Unit 2- Internet Address

1. **To print the address of** [**www.hippo.com**](http://www.hippo.com)

public class InetClass2 {  
  
 public static void main(String []args) throws UnknownHostException{  
 InetAddress obj= InetAddress.*getByName*("www.hippo.com");  
 System.*out*.println("IP address: "+obj.getHostAddress());  
 }  
}

Output:

IP address: 104.18.7.243

1. **To find the address of local machine**.

public class InetClass2 {  
  
 public static void main(String []args) throws UnknownHostException{  
 InetAddress obj= InetAddress.*getLocalHost*();  
 System.*out*.println("IP address: "+obj.getHostAddress());  
  
  
 }  
}

Output:

IP address: 192.168.1.85

1. **To determine whether an ip address is ipV4 or ipV6**

import java.net.InetAddress;  
import java.net.UnknownHostException;

public class IPAddressDetermination {  
 public static void main(String []args) throws UnknownHostException {  
 InetAddress obj= InetAddress.*getByName*("142.250.192.100");  
 byte[] address = obj.getAddress();  
*// first way* if(address.length==4){  
 System.*out*.println("Its IP V4");  
 }  
 else if(address.length==16){  
 System.*out*.println("Its Ip V6");  
 }  
 else{  
 System.*out*.println("None");  
 }  
*//second way*if(obj instanceof java.net.Inet4Address){  
 System.*out*.println("The address is IPV4");  
}  
else if(obj instanceof java.net.Inet6Address){  
 System.*out*.println("The address is IPV6");  
 }  
else{  
 System.*out*.println("Not an IP address");  
}  
  
 }  
}

Output: Its IP V4

The address is IPV4

1. **To get IPV4 and IPV6 address of a given web address** [**www.google.com**](http://www.google.com)**.**

import java.net.InetAddress;  
import java.net.UnknownHostException;  
public class GetIPAddresses {  
 public static void main(String[] args) throws UnknownHostException {  
 InetAddress[] addresses = InetAddress.*getAllByName*("www.google.com");  
  
 for (InetAddress address : addresses) {  
 if (address instanceof java.net.Inet4Address) {  
 System.*out*.println("IPV4 address of www.google.com "+address.getHostAddress());  
 } else if (address instanceof java.net.Inet6Address) {  
 System.*out*.println("IPV6 address of www.google.com "+address.getHostAddress());  
 } else {  
 System.*out*.println("Not an IP address");  
 }  
  
 }  
 }

}

Output:

IPV4 address of www.google.com 142.250.77.228

IPV6 address of www.google.com 2404:6800:4002:814:0:0:0:2004

1. **To list all the network interfaces of your system.**

import java.net.NetworkInterface;  
import java.net.SocketException;  
import java.net.UnknownHostException;  
import java.util.Enumeration;

public class InterfaceLister {  
 public static void main(String []args) throws SocketException,UnknownHostException {  
 Enumeration<NetworkInterface> address= NetworkInterface.*getNetworkInterfaces*();  
 while(address.hasMoreElements()){  
 NetworkInterface ni = address.nextElement();  
 System.*out*.println(ni.getName()+" "+ni.getMTU());  
 }  
 }  
}

Output:

lo -1

eth0 -1

wlan0 1500

net0 -1

net1 -1

net2 -1

net3 -1

wlan1 1500

eth1 1500

wlan2 -1

ppp0 -1

wlan3 -1

eth2 -1

net4 -1

net5 -1

net6 -1

eth3 -1

wlan4 1500

eth4 -1

ppp1 -1

eth5 -1

wlan5 -1

wlan6 -1

wlan7 -1

wlan8 -1

Unit 3 URLs and URIs

1. **Program that splits the parts of a URL(get the link of your fb profile).**

public class URLParser {  
  
 public static void main(String[] args) {  
 String url = "https://www.facebook.com/profile.php";  
  
 try {  
 URL u = new URL(url);  
 String protocol = u.getProtocol();  
 String domain = u.getHost();  
 String path = u.getPath();  
 String profile = path.substring(1);  
  
 if (domain.equals("www.facebook.com")) {  
 System.*out*.println("Your Facebook profile link is: https://" + domain + "/" + profile);  
 System.*out*.println("path is: " + path.substring(1));  
  
 } else {  
 System.*out*.println("This is not a valid Facebook URL.");  
 }  
  
 } catch (MalformedURLException e) {  
 System.*out*.println("Error: " + e.getMessage());  
 }  
 }  
}

Output:

