

Unit - 2

1. WAP that prints the address of 'www.tufohss.edu.np'.

```
1 // 1. WAP that prints the address of 'www.tufohss.edu.np'.
2 import java.net.*;
3 class GetAddress{
4     public static void main(String[] args) {
5         try{
6             InetAddress address = InetAddress.getByName("www.tufohss.edu.np");
7             System.out.println("Address : "+address);
8         }catch(Exception ex){
9             System.out.println(ex);
10        }
11    }
12 }
```

2. WAP that finds the address of local machine.

```
1 // 2. WAP that finds the address of local machine.
2
3 import java.net.*;
4
5 public class LocalMachineAddress {
6     public static void main(String[] args) {
7         try{
8             InetAddress address = InetAddress.getLocalHost();
9             System.out.println("Address : "+address);
10        }catch(Exception ex){
11            System.out.println(ex);
12        }
13    }
14 }
15
```

3. WAP that find the canonical host name of the given address.

```
1 // 3. WAP that find the canonical host name of the given address.
2
3 import java.net.*;
4 public class CanonicalHostName {
5     public static void main(String[] args) {
6         try {
7             InetAddress address = InetAddress.getByName("www.facebook.com");
8             System.out.println(address.getCanonicalHostName());
9
10        } catch (Exception ex) {
11            System.out.println(ex);
12        }
13    }
14 }
15
```

4. WAP to find the IP Address and Host Name of a local machine.

```
1 // 4. WAP to find the IP Address and Host Name of a local machine.
2
3 import java.net.*;
4
5 public class IPAndHostNameOfLocalMachine {
6     public static void main(String[] args) {
7         try {
8             InetAddress address = InetAddress.getLocalHost();
9             System.out.println("IP Address : "+address.getHostAddress());
10            System.out.println("Host Name : "+address.getHostName());
11
12        } catch (Exception ex) {
13            System.out.println(ex);
14        }
15    }
16 }
17
18 }
19
```

5. WAP to get IP Address of IPV4 & IPV6 of a given Web Address.

```
1 // 5. WAP to get IP Address of IPV4 & IPV6 of a given Web Address.
2 import java.net.*;
3 public class CheckIPV4orIPV6 {
4     public static void main(String[] args) {
5         try {
6             InetAddress address = InetAddress.getByName("www.facebook.com");
7             String ip=address.getHostAddress();
8             System.out.println("IP Address : "+ip);
9             if(address instanceof Inet4Address){
10                 System.out.println(ip+" is IPV4 Address");
11             }
12             if(address instanceof Inet6Address){
13                 System.out.println(ip+" is IPV6 Address");
14             }
15         } catch (Exception ex) {
16             System.out.println(ex);
17         }
18     }
19 }
```

6. WAP for determining whether an IP Address is IPV4 or IPV6.

```
1 // 6. WAP for determining whether an IP Address is IPV4 or IPV6.
2 public class DetermineIPV4orIPV6 {
3     public static void main(String[] args) {
4         try {
5             String ip = "127.0.0.1";
6             if(ip.contains(".")){
7                 System.out.println(ip+" is IPV4.");
8             }
9             if(ip.contains(":")){
10                 System.out.println(ip+" is IPV6.");
11             }
12         } catch (Exception ex) {
13             System.out.println(ex);
14         }
15     }
16 }
17
18 }
19 }
```

7. WAP for testing the Characteristics of an IP Address.

```
1 // 7. WAP for testing the Characteristics of an IP Address.
2 import java.net.*;
3 public class CharacteristicsOfIP {
4     public static void main(String[] args) {
5         String ip = "127.0.0.1";
6         try{
7             InetAddress address = InetAddress.getByName(ip);
8             if(address.isLoopbackAddress()){
9                 System.out.println(address + " is Loopback Address");
10            }
11            if (address.isAnyLocalAddress()) {
12                System.out.println(address + " is Wildcard Address");
13            }
14            if (address.isSiteLocalAddress()) {
15                System.out.println(address + " is a Site Local Address");
16            }
17            if(address.isLinkLocalAddress()){
18                System.out.println(address + " is a Link Local Address");
19            }
20            if (address.isMulticastAddress()) {
21                System.out.println(address + " is Multicast Address");
22            }
23        }catch(Exception ex){
24            System.out.println(ex);
25        }
26    }
27 }
```

