buffer, buffer.length);
socket.receive(packet);

// Extract the received data from the packet
byte[] receivedData = packet.getData();
int receivedLength = packet.getLength();
String message = new String(receivedData, 0, receivedLength);

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

// Add your logic to process the received message here

// Check if you want to exit the loop and leave the multicast group
// For example, if a certain condition is meet
if (message.equals("exit")) {
break;
}
}

// Leave the multicast group and close the socket
socket.leaveGroup(multicastAddress);
socket.close();

System.out.println("Left multicast group");
} catch (Exception e) {
e.printStackTrace();
}
}
}
```

Multicast :

```
package lab_multicast;
import java.net.DatagramPacket;
import java.net.InetAddress;
import java.net.MulticastSocket;
public class multicast {
public static void main(String[] args) {
try {
// Define the multicast group address and port
InetAddress multicastAddress = InetAddress.getByName("224.0.0.1");
int multicastPort = 8888;

// Create a multicast socket
MulticastSocket socket = new MulticastSocket();

// Create the message to be sent
String message = "Hello, multicast!";
byte[] data = message.getBytes();

// Create a datagram packet with the message and destination address/port
DatagramPacket packet = new DatagramPacket(data, data.length,
multicastAddress, multicastPort);

// Send the packet
socket.send(packet);

// Close the socket
socket.close();
} catch (Exception e) {
e.printStackTrace();
}
}
}
```

Output :

```
multicastjoiner1_swosti [Java Application] C:\Program Files\Java\jdk-24\bin\javaw.exe
Joined multicast group: 224.0.0.1

multicastjoiner2_swosti [Java Application] C:\Program Files\Java\jdk-24\bin\javaw.exe
Joined multicast group: 224.0.0.1

multicastjoiner2_swosti [Java Application] C:\Program Files\Java\jdk-24\bin\javaw.exe
Joined multicast group: 224.0.0.1
Received message: Hello, multicast!
Received message: Hello, multicast!
Received message: Hello, multicast!
```