Your Facebook profile link is: https://www.facebook.com/profile.php

path is: profile.php

1. **Program to download a web page of a given web address.**

public class WebPage {  
  
 public static void main(String[] args) {  
 String url = "https://www.example.com";  
 String filename = "example.html";  
  
 try {  
 URL webpage = new URL(url);  
 URLConnection connection = webpage.openConnection();  
 BufferedReader reader = new BufferedReader(new InputStreamReader(connection.getInputStream()));  
  
 String line;  
 StringBuilder content = new StringBuilder();  
 while ((line = reader.readLine()) != null) {  
 content.append(line);  
 content.append(System.*lineSeparator*());  
 }  
  
 reader.close();  
  
 FileWriter writer = new FileWriter(filename);  
 writer.write(content.toString());  
 writer.close();  
  
 System.*out*.println("Web page downloaded and saved to " + filename);  
 } catch (MalformedURLException e) {  
 System.*out*.println("Invalid URL: " + url);  
 e.printStackTrace();  
 } catch (IOException e) {  
 System.*out*.println("Error downloading web page: " + e.getMessage());  
 e.printStackTrace();  
 }  
 }  
}

Output:

Web page downloaded and saved to example.html

1. **Program for resolving relatives URI**

import java.net.URI;  
import java.net.URISyntaxException;  
  
public class RelativeUriResolver {  
 public static void main(String[] args) {  
 try {  
 URI baseUri = new URI("https://example.com/path/to/file");  
  
 URI relativeUri = new URI("newfile.html");  
  
 URI resolvedUri = baseUri.resolve(relativeUri);  
  
 System.*out*.println(resolvedUri.toString());  
 } catch (URISyntaxException e) {  
 System.*err*.println("Invalid URI syntax: " + e.getMessage());  
 }  
 }  
}

Output:

<https://example.com/path/to/newfile.html>

1. **Program that checks which protocols a virtual machine support or not.**

public class ProtocolChecker {  
 public static void main(String[] args) {  
 *checkProtocols*();  
 }  
  
 public static void checkProtocols() {  
 String[] protocols = {"http", "https", "ftp", "smtp", "pop3", "imap"};  
  
 for (String protocol : protocols) {  
 try {  
 URL url = new URL(protocol + "://www.google.com");  
 URLConnection connection = url.openConnection();  
 connection.connect();  
   
 System.*out*.println(protocol + " supported");  
 } catch (MalformedURLException e) {  
 System.*out*.println(e);  
  
 System.*out*.println(protocol + " not supported");  
 }catch(Exception e){  
 System.*out*.println(e);  
 }  
 }  
 }  
}

Output:

http supported

https supported

java.net.ConnectException: Connection timed out: no further information

java.net.MalformedURLException: unknown protocol: smtp

smtp not supported

java.net.MalformedURLException: unknown protocol: pop3

pop3 not supported

java.net.MalformedURLException: unknown protocol: imap

imap not supported

1. **Program demonstrating UrlEncoding and UrlDecoding.**

import java.net.URISyntaxException;  
import java.io.UnsupportedEncodingException;  
import java.net.URLEncoder;  
  
public class Urlencode {  
 public static void main(String args []) throws URISyntaxException, UnsupportedEncodingException  
 {  
  
 String data = "Simple text data!@#$";  
  
 String encodeData = URLEncoder.*encode*(data, "UTF-8");  
 System.*out*.println(encodeData);  
 }  
}

Output:

Simple+text+data%21%40%23%24

URLDecoding:

public class Urldecode{  
 public static void main(String args[]) throws UnsupportedEncodingException  
 { String data = "http://www.mysite.com/?video=funny%20cat%20plays%20piano.";  
 String result = URLDecoder.*decode*(data, "UTF-8");  
 System.*out*.println(result);  
  
 }  
}

Output:

http://www.mysite.com/?video=funny cat plays piano.

