

# SOCKET PROGRAMMING

**Date - 14-12-18**

**Experiment – 1**

**PROGRAM 1:**

**AIM :**

To implement simple ping pong application between two or more different machines.

**PROGRAM:**

```
import java.io.*;

import java.net.*;

class Ping{

    public static void sendPingRequest(String ipAddress) throws
    UnknownHostException, IOException{

        InetAddress geek = InetAddress.getByName(ipAddress);

        System.out.println("Sending Ping Request to " + ipAddress);

        if (geek.isReachable(5000))

            System.out.println("Host " + geek.getHostName() + " is
reachable");

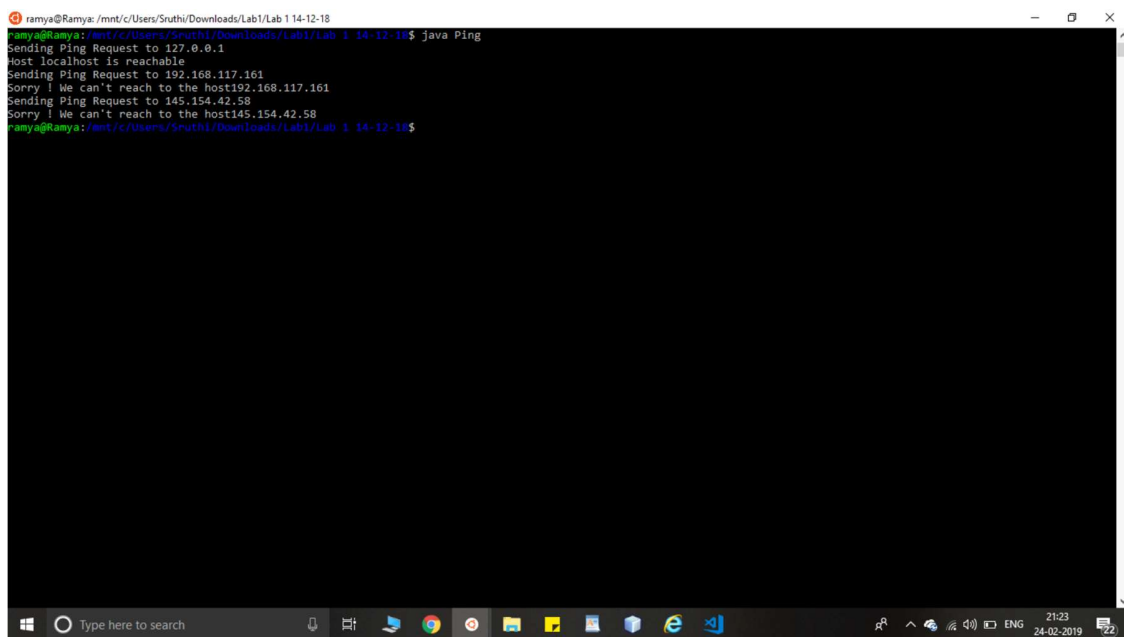
        else

            System.out.println("Sorry ! We can't reach to the host" +
geek.getHostName());}

    public static void main(String[] args) throws UnknownHostException, IOException

        sendPingRequest(ipAddress);
```

```
        ipAddress = "145.154.42.58";  
  
        sendPingRequest(ipAddress);  
  
    }  
  
}
```

**OUTPUT:**

```
ramya@Ramya: /mnt/c/Users/Sruthi/Downloads/Lab1/Lab 1 14-12-18$ java Ping  
Sending Ping Request to 127.0.0.1  
Host localhost is reachable  
Sending Ping Request to 192.168.117.161  
Sorry ! We can't reach to the host192.168.117.161  
Sending Ping Request to 145.154.42.58  
Sorry ! We can't reach to the host145.154.42.58  
ramya@Ramya: /mnt/c/Users/Sruthi/Downloads/Lab1/Lab 1 14-12-18$
```

**RESULT:**

Thus the source code has been compiled and executed successfully.

**PROGRAM 2:****AIM :**

To implement date and time display from client to server using Sockets.

**PROGRAM:****CLIENT**

```
import java.net.*;

import java.util.*;

import java.io.*;

class Client{

    private DataInputStream inp = null;

    private DataOutputStream out = null;

    private Socket socket = null;

    public Client(String ip, int port)    {

        try    {

            socket = new Socket(ip,port);

            inp = new DataInputStream(System.in);

            out = new DataOutputStream(socket.getOutputStream());    }

        catch(Exception e)    {

            System.out.println(e);    }

    }
```

```
void sendDateTime()
{
    try {
        Date d = new Date();
        out.writeUTF(d.toString());
        inp.close();
        out.close();
        socket.close();
    }
    catch(Exception e) {
        System.out.println(e);
    }
}

public static void main(String []args) {
    Client client = new Client("127.0.0.1", 5000);
    client.sendDateTime();
}
}
```

**SERVER**

```
import java.net.*;
import java.util.*;
import java.io.*;
class Server {
```

```
private DataInputStream inp = null;

private ServerSocket server = null;

private Socket socket = null;

private String dateTime = "";

public Server(int port)    {

    try    {

        server = new ServerSocket(port);

        socket = server.accept();

        inp = new DataInputStream(socket.getInputStream());    }

    catch(Exception e)    {

        System.out.println(e);        }

}

void getDateTime() {

    try    {

        dateTime = inp.readUTF();

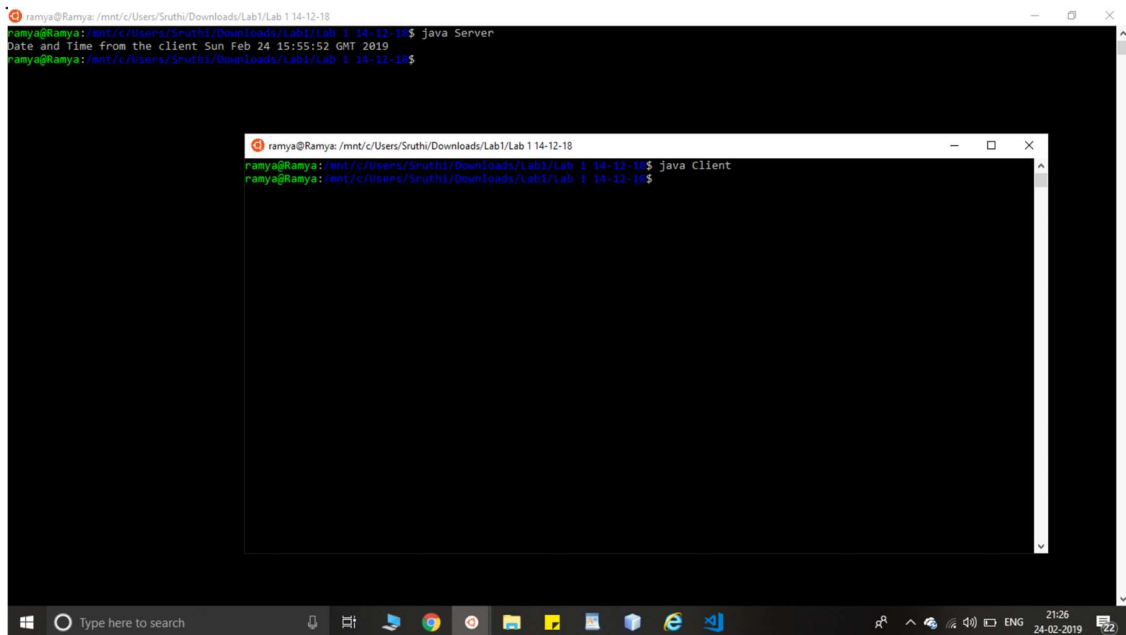
        System.out.println("Date and Time from the client " +
dateTime);

        socket.close();

        inp.close();        }

    catch(Exception e)    {
```

```
        System.out.println(e);    }  
    }  
  
    public static void main(String []args)    {  
  
        Server server = new Server(5000);  
  
        server.getDateTime();  
  
    }
```

**OUTPUT:****RESULT:**

Thus the source code has been compiled and executed successfully.

**PROGRAM 3:****AIM :**

To write a ping pong client and server application. When a client sends a ping message to the server, the server will respond with a pong message. Other messages sent by the client can be safely dropped by the client.

**PROGRAM :****CLIENT**

```
import java.net.*;

import java.util.*;

import java.io.*;

class Client {

    private DataInputStream inp = null;

    private DataOutputStream out = null;

    private DataInputStream serverInput = null;

    private Socket socket = null;

    private String dataToBeSent = "";

    private String dataReceived = "";

    public Client(String ip, int port) {

        try {

            socket = new Socket(ip,port);

            inp = new DataInputStream(System.in);
```



2016503543

```
serverInput = new DataInputStream(socket.getInputStream());
out = new DataOutputStream(socket.getOutputStream());    }

catch(Exception e)  {

    System.out.println(e);    }

}

void chat()  {

    try  {

        System.out.println("Enter the data to be sent to the Server ");

        Scanner sc = new Scanner(System.in);

        dataToBeSent = sc.nextLine();

        while(!dataToBeSent.equals("Over"))    {

            out.writeUTF(dataToBeSent);

            if(dataToBeSent.equals("ping"))    {

                dataReceived = serverInput.readUTF();

                System.out.println("Data Received from the
Server is " + dataReceived);    }

            dataToBeSent = sc.nextLine();    }

        inp.close();

        out.close();

        socket.close();}
```

```
        catch(Exception e) {  
            System.out.println(e);  
        }  
  
    public static void main(String []args) {  
        Client client = new Client("127.0.0.1", 4000);  
        client.chat(); }  
    }
```

**SERVER:**

```
import java.net.*;  
import java.util.*;  
import java.io.*;  
  
class Server {  
    private DataInputStream inp = null;  
    private ServerSocket server = null;  
    private Socket socket = null;  
    private DataOutputStream out = null;  
    private String inputData = "";  
    public Server(int port) {  
        try {  
            server = new ServerSocket(port);  
            socket = server.accept();
```

2016503543

```
        inp = new DataInputStream(socket.getInputStream());
        out = new DataOutputStream(socket.getOutputStream());    }

    catch(Exception e)    {

        System.out.println(e);        }

    }

    void chat()    {

        try    {

            inputData = inp.readUTF();

            while(!inputData.equals("Over"))    {

                System.out.println("Data received from the client " +
inputData);

                if(inputData.equals("ping")) {

                    System.out.println("Data receiver from the client
is ping");

                    out.writeUTF("pong");        }

                Else    {

                    inputData = "";        }

                inputData = inp.readUTF();

            socket.close();

            inp.close();

            out.close();    }
```

```
        catch(Exception e)

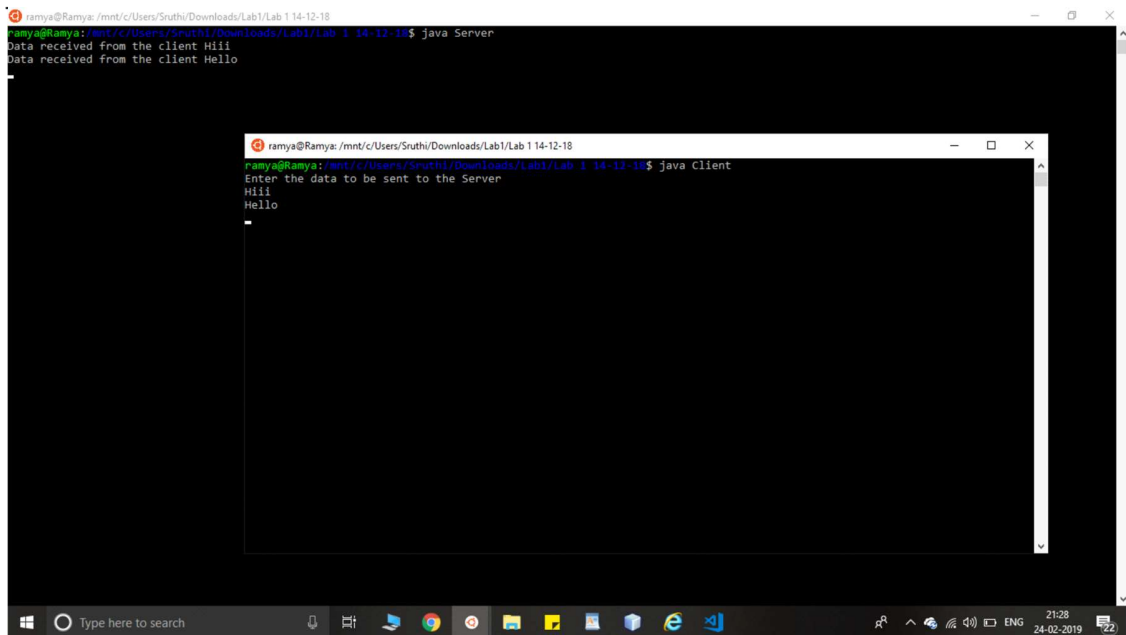
            System.out.println(e);

    }

    public static void main(String []args)    {

        Server server = new Server(4000);

        server.chat(); }    }
```

**OUTPUT:**

```
ramya@Ramya: /mnt/c/Users/Sruthi/Downloads/Lab1/Lab 1 14-12-18
ramya@Ramya: /mnt/c/Users/Sruthi/Downloads/Lab1/Lab 1 14-12-18$ java Server
Data received from the client Hi!!
Data received from the client Hello

ramya@Ramya: /mnt/c/Users/Sruthi/Downloads/Lab1/Lab 1 14-12-18$ java Client
Enter the data to be sent to the Server
Hi!!
Hello
```

**RESULT :**

Thus, the program is executed and output is obtained.

**PROGRAM 4:****AIM:**

To write a socket based Java server program that responds to client messages as follows: When it receives a message from client, it simply converts the message into all uppercase letters and sends back the same to the client.

**PROGRAM:****CLIENT:**

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

class Client {

    private DataInputStream inp = null;
    private DataOutputStream out = null;
    private DataInputStream serverInput = null;
    private Socket socket = null;
    private String dataToBeSent = "";
    private String dataReceived = "";

    public Client(String ip, int port) {

        try{

            socket = new Socket(ip,port);

            inp = new DataInputStream(System.in);
```

2016503543

```
serverInput = new DataInputStream(socket.getInputStream());  
out = new DataOutputStream(socket.getOutputStream());  
catch(Exception e) {  
    System.out.println(e); }  
}  
void chat(){  
    try{  
        System.out.println("Enter the data to be sent to the Server ");  
        Scanner sc = new Scanner(System.in);  
        dataToBeSent = sc.nextLine();  
        while(!dataToBeSent.equals("Over"))    {  
            out.writeUTF(dataToBeSent);  
            dataReceived = serverInput.readUTF();  
            System.out.println("Data Received from the Server is " +  
dataReceived);  
            dataToBeSent = sc.nextLine();  
        }  
        inp.close();  
        out.close();  
        socket.close();}
```

```
        catch(Exception e) {  
            System.out.println(e);  
        }  
    }  
  
    public static void main(String []args)    {  
        Client client = new Client("127.0.0.1", 4000);  
        client.chat();  
    }  
}
```

**SERVER:**

```
import java.net.*;  
import java.util.*;  
import java.io.*;  
  
class Server {  
    private DataInputStream inp = null;  
    private ServerSocket server = null;  
    private Socket socket = null;  
    private DataOutputStream out = null;  
    private String inputData = "";  
    public Server(int port)    {
```

```
try    {

    server = new ServerSocket(port);

    socket = server.accept();

    inp = new DataInputStream(socket.getInputStream());

    out = new DataOutputStream(socket.getOutputStream());    }

catch(Exception e)    {

    System.out.println(e);    }

}

void chat(){

    try{

        inputData = inp.readUTF();

        while(!inputData.equals("Over")){

            System.out.println("Data received from the client " +
            inputData);

            out.writeUTF(inputData.toUpperCase());

            inputData = inp.readUTF(); }

        socket.close();

        inp.close();

        out.close();    }

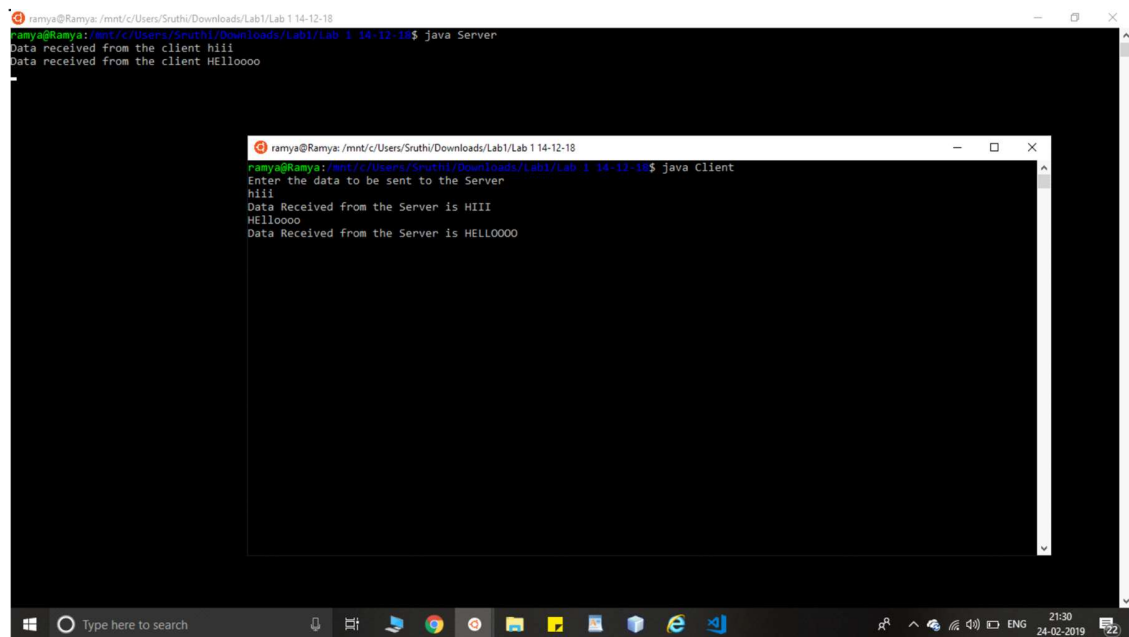
    catch(Exception e){
```



```
        System.out.println(e);    }  
    }  
}
```

```
public static void main(String []args)    {  
    Server server = new Server(4000);  
    server.chat(); }    }
```

### OUTPUT:



```
ramya@Ramya: /mnt/c/Users/Sruthi/Downloads/Lab1/Lab 1 14-12-18  
ramya@Ramya: /mnt/c/Users/Sruthi/Downloads/Lab1/Lab 1 14-12-18$ java Server  
Data received from the client hiii  
Data received from the client Helloooo  
  
ramya@Ramya: /mnt/c/Users/Sruthi/Downloads/Lab1/Lab 1 14-12-18$ java Client  
Enter the data to be sent to the Server  
Hiii  
Data Received from the Server is HIII  
Helloooo  
Data Received from the Server is HELLOOOO
```

### RESULT:

Thus, the source code is executed and output is executed successfully.

**PROGRAM 5:****AIM:**

To write a socket based Java server program that responds to clients messages as follows: When it receives a message from clients, it simply converts the message into all uppercase letters and sends back the same to the clients. Broadcasting messages from server to several clients.

**CODE :****CLIENT:**

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

public class p4client {

    private Socket socket = null;

    private DataInputStream input = null;

    private DataOutputStream out = null;

    public p4client(String address, int port) {

        try{

            socket = new Socket(address, port);

            System.out.println("Connected");

            input = new DataInputStream(new
BufferedInputStream(socket.getInputStream()));

            out = new DataOutputStream(socket.getOutputStream());}
```

2016503543

```
catch(Exception e) {  
    System.out.println(e); }  
  
String line = "", r = "";  
  
Scanner sc = new Scanner(System.in);  
  
while (!line.equals("Over")){  
    try{  
        line = sc.next();  
        r = input.readUTF();  
        System.out.println("UPPERCASE VERSION:" + r); }  
    catch(IOException i) {  
        System.out.println();}  
    }  
    try  
    {  
        input.close();  
        out.close();  
        socket.close();  
    }  
    catch(IOException i)  
    {
```

```
        System.out.println(i); }  
  
    }  
  
    public static void main(String args[]){  
        p4client client = new p4client("192.168.117.70", 5000); }  
    }
```