8. WAP that compares the domain names 'www.ibiblio.org' and 'helios.ibiblio.org'.

```
1 // 8. WAP that compares the domain names 'www.ibiblio.org' and 'helios.ibiblio.org'.
2 import java.net.*;
3 public class URLEqualityChecker {
4     public static void main(String[] args) {
5         try {
6             URL url1=new URL("http://www.ibiblio.org");
7             URL url2=new URL("https://helios.ibiblio.org");
8             if(url1.equals(url2)){
9                 System.out.println("Both are Equals");
10            }else{
11                System.out.println("URL are not Equals");
12            }
13        } catch (Exception ex) {
14            System.out.println(ex);
15        }
16    }
17 }
18 }
```

9. WAP that list all the network interface.

```
1 // 9. WAP that list all the network interface.
2 import java.net.*;
3 import java.util.Enumeration;
4 public class ListAllNetworkInterface {
5     public static void main(String[] args) {
6         try {
7             Enumeration<NetworkInterface> inet = NetworkInterface.getNetworkInterfaces();
8             while (inet.hasMoreElements()){
9                 NetworkInterface i = inet.nextElement();
10                System.out.println(i);
11            }
12        } catch (Exception ex) {
13            System.out.println(ex);
14        }
15    }
16 }
17 }
18 }
```

10. WAP to use of Network Interface using getter method.

```
1 // 10. WAP to use of Network Interface using getter method
2 import java.net.*;
3 import java.util.Enumeration;
4 public class useOfNetworkInterface {
5     public static void main(String[] args) {
6         try {
7             Enumeration<NetworkInterface> inet = NetworkInterface.getNetworkInterfaces();
8             NetworkInterface iface = inet.nextElement();
9             System.out.println(iface.getDisplayName());
10            System.out.println(iface.getHardwareAddress());
11            System.out.println(iface.getIndex());
12            System.out.println(iface.getMTU());
13            System.out.println(iface.hashCode());
14            System.out.println(iface.isLoopback());
15            System.out.println(iface.isUp());
16            System.out.println(iface.isPointToPoint());
17        } catch (Exception ex) {
18            System.out.println(ex);
19        }
20    }
21 }
22 }
```


11. WAP to check remote system is reachable or not.

```
1 // 11. WAP to check remote system is reachable or not.
2 import java.net.*;
3 public class CheckSystemIsReachableOrNot {
4     public static void main(String[] args) {
5
6         try {
7             InetAddress address = InetAddress.getByName("www.facebook.com");
8             if(address.isReachable(500)){
9                 System.out.println(address + " is Reachable.");
10            }else{
11                System.out.println(address + " is not Reachable.");
12            }
13        } catch (Exception ex) {
14            System.out.println(ex);
15        }
16    }
17 }
18 }
19 }
```

12. WAP that demonstrate the Spam Check.

```
1 // 12. WAP that demonstrate the Spam Check.
2 import java.net.*;
3 public class SpamCheckIP {
4     private static boolean CheckSpam(String ip){
5         try {
6             InetAddress address = InetAddress.getByName(ip);
7             byte[] add = address.getAddress();
8             String query = "sbl.spamhaus.org";
9             for(byte octet : add){
10                 int unsignedByte = octet<0 ? octet+256 : octet;
11                 query=unsignedByte+"."+query;
12             }
13             System.out.println(query);
14             InetAddress.getByAddress(query);
15             return true;
16         } catch (Exception ex) {
17             System.out.println(ex);
18             return false;
19         }
20     }
21     public static void main(String[] args) {
22         String ip = "127.0.0.1";
23         if(CheckSpam(ip)){
24             System.out.println(ip + " is a Spam IP.");
25         }else{
26             System.out.println(ip + " is not found Spam IP List.");
27         }
28     }
29 }
30 }
31 }
```

13. WAP to process Web Server log file.

```
1 // 13. WAP to process Web Server log file.
2 import java.io.*;
3 import java.net.*;
4 public class ProcessWebServerLogFile {
5     public static void main(String[] args) {
6         String file="logfiles.txt";
7         try{
8             FileInputStream fin = new FileInputStream(file);
9             Reader in = new InputStreamReader(fin);
10            BufferedReader bin = new BufferedReader(in);
11            while(bin.readLine()!=null) {
12                // separate out the IP address
13                String entry = bin.readLine();
14                int index = entry.indexOf(' ');
15                String ip = entry.substring(0, index);
16                String theRest = entry.substring(index);
17                // Ask DNS for the hostname and print it out
18                try {
19                    InetAddress address = InetAddress.getByName(ip);
20                    System.out.println(address.getHostName()+ theRest);
21                } catch (UnknownHostException ex) {
22                    System.out.println(ex);
23                }
24            }
25        } catch (IOException ex) {
26            System.out.println("Exception: " + ex);
27        }
28    }
29 }
```