1. **Program communicating with server side programs through GET**

import java.io.BufferedInputStream;  
import java.io.IOException;  
import java.io.InputStream;  
import java.io.InputStreamReader;  
import java.net.MalformedURLException;  
import java.net.URL;  
import java.util.Scanner;  
  
public class GetExample  
{  
 public static void main(String[] args)  
 {  
 System.*out*.println("Enter YourKeyword:");  
 Scanner ashish=new Scanner(System.*in*);  
 String search= ashish.nextLine();  
 try  
 {  
 URL u=new URL("https://www.google.com/search?q");  
 try(InputStream in=new BufferedInputStream(u.openStream()))  
 {  
 InputStreamReader HTML=new InputStreamReader(in);  
 int c;  
 while((c=HTML.read())!=-1)  
 {  
 System.*out*.println((char)c);  
 }  
 }  
 ashish.close();  
 }  
 catch (MalformedURLException ex)  
 {  
 System.*out*.println(ex);  
 }  
 catch (IOException ax)  
 {  
 System.*out*.println(ax);  
 }  
 }  
}

1. **Program to get the object.**

import java.io.IOException;  
import java.net.MalformedURLException;  
import java.net.URL;  
  
public class ObjDownload *//ContentGettter*{  
 public static void main(String[] args)  
 {  
 *//open the URL for reading* try  
 {  
 URL url=new URL("http://www.google.com");  
 Object o=url.getContent();  
 System.*out*.println("I got:"+o.getClass().getName());  
 }  
 catch (MalformedURLException mle)  
 {  
 mle.printStackTrace();  
 }  
 catch (IOException ioe)  
 {  
 ioe.printStackTrace();  
 }  
 }  
}

Output:

I got:sun.net.www.protocol.http.HttpURLConnection$HttpInputStream

Unit 4 – HTTP

1. **Program to implement the cookie store methods(add, read ,delete).**

public class CookiesManager {  
 public static void main(String[] args) {  
 CookieManager cm = new CookieManager();  
 CookieStore cs = cm.getCookieStore();  
*//createing cookies and URI* HttpCookie c1 = new HttpCookie("user1", "1");  
 HttpCookie c2 = new HttpCookie("user2", "2");  
 HttpCookie c3 = new HttpCookie("user3", "3");  
 URI uri1 = URI.*create*("http://spm.com.np");  
 URI uri2 = URI.*create*("http://spm1.com.np");  
*// Add cookies into cookiestore* cs.add(uri1, c1);  
 cs.add(uri2,  
 c2);  
 cs.add(null, c3);  
*//read stored cookies* List cl = cs.getCookies();  
 System.*out*.println("cookies lst store" + cl +  
 "\n"); *//remove cookiestore of uri* cs.remove(uri1, c1);  
 List cr = cs.getCookies();  
 System.*out*.println("remaining CS" + cr + "\n");  
*// remove all cookies* cs.removeAll();  
 List empty = cs.getCookies();  
 System.*out*.println("remove all cs" + empty);  
 }  
}

Output:

cookies lst store[user1="1", user2="2", user3="3"]

remaining CS[user2="2", user3="3"]

remove all cs[]

1. **Program that shows a simple cookie policy that blocks cookies from .edu domain but allow others.**

public class Main implements CookiePolicy {  
 @Override  
 public boolean shouldAccept(URI uri, HttpCookie cookie) {  
 if (cookie.getDomain().endsWith(".edu")) {  
 return false;  
 } else {  
 return true;  
 }  
 }  
 public static void main(String[]args){  
 Main cp=new Main();  
  
 URI uri = URI.*create*("https://www.example.com");  
 HttpCookie cookie = new HttpCookie("name", "value");  
 cookie.setDomain(".edu");  
 boolean accept = cp.shouldAccept(uri, cookie);  
 System.*out*.println("Should accept cookie: " + accept);  
  
 }  
}

Output:

Should accept cookie: false

Unit 5 - URLConnection

1. **To download the web page using URLConnection.**

public class DownloadPageExample {  
 public static void main(String []args){  
 try{  
 URL resource = new URL("http://www.example.com");  
 URLConnection ur = resource.openConnection();  
 BufferedReader reader = new BufferedReader(new InputStreamReader(ur.getInputStream()));  
 String line;  
 StringBuilder content = new StringBuilder();  
 while((line=reader.readLine())!=null){  
 content.append(line).append("\n");  
 }  
 reader.close();  
 System.*out*.println(content);  
 }catch(MalformedURLException e){  
 System.*out*.println(e);  
 }catch(Exception e){  
 System.*err*.println(e);  
 }  
 }  
}

Output:

<!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", Helvetica, 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>

1. **To read the values of Http Header fields.**

public class HeaderExample {  
 public static void main(String []args){  
 try {  
 String url = "http://spm.com.np/";  
 URL u = new URL(url);  
 URLConnection ur = u.openConnection();  
 System.*out*.println(ur.getURL());  
 for(int i=1; ; i++){  
 String header = ur.getHeaderField(i);  
 if(header==null)  
 break;  
 System.*out*.println(ur.getHeaderFieldKey(i) + ": " +header);  
  