**SERVER:**

```
import java.net.*;  
import java.util.*;  
import java.io.*;  
class Server {  
    private DataInputStream inp = null;  
    private ServerSocket server = null;  
    private Socket socket = null;  
    private DataOutputStream out = null;  
    private String inputData = "";  
    public Server(int port) {  
        try {  
            server = new ServerSocket(port);  
            socket = server.accept();  
            inp = new DataInputStream(socket.getInputStream());
```

2016503543

```
        out = new DataOutputStream(socket.getOutputStream());    }

    catch(Exception e)    {

        System.out.println(e);        }

    }

    void chat()    {

        try{

            inputData = inp.readUTF();

            while(!inputData.equals("Over"))        {

                System.out.println("Data received from the client " +
                    inputData);

                out.writeUTF(inputData.toUpperCase());

                inputData = inp.readUTF(); }

            socket.close();

            inp.close();

            out.close();

        }

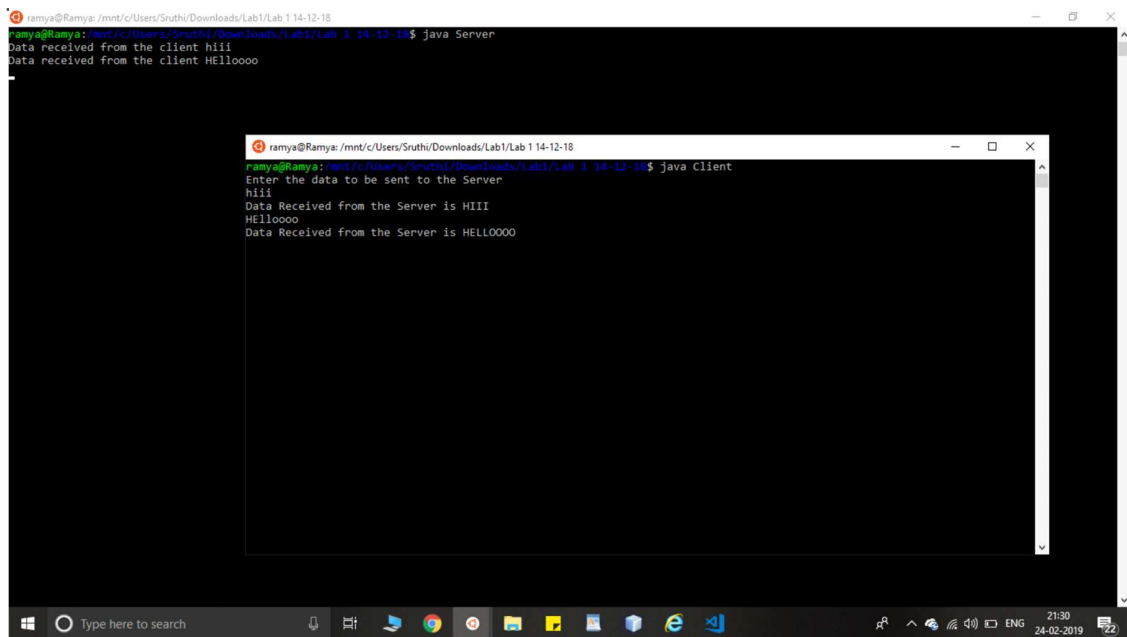
        catch(Exception e)    {

            System.out.println(e)}

    public static void main(String []args)        {

        Server server = new Server(4000);
```

```
server.chat();}  
}
```

**OUTPUT:**

```
ramya@Ramya: /mnt/c/Users/Sruthi/Downloads/Lab1/Lab 1 14-12-18  
ramya@Ramya: /mnt/c/Users/Sruthi/Downloads/Lab1/Lab 1 14-12-18$ java Server  
Data received from the client hiii  
Data received from the client Helloooo  
  
ramya@Ramya: /mnt/c/Users/Sruthi/Downloads/Lab1/Lab 1 14-12-18$ java Client  
Enter the data to be sent to the Server  
hiii  
Data Received from the Server is HIII  
Helloooo  
Data Received from the Server is HELLOOOO
```

**RESULT :**

Thus, the source code is executed and output is executed successfully.

# REMOTE METHOD INVOCATION

**Date - 21-12-18**

**Experiment – 2**

**Program-1**

**AIM :**

To write a Java program to implement Client Server communication using RPC.

**PROGRAM :**

**Server:**

```
import java.io.*;

import java.net.*;

class ser    {

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

        ServerSocket sersock = new ServerSocket(3000);

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

        Socket sock = sersock.accept();

        BufferedReader keyRead = new BufferedReader(new
        InputStreamReader(System.in));

        OutputStream ostream = sock.getOutputStream();

        PrintWriter pwrite = new PrintWriter(ostream, true);

        InputStream istream = sock.getInputStream();

        BufferedReader receiveRead = new BufferedReader(new
```



```
InputStreamReader(istream));  
  
String receiveMessage, sendMessage, fun;  
  
int a,b,c;  
  
while(true) {  
  
    fun = receiveRead.readLine();  
  
    if(fun != null)  
  
        System.out.println("Data: "+fun);  
  
        System.out.flush();} }  
}
```

**Client:**

```
import java.io.*;  
  
import java.net.*;  
  
class cli    {  
  
    public static void main(String[] args) throws Exception {  
  
        Socket sock = new Socket("127.0.0.1", 3000);  
  
        BufferedReader keyRead = new BufferedReader(new  
        InputStreamReader(System.in));  
  
        OutputStream ostream = sock.getOutputStream();  
  
        PrintWriter pwrite = new PrintWriter(ostream, true);  
  
        InputStream istream = sock.getInputStream();
```

```
BufferedReader receiveRead = new BufferedReader(new
InputStreamReader(istream));

System.out.println("Client ready, type and press Enter key");

String receiveMessage, sendMessage,temp;

while(true) {

    System.out.println("\nEnter input");

    temp = keyRead.readLine();

    sendMessage=temp.toLowerCase();

    pwrite.println(sendMessage);

    System.out.flush(); }}
```

**OUTPUT:**

Server:

Data: hello

Client:

Enter input

hello

**RESULT:**

Thus the program is executed and output is obtained.

**Program 2****AIM :**

To implement Remote command execution using RMI.

**PROGRAM :**

Command.java

```
import java.rmi.*;
```

```
public interface Command extends Remote {
```

```
    public void execute(String cmd) throws RemoteException;    }
```

CommandRemote.java

```
import java.rmi.*;
```

```
public class CommandRemote extends UnicastRemoteObject implements Command
```

```
{
```

```
    CommandRemote() throws RemoteException {
```

```
        super();    }
```

```
    public void execute(String cmd)    {
```

```
        try    {
```

```
            Runtime run = Runtime.getRuntime();
```

```
            Process p = run.exec(cmd);
```

```
            System.out.println(p);
```

```
System.out.println("This " + cmd + " has been executed.");    }
```

```
catch(Exception e) {
```

```
System.out.println(e);    }
```

MyServer.java

```
import java.rmi.*;
```

```
import java.rmi.registry.*;
```

```
public class MyServer {
```

```
public static void main(String[] args) {
```

```
try{
```

```
Command stub = new CommandRemote();
```

```
Naming.rebind("rmi://localhost:5001/command", stub);}
```

```
catch(Exception e)  {
```

```
System.out.println(e);    }
```

MyClient.java

```
import java.rmi.*;
```

```
public class MyClient{
```

```
public static void main(String args[]){
```

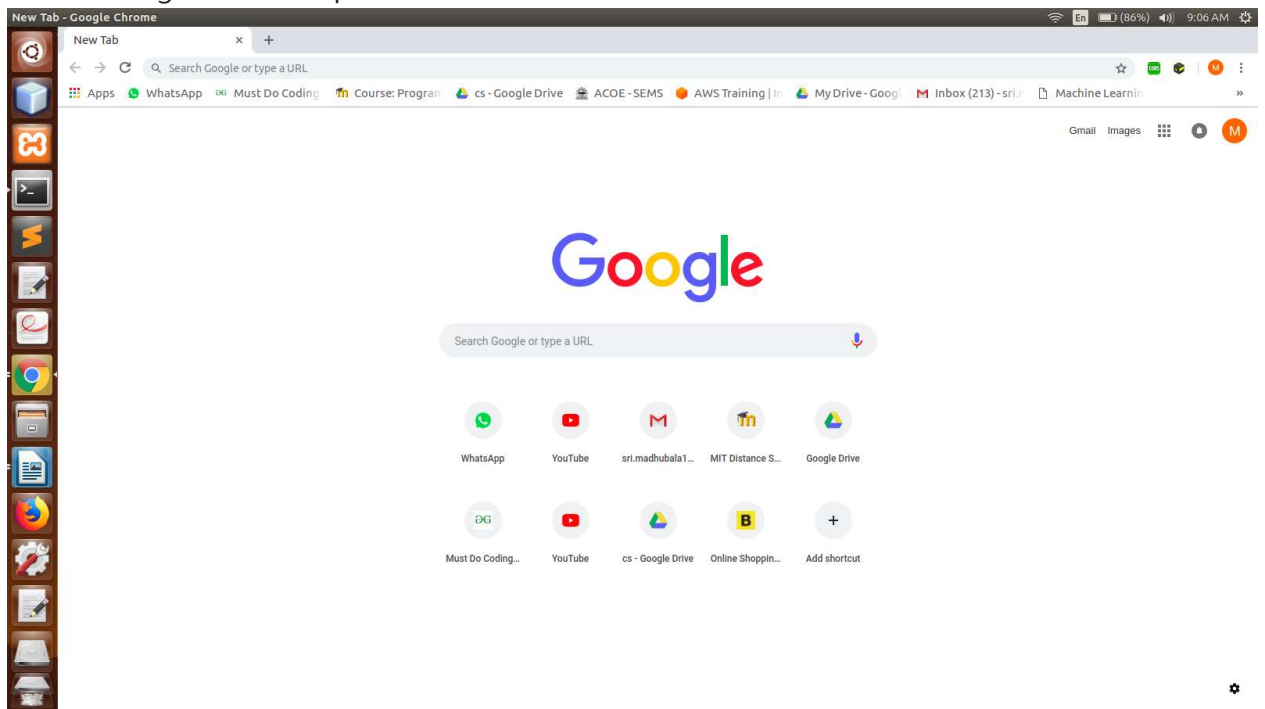
```
try{
```

```
Command stub=(Command)Naming.lookup("rmi://localhost:5001/command");  
stub.execute("google-chrome");  }  
  
catch(Exception e){}  
}
```

**OUTPUT:**

Server: google-chrome

Client: Google chrome opened

**RESULT:**

Thus the program is executed and output is obtained.

**Program 3****AIM :**

To create RMI to calculate factorial of given number.

**PROGRAM :**

Factorial.java

```
import java.rmi.*;
```

```
public interface Factorial extends Remote{  
    public int add(int x)throws RemoteException; }
```

FactorialRemote.java

```
import java.rmi.*;
```

```
import java.rmi.server.*;
```

```
public class FactorialRemote extends UnicastRemoteObject implements Adder{  
    FactorialRemote()throws RemoteException{  
        super();}
```

```
    public int add(int x){  
        int p=1;for(int i=1;i<=x;i++)p*=i;return p;}    }
```

MyServer.java

```
import java.rmi.*;
```

```
import java.rmi.registry.*;
```

```
public class MyServer{  
  
    public static void main(String args[]){  
  
        Factorail stub=new FactorialRemote();  
  
        Naming.rebind("rmi://localhost:5000/sonoo",stub);  
  
    }  
}
```

MyClient.java

```
import java.rmi.*;  
  
import java.util.*;  
  
public class MyClient{  
  
    public static void main(String args[]){  
  
        Factorial stub = (Factorial)Naming.lookup("rmi://localhost:5000/sonoo");  
  
        System.out.println("enter number for factorial");  
  
        Scanner sc=new Scanner(System.in);  
  
        int k=sc.nextInt();  
  
        System.out.println(stub.add(k));  
    }  
}
```

**OUTPUT:**

enter number for factorial 3      Ans : 6

**RESULT:**

Thus the program is executed and output is obtained.

**Program -4****AIM :**

To create RMI to perform arithmetic operations using RMI.

**PROGRAM :**

Arithmetic.java

```
import java.rmi.*;

public interface Arithmetic extends Remote {

    public int add(int x, int y) throws RemoteException;

    public int sub(int x, int y) throws RemoteException;

    public int mul(int x, int y) throws RemoteException;

    public int div(int x, int y) throws RemoteException;

    public int mod(int x, int y) throws RemoteException;

}
```

ArithmeticRemote.java

```
import java.rmi.*;

import java.rmi.server.*;

public class ArithmeticRemote extends UnicastRemoteObject implements
Arithmetic{

    ArithmeticRemote() throws RemoteException {
```



```
super();

}

public int add(int x, int y) {

return x+y;  }

public int sub(int x, int y) {

return x-y;  }

public int mul(int x, int y) {

return x*y;  }

public int div(int x, int y) {

return x/y;  }

public int mod(int x, int y) {

return x%y;  }

}

MyServer.java

import java.rmi.*;

import java.rmi.registry.*;

public class MyServer {

public static void main(String[] args) {
```

```
try    {  
  
    Arithmetic stub = new ArithmeticRemote();  
  
    Naming.rebind("rmi://localhost:5001/arithmetic", stub); }  
  
catch(Exception e)  {  
  
    System.out.println(e);      }  
  
}
```

MyClient.java

```
import java.rmi.*;  
  
public class MyClient{  
  
    public static void main(String args[]){  
  
        try{  
  
            Arithmetic stub=(Arithmetic)Naming.lookup("rmi://localhost:5001/arithmetic");  
  
            System.out.println(stub.add(4,3));  
  
            System.out.println(stub.sub(4,3));  
  
            System.out.println(stub.mul(4,3));  
  
            System.out.println(stub.div(4,3));  
  
            System.out.println(stub.mod(4,3)); }  
  
        catch(Exception e){}
```

```
}
```

```
}
```

**OUTPUT:**

Enter the 2 numbers

2

3

The addition answer is 5

The subtraction answer is -1

The multiplication answer is 6

The division answer is 0

The modulus answer is 0

**RESULT:**

Thus the program is executed and output is obtained.

**Program 5****AIM :**

Implement Domain name server : It converts IP address for given textual name.

**PROGRAM :**

DNS.java

```
import java.rmi.*;
```

```
public interface DNS extends Remote{
```

```
    public int find(String x)throws RemoteException;
}
```

DNSRemote.java

```
import java.rmi.*;
```

```
import java.rmi.server.*;
```

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

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

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

```
public class DNSRemote extends UnicastRemoteObject implements DNS{
```

```
    DNSRemote()throws RemoteException{
    }
}
```

```
public int find(String x){
```

```
try{

    InetAddress ad = java.net.InetAddress.getByName(x);

    String address = ad.getHostAddress();

    System.out.println(address);

}

catch(Exception e)  {

    System.out.println(e);

}

return 0;

}

}
```

Server.java

```
import java.rmi.*;

import java.rmi.registry.*;

public class Server{

public static void main(String args[]){

    DNS stub=new DNSRemote();
```

```
Naming.rebind("rmi://localhost:5000/sonoo",stub);
```

Client.java

```
import java.rmi.*;
```

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

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

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

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

```
public class Client{
```

```
    public static void main(String args[]){
```

```
        try{
```

```
            Scanner in = new Scanner(System.in);
```

```
            DNS stub=(DNS)Naming.lookup("rmi://localhost:5000/sonoo");
```

```
            System.out.println("Enter the DOMAIN NAME :");
```

```
            String str=in.next();
```

```
            stub.find(str);
```

```
        }
```

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

}
```

**OUTPUT:**

Server:

216.58.197.46

Client:

Enter the DOMAIN NAME :

google.com

**RESULT:**

Thus the program is executed and output is obtained.

# CLIENT SIDE SCRIPTING



**Date - 28-12-18**

**Experiment – 3**

**Program – 1**

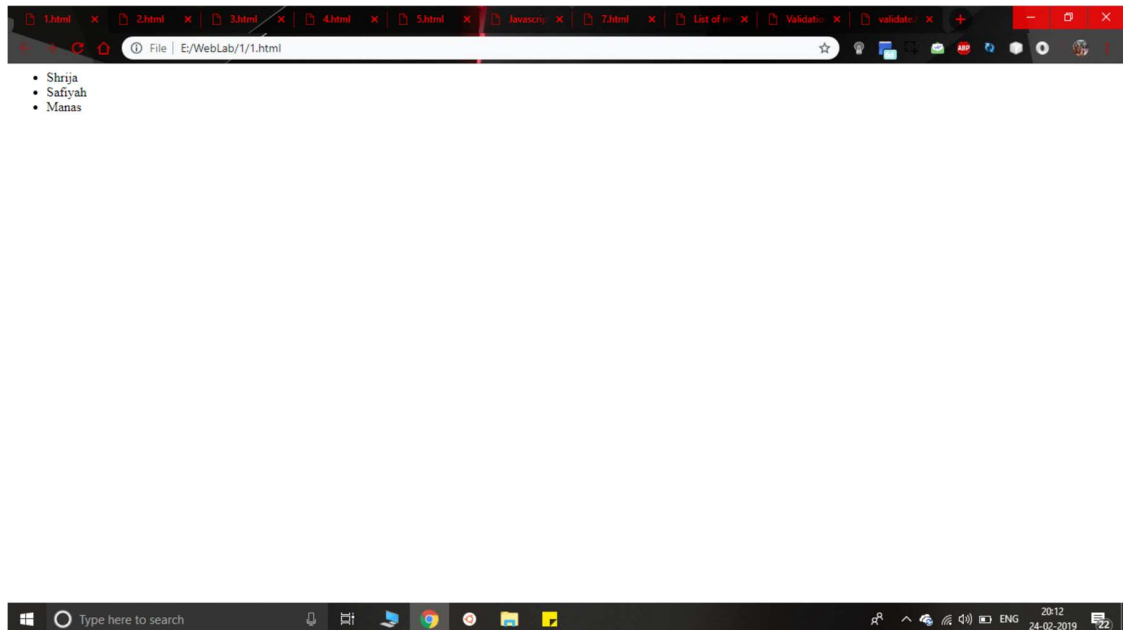
**AIM :**

Create a Web page HTML CSS that holds a bulleted list of the names of your friends. Make sure that the bullets are in plain circle.

**CODE :**

```
<ul>  
  
<li>Shrija</li>  
  
<li>Safiyah</li>  
  
<li>Manas</li>  
  
</ul>
```

**OUTPUT:**

**RESULT:**

Thus the program is successfully executed and output is verified.

**Program – 2****AIM :**

Create a web page in HTML CSS to display the maximum and minimum temperature of 5 cities using table.

**CODE :**

<style>

table, th, td {

```
border: 1px solid black;
border-collapse: collapse;
}
</style>
<table>
<tr>
<th> City </th>
<th> Max temperature(in celsius)</th>
<th> Min temperature(in celsius)</th>
</tr>
<tr>
<td>
Chennai
</td>
<td>
15
</td>
<td>
10
</td>
```