Unit - 3

1. WAP that split the part of URL.

```
1 // 1. WAP that split the part of URL.
2 import java.net.*;
3 public class SplitURLPart {
4     public static void main(String[] args) {
5         try {
6             URL url = new URL("https://www.facebook.com/login");
7             System.out.println("Protocol : "+url.getProtocol());
8             System.out.println("Host : "+url.getHost());
9             System.out.println("Port : "+url.getDefaultPort());
10            System.out.println("Path : "+url.getPath());
11            System.out.println("Query : "+url.getQuery());
12            System.out.println("Fragment : "+url.getRef());
13
14        } catch (Exception ex) {
15            System.out.println(ex);
16        }
17    }
18 }
19
```

2. WAP that check which protocol does a virtual machine support or not.

```
1 // 2. WAP that check which protocol does a virtual machine support or not.
2 import java.net.*;
3 public class CheckProtocol {
4     public static void main(String[] args) {
5         try {
6             URL url = new URL("https://google.com");
7             System.out.println("Protocol : "+url.getProtocol());
8         } catch (Exception ex) {
9             System.out.println(ex);
10        }
11    }
12 }
13
```


3. WAP to download a webpage of a given web address.

```
1 // 3. WAP to download a webpage of a given web address.
2 import java.io.*;
3 import java.net.*;
4 public class DownloadWebPage {
5     public static void main(String[] args) {
6         String url ="https://bhandari-santosh.com.np";
7         try {
8             URL u = new URL(url);
9             BufferedReader bf = new BufferedReader(new InputStreamReader(u.openStream()));
10             while(bf.readLine()!=null){
11                 System.out.println(bf.readLine());
12             }
13         } catch (Exception ex) {
14             System.out.println(ex);
15         }
16     }
17 }
18
```

4. WAP to download an objects.

```
1 // 4. WAP to download an objects.
2 import java.net.*;
3 import java.io.*;
4 public class DownloadObjects {
5     public static void main(String[] args) {
6         String url = "http://127.0.0.1:8080/Readme.md";
7         String filename="readme.md";
8         try {
9             URL u = new URL(url);
10             BufferedInputStream bis=new BufferedInputStream(u.openStream());
11             FileOutputStream fos = new FileOutputStream(filename);
12             byte[] b = new byte[1024];
13             int i;
14             while((i=bis.read(b,0,1024))!=-1){
15                 fos.write(b,0,i);
16             }
17         } catch (Exception ex ) {
18             System.out.println(ex);
19         }
20     }
21 }
22
```

5. WAP to demonstrate x-www-form-urlencoded String.

```
1 // 5. WAP to demonstrate x-www-form-urlencoded String.
2 import java.net.*;
3 public class URLEncoding {
4     public static void main(String[] args) {
5         String data = "?query=data1 & data2";
6         try {
7             String encodeData = URLEncoder.encode(data,"UTF-8");
8             System.out.println("Encoded Data : "+encodeData);
9             String decodedData = URLDecoder.decode(encodeData,"UTF-8");
10            System.out.println("Decoded Data : "+decodedData);
11        } catch (Exception ex ) {
12            System.out.println(ex);
13        }
14    }
15 }
16
```

6. WAP to communicate with server side program through get.

```
1 // WAP to communicate with server side program through get.
2 import java.io.BufferedReader;
3 import java.io.InputStreamReader;
4 import java.net.*;
5 public class GetRequest {
6     public static void main(String[] args) {
7         String url = "https://bhandari-santosh.com.np";
8         try {
9             URL u = new URL(url);
10            HttpURLConnection con = (HttpURLConnection) u.openConnection();
11            con.setRequestMethod("GET");
12            BufferedReader br = new BufferedReader(new InputStreamReader(con.getInputStream()));
13            while(br.readLine()!=null){
14                System.out.println(br.readLine());
15            }
16        } catch (Exception ex) {
17            System.out.println(ex);
18        }
19    }
20 }
21
```

7. WAP to resolve relative URI.

```
1 // 7. WAP to resolve relative URI.
2 import java.net.*;
3 public class ResolveRelativeURI {
4     public static void main(String[] args) {
5         try {
6             URI baseUrl=new URI("https://bhandari-santosh.com.np");
7             URI relativeUrl = new URI("/photo.png");
8             URI resolvedUrl = baseUrl.resolve(relativeUrl);
9             System.out.println(resolvedUrl);
10        } catch (Exception ex) {
11            System.out.println(ex);
12        }
13    }
14 }
15
```

Unit - 5

1. WAP to download a Webpage using URL Connection.

```
1 // 1. WAP to download a Webpage using URL Connection.
2 import java.io.*;
3 import java.net.*;
4
5 public class DownloadWebpageURLConnection {
6     public static void main(String[] args) {
7         try {
8             URL url = new URL("https://bhandari-santosh.com.np/");
9             URLConnection con = url.openConnection();
10            InputStream data = con.getInputStream();
11            int res;
12            while((res=data.read())!=-1){
13                System.out.print((char) res);
14            }
15        } catch (Exception ex) {
16            System.out.println(ex);
17        }
18    }
19 }
```

2. WAP to read the Value of HTTP Header Fields.

```
1 // 2. WAP to read the Value of HTTP Header Fields.
2 import java.net.*;
3 public class ValueOfHTTPHeader {
4     public static void main(String[] args) {
5         try {
6             URL u = new URL("https://bhandari-santosh.com.np");
7             URLConnection con = u.openConnection();
8             System.out.println("Content Type : "+con.getContentType());
9             System.out.println("Content Length : "+con.getContentLength());
10            System.out.println("Content Encoding : "+con.getContentEncoding());
11            System.out.println("Document Sent : "+con.getDate());
12            System.out.println("Last Modified : "+con.getLastModified());
13        } catch (Exception ex) {
14            System.out.println(ex);
15        }
16    }
17 }
18 }
```

3. WAP to print entire HTTP Headers.

```
1 // 3. WAP to print entire HTTP Headers.
2 import java.net.*;
3 class ArbitraryHeaderFields{
4     public static void main(String[] args) {
5         try {
6             URL url = new URL("https://bhandari-santosh.com.np");
7             URLConnection con = url.openConnection();
8             for(int i=1; ; i++){
9                 if(con.getHeaderField(i)==null)
10                     break;
11                 System.out.println(con.getHeaderFieldKey(i)+ " : "+con.getHeaderField(i));
12             }
13         } catch (Exception ex) {
14             System.out.println(ex);
15         }
16     }
17 }
```

4. WAP for HTTP request Method. - Not Done

5. WAP to print URL of a URL Connection to "tufohss.edu.np".

```
1 // 5. WAP to print URL of a URL Connection to "tufohss.edu.np".
2 import java.net.*;
3 public class PrintURLOfURLConnection {
4     public static void main(String[] args) {
5         String url = "https://tufohss.edu.np";
6         try {
7             URL u = new URL(url);
8             System.out.println("URL : "+u);
9         } catch (Exception ex) {
10             System.out.println(ex);
11         }
12     }
13 }
14 }
15 }
```

6. WAP to get a time when a URI was last change.

```
1 // 6. WAP to get a time when a URI was last change.
2 import java.net.*;
3 public class LastChangeURI {
4     public static void main(String[] args) {
5         try {
6             URL url = new URL("https://bhandari-santosh.com.np");
7             URLConnection con = url.openConnection();
8             System.out.println("Last Modified Date : "+con.getLastModified());
9         } catch (Exception ex) {
10             System.out.println(ex);
11         }
12     }
13 }
14
```


Unit - 6

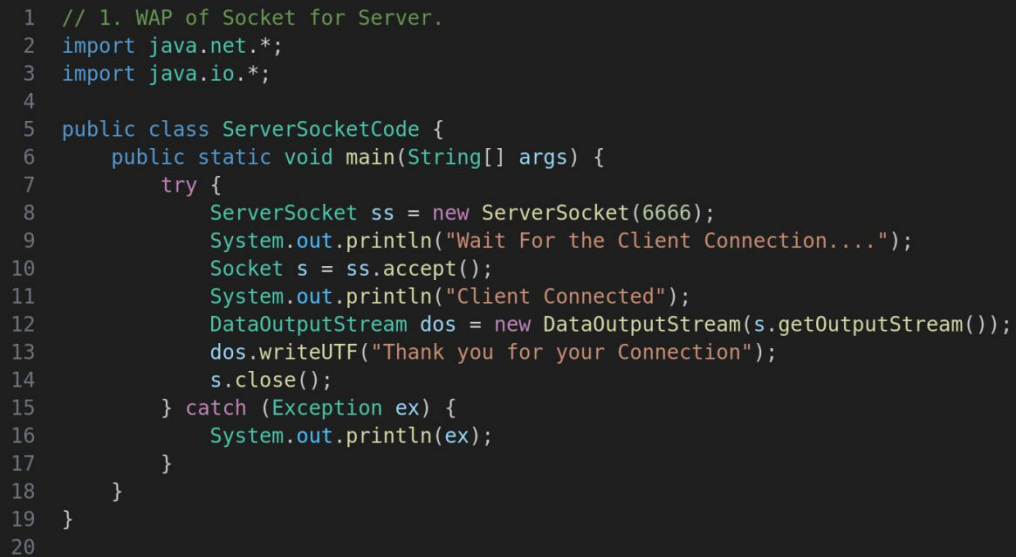
1. WAP of Socket to Client.



```
1 // 1. WAP of Socket to Client.
2 import java.net.*;
3 import java.io.*;
4
5 public class ClientSocketCode {
6     public static void main(String[] args) {
7         try {
8             Socket s = new Socket("localhost",6666);
9             System.out.println("Connected to the Server");
10            DataInputStream dis = new DataInputStream(s.getInputStream());
11            System.out.println((String)dis.readUTF());
12        } catch (Exception ex) {
13            System.out.println(ex);
14        }
15    }
16 }
17
```

Unit - 7

1. WAP of Socket for Server.



```
1 // 1. WAP of Socket for Server.
2 import java.net.*;
3 import java.io.*;
4
5 public class ServerSocketCode {
6     public static void main(String[] args) {
7         try {
8             ServerSocket ss = new ServerSocket(6666);
9             System.out.println("Wait For the Client Connection...");
10            Socket s = ss.accept();
11            System.out.println("Client Connected");
12            DataOutputStream dos = new DataOutputStream(s.getOutputStream());
13            dos.writeUTF("Thank you for your Connection");
14            s.close();
15        } catch (Exception ex) {
16            System.out.println(ex);
17        }
18    }
19 }
20
```

Client Server Communication

```
1 import java.net.*;
2 import java.io.*;
3 import java.util.Scanner;
4 class ServerCode{
5     public static void main(String args[]){
6         try{
7             ServerSocket ss = new ServerSocket(9999);
8             Scanner sc = new Scanner(System.in);
9             String servermsg,clientmsg;
10            System.out.println("Waiting For Client Connection....");
11            Socket s = ss.accept();
12            System.out.println("Client Connected.");
13            DataInputStream dis = new DataInputStream(s.getInputStream());
14            DataOutputStream dos = new DataOutputStream(s.getOutputStream());
15            while(true){
16                System.out.println("Waiting Client Message.....");
17                clientmsg = (String) dis.readUTF();
18                System.out.println("Client : "+clientmsg);
19                System.out.print("Enter a Message(e for Exit): ");
20                servermsg=sc.nextLine();
21                if (clientmsg.equalsIgnoreCase("e") || servermsg.equalsIgnoreCase("e")){
22                    break;
23                }
24                System.out.println("Server : "+servermsg);
25                dos.writeUTF(servermsg);
26            }
27            dos.flush();
28            dos.close();
29            s.close();
30        }catch(Exception ex){
31            System.out.println(ex);
32        }
33    }
34 }
```

```
1 import java.net.*;
2 import java.io.*;
3 import java.util.Scanner;
4 class ClientCode{
5     public static void main(String args[]){
6         try{
7             Socket s = new Socket("localhost",9999);
8             Scanner sc = new Scanner(System.in);
9             String clientmsg,servermsg;
10            DataOutputStream dos = new DataOutputStream(s.getOutputStream());
11            DataInputStream dis = new DataInputStream(s.getInputStream());
12            System.out.println("Connected to the Server.");
13            while(true){
14                System.out.print("Enter a Message(e for Exit): ");
15                clientmsg = sc.nextLine();
16                if (clientmsg.equalsIgnoreCase("e")){
17                    break;
18                }
19                dos.writeUTF(clientmsg);
20                System.out.println("Waiting Server Message.....");
21                servermsg=(String)dis.readUTF();
22                System.out.println("Server : "+ servermsg);
23            }
24            dos.flush();
25            dos.close();
26            s.close();
27        }catch(Exception ex){
28            System.out.println(ex);
29        }
30    }
31 }
```