 }  
  
 } catch (MalformedURLException e) {  
 System.*err*.println(e);  
 } catch (IOException ex) {  
 System.*err*.println(ex);  
 }  
  
 }  
}

Output:

http://spm.com.np/

Connection: Keep-Alive

Keep-Alive: timeout=5, max=100

content-type: text/html; charset=UTF-8

content-length: 56413

date: Thu, 20 Jul 2023 15:22:03 GMT

server: LiteSpeed

vary: User-Agent

1. **To print the entire http header.**

public class HttpHeaderExample {  
  
 public static void main(String []args){  
 try{  
 URL resource = new URL("http://www.example.com");  
 HttpURLConnection connection = ( HttpURLConnection) resource.openConnection();  
 connection.setRequestMethod("GET");  
 for(int i =0; ; i++){  
 String header = connection.getHeaderField(i);  
 if(header==null)  
 break;  
 System.*out*.println(connection.getHeaderFieldKey(i)+" : "+header);  
 }  
 }catch(MalformedURLException e){  
 System.*err*.println(e);  
 }catch(Exception ex){  
 System.*out*.println(ex);  
 }  
 }  
  
  
}

Output:

null : HTTP/1.1 200 OK

Age : 574922

Cache-Control : max-age=604800

Content-Type : text/html; charset=UTF-8

Date : Thu, 20 Jul 2023 15:23:03 GMT

Etag : "3147526947+ident"

Expires : Thu, 27 Jul 2023 15:23:03 GMT

Last-Modified : Thu, 17 Oct 2019 07:18:26 GMT

Server : ECS (nyb/1D2B)

Vary : Accept-Encoding

X-Cache : HIT

Content-Length : 1256

1. **To get the time when a URI was last changed or updated.**

public class LastModifiedExample {  
 public static void main(String []args){  
 try{  
 URL resource = new URL("http://www.example.com");  
 HttpURLConnection connection = (HttpURLConnection) resource.openConnection();  
 connection.setRequestMethod("HEAD");  
 long lastmodified= connection.getLastModified();  
 System.*out*.println(lastmodified);  
 if(lastmodified==0)  
 System.*out*.println("last modified not found in system");  
 else{  
 Date date = new Date(lastmodified);  
 System.*out*.println(date);  
  
 SimpleDateFormat formatter = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss z");  
 String formattedDate = formatter.format(date);  
 System.*out*.println(formattedDate);  
  
 }  
 }catch(MalformedURLException ex){  
 System.*err*.println(ex);  
 }catch(Exception e){  
 System.*err*.println(e);  
 }  
 }  
}

Output:

1571296706000

Thu Oct 17 13:03:26 NPT 2019

2019-10-17 13:03:26 NPT

1. **Get permission related programs.**

public class SecurityGetPermission {  
 public static void main(String[] args) {  
 try {  
 URL url = new URL("http://example.com");  
 URLConnection connection = url.openConnection();  
  
 *// Get the permission required for the connection* Permission permission = connection.getPermission();  
  
 *// Print the permission details* if (permission != null) {  
 System.*out*.println("Permission: " + permission.toString());  
 System.*out*.println("Permission Actions: " + permission.getActions());  
 } else {  
 System.*out*.println("No permission required for the connection.");  
 }  
  
 connection.connect();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
}

Output:

Permission: ("java.net.SocketPermission" "example.com:80" "connect,resolve")

Permission Actions: connect,resolve

1. **Arbitrary Header field example.**

public class ArbitraryHeaderFieldsExample {  
 public static void main(String[] args) throws IOException {  
 URL url = new URL("http://example.com");  
 HttpURLConnection connection = (HttpURLConnection) url.openConnection();  
  
 *// Set arbitrary header fields* connection.setRequestProperty("X-Custom-Header", "ArbitraryValue");  
 connection.setRequestProperty("X-Another-Header", "SomeValue");  
  
 *// Get arbitrary header fields* String customHeaderValue = connection.getRequestProperty("X-Custom-Header");  
 String anotherHeaderValue = connection.getRequestProperty("X-Another-Header");  
  