```
</tr>

<tr>

<td>

        Chennai
```

```
</td>
```

```
<td>

        15
```

```
</td>
```

```
<td>

        10
```

```
</td>
```

```
</tr>
```

```
<tr>
```

```
<td>

        Madurai
```

```
</td>
```

```
<td>

        45
```

```
</td>
```

&lt;td&gt;

17

&lt;/td&gt;

&lt;/tr&gt;

&lt;tr&gt;

&lt;td&gt;

Neywork

&lt;/td&gt;

&lt;td&gt;

30

&lt;/td&gt;

&lt;td&gt;

-20

&lt;/td&gt;

&lt;/tr&gt;

&lt;tr&gt;

&lt;td&gt;

Manhattan

&lt;/td&gt;

&lt;td&gt;

30

&lt;/td&gt;

&lt;td&gt;

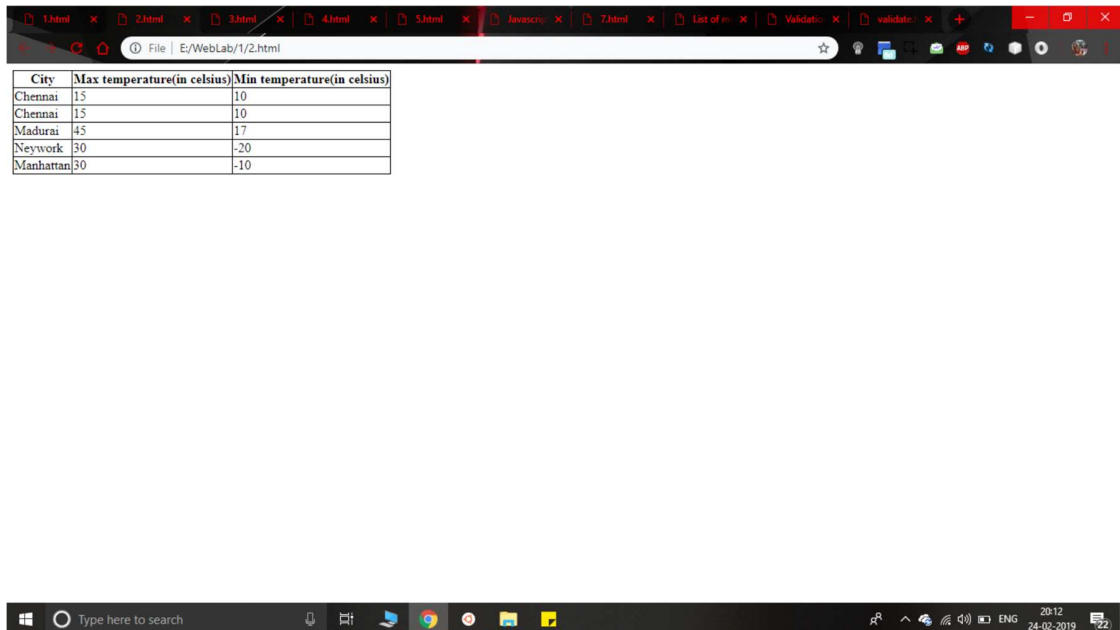
-10

&lt;/td&gt;

&lt;/tr&gt;

&lt;/table&gt;

**OUTPUT:**

**RESULT:**

Thus the program is successfully executed and output is verified.

**Program – 3****AIM :**

To design a web page in HTML that accepts username and password. Opens a new window when the password corresponds to a particular value is set by the developer.

**CODE :**

```
<form name = "RegForm" action = "validate.html" onsubmit="return func()"
method="post">
```

```
Username : <input type = "text" name = "uname">

Password :   <input type = "password" name = "pass">

<input type="submit" value="Login">

</form>

<script>

function func() {

    var x = document.forms["RegForm"]["pass"];

    if(x.value === "ramya123") {

        return true    }

    else {

        window.alert("Password mismatch");

        return false; }

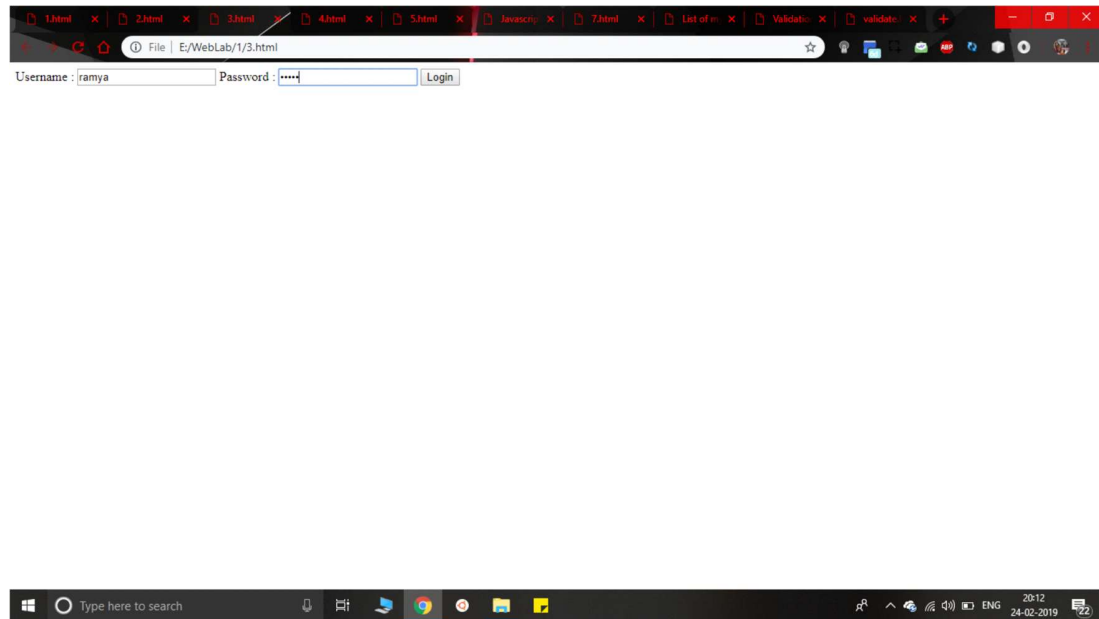
    }

</script>
```

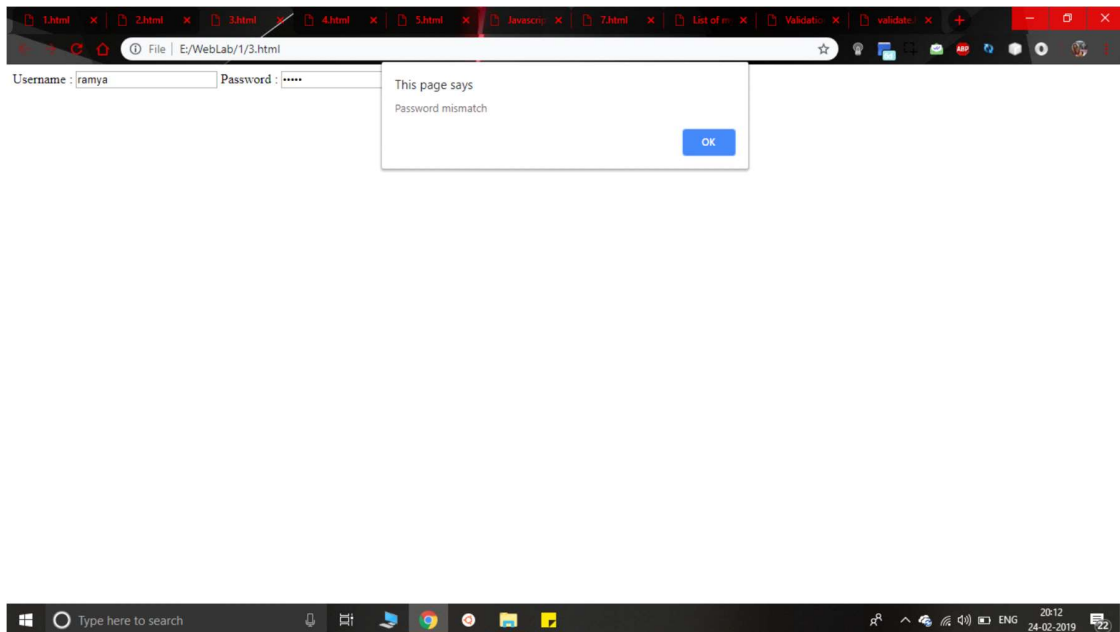
**OUTPUT:**

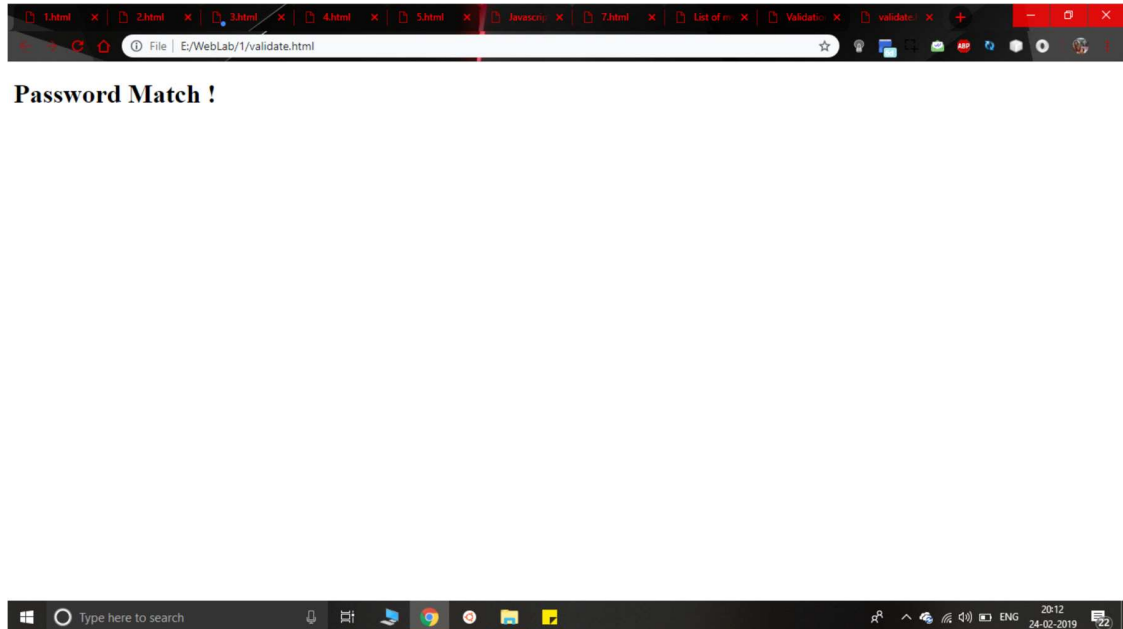


2016503543



## FAILURE CASE :



**SUCCESS CASE:****RESULT:**

Thus the program is successfully executed and output is verified.

**Program – 4****AIM :**

To design a web page in HTML that consists of 2 text boxes. When the page is first loaded set the focus to the first text box. The user should not be allowed to leave the box unless enters a value in it.

**CODE :**

```
<!DOCTYPE html>
```

```
<html>
```

```
<body>
```

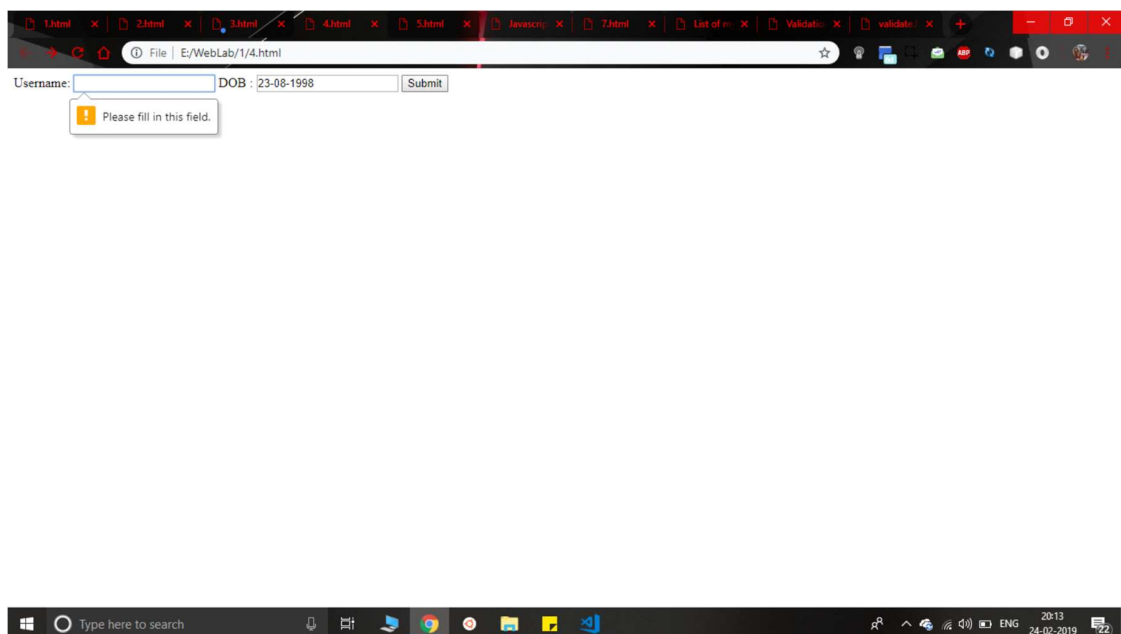
```
<form>
```

```
Username: <input type="text" name="username" required>
```

```
DOB : <input type="text">
```

```
<input type="submit">
```

```
</form>
```

**OUTPUT:****RESULT:**

Thus the program is successfully executed and output is verified.

**Program – 5****AIM :**

To display an alert box to alert the x and y co-ordinates of the cursor in JavaScript.

**CODE :**

```
<!DOCTYPE html>

<html>

<body>

<h2 onclick="showCoords(event)">Click here to get the x (horizontal) and y
(vertical) coordinates of the mouse pointer when it was clicked.</h2>

<script>

function showCoords(event) {

    var x = event.clientX;

    var y = event.clientY;

    var coords = "X coords: " + x + ", Y coords: " + y;

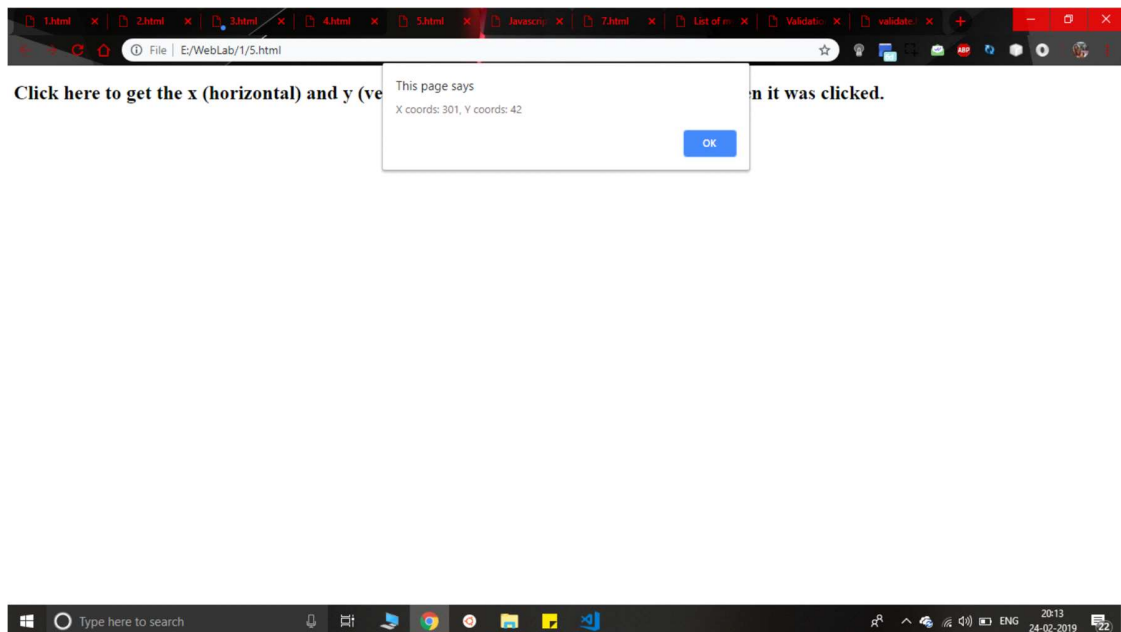
    window.alert(coords);}

</script>

</body>

</html>
```

**OUTPUT :**

**RESULT:**

Thus the program is successfully executed and output is verified.

**Program – 6****AIM :**

To design a simple arithmetic calculator in JavaScript.

**CODE :**

```
<html>
```

```
<head>
```

```
<title>Javascript Calculator</title>
```

```
<script language="javascript" type="text/javascript">
function multiply(){
a=Number(document.calculator.number1.value);
b=Number(document.calculator.number2.value);
c=a*b;
document.calculator.total.value=c;
}
</script>

<script language="javascript" type="text/javascript">
function addition(){
a=Number(document.calculator.number1.value);
b=Number(document.calculator.number2.value);
c=a+b;
document.calculator.total.value=c;
}
</script>

<script language="javascript" type="text/javascript">
function subtraction(){
a=Number(document.calculator.number1.value);
b=Number(document.calculator.number2.value);
```

```
c=a-b;

document.calculator.total.value=c;

}

</script>

<script language="javascript" type="text/javascript">

function division(){

a=Number(document.calculator.number1.value);

b=Number(document.calculator.number2.value);

c=a/b;

document.calculator.total.value=c;

}

</script>

<script language="javascript" type="text/javascript">

function modulus(){

a=Number(document.calculator.number1.value);

b=Number(document.calculator.number2.value);

c=a%b;

document.calculator.total.value=c;

}

</script>
```

```
</head>

<body>

<form name="calculator">

Number 1: <input type="text" name="number1">

Number 2: <input type="text" name="number2">

Get Result: <input type="text" name="total">

<input type="button" value="ADD" onclick="javascript:addition();">

<input type="button" value="SUB" onclick="javascript:subtraction();">

<input type="button" value="MUL" onclick="javascript:multiply();">

<input type="button" value="DIV" onclick="javascript:division();">

<input type="button" value="MOD" onclick="javascript:modulus();">

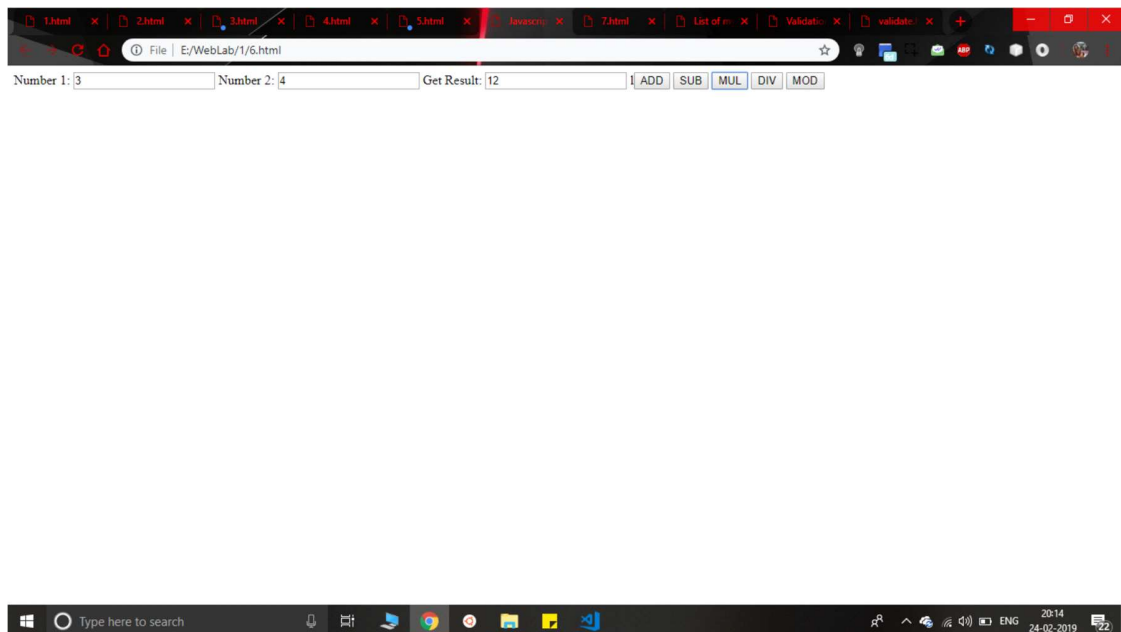
</form>

</body>

</html>
```

**OUTPUT :**



**RESULT:**

Thus the program is successfully executed and output is verified.

**Program – 7****AIM :**

To design a webpage to display a digital clock in JavaScript.

**CODE :**

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
<h1><i>DIGITAL CLOCK</i></h1>

<script>

function startTime() {

    var today = new Date();

    var h = today.getHours();

    var m = today.getMinutes();

    var s = today.getSeconds();

    m = checkTime(m);

    s = checkTime(s);

    document.getElementById('txt').innerHTML =

    h + ":" + m + ":" + s;

    var t = setTimeout(startTime, 500); }

function checkTime(i) {

    if (i < 10) {i = "0" + i}; // add zero in front of numbers < 10

    return i;}

</script>

</head>

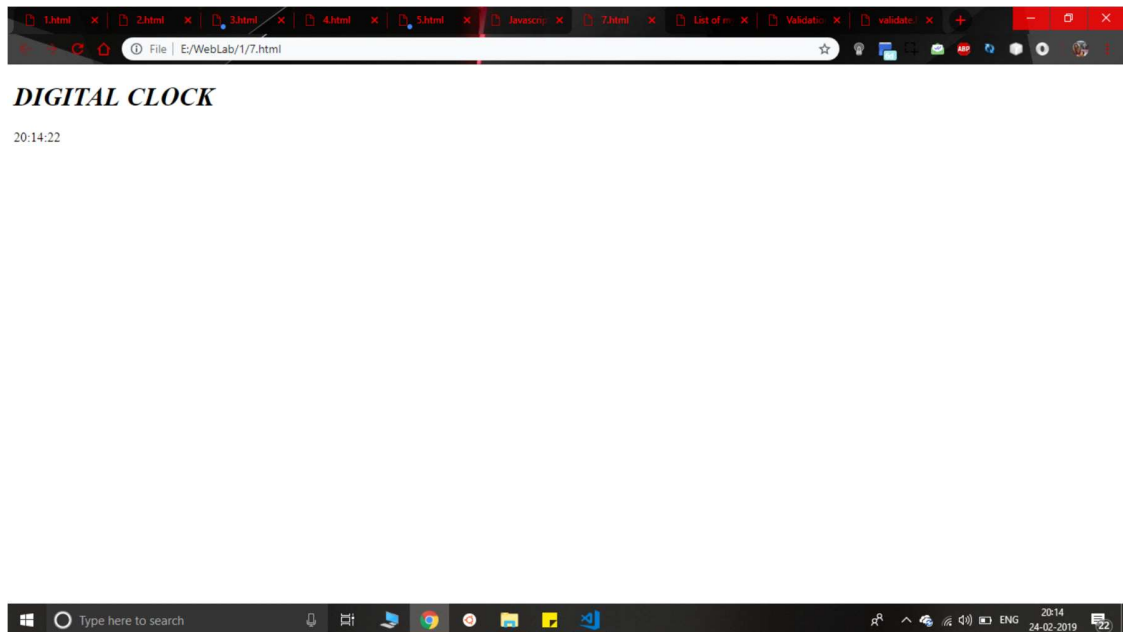
<body onload="startTime()">

<div id="txt"></div>

</body>
```

</html>

## OUTPUT :



## RESULT:

Thus the program is successfully executed and output is verified.

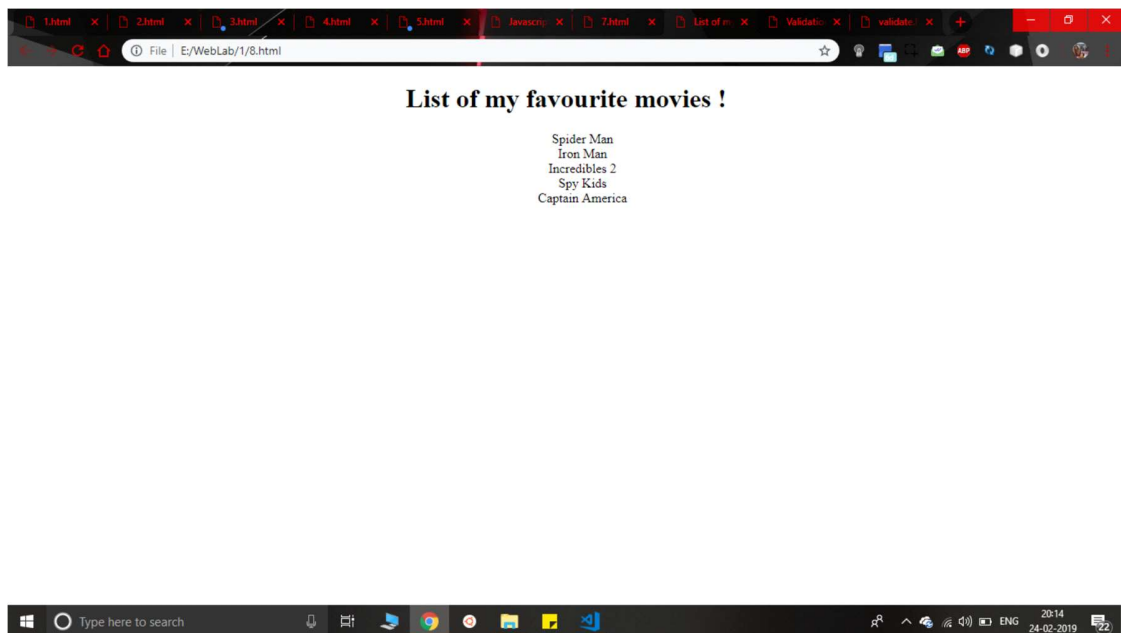
**Program – 8****AIM :**

To create, test and validate an XHTML document that describes an ordered list of 5 movies.

**CODE :**

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>List of my favourite movies</title>
<style>
ul {
    text-align: center;
    list-style-position: inside; }
</style>
</head>
<body>
<center>
<h1> List of my favourite movies !</h1>
```

```
<ol style="list-style-type:none">  
<li>Spider Man</li>  
<li>Iron Man</li>  
<li>Captain America</li>  
</ol>  
</center>  
</body>  
</html>
```

**OUTPUT:****RESULT:**

Thus the program is successfully executed and output is verified.

**Program – 9****AIM :**

To create, test and validate an XHTML document that has a form with :

- (i) A textbox to collect the user names.
- (ii) Four check boxes.
- (iii) A collection of 3 radio buttons

**CODE :**

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
```

```
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

```
<html xmlns="http://www.w3.org/1999/xhtml">
```

```
<head>
```

```
<title>Validation</title>
```

```
</head>
```

```
<body>
```

```
<form>
```

```
Username : <input type = "text" name="uname" /> <br/> <br/>
```

```
<input type="checkbox" name="vehicle1" /> Voter ID <br/>
```

```
<input type="checkbox" name="vehicle2" /> Aadhar Card <br/>
```

```
<input type="checkbox" name="vehicle3"/> Visa <br/>
```

```
<input type="checkbox" name="vehicle3"/>Passport<br/>
```

```
Gender: <br/> <input type="radio" name="gender" value="male" checked>  
Male<br>
```

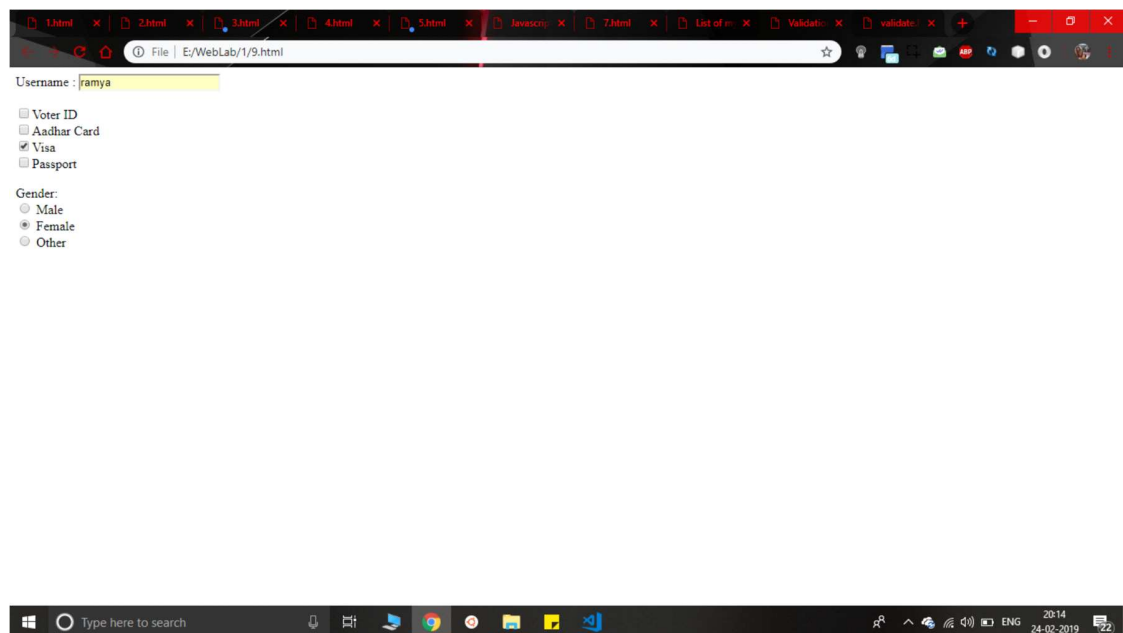
```
<input type="radio" name="gender" value="female"> Female<br>
```

```
<input type="radio" name="gender" value="other"> Other
```

```
</form>
```

```
</body>
```

```
</html>
```

**OUTPUT:****RESULT:**

Thus the program is successfully executed and output is verified.

# PHP



**Date : 04-01-19**

#### **Experiment-4**

##### **PROGRAM 1A**

##### **AIM:**

To write a PHP script that take in an array of strings and returns the list of unique strings in the parameter array

##### **PROGRAM:**

```
<?php
function compute($a1){
    $a2 = array_unique($a1);
    $a3 = array_values($a2);
    return $a3;
}
$a1 = array("a","b","c","a","d","e");
$a2 = compute($a1);
print "The unique elements are\n";
for($x1=0;$x1<count($a2);$x1++){
    print $a2[$x1]."";
}
?>
```

##### **OUTPUT:**

The unique elements are  
a b c d e

##### **RESULT:**

Thus, the program is executed and output is obtained.

**PROGRAM 1B****AIM:**

To write a PHP script that take in an array of numbers and returns average and median of parameter array.

**PROGRAM:**

```
<?php
function Average($numbers)
{
    # code...
    $k = array_sum($numbers);
    $n = count($numbers);
    $ans = $k/$n;
    return $ans;
}
function Median($numbers)
{
    if (count($numbers)%2 == 0) {
        # code...
        $mid=count($numbers)/2;
        return (($numbers[$mid-1]+$numbers[$mid])/2);
    }
    else {
        $mid=(count($numbers)-1)/2;
        return $numbers[$mid];
    }
}
$numbers=array(1,2,3,4,5,6,7,8,9,10);
echo ("THE AVERAGE IS: " .Average($numbers)."\n");
echo ("THE MEDIAN IS: ".Median($numbers)."\n");
?>
```

**OUTPUT:**

THE AVERAGE IS: 5.5

THE MEDIAN IS: 5.5

**RESULT:**

Thus the program is successfully executed and output is obtained.

**PROGRAM 1C****AIM:**

To write a PHP script that take in an array of strings and returns the list of three strings that occur most frequently in parameter array.

**PROGRAM:**

```
<?php
function compute($a1){
    $map = array_count_values($a1);
    arsort($map);
    $a2 = array_keys($map);
    return $a2;
}
$a1 = array("a","b","c","a","b","a","b","a","b","a");
$a2 = compute($a1);
print "The top 3 elements are\n";
for($x1=0;$x1<3;$x1++){
    print $a2[$x1];
    print " ";
}
?>
```

**OUTPUT:**

The top 3 elements are

a b c

**RESULT:**

Thus, the program is executed and output is obtained.

**PROGRAM 1D****AIM:**

To write a PHP script that take in an array of numbers (pass by value) and two arrays (pass by reference). The first pass by reference must have numbers less than zero and second must have numbers greater than 0.

**PROGRAM:**

```
<?php
function filter($a, &$amp;p, &$amp;n)
{
    # code...
    $num = count($a);
    for ($i=0; $i < $num; $i++) {
        # code...
        if($a[$i] > 0)
        {
            array_push($p,$a[$i]);
        }
        elseif($a[$i] < 0)
        {
            array_push($n,$a[$i]);
        }
    }
}
$numbers=array(-1,2,-3,4,-5,-10,11,100,200);
$pos = array();
```

```
$neg = array();
filter($numbers,$pos,$neg);
echo "Positive array values are:\n";
for ($i=0; $i < count($pos); $i++) {
    # code...
    echo($pos[$i]."\n");
}
echo "\n";
echo "Negative array values are:\n";
for ($i=0; $i < count($neg); $i++) {
    # code...
    echo($neg[$i]."\n");
}
echo "\n";
?>
```

**OUTPUT:**

Positive array values are:

2 4 11 100 200

Negative array values are:

-1 -3 -5 -10

**RESULT:**

Thus, the program is executed and output is obtained.

**PROGRAM 1E****AIM:**

To write a PHP script that take in a string of numbers separated by spaces and returns first four-digit number in the string, else return none.

**PROGRAM:**

```
<?php
```

```
function compute($a1){
    $a3 = explode(' ', $a1);
    $a2 = array_values($a3);
    for($x1=0;$x1<count($a2);$x1++){
        if(strlen($a2[$x1]) == 4){
            $ans = $a2[$x1];
            return $ans;
        }
    }
    return "false";
}
$a1 = "1 2 3 4 5 6 7";
$a2 = compute($a1);
print $a2 . "\n";
$a1 = "1 2 3 4 5 6 7 8 8999 19919";
$a2 = compute($a1);
print $a2 . "\n";
?>
```

**OUTPUT:**

false  
8999

**RESULT:**

Thus, the program is executed and output is obtained.

**PROGRAM 1F****AIM:**

To write a PHP script that take in a file variable of a file of text where the words are separated by spaces or colons and returns the word that appears most often in the file.

**PROGRAM:**

```
<?php
function findMax($file,$fname)
{
    # code...
    $filecontents = file_get_contents($fname);
    $words = preg_split('/[\s:]+/', $filecontents, -1, PREG_SPLIT_NO_EMPTY);
    $arr = array_count_values($words);
    print_r($words);
    $max = 0;
    $word = "";
    foreach ($arr as $key => $value) {
        # code...
        if($value > $max)
        {
            $max = $value;
            $word = $key;
        }
    }
    return $word;
}
```

```
$fname = "p7.txt";
$file = fopen($fname,"r") or die("Unable to open file");
echo findMax($file,$fname);
?>
```

TXT

shrija ramya safiya ja:rum:saf:madhu manas

OUTPUT:

Array

(

- [0] => shrija
- [1] => ramya
- [2] => safiya
- [3] => ja
- [4] => rum
- [5] => saf
- [6] => madhu
- [7] => manas

)

shrija

**RESULT:**

Thus, the program is executed and output is obtained.

## PROGRAM 1G

**AIM:**

To write a PHP script that take in a string containing words that are delimited on the left by spaces and on the left with spaces, commas, periods or question marks and returns three most common words in the string that has 3 or more letters.

**PROGRAM:**

```
<?php
function compute($a1){
    $a3 = preg_split('/[\s]+|\.\./,$a1);
    $a2 = array_values($a3);
    $ff = array();
    for($x=0;$x<count($a2);$x++){
        if(strlen($a2[$x])>3){
            array_push($ff,$a2[$x]);
        }
    }
}
```



```
    }  
  }  
  $map = array_count_values($ff);  
  arsort($map);  
  $ans = array_keys($map);  
  return $ans;  
}  
$a1 = "hello this is shrijasathyanarayanan..., welcome.";  
$a2 = compute($a1);  
print "The top THREE elements are\n";  
for($x1=0;$x1<3;$x1++){  
    print $a2[$x1];  
    print "";  
}  
?>
```

**OUTPUT:**

The top THREE elements are  
hello this shrijasathyanarayanan

**RESULT:**

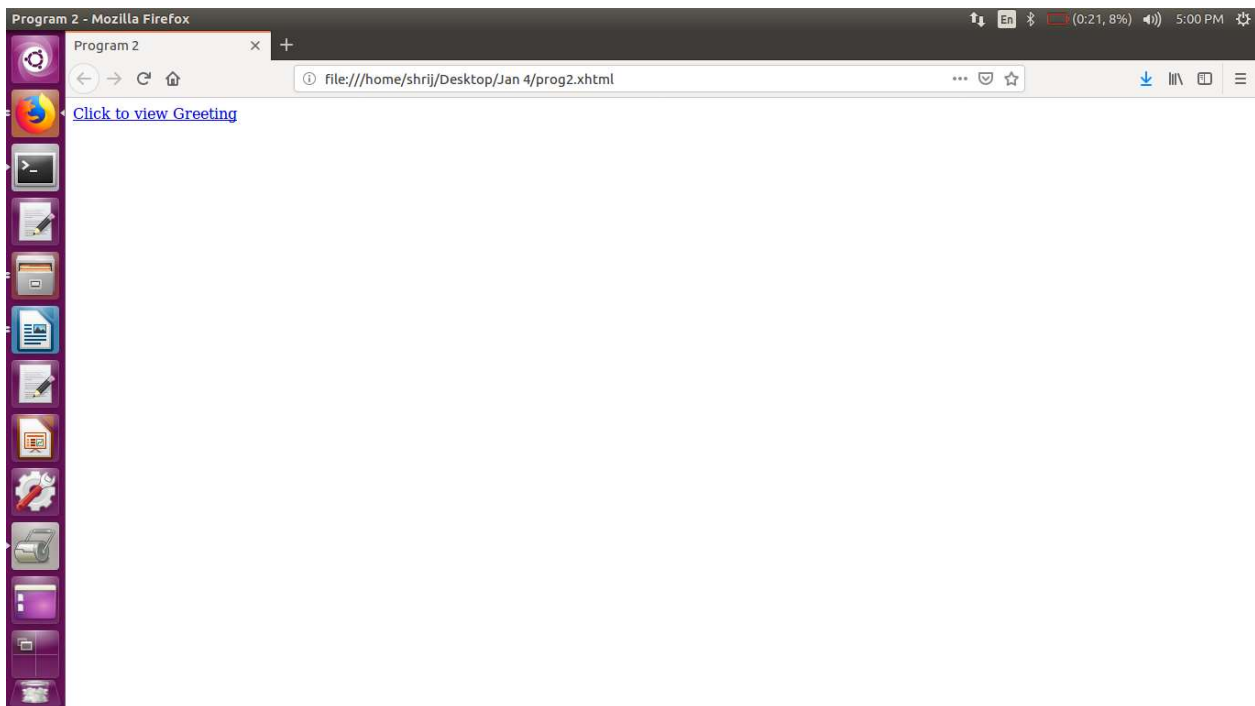
Thus, the program is executed and output is obtained.

**PROGRAM 2****AIM:**

To write an XHTML document that includes an anchor tag that calls a PHP document. Also write the PHP document, which returns a randomly chosen greeting of five different greetings. The greetings must be stored as constant strings in the script. A random number between 0 and 4 can be computed by a random function.

**PROGRAM:**

```
<?php
function greetingCall(){
    define("GREETING1","Have a nice day.");
    define("GREETING2","Thank you.");
    define("GREETING3","Welcome.");
    define("GREETING4","Nice meeting you.");
    define("GREETING5","Hope to see you again.");
    $greet = array(GREETING1,GREETING2,GREETING3,GREETING4,GREETING5);
    $index = rand(0,4);
    echo "<div>".$greet[$index]."</div>";
}
greetingCall();
?>
```

**OUTPUT:**

**RESULT:**

Thus, the program is executed and output is obtained.

**PROGRAM 3****AIM:**

To write an XHTML document to create a form that collects favourite popular songs, including the name of the song, the composer, and the performing artist or group. The document must call one PHP script where the form is submitted and another to request a current list of survey results.

**PROGRAM:**

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Transitional//EN""http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>
    Program 3
</title>
</head>
<body>
<?php
    $display = "";
    $content = $artist = $name = $song = $composer = "";
    if ($_SERVER["REQUEST_METHOD"] == "POST")
    {
        $file = fopen("prog3.txt","r");
        $content = "Name: " . $_POST["name"] . "\n" . "Favorite Song: " .
$_POST["song"] . "\n" . "Composer: " . $_POST["composer"] . "\n" . "Artist/Group: " .
$_POST["artist"] . "\n\n";
```

2016503543

```

        $myfile = file_put_contents('prog3.txt', $content.PHP_EOL , FILE_APPEND |
LOCK_EX);
        $display = file('prog3.txt');
    }
?>

```

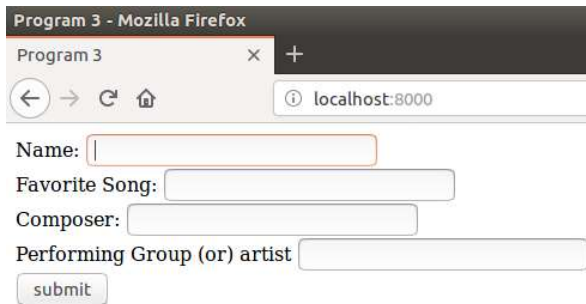
```

<form method="post" action="<?php
htmlspecialchars($_SERVER["PHP_SELF"]);?>">
    Name: <input type="text" name="name" id="name" value="<?php echo
$name;?>" /> <br />
    Favorite Song: <input type="text" name="song" id="song" value="<?php echo
$song;?>" /> <br />
    Composer: <input type="text" name="composer" id="composer"
value="<?php echo $composer;?>" /> <br />
    Performing Group (or) artist <input type="text" name="artist" id="artist"
value="<?php echo $artist;?>" /> <br />
    <input type="submit" value="submit" /> <br />
<br />
<br />
<div><?php
    for ($i=0; $i < count($display); $i++) {
        # code...
        echo $display[$i] . "<br>";
    }
?></div>
</form>
</body>
</html>

```

**OUTPUT:**

2016503543



Name: a  
 Favorite Song: b  
 Composer: c  
 Artist/Group: d

Name:  
 Favorite Song:  
 Composer:  
 Artist/Group:

**RESULT:**

Thus, the program is executed and output is obtained.

**PROGRAM 4****AIM:**

To write an XHTML document to create a form that collects favourite popular songs, including the name of the song, the composer, and the performing artist or group. The document must call one PHP script where the form is submitted and another to request a current list of survey results.

**PROGRAM:**

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Transitional//EN""http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>
    Program 4
</title>
</head>
```

2016503543

```
<body>
<?php
    $mobileErr = "";
    $mobile = "";
    $name = "";
    if ($_SERVER["REQUEST_METHOD"] == "POST")
    {
        if(empty($_POST["mobile"])) {
            $mobileErr = "Mobile number is required";
        } else {
            $mobile = test_input($_POST["mobile"]);
            if(!preg_match("/^\+\d{2}-\d{4}-\d{6}$/", $mobile))
            {
                $mobileErr = "invalid Format";
            }
        }
    }
    function test_input($data) {
        $data = trim($data);
        $data = stripslashes($data);
        $data = htmlspecialchars($data);
        return $data;
    }
?>
```

<h2>Form Validation</h2>

<form method="post" action="<?php

htmlspecialchars(\$\_SERVER["PHP\_SELF"]);?>">

    Name: <input type="text" name="name" value="<?php echo \$name;?>"

/> <br><br>

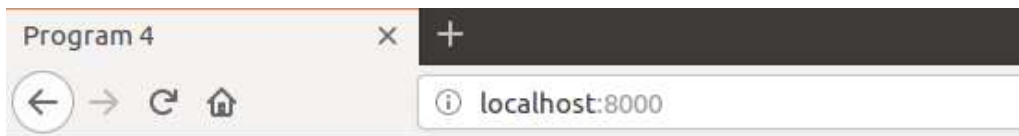
2016503543

```
Mobile: <input type="text" name="mobile" value="<?php echo $mobile;?>" />
<span class="error">* <?php echo $mobileErr;?> </span>
<br /> <br />
<input type="submit" name="submit" value="Submit" />
</form>
```

&lt;/body&gt;

&lt;/html&gt;

OUTPUT:



## Form Validation

Name: Mobile:  \* Mobile number is required

RESULT:

Thus, the program is executed and output is obtained.

PROGRAM 5

AIM:

To modify the PHP script from Exercise 2 to count the number of visitors and display the numbers for each visitor.

PROGRAM:

&lt;?php

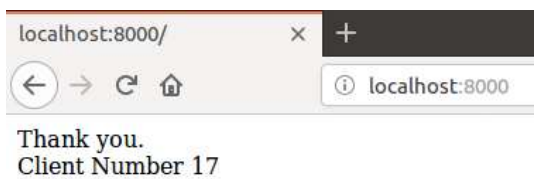
2016503543

```
function greetingCall(){
    define("GREETING1","Have a nice day.");
    define("GREETING2","Thank you.");
    define("GREETING3","Welcome.");
    define("GREETING4","Nice meeting you.");
    define("GREETING5","Hope to see you again.");
    $greet = array(GREETING1,GREETING2,GREETING3,GREETING4,GREETING5);
    $index = rand(0,4);
    $fp = file_get_contents("prog5.txt");
    $val = (int)$fp;
    $val+=1;
    $fp = (string)$val;
    file_put_contents("prog5.txt",$fp);
    print "<div>".$greet[$index]."</div>";
    print "<div> Client Number ".$val."</div>";
}
greetingCall();
?>
```

TXT

10

OUTPUT:



RESULT:

Thus, the program is executed and output is obtained.



## PROGRAM 6

## AIM:

To implement the following modules using Server-Side Scripting (PHP)

- (i) Gathering form data.
- (ii) Querying the database.
- (iii) Response generation.
- (iv) Session management.
- (v) Use MySQL or JDBS or Oracle.

## PROGRAM:

```
<html>
<head>
<style>
    body{
        display: inline-block;
        width:100%;
        height:100%;
        text-align:center;
    }
</style>
</head>
<body>
<?php
    session_start();
    ?>
<?php
    $regno = "";
    $pwd = "";
    $regErr = "";
```

```
$pwdErr = "";
$name = "";
$address = "";
$mobile = "";
$time = $_SERVER['REQUEST_TIME'];
$timeout_duration = 60;
if (isset($_SESSION['LAST_ACTIVITY']) && ($time -
$_SESSION['LAST_ACTIVITY']) > $timeout_duration) {
    session_unset();
    session_destroy();
    session_start();
}
if ($_SERVER["REQUEST_METHOD"] == "POST")
{
    if(empty($_POST["regno"])) {
        $regErr = "Student Registration number is required";
    }
    if(empty($_POST["pwd"])) {
        $pwdErr = "Password is required";
    }
    if(isset($_POST["reg"]) && isset($_POST["pwd"])){
        $servername = "localhost";
        $username = "root";
        $password = "srini1998";
        $dbname = "Base1";
        $conn = new mysqli($servername,$username,$password,$dbname);
        if($conn -> connect_error){
            die("Connection failed: ".$conn->connect_error);
        }
    }
}
```

```
$regno = $_POST['regno'];
$pwd = $_POST['pwd'];

$sql = "SELECT name,address,mobile,id from Tab1 where id = '$regno' and
password = '$pwd'";
$result = $conn->query($sql);
if($result->num_rows > 0){
    while($row = $result->fetch_assoc()){
        $name = $row["name"];
        $address = $row["address"];
        $mobile = $row["mobile"];
        $regno = $row["id"];
    }
}
else{
    print "Data mismatch. Please try again.";
    $regno = "";
    $pwd = "";
}

$conn ->close();
}
$_SESSION['LAST_ACTIVITY'] = $time;
}
?>
<div><h1>Depatment of Computer Technology</h1></div>
<form class = "form1" method="post" action="<?php
htmlspecialchars($_SERVER["PHP_SELF"]);?>">
<h3>
```

2016503543

```

Registration No: <input type="text" name="regno" id="regno"
value="<?php echo $regno;?>" />
<span class="error">* <?php echo $regErr;?> </span>
Password: <input type="password" name="pwd" id="pwd" value="<?php
echo $pwd;?>" />
<span class="error">* <?php echo $pwdErr;?> </span>
<input type="submit" name="submit" value="submit" />
<div>
Student Name: <input type="text" name="name" id="name"
value="<?php echo $name;?>" />
Registration No: <input type="text" name="reg" id="reg" value="<?php
echo $regno;?>" />
Address: <input type="text" name="address" id="address"
value="<?php echo $address;?>" />
Student Name: <input type="text" name="mobile" id="mobile"
value="<?php echo $mobile;?>" />
</div>
</h3>
</form>
</body>
</html>

```

**OUTPUT:**


---

**Department of Computer Technology**

Registration No:  \*

Password:  \*

Student Name:  Registration No:  Address:  Mobile:

**RESULT:**

Thus, the program is executed and output is obtained.

# PYTHON

**Date - 11-01-19**

**Experiment-5**

**PROGRAM 1**

**AIM:**

To create a new program called HelloWorld.py. This file should be used to write "Hello World!" program.

**PROGRAM:**

```
print("Hello World")
```

**OUTPUT:**

Hello World

**RESULT:**

Thus, the program is executed and output is obtained.

**PROGRAM 2**

**AIM:**

To write a function reverse to reverse a list without using the reverse function.

**PROGRAM:**

```
def reverse_list(a):  
    b = list()  
    for i in range(0,len(a)):  
        b.append(a[len(a)-i-1])  
    return b  
a = [1,2,3,4,5]  
b = reverse_list(a)  
print(b)
```

**OUTPUT:**

[5, 4, 3, 2, 1]

**RESULT:**

Thus, the program is executed and output is obtained.

**PROGRAM 3****AIM:**

To write a method fact that takes a number from the user and prints the factorial.

**PROGRAM:**

```
from math import factorial
def fact(n):
    return factorial(n)
a = 17
print(fact(a))
```

**OUTPUT:**

355687428096000

**RESULT:**

Thus, the program is executed and output is obtained.

**PROGRAM 4****AIM:**

To write a GUI for the expression calculator using tk.

**PROGRAM:**

```
from tkinter import *
expression = ""
def press(num):
    global expression
    expression = expression + str(num)
    equation.set(expression)

def equalpress():
    try:
        global expression
        total = str(eval(expression))
```

2016503543

```
        equation.set(total)
        expression = ""
    except:
        equation.set(" error ")
        expression = ""

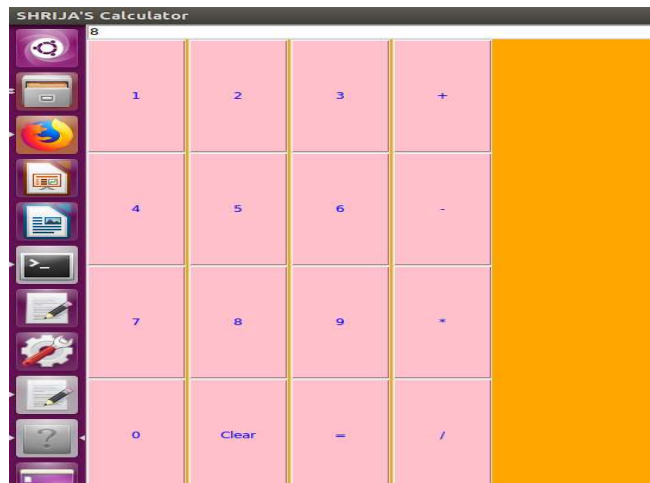
def clear():
    global expression
    expression = ""
    equation.set("")

if __name__ == "__main__":
    gui = Tk()
    gui.configure(background = "orange")
    gui.title("SHRIJA'S Calculator")
    gui.geometry("265x1025")
    equation = StringVar()
    expression_field = Entry(gui, textvariable = equation)
    expression_field.grid(columnspan=1000, ipadx=1000)
    equation.set('Enter expression')
    button2 = Button(gui, text=' 2 ', fg='blue', bg='pink', command=lambda: press(2),
height=10, width=7)
    button2.grid(row=2, column=1)
    button3 = Button(gui, text=' 3 ', fg='blue', bg='pink', command=lambda: press(3),
height=10, width=7)
    button3.grid(row=2, column=2)
    button4 = Button(gui, text=' 4 ', fg='blue', bg='pink', command=lambda: press(4),
height=10, width=7)
    button4.grid(row=3, column=0)
    button5 = Button(gui, text=' 5 ', fg='blue', bg='pink', command=lambda: press(5),
```



2016503543

```
height=10, width=7)
    button5.grid(row=3, column=1)
    button9 = Button(gui, text=' 9 ', fg='blue', bg='pink', command=lambda: press(9),
height=10, width=7)
    button9.grid(row=4, column=2)
    button0 = Button(gui, text=' 0 ', fg='blue', bg='pink', command=lambda: press(0),
height=10, width=7)
    divide.grid(row=5, column=3)
    equal = Button(gui, text=' = ', fg='blue', bg='pink', command=equalpress,
height=10, width=7)
    equal.grid(row=5, column=2)
    clear = Button(gui, text='Clear', fg='blue', bg='pink', command=clear, height=10,
width=7)
    clear.grid(row=5, column='1')
    gui.mainloop()
```

**OUTPUT:****RESULT:**

Thus, the program is executed and output is obtained.

**PROGRAM 5****AIM:**

To write a procedure to install packages requests, flask and explore them using pip.

**PROGRAM:****CLIENT**

```
import httpplib2
http = httpplib2.Http()
a = input("Enter a:")
b = input("Enter b:")
operation = "/add/" + str(a) + "&" + str(b)
url = "http://localhost:3006" + operation
(response_headers, content) = http.request(url, method="GET")
print(str(content))
```

**SERVER**

```
from flask import Flask
app = Flask(__name__, static_url_path = "")
@app.route('/add/<int:x>&<int:y>', methods = ['GET'])
def add(x,y):
    print("result:",str(x+y))
    return str(x+y)

if __name__ == '__main__':
    app.run(host='127.0.0.1',port = 3006,debug = True)
```

**OUTPUT:**

```
ramya@Ramya: /mnt/c/Users/Sruthi/Downloads
ramya@Ramya:/mnt/c/Users/Sruthi/Downloads$ python prog5client.py
Enter a:3
Enter b:4
7
ramya@Ramya:/mnt/c/Users/Sruthi/Downloads$
```

```
ramya@Ramya: /mnt/c/Users/Sruthi/Downloads
ramya@Ramya:/mnt/c/Users/Sruthi/Downloads$ python prog5server.py
* Serving Flask app "prog5server" (lazy loading)
* Environment: production
  WARNING: Do not use the development server in a production environment.
  Use a production WSGI server instead.
* Debug mode: on
* Running on http://localhost:3006/ (Press CTRL+C to quit)
* Restarting with stat
* Debugger is active!
* Debugger PIN: 668-146-386
('result:', '7')
127.0.0.1 - - [26/Feb/2019 18:48:55] "GET /add/3&4 HTTP/1.1" 200 -
```

**RESULT:**

Thus, the program is executed and output is obtained.

**PROGRAM 6****AIM:**

To write a script that imports Requests and fetches the content from a page.

**PROGRAM:**

```
from bs4 import BeautifulSoup
import requests
def scrapeSite(url,out):
    r = requests.get(url)
    txt = r.text
    htmlCode = BeautifulSoup(txt,'html.parser')
    if htmlCode.h1:
        ans = htmlCode.find_all('h1')
        for i in ans:
            out.write(str(i.string) + "\n")
    if htmlCode.p:
        ans = htmlCode.find_all('p')
        for i in ans:
            out.write(str(i.string) + "\n")
```

**try:**

```
url = input("Enter a url:")
out = open("out6.txt","w")
scrapeSite(url,out)
out.close()
except IOError as e: print(e)
```

**OUTPUT:**

2016503543

Enter a url:<https://stackoverflow.com/questions/37541718/how-to-run-a-php-file-from-ubuntu>

OUT6.TXT

How to run a PHP file from ubuntu?

None

Q&A for work. A dedicated place to share your team's knowledge.

How to run a php file from ubuntu platform in the localhost?

I have also installed LAMP in my system.

When I try to run the php file, in the browser, it says "The requested URL is not found-404 ERROR found".

I do not know how to proceed with this.

My php files are in the directory as shown here "/usr/var/html/a.php".

There are two options.

Access the php file through a local webserver(ie thru a local website). The web-server will deal with the requested php file. It will use either,

None

None

Use the php binary directly from a terminal.

Thanks for contributing an answer to Stack Overflow!

None

None

Required, but never shown

Required, but never shown

None

asked

None

viewed

None

active

2016503543

2 years, 8 months ago

None

**RESULT:**

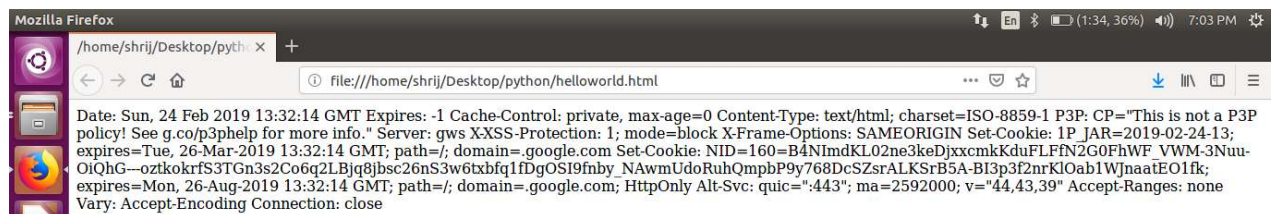
Thus, the program is executed and output is obtained.

**PROGRAM 7****AIM:**

To write a simple script that serves a simple HTTP response and a simple HTML page using Python.

**PROGRAM:**

```
import webbrowser
from urllib.request import urlopen
f = open("helloworld.html","w")
res = urlopen("https://www.google.com")
print(res.info())
res.close()
message = """<html>
<head> </head>
<body> <p>""" + str(res.info()) + """</p> </body>
</html>"""
f.write(message)
f.close()
webbrowser.open_new_tab('helloworld.html')
```

**OUTPUT:****RESULT:**

Thus, the program is executed and output is obtained.

**PROGRAM 8****AIM:**

To implement the following modules using Server Side Scripting (Python)

- (i) Gathering form data.
- (ii) Querying the database.
- (iii) Response generation.
- (iv) Session management.
- (v) Use MySQL or JDBS or Oracle .

**PROGRAM:**

A)

```
import mysql.connector
from flask import Flask, request
app = Flask(__name__, static_url_path = "")
@app.route('/result/<string:x>&<string:y>', methods = ['GET'])
def result(x,y):
    try:
        db =
mysql.connector.connect(host="localhost",user="root",password="srini1998",database="Base1")
        print(db)
        cursor = db.cursor()
        sql = "SELECT * from Tab2 where id = '"+x+"' and password='"+y+"';"
        print(sql)
        # return "S"
        cursor.execute(sql)
        result = cursor.fetchall()
        print(result)
        if len(result) == 0:
            return "Please Verify Credentials"
```

```
ans = ""
for a in result:
    for b in range(0,len(a)-1):
        ans=ans+"#"+a[b]
    # return "S"
return ans
except mysql.connector.Error as e:
    print(e)
if __name__ == '__main__':
    app.run(host='127.0.0.1',port=3005,debug = True)
```

B)

```
import webbrowser
f = open("helloworld1.html","w")
```

```
msg="""<!DOCTYPE html>
<html>
<head>
<title>Result page </title>
<style>
    .intro{
        text-align:center;
    }
    #result-login{
        display:inline-block;
        width: 100%;
        text-align:center;
    }
    #details{
        text-align:center;
    }
    """
```



```
#publish-result{
    visibility: hidden;
}
table{
    width: 70%;
}
table,th,td{
    border: 1px solid black;
}
th, td {
    padding: 15px;
    text-align: center;
}
</style>
<script>
function myFunction(){
    var reg = document.getElementById("roll_no").value;
    var pass = document.getElementById("pwd").value;
    if(reg.length ==0 && pass.length == 0){
        alert("Please fill in both Register number and password");
    }
    if(reg.length== 0){
        alert("Roll No is required");
    }
    if(pass.length ==0){
        alert("Password is required")
    }
    var url = "http://127.0.0.1:3005/result/"+reg+"&"+pass;
    var xmlhttp = new XMLHttpRequest();
    xmlhttp.open("GET",url,false);
```

2016503543

```
xmlHttp.send();
var ans = xmlHttp.responseText;
console.log(ans);
var arr = ans.split("#");
if(arr.length == 1){
    alert(arr);
    document.getElementById("roll_no").value = "";
    document.getElementById("pwd").value = "";
}else{
    arr = arr.slice(1);
    console.log(arr)
    document.getElementById("student-name").value = arr[0];
    document.getElementById("student-reg").value = arr[1];
    document.getElementById("student-sem").value = arr[2];
    document.getElementById("sub1-id").innerHTML = arr[3];
    document.getElementById("sub1-name").innerHTML = arr[4];
    document.getElementById("sub1-grade").innerHTML = arr[5];
    document.getElementById("sub2-id").innerHTML = arr[6];
    document.getElementById("sub2-name").innerHTML = arr[7];
    document.getElementById("sub2-grade").innerHTML = arr[8];
    document.getElementById("result-login").style.display = "none";
    document.getElementById("publish-result").style.visibility = "visible";
}
}
</script>
</head>
<body>
<div class="intro">
<h1>Madras Institue of Technology</h1>
</div>
```

```
<div id="result-login">
    Roll No : <input type="text" id="roll_no" />
<br><br>
    Password : <input type="password" id="pwd" />
<br><br>
<button onclick="myFunction()">Login</button>
</div>
<div id="publish-result">
<div id = "details">
    Name : <input type = "text" id="student-name" value="" />
<br><br>
    Register Number : <input type = "text" id="student-reg" value="" />
<br><br>
    Sem : <input type = "text" id="student-sem" value="" />
<br><br>
</div>
<table class="result" align="center">
<tr>
<th>Course Id</th>
<th>Course Name</th>
<th>Grade</th>
</tr>
<tr>
<td id="sub1-id"> </td>
<td id="sub1-name"> </td>
<td id="sub1-grade"> </td>
</tr>
<tr>
<td id="sub2-id"> </td>
<td id="sub2-name"> </td>
```

```
<td id="sub2-grade"> </td>
</tr>
</table>
</div>
</body>
</html>"""
f.write(msg)
f.close()
webbrowser.open_new_tab("helloworld1.html")
```

**OUTPUT:****RESULT:**

Thus, the program is executed and output is obtained.

# JAVA SERVLETS

**Date - 08-02-19**

**Experiment-6**

### **PROGRAM 1**

#### **AIM:**

To create a servlet program that makes Ordered list of four random numbers

#### **PROGRAM:**

Index.html

```
<!DOCTYPE html>
<html>
<head>
</head>
<body>
<a href= "/RandomNumbers/Random1">Check Out</a>
</body>
</html>
```

#### **SERVLET :**

```
package com.hello.random;
import java.io.IOException;
import java.io.PrintWriter;

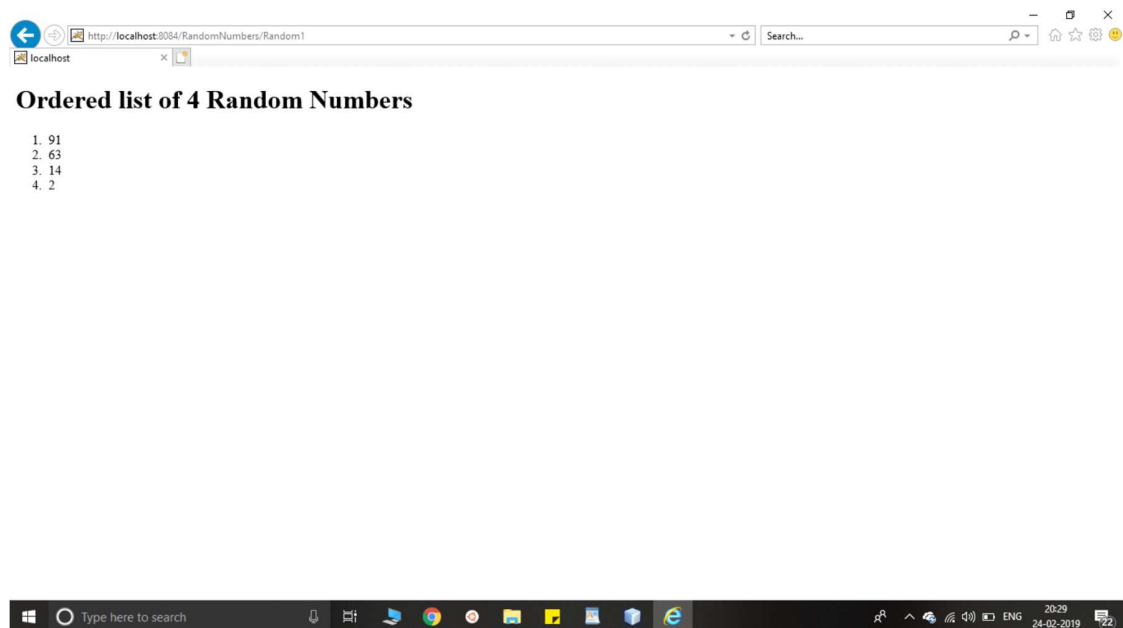
public class Random1 extends HttpServlet {
protected void processRequest(HttpServletRequest request, HttpServletResponse
response)
throws ServletException, IOException {
response.setContentType("text/html;charset=UTF-8");
try (PrintWriter out = response.getWriter()) {
out.println("<!DOCTYPE html>");
out.println("<html>");
out.println("<head>");
```

2016503543

```
        out.println("<h1>Ordered list of 4 Random Numbers</h1>");
        out.println("</head>");
        out.println("<body>");
        out.println("<ol>");
        Random rand = new Random();
        for(int i=0;i<4;i++) {
            out.println("<li>" + rand.nextInt(100) + "</li>");
        }
        out.println("</ol>");
        out.println("</body>");
        out.println("</html>");
    }
}

@Override
protected void doGet(HttpServletRequest request, HttpServletResponse
response)
    throws ServletException, IOException {
    processRequest(request, response);
}
}
```

**OUTPUT:**

**RESULT :**

Thus the program is successfully executed and output is obtained.

**PROGRAM 2****AIM:**

To create a servlet program that uses a loop to output an HTML table with 25 Rows and 3 columns.

**PROGRAM:**

index.html

```
<!DOCTYPE html>
<html>
<head>
</head>
<body>
<a href="/htmltable/table">Check Out</a>
</body>
```



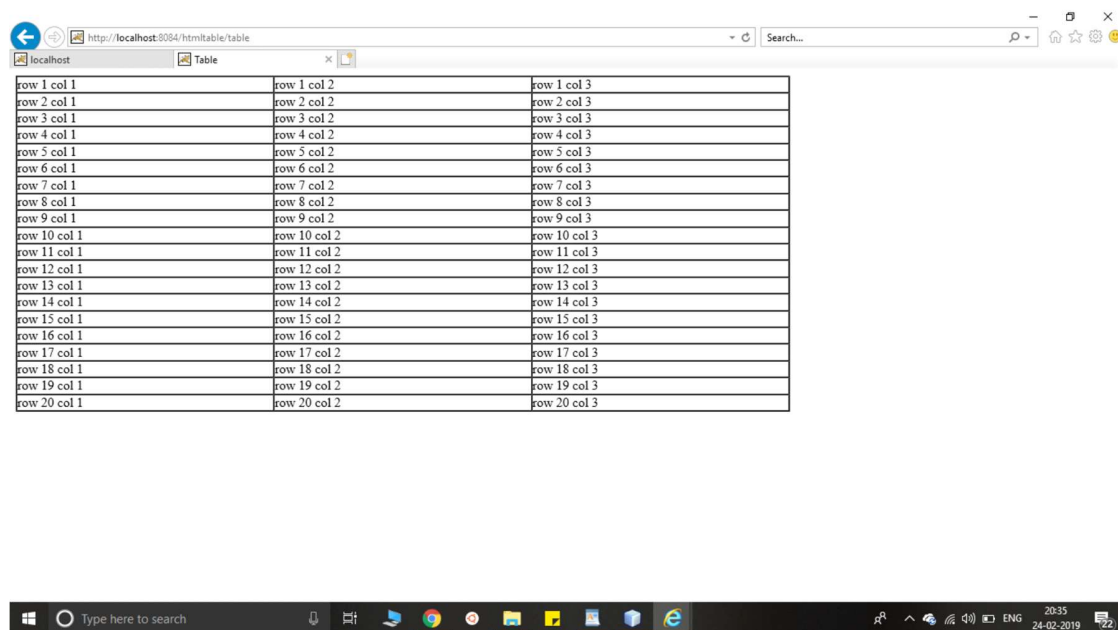
```
</html>
SERVLET:
package com.table;
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
public class table extends HttpServlet{
    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse
response)
        throws ServletException, IOException {
        PrintWriter out = response.getWriter();
        response.setContentType("text/html");
        String title = "Table";
        out.println("<html>");
        out.println("<head>");
        out.println("<title>" + title + "</title>");
        out.println("</head>");
        out.println("<body>");
        out.println("<center>");
        out.println("<table border=1 cellpadding=0 cellspacing=0 width=70%>");
        for(int i=1;i<=20;i++) {
            out.println("<tr>");
            for(int j=1;j<=3;j++) {
                out.print("<td> row "+i+" col "+j+"</td>");
            }

            out.println(" </tr>");
        }
        out.println("</table>");
    }
}
```

```

        out.println("</center>");
        out.println("</body>");
        out.println("</html>");
    }
}

```

**OUTPUT:**


row 1 col 1	row 1 col 2	row 1 col 3
row 2 col 1	row 2 col 2	row 2 col 3
row 3 col 1	row 3 col 2	row 3 col 3
row 4 col 1	row 4 col 2	row 4 col 3
row 5 col 1	row 5 col 2	row 5 col 3
row 6 col 1	row 6 col 2	row 6 col 3
row 7 col 1	row 7 col 2	row 7 col 3
row 8 col 1	row 8 col 2	row 8 col 3
row 9 col 1	row 9 col 2	row 9 col 3
row 10 col 1	row 10 col 2	row 10 col 3
row 11 col 1	row 11 col 2	row 11 col 3
row 12 col 1	row 12 col 2	row 12 col 3
row 13 col 1	row 13 col 2	row 13 col 3
row 14 col 1	row 14 col 2	row 14 col 3
row 15 col 1	row 15 col 2	row 15 col 3
row 16 col 1	row 16 col 2	row 16 col 3
row 17 col 1	row 17 col 2	row 17 col 3
row 18 col 1	row 18 col 2	row 18 col 3
row 19 col 1	row 19 col 2	row 19 col 3
row 20 col 1	row 20 col 2	row 20 col 3

**RESULT:**

Thus the program is successfully executed and output is verified.

**PROGRAM 3:****AIM:**

To create a servlet program to make a registration form that collects a Name, Register Number, and email address. Send the data to the servlet that displays it.

**PROGRAM:**

index.html

<html>

```
<head>
<title>AuthServlet</title>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
</head>
<body>
<h3>Authorization</h3>
<form action = "AuthServlet" method = "post">
    Enter Name : <input type = "text" name = "name" required="">
    Enter Register No : <input type = "text" name = "regno" required="">
    Enter Email ID : <input type = "email" name = "email" required="">
<center> <input type = "submit" value = "Submit"> </center>
</form>
</body>
</html>
```

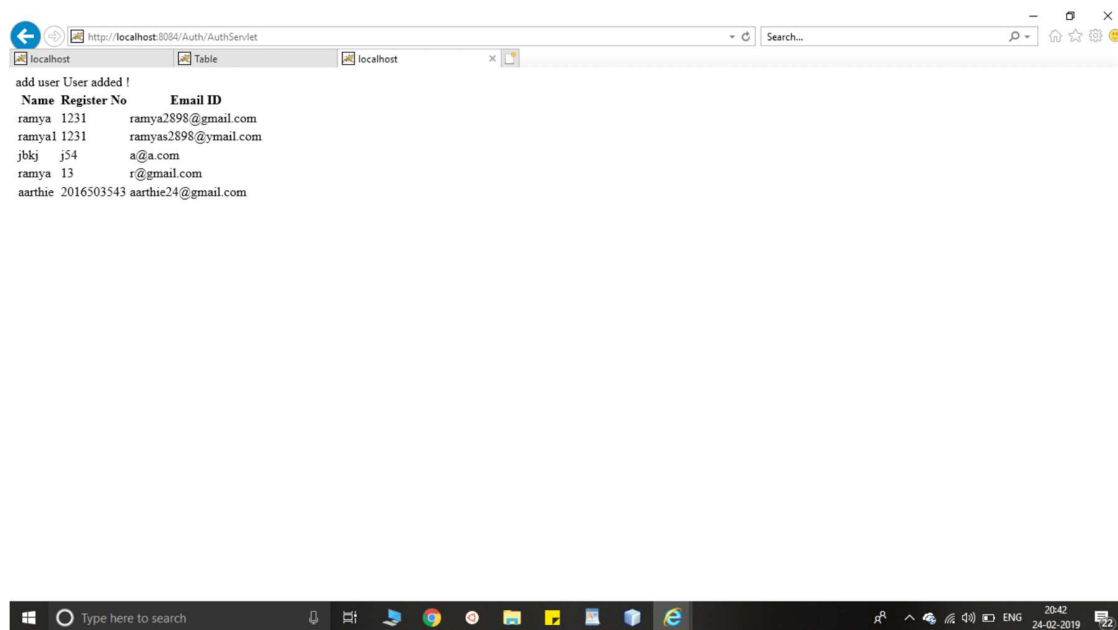
SERVLET :

```
public class AuthServlet extends HttpServlet{
    @Override
    protected void doPost(HttpServletRequest request, HttpServletResponse
response)
        throws ServletException, IOException {
    PrintWriter out = response.getWriter();
    LoginDAO dao = new LoginDAO();
    try {
    dao.addUser(request.getParameter("name"),request.getParameter("regno"),request.
getParameter("email"),response);
    } catch (ClassNotFoundException ex) {
        Logger.getLogger(AuthServlet.class.getName()).log(Level.SEVERE, null, ex);
    } catch (SQLException ex) {
        Logger.getLogger(AuthServlet.class.getName()).log(Level.SEVERE, null, ex);
    }
```

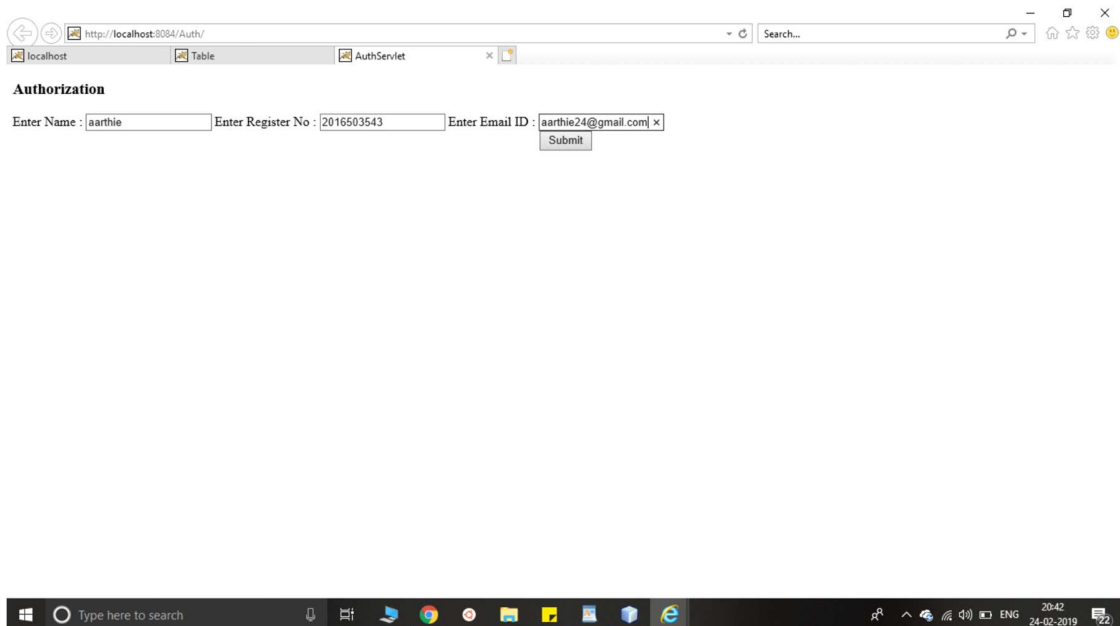
```
    }  
  }  
}  
LOGINDAO.java  
public class LoginDAO {  
    String url = "jdbc:mysql://localhost:3306/ramya";  
    String username = "root";  
    String password = "ramya2898";  
    public void addUser(String name, String regno,String email,HttpServletResponse  
response) throws ClassNotFoundException, SQLException, IOException {  
        PrintWriter out = response.getWriter();  
        out.println("add user");  
        String sql = "insert into student1 values(?,?,?)";  
        response.setContentType("text/html");  
        Class.forName("com.mysql.jdbc.Driver");  
        Connection con = DriverManager.getConnection(url,username,password);  
        PreparedStatement st = con.prepareStatement(sql);  
        st.setString(1,name);  
        st.setString(2,regno);  
        st.setString(3,email);  
        st.executeUpdate();  
        String sql1 = "select * from student1";  
        PreparedStatement st1 = con.prepareStatement(sql1);  
        ResultSet rs1 = st1.executeQuery();  
        out.print("<html>");  
        out.print("<table>");  
        out.print("<tr> <th> Name </th> <th> Register No </th> <th> Email ID  
</th> </tr>");  
        while(rs1.next()) {  
            out.print("<tr>");
```

2016503543

```
        out.print("<td>");
        out.print(rs1.getString(1)+ "</td><td>" + rs1.getString(2) +"</td><td>"
+ rs1.getString(3));
        out.print("</td>");
        out.print("</tr>");
    }
    out.print("</table>");
    out.print("</html>");
}
}
```

**OUTPUT:**

2016503543

**RESULT :**

Thus the program is successfully executed and output is obtained.

**PROGRAM 4:****AIM:**

To use session tracking to the servlet that says "Welcome Guest" to first-time visitors (with browsing session) and "Welcome back" to repeat visitors

**PROGRAM:****INDEX.HTML**

```
<html>
<head>
<title>Session</title>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
</head>
<body>
```

```
<h1>Session</h1>
<form action="Welcome" method="post">
    Your name : <input type="text" name="name">
    <input type="submit" value="Session" name="clicked">
</form>
</body>
</html>
```

SERVLET :

```
public class Welcome extends HttpServlet {
    ArrayList<String> names = new ArrayList<String>();
    protected void processRequest(HttpServletRequest request, HttpServletResponse
response)
        throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        try (PrintWriter out = response.getWriter()) {
            String button1 = request.getParameter("Session");
            String button2 = request.getParameter("Cookie");
            out.println("<!DOCTYPE html>");
            out.println("<html>");
            out.println("<head>");
            out.println("<title>Servlet Welcome</title>");
            out.println("</head>");
            out.println("<body>");
            String name = request.getParameter("name");
            if((request.getParameter("clicked")).equals("Session")) {
                HttpSession session = request.getSession();
                session.setAttribute("name",name);
                out.println("<a href = '/SessionManagement/ViewRiddle'>Click me for a
riddle</a><br>");
            }
        }
    }
}
```

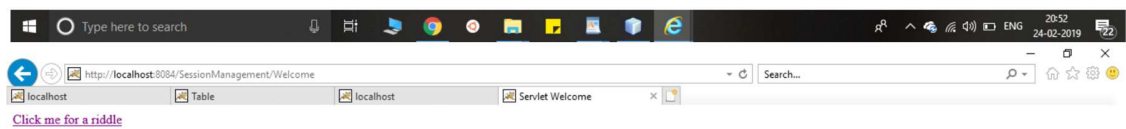
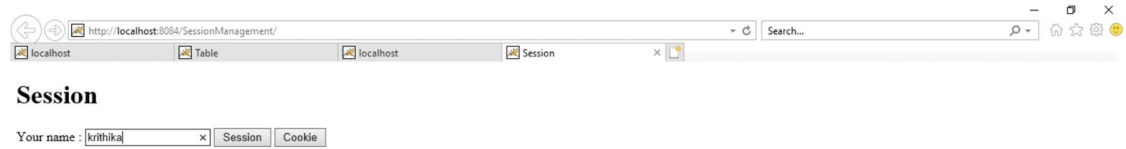
2016503543

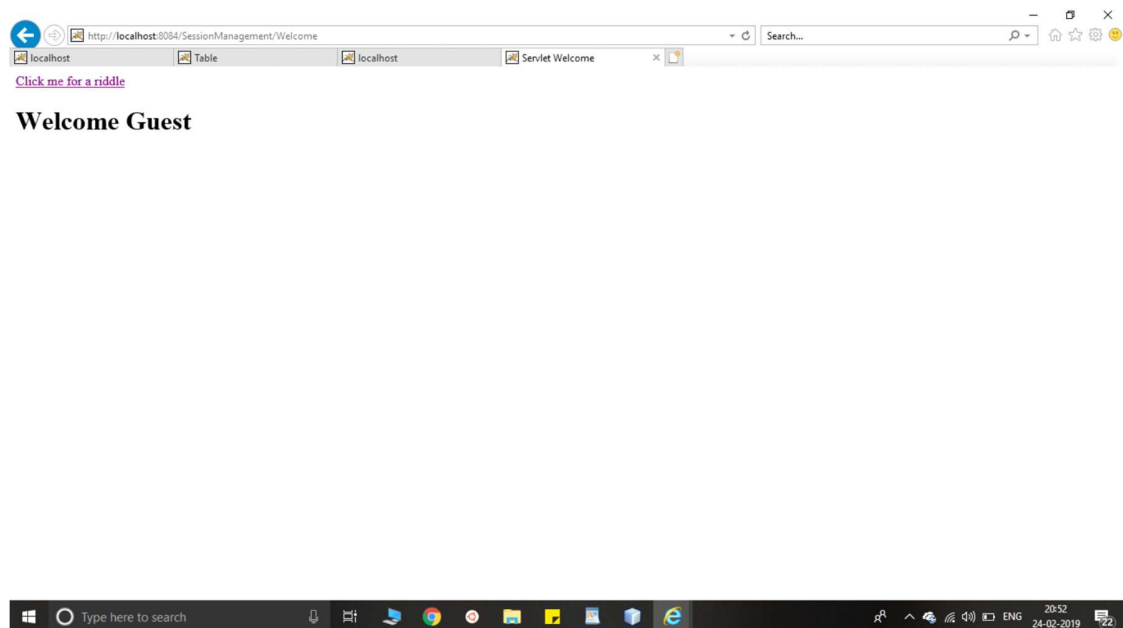
```
        else
        {
            Cookie cookie = new Cookie("name",name);
            cookie.setMaxAge(60*30);
            response.addCookie(cookie);
            out.println("<a href = '/SessionManagement/ViewQuote'>Click me for a
quote</a><br>");
        }
        if(names.contains(name)) {
            out.println("<h1>Welcome Back</h1>");
        }
        else
        {
            names.add(name);
            out.println("<h1>Welcome Guest</h1>");
        }
        out.println("</body>");
        out.println("</html>");
    }
}

@Override
protected void doGet(HttpServletRequest request, HttpServletResponse
response)
    throws ServletException, IOException {
    processRequest(request, response);
}
}
```

**OUTPUT:**





**RESULT :**

Thus the program is successfully executed and output is obtained.

**PROGRAM 5****AIM:**

To write a servlet that displays the values of Name, Register Number, and email-address request parameters. If a parameter is missing and the client is a first-time visitor, have the servlet list "Unknown for the missing values. If a parameter is missing and the client is a repeat visitor, have the servlet use previously entered values for the missing values.

**PROGRAM:**

index.html

```
<html>
```

```
<head>
```

```
<title>AuthServlet</title>
</head>
<body>
<h3>Authorization</h3>

<form action = "Cookie1" method = "post">
    Enter Name : <input type = "text" name = "name">
    Enter Register No : <input type = "text" name = "regno">
    Enter Email ID : <input type = "text" name = "email">

<center> <input type = "submit" value = "Submit"> </center>
</form>
</body>
</html>
```

SERVLET:

```
public class Cookie1 extends HttpServlet {
    protected void processRequest(HttpServletRequest request, HttpServletResponse
response)
        throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        try (PrintWriter out = response.getWriter()) {
            out.println("<!DOCTYPE html>");
            out.println("<html>");
            out.println("<head>");
            out.println("<title>Servlet Cookie</title>");
            out.println("</head>");
            out.println("<body>");
            String name = request.getParameter("name");
            String regno = request.getParameter("regno");
            String email = request.getParameter("email");
```

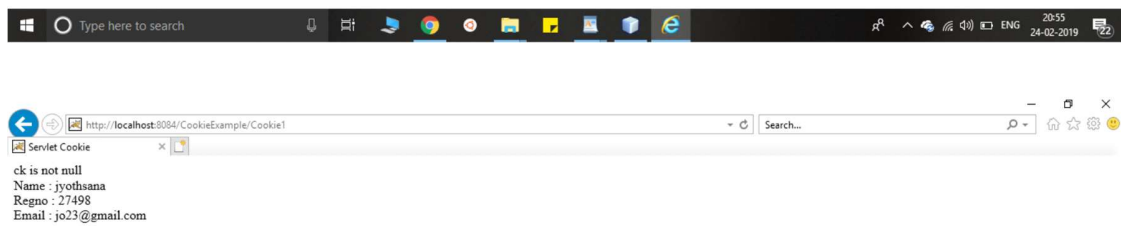
```
Cookie ck[] = request.getCookies();
if(ck != null) {
    out.println("ck is not null");
    if(name == "") {
        out.print("<br> Name : "+ ck[0].getValue());
    }
    else
    {
        out.print("<br> Name : "+ name);
    }
    if(regno == "") {
        out.print("<br> Regno : "+ ck[1].getValue());
    }
    else
    {
        out.print("<br> Regno : "+ regno);
    }
    if(email == "") {
        out.print("<br> Email : "+ ck[2].getValue());
    }
    else
    {
        out.print("<br> Email : "+ email);
    }
}
else {
    out.println("ck is null");
    Cookie cookie = new Cookie("name",name);
    Cookie cookie1 = new Cookie("regno",regno);
    Cookie cookie2 = new Cookie("email",email);
```

2016503543

```
response.addCookie(cookie);
response.addCookie(cookie1);
response.addCookie(cookie2);
if(name == "") {
    out.print("<br> Name : "+ "Unknown");
}
else
{
    out.print("<br> Name : "+ name);
}
if(regno == "") {
    out.print("<br> Regno : "+ "Unknown");
}
else
{
    out.print("<br> Regno : "+ regno);
}
if(email == "") {
    out.print("<br> Email : "+ "Unknown");
}
else
{
    out.print("<br> Email : "+ email);
}

}
out.println("</body>");
out.println("</html>");
}
```

**OUTPUT:**

**RESULT:**

Thus, the program is executed and output is obtained.

**PROGRAM 6****AIM:**

To write a servlet program that shows all the request headers. Use a red background and a yellow foreground for Google Chrome users; use a yellow background and a red foreground for Firefox and other users.

**PROGRAM:**

```
<html>
<body>
<div><a href = "/RequestHeaders/RequestHeader">Get the request
headers!</a></div>
</body>
</html>
```

**SERVLET:**

```
public class RequestHeader extends HttpServlet {
    protected void processRequest(HttpServletRequest request, HttpServletResponse
response)
        throws ServletException, IOException {
    response.setContentType("text/html;charset=UTF-8");
    try (PrintWriter out = response.getWriter()) {
        out.println("<!DOCTYPE html>");
        out.println("<html>");
        out.println("<head>");
        out.println("<title>Servlet RequestHeader</title>");
        out.println("</head>");
        final String clientBrowser = getClientBrowser(request);
        if(clientBrowser.equals("IE"))
            out.println("<body bgcolor= 'yellow' text='red'>");
        else
            out.println("<body bgcolor = 'red' text='yellow'>");
        out.println("<h1>Servlet RequestHeader at " + request.getContextPath() +
```

```

"</h1>");
    Enumeration<String> headerNames = request.getHeaderNames();
    while (headerNames.hasMoreElements()) {
        String headerName = headerNames.nextElement();
        out.print("Header Name: <em>" + headerName);
        String headerValue = request.getHeader(headerName);
        out.print("</em>, Header Value: <em>" + headerValue);
        out.println("</em><br/>");
    }
    out.println("<h3> The client browser is : " + clientBrowser + "</h3>");
    out.println("</body>");
    out.println("</html>");
}
}

public String getClientBrowser(HttpServletRequest request) {
    final String browserDetails = request.getHeader("User-Agent");
    final String user = browserDetails.toLowerCase();
    String browser = "";
    if (user.contains("msie")) {
        String substring =
browserDetails.substring(browserDetails.indexOf("MSIE")).split(";")[0];
        browser = substring.split(" ")[0].replace("MSIE", "IE") + "-" +
substring.split(" ")[1];
    } else if (user.contains("safari") && user.contains("version")) {
        browser =
(browserDetails.substring(browserDetails.indexOf("Safari")).split(" ")[0]).split(
"/")[0] + "-" + (browserDetails.substring(
        browserDetails.indexOf("Version")).split(" ")[0]).split("/")[1];
    } else if (user.contains("opr") || user.contains("opera")) {
        if (user.contains("opera"))

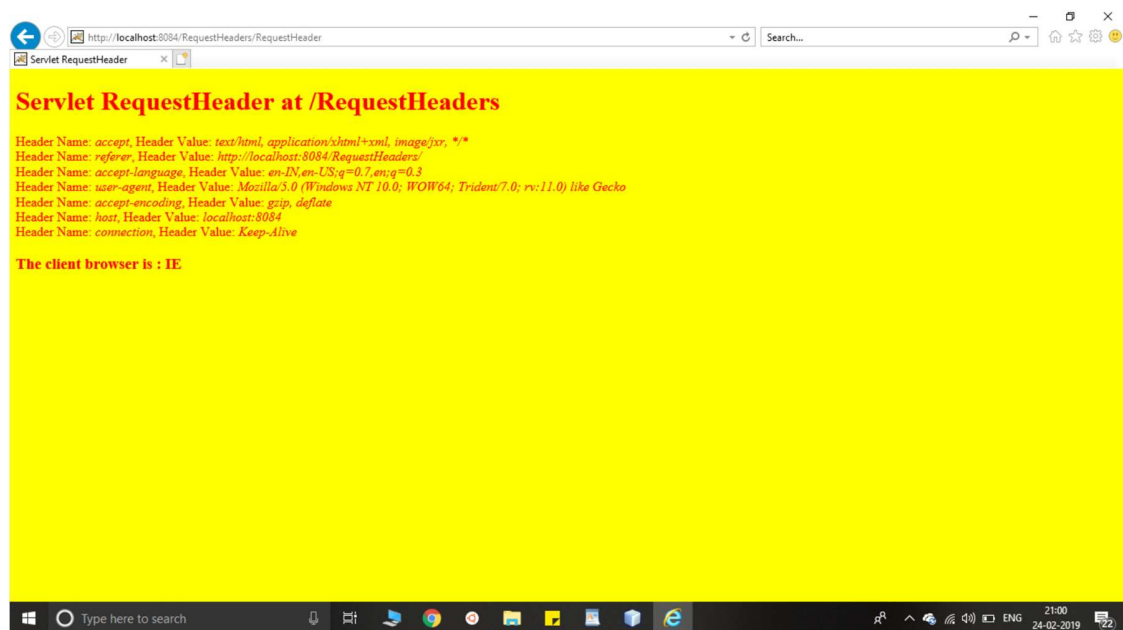
```



```

        browser =
(browserDetails.substring(browserDetails.indexOf("Opera")).split(" ")[0]).split(
"/")[0] + "-" + (browserDetails.substring(
        browserDetails.indexOf("Version")).split(" ")[0]).split("/")[1];
    else if (user.contains("opr"))
        browser =
((browserDetails.substring(browserDetails.indexOf("OPR")).split(" ")[0]).replace("/",
"-")).replace(
        return browser;
    }

```

**OUTPUT:****RESULT:**

Thus, the program is executed and output is obtained.

**PROGRAM 7****AIM:**

To write a servlet that returns a Bad Request error page(400) unless the user supplies email-id without @ symbol in the form.

**PROGRAM:**

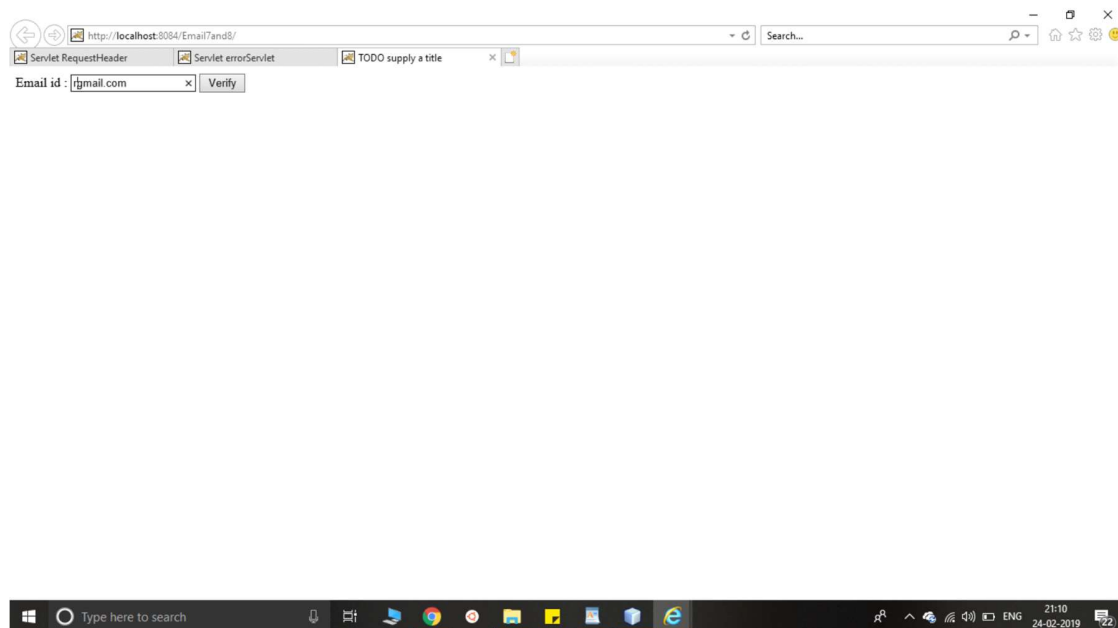
```
<html>
<head> </head>
<body>
<form action="errorServlet" method="post">
    Email id : <input type="text" name="email" required="">
<input type="submit" value="Verify" name="clicked">
</form>
</body>
</html>
```

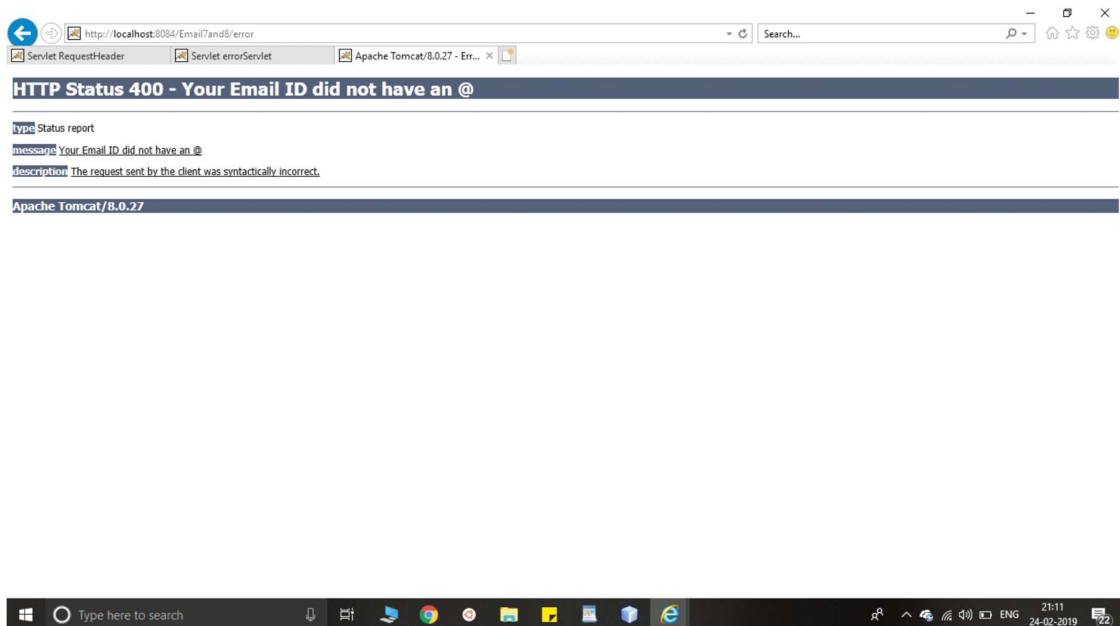
**SERVLET:**

```
public class error extends HttpServlet {
    protected void processRequest(HttpServletRequest request, HttpServletResponse
response)
        throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        try (PrintWriter out = response.getWriter()) {
            if(request.getParameter("email").contains("@")) {
                out.println("Email Verified");
            }
            else
            {
                response.sendError(400,"Your Email ID did not have an @");
            }
        }
    }
    @Override
```

2016503543

```
protected void doGet(HttpServletRequest request, HttpServletResponse
response)
    throws ServletException, IOException {
    processRequest(request, response);
}
@Override
protected void doPost(HttpServletRequest request, HttpServletResponse
response)
    throws ServletException, IOException {
    processRequest(request, response);
}
}
```

**OUTPUT:**

**RESULT:**

Thus, the program is executed and output is obtained.

**PROGRAM 8****AIM:**

To use session tracking to the servlet that says "Welcome Guest" to first-time visitors (with browsing session) and "Welcome back" to repeat visitors

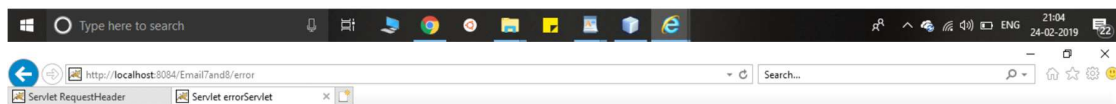
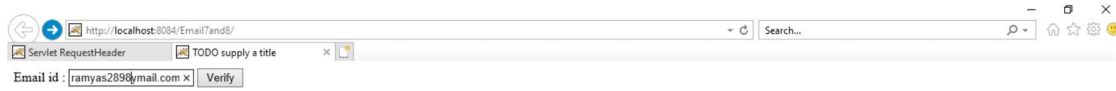
**PROGRAM:****INDEX.HTML**

```
<html>
<head>
</head>
<body>
<form action="error" method="post">
    Email id : <input type="text" name="email" required="">
    <input type="submit" value="Verify" name="clicked">
</form>
```

```
</body>
</html>
SERVLET:
public class error extends HttpServlet {
    protected void processRequest(HttpServletRequest request, HttpServletResponse
response)
        throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        try (PrintWriter out = response.getWriter()) {
            if(request.getParameter("email").contains("@")) {
                out.println("Email Verified");
            }
            else
            {
                RequestDispatcher rd=request.getRequestDispatcher("errorServlet");
                rd.forward(request, response);
            }
        }
    }
    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse
response)
        throws ServletException, IOException {
        processRequest(request, response);
    }
    @Override
    protected void doPost(HttpServletRequest request, HttpServletResponse
response)
        throws ServletException, IOException {
        processRequest(request, response);    }
```

SERVLET2:

```
public class errorServlet extends HttpServlet {
    protected void processRequest(HttpServletRequest request, HttpServletResponse
response)
        throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        try (PrintWriter out = response.getWriter()) {
            out.println("<!DOCTYPE html>");
            out.println("<html>");
            out.println("<head>");
            out.println("<title>Servlet errorServlet</title>");
            out.println("</head>");
            out.println("<body>");
            out.println("<h1>Servlet errorServlet </h1>");
            out.println("</body>");
            out.println("</html>");
        }
    }
    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse
response)
        throws ServletException, IOException {
        processRequest(request, response);
    }
    @Override
    protected void doPost(HttpServletRequest request, HttpServletResponse
response)
        throws ServletException, IOException {
        processRequest(request, response);
    }
}
```

**OUTPUT:****Servlet errorServlet****RESULT:**

Thus, the program is executed and output is obtained.

# AJAX, JSON, JQUERY



**Date - 15-02-19**

**Experiment-7**

### **PROGRAM 1**

#### **AIM:**

To create a DTD for a catalogue of cars, here each car has the child elements make, model, year, color, engine, number of doors, transmission type, and accessories. The engine element has the child elements number of cylinders and fuel system (carbureted or fuel injected). The accessories element has the attributes radio, air conditioning, power windows, power steering, and power brakes, each of which is required and has the possible values yes and no. Entities must be declared for the names of popular car models.

#### **PROGRAM:**

```
<!ELEMENT catalog (car)>
<!ELEMENT car
(make,model,year,color,engine,number_of_doors,transmission_type,accessories)>
<!ELEMENT make (#PCDATA)>
<!ELEMENT model (#PCDATA)>
<!ELEMENT year (#PCDATA)>
<!ELEMENT color (#PCDATA)>
<!ELEMENT engine (number_of_cylinders,fuel_system)>
<!ELEMENT number_of_cylinders (#PCDATA)>
<!ELEMENT fuel_system (#PCDATA)>
<!ELEMENT number_of_doors (#PCDATA)>
<!ELEMENT transmission_type (#PCDATA)>
<!ELEMENT accessories (#PCDATA)>
<!ATTLIST accessories radio (yes|no)>
<!ATTLIST accessories air_conditioning (yes|no)>
```

```
<!ATTLIST accessories power_windows (yes|no)>
```

```
<!ATTLIST accessories power_steering (yes|no)>
```

```
<!ATTLIST accessories power_brakes (yes|no)>
```

**RESULT:**

Thus, the program is executed and output is obtained.

**PROGRAM 2****AIM:**

To create an XML document with atleast three instances of the car element defined in the DTD of Program 1. Process the document by using the DTD of Program 1, and produce a display of raw XML document.

**PROGRAM:**

XML

```
<?xml version="1.0"?>
```

```
<!DOCTYPE catalog SYSTEM "prog1.dtd">
```

```
<?xml-stylesheet type="text/xsl" href="prog2.xsl"?>
```

```
<catalog>
```

```
<car>
```

```
<year>1987</year>
```

```
<make>JAPAN</make>
```

```
<model>NISSAN</model>
```

```
<color>WHITE</color>
```

```
<engine>
```

```
<number_of_cylinders>6</number_of_cylinders>
```

```
<fuel_system>fuel_injected</fuel_system>
```

```
</engine>
```

```
<transmission_type>auto</transmission_type>
```

```
<number_of_doors>4</number_of_doors>
```

```
<accessories radio = "yes" air_conditioning = "no" power_windows = "yes"
```

2016503543

```
power_steering = "no" power_brakes = "yes"> </accessories>  
</car>
```

```
<car>  
<make>CHINA</make>  
<model>HONDA</model>  
<year>1991</year>  
<color>Black</color>  
<engine>  
<number_of_cylinders>8</number_of_cylinders>  
<fuel_system>carbureted</fuel_system>  
</engine>  
<number_of_doors>5</number_of_doors>  
<transmission_type>manual</transmission_type>  
<accessories radio = "yes" air_conditioning = "no" power_windows = "yes"  
power_steering = "no" power_brakes = "yes"> </accessories>  
</car>
```

```
<car>  
<make>INDIA</make>  
<model>MARUTI SUZUKI</model>  
<year>1911</year>  
<color>WHITE</color>  
<engine>  
<number_of_cylinders>4</number_of_cylinders>  
<fuel_system>carbureted</fuel_system>  
</engine>  
<number_of_doors>4</number_of_doors>  
<transmission_type>auto</transmission_type>  
<accessories radio = "yes" air_conditioning = "no" power_windows = "yes"
```

```

power_steering = "no" power_brakes = "yes"> </accessories>
</car>
</catalog>

```

## XSL

```

<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">
<html>
<body bgcolor="orange">
<h2 align="left" style="color:blue font-family:helvetica">Cars! </h2>
<table border="10" align="left">
<tr bgcolor="red">
<th style="text-align:left font-size:700%">Year</th>
<th style="text-align:left font-size:700%">Make</th>
    <th style="text-align:left font-size:700%">Model</th>
    <th style="text-align:left font-size:700%">Color</th>
    <th style="text-align:left font-size:700%">Engine Cylinder</th>
    <th style="text-align:left font-size:700%">Engine Fuel</th>
    <th style="text-align:left font-size:700%">Door</th>
    <th style="text-align:left font-size:700%">Transmission</th>
    <th style="text-align:left font-size:700%">Radio</th>
    <th style="text-align:left font-size:700%">AC</th>
    <th style="text-align:left font-size:700%">PWindow</th>
    <th style="text-align:left font-size:700%">PSteering</th>
    <th style="text-align:left font-size:700%">PBrakes</th>
</tr>
<xsl:for-each select="catalog/car">
<tr>
<td><xsl:value-of select="year"/> </td>

```

```

<td><xsl:value-of select="make"/></td>
    <td><xsl:value-of select="model"/></td>
    <td><xsl:value-of select="color"/></td>
    <td><xsl:value-of select="engine/number_of_cylinders"/></td>
    <td><xsl:value-of select="engine/fuel_system"/></td>
    <td><xsl:value-of select="number_of_doors"/></td>
    <td><xsl:value-of select="transmission_type"/></td>
    <td><xsl:value-of select="accessories/@radio"/></td>
    <td><xsl:value-of select="accessories/@air_conditioning"/></td>
    <td><xsl:value-of select="accessories/@power_windows"/></td>
    <td><xsl:value-of select="accessories/@power_steering"/></td>
    <td><xsl:value-of select="accessories/@power_brakes"/></td>
</tr>
</xsl:for-each>
</table>
</body>
</html>
</xsl:template>
</xsl:stylesheet>

```

**OUTPUT:**

Year	Make	Model	Color	Engine Cylinder	Engine Fuel	Door	Transmission	Radio	AC	PWindow	PSteering	PBrakes
1987	JAPAN	NISSAN	WHITE	6	fuel injected	4	auto	yes	no	yes	no	yes
1991	CHINA	HONDA	Black	8	carbureted	5	manual	yes	no	yes	no	yes
1911	INDIA	MARUTI SUZUKI	WHITE	4	carbureted	4	auto	yes	no	yes	no	yes

**RESULT:**

Thus, the program is executed and output is obtained.

**PROGRAM 3****AIM:**

To modify the example application of Program 1 to allow the user to select a make and model of used cars. The make must be in a menu. When a model is chosen, an AJAX request must be made to get a list of the years and colors of the chosen make and model that are available. Make up a server-resident script to produce the data from an example array or hash.

**PROGRAM:****HTML**

```
<html>
<head>
<title> Program 3 </title>
<style>
    #display{
        visibility: hidden;
    }
</style>
<script>
    function getData(){
        var xml_load1 = new XMLHttpRequest();
        xml_load1.onreadystatechange = function(){
            if(this.readyState == 4 && this.status == 200){
                getMakeElement(this);
            }
        };
        xml_load1.open("GET","index.xml",true);
        xml_load1.send();}
    function getMakeElement(xml){
        var xmlDoc = xml.responseXML;
```

```

var content = "";
var x = xmlDoc.getElementsByTagName("make");
for(i = 0;i < x.length;i++){
    var name = "button"+i.toString();
    var id1 = "value"+i.toString();
    content += "<br><input type = 'text' + "value='" +
x[i].childNodes[0].nodeValue + "' id=" + id1+ " /><button id="+name+" onclick =
displayUtil('"+x[i].childNodes[0].nodeValue+"')> Click </button><br>";
    document.getElementById("list").innerHTML = content;
}
function getContentData(data){
    var xml_load1 = new XMLHttpRequest();
    xml_load1.onreadystatechange = function(){
        if(this.readyState == 4 && this.status == 200){
            loadData(this,data);
        }
    };
    xml_load1.open("GET","index.xml",true);
    xml_load1.send();
}
function loadData(xml,data){
    var xmlDoc = xml.responseXML;
    var x = xmlDoc.getElementsByTagName("car");
    for(var i = 0;i<x.length;i++){
        if(x[i].getElementsByTagName("make")[0].childNodes[0].nodeValue ==
data){
            content+="

```

2016503543

```
    }  
    content+="/table>";  
    document.getElementById("display").innerHTML = content;  
    document.getElementById("display").style.visibility = "visible";  
  }  
</script>  
</head>  
</body>  
</html>
```

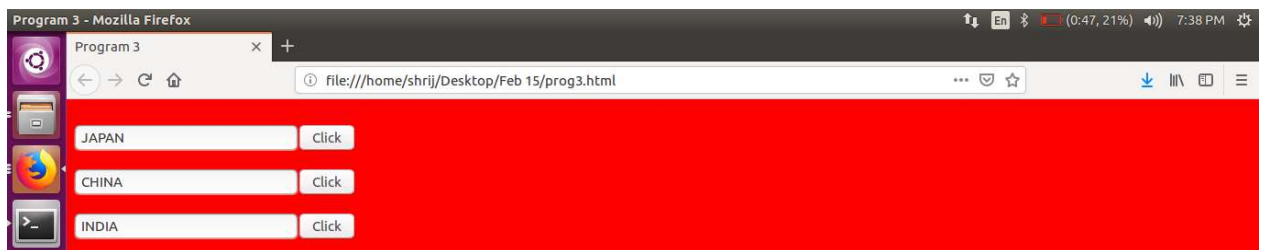
**XML**

```
<catalog>  
  <car>  
    <year>1987</year>  
    <make>JAPAN</make>  
    <model>NISSAN</model>  
    <color>WHITE</color>  
    <engine>  
      <number_of_cylinders>6</number_of_cylinders>  
      <number_of_cylinders>8</number_of_cylinders>  
      <fuel_system>carbureted</fuel_system>  
    </engine>  
    <number_of_doors>5</number_of_doors>  
    <transmission_type>manual</transmission_type>  
    <accessories radio = "yes" air_conditioning = "no" power_windows = "yes"  
    power_steering = "no" power_brakes = "yes"> </accessories>  
  </car>  
  
  <car>  
    <make>INDIA</make>
```



2016503543

```
<model>MARUTI SUZUKI</model>
<year>1911</year>
<color>WHITE</color>
<engine>
<number_of_cylinders>4</number_of_cylinders>
<fuel_system>carbured</fuel_system>
</engine>
<number_of_doors>4</number_of_doors>
<transmission_type>auto</transmission_type>
<accessories radio = "yes" air_conditioning = "no" power_windows = "yes"
power_steering = "no" power_brakes = "yes"> </accessories>
</car>
</catalog>
```

**OUTPUT:****RESULT:**

Thus, the program is executed and output is obtained.

**PROGRAM 4****AIM:**

To modify the example application of Program 1 to have it provide the addresses of repeat customers, using a hash of names and addresses.

**PROGRAM:****DTD**

```
<!ELEMENT catalog (car,customers)>
<!ELEMENT car
(make,model,year,color,engine,number_of_doors,transmission_type,accessories)>
<!ELEMENT make (#PCDATA)>
<!ELEMENT model (#PCDATA)>
<!ELEMENT year (#PCDATA)>
<!ELEMENT color (#PCDATA)>
<!ELEMENT engine (number_of_cylinders,fuel_system)>
<!ELEMENT number_of_cylinders (#PCDATA)>
<!ELEMENT fuel_system (#PCDATA)>
<!ELEMENT number_of_doors (#PCDATA)>
<!ELEMENT transmission_type (#PCDATA)>
<!ELEMENT accessories (#PCDATA)>
<!ATTLIST accessories radio (yes|no)>
<!ATTLIST accessories air_conditioning (yes|no)>
<!ATTLIST accessories power_windows (yes|no)>
<!ATTLIST accessories power_steering (yes|no)>
<!ATTLIST accessories power_brakes (yes|no)>
<!ELEMENT customers (id,name,address,count)>
<!ELEMENT id (#PCDATA)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT address (city,state,zipcode)>
<!ELEMENT city (#PCDATA)>
```

```
<!ELEMENT catalog (#PCDATA)>
<!ELEMENT catalog (#PCDATA)>
<!ELEMENT count (#PCDATA)>
```

## XML

```
<?xml version="1.0"?>
<!DOCTYPE catalog SYSTEM "prog4.dtd">
<catalog>
  <car>
    <year>1998</year>
    <make>Indian</make>
    <model>BMW</model>
    <color>Blue</color>
    <engine>
      <number_of_cylinders>6</number_of_cylinders>
      <fuel_system>fuel_injected</fuel_system>
    </engine>
    <transmission_type>auto</transmission_type>
    <number_of_doors>4</number_of_doors>
    <accessories radio = "yes" air_conditioning = "no" power_windows = "yes"
power_steering = "no" power_brakes = "yes"> </accessories>
  </car>
  <car>
    <make>English</make>
    <model>Benz</model>
    <year>1912</year>
    <color>Black</color>
    <engine>
      <number_of_cylinders>8</number_of_cylinders>
      <fuel_system>carbureted</fuel_system>
```

```
</engine>
<number_of_doors>2</number_of_doors>
<transmission_type>manual</transmission_type>
<accessories radio = "yes" air_conditioning = "no" power_windows = "yes"
power_steering = "no" power_brakes = "yes"> </accessories>
</car>
<number_of_doors>6</number_of_doors>
<transmission_type>auto</transmission_type>
<accessories radio = "yes" air_conditioning = "no" power_windows = "yes"
power_steering = "no" power_brakes = "yes"> </accessories>
</car>
<customer>
<id>1</id>
<name>Maya</name>
<address>
<city>Chennai</city>
<state>TN</state>
<zip>600073</zip>
</address>
</customer>
</catalog>
```

**HTML**

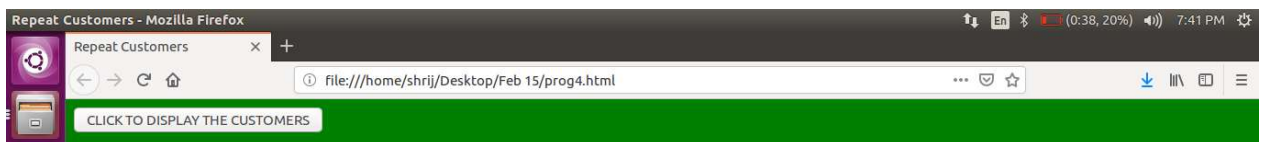
```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<title>Repeat Customers</title>
<meta name="viewport" content="width=device-width, initial-scale=1">
</head>
```

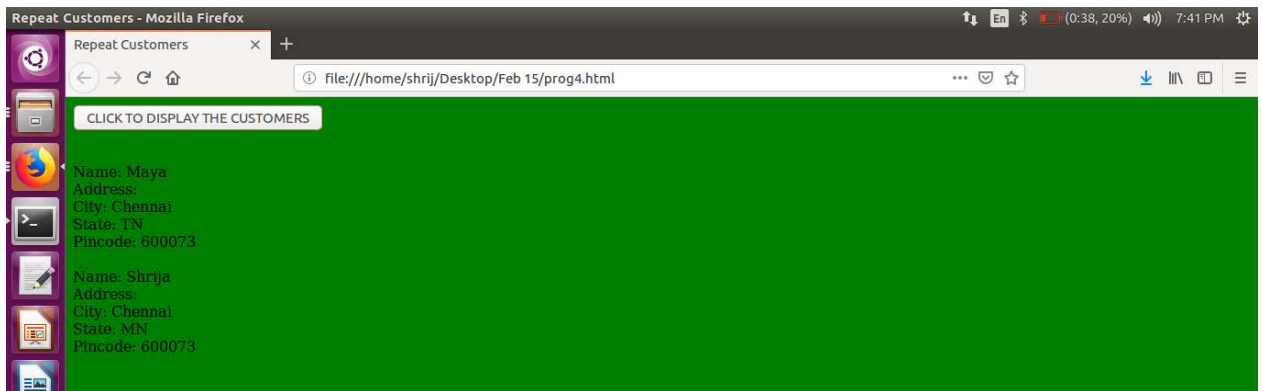
```
<body bgcolor="green">
<!-- ID: <input type="text" name="id" id="id"> <br>
      Name: <input type="text" name="name" id="name"> <br>
      City: <input type="text" name="city" id="city"> <br>
      State: <input type="text" name="state" id="state"> <br>
      Zipcode: <input type="text" name="zip" id="zip"> <br>
      <input type="button" value="Submit" onclick=inputData()> -->
      <input type="button" value="CLICK TO DISPLAY THE CUSTOMERS"
      onclick=dispData()>
<script>
    function inputData() {
        var id = document.getElementById("id");
        var name = document.getElementById("name");
        var city = document.getElementById("city");
        var state = document.getElementById("state");
        var zip = document.getElementById("zip");

        var xhttp = new XMLHttpRequest();
        xhttp.onreadystatechange = function() {
            if(this.readyState == 4 && this.status == 200) {
                alert(this.responseText);
                document.getElementById("demo").innerHTML = this.responseText;
            }
        };
        xhttp.open("GET","prog4.php?id=" + id,true);
        xhttp.send();
    }
    function dispData() {
        var xhttp = new XMLHttpRequest();
        xhttp.onreadystatechange = function() {
```

2016503543

```
        if(this.readyState == 4 && this.status == 200) {  
            dispXML(this);  
        }  
    };  
    xhttp.open("GET","prog4.xml",true);  
    xhttp.send();  
}  
  
function dispXML(xml) {  
    var xmlDoc = xml.responseXML;  
    var x = xmlDoc.getElementsByTagName("customer");  
    var content = "";  
    alert(x.length);  
}  
</script>  
</body>  
</html>
```

**OUTPUT:**

**RESULT:**

Thus, the program is executed and output is obtained.

**PROGRAM 5****AIM:**

To modify the example application of Program 4 with validate the zip code when it is entered, to ensure that it is a valid zip code for the given city and state. The response document can be a PHP script that looks up the zip code and the city and state in a small table of examples.

**PROGRAM:****HTML**

```
<html>
<head>
<title>Program 5</title>
<meta charset="utf-8">
<script>
    function myfunc(){
        var x = document.getElementById("city").value;
        var y = document.getElementById("state").value;
        var z = document.getElementById("zipcode").value;
```

```

        console.log(x);
        var req = x + "" + y + "" + z;
        var xml_load1 = new XMLHttpRequest();
        xml_load1.onreadystatechange = function(){
            if(this.readyState == 4 && this.status == 200){
                alert(this.responseText);
            }
        };
        xml_load1.open("GET","prog5.php?val="+req,true);
        xml_load1.send();
    }
</script>
</head>
<body>
<form>
    CITY:<input id="city" type="text" value="" /> <br> <br> <br> <br>
    STATE:<input id="state" type="text" value="" /> <br> <br> <br> <br>
    PIN NUMBER:<input id="zipcode" type="text" value=""
/> <br> <br> <br> <br>
    <input type="button" value="Submit" onclick=myfunc() />
</form>
</body>
</html>
PHP
<?php
$address[] = "ch tn 1";
$address[] = "salem tn 2";
$address[] = "pondy tn 3";

$val = $_REQUEST["val"];

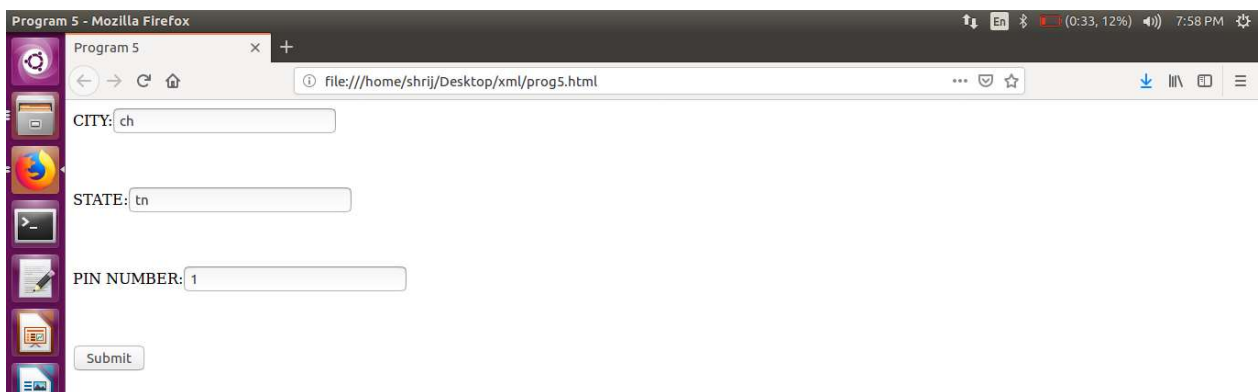