 System.*out*.println("Custom Header Value: " + customHeaderValue);  
 System.*out*.println("Another Header Value: " + anotherHeaderValue);  
  
 connection.disconnect();  
 }  
}

Output: Custom Header Value: ArbitraryValue

Another Header Value: SomeValue

Unit 6 and 7 – Socket for Client and Servers

1. **WAP reading from servers with socket.**

import java.io.BufferedReader;  
import java.io.IOException;  
import java.io.InputStreamReader;  
import java.net.Socket;  
  
public class SocketReader {  
 public static void main(String[] args) {  
 String serverAddress = "time.nist.gov";  
 int serverPort = 13;  
  
 try (Socket socket = new Socket(serverAddress, serverPort)) {  
 BufferedReader reader = new BufferedReader(new InputStreamReader(socket.getInputStream()));  
  
 String line;  
 while ((line = reader.readLine()) != null) {  
 System.*out*.println(line);  
 }  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
}

Output:

60145 23-07-20 11:17:40 50 0 0 506.2 UTC(NIST) \*

1. **WAP to extract the socket information from a socket.**

public class SocketInfoExtractor {  
 public static void main(String[] args) {  
 String serverAddress = "example.com";  
 int serverPort = 80;  
  
 try (Socket socket = new Socket(serverAddress, serverPort)) {  
 *// Extract socket information* InetAddress localAddress = socket.getLocalAddress();  
 int localPort = socket.getLocalPort();  
 InetAddress remoteAddress = socket.getInetAddress();  
 int remotePort = socket.getPort();  
  
 *// Print the extracted socket information* System.*out*.println("Local Address: " + localAddress);  
 System.*out*.println("Local Port: " + localPort);  
 System.*out*.println("Remote Address: " + remoteAddress);  
 System.*out*.println("Remote Port: " + remotePort);  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 }  
}

Output:

Local Address: /192.168.1.73

Local Port: 52661

Remote Address: example.com/93.184.216.34

Remote Port: 80

1. **WAP socket to LocalPortScanner.**
2. public class LocalPortScanner {  
    public static void main(String[] args) throws UnknownHostException  
    { InetAddress remote = InetAddress.*getLocalHost*();  
    String hostname = remote.getHostName();  
    for (int port = 1; port < 65536; port++) {  
    try {  
    Socket s = new Socket("localhost", port);  
    System.*out*.println("A server is listening on port "+ port + " of " + hostname);  
    s.close();  
    }  
    catch (IOException ex) {  
   *// The remote host is not listening on this port* }  
    }  
    }  
   }

Output:

A server is listening on port 49664 of LAPTOP-A6MBUV3O

A server is listening on port 49665 of LAPTOP-A6MBUV3O

A server is listening on port 49666 of LAPTOP-A6MBUV3O

A server is listening on port 49667 of LAPTOP-A6MBUV3O

A server is listening on port 49668 of LAPTOP-A6MBUV3O

1. **WAP socket to read TimeClient.**

public class TimeClient {  
 public static void main(String[] args) {  
 try {  
 Socket s = new Socket("time.nist.gov" , 13);  
 InputStream in = s.getInputStream();  
 InputStreamReader isr = new InputStreamReader(in, "ASCII");  
 BufferedReader br = new BufferedReader(isr);  
 br.lines().forEach(System.*out*::println);  
 int line;  
 while ((line = br.read()) != -1) {  
 System.*out*.print((char) line);  
 }  
 System.*out*.println();  
 }  
 catch (IOException e) {  
 System.*out*.println(e);  
 }  
 }  
}

Output:

60145 23-07-20 11:20:16 50 0 0 0.0 UTC(NIST) \*

Unit 8 – Secure Socket

import javax.net.ssl.SSLSocketFactory;  
import java.net.Socket;  
public class ClientSocketExample {  
 public static void main ( String[] args ) {  
 try{  
 SSLSocketFactory factory = (SSLSocketFactory)SSLSocketFactory.*getDefault*();  
 Socket socket = factory.createSocket("localhost",1422);  
 System.*out*.println("Server Connected: " + socket);  
 socket.close();  
 }catch (Exception e){  
 }  
 }  
}

import javax.net.ssl.SSLServerSocketFactory;  
import java.net.ServerSocket;  
import java.net.Socket;  
  
public class ServerSocketExample {  
 public static void main ( String[] args )  
 {  
 try{  
 SSLServerSocketFactory factory = (SSLServerSocketFactory)SSLServerSocketFactory.*getDefault*();  
 ServerSocket serverSocket = factory.createServerSocket(1422);  
 Socket s = serverSocket.accept();  
 System.*out*.println(s+" Client accepted and connected");  
 s.close();  
 }catch (Exception e){  
 }  
 }  
}

Running ServerSocketExample.java

Output:

SSLSocket[hostname=127.0.0.1, port=53725, Session(1689870659392|SSL\_NULL\_WITH\_NULL\_NULL)] Client accepted and connected

Running ClientSocketExample.java

Output:

Server Connected: SSLSocket[hostname=localhost, port=1422, Session(1689870672287|SSL\_NULL\_WITH\_NULL\_NULL)

Unit 9 -Non Blocking I/O

import java.io.RandomAccessFile;

import java.nio.\*;

public class ChannelDemo{

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

        try (RandomAccessFile file = new RandomAccessFile("test.txt","rw")) {

            java.nio.channels.FileChannel fileChannel  =  file.getChannel();

            ByteBuffer byteBuffer  =  ByteBuffer.allocate(512);

            fileChannel.read(byteBuffer);

            byteBuffer.flip();

            while(byteBuffer.hasRemaining()){

                System.out.print((char)byteBuffer.get());

            }

            byteBuffer.clear();

            String data = "Hello, world!";

            byteBuffer.put(data.getBytes());

            byteBuffer.flip();

            fileChannel.write(byteBuffer);

            fileChannel.close();

            file.close();

        }

    }

}

Output:

BCA 6th Hello, World //Content written in test file is “BCA 6th”

Unit 11 – IP Multicast

public class MulticastServer {   
 final static String *INET\_ADDR* = "224.0.0.3";  
 final static int *PORT* = 8888;  
 public static void main(String[] args) throws UnknownHostException {  
 InetAddress address = InetAddress.*getByName*(*INET\_ADDR*);  
 try(DatagramSocket ss = new DatagramSocket()){  
 for(int i=0;i<5;i++){  
 String msg = "Sent Message server no" +i;  
 DatagramPacket msgPacket = new DatagramPacket(msg.getBytes(),  
 msg.getBytes().length,address,*PORT*);  
 ss.send(msgPacket);  
 System.*out*.println("Socket 1 received msg:"+msg);  
*// Thread.sleep(500);* }  
  
 }catch(IOException ex){  
 ex.printStackTrace();  
  
  
 }  
  
  
 }  
}

public class MulticastClient {  
 final static String *INET\_ADDR* = "224.0.0.3";  
 final static int *PORT* = 8888;  
 public static void main(String[] args)throws UnknownHostException{  
 InetAddress address = InetAddress.*getByName*(*INET\_ADDR*);  
 byte[] buf = new byte[256];  
 try(MulticastSocket cs = new MulticastSocket(*PORT*)){  
 cs.joinGroup(address);  
 while(true){  
 DatagramPacket msgPacket = new DatagramPacket(buf,buf.length);  
 cs.receive(msgPacket);  
 String msg = new String(buf,0,buf.length);  
 System.*out*.println("Socket 1 received msg:"+msg);  
  
 }  
 } catch (IOException e) {  
 throw new RuntimeException(e);  
 }  
 }  
  
}

Output:

Socket 1 received msg:Sent Message server no0

Socket 1 received msg:Sent Message server no1

Socket 1 received msg:Sent Message server no2

Socket 1 received msg:Sent Message server no3

Socket 1 received msg:Sent Message server no4

Unit -12 RMI

Q) **RMI program to find the factorial of a number.**

import java.rmi.\*;

public interface Calc extends Remote

{

int factorial(int a) throws RemoteException;

}

import java.rmi.\*;

import java.rmi.server.\*;

public class CalcRemote extends UnicastRemoteObject implements Calc

{

    int ans = 1;

public CalcRemote() throws RemoteException{

super();

}

public int factorial(int a)

{

for (int i = a;i >=1 ;i--){

    ans = i\*ans;

}

return(ans);

}

}

import java.rmi.registry.\*;

public class Server{

public static void main(String[] args)

{

try{

CalcRemote obj=new CalcRemote();

Registry rgsty=LocateRegistry.createRegistry(9003);

rgsty.rebind("fact",obj);

System.out.println("Server Ready");

}

catch(Exception e)

{

System.out.println("Server Failed");

import java.rmi.registry.\*;

public class Client{

public static void main(String[] args)

{

try{

Registry rgsty=LocateRegistry.getRegistry(9003);

Calc obj=(Calc)rgsty.lookup("fact");

int value=obj.factorial(8);

System.out.println("The factorial is: "+value);

}

catch(Exception e)

{

System.out.println("Failed to connect Server!!");

}

}

}

**Running Server.java**

Output: Server is Ready.

**Running Client.java**

Output: The factorial is: 40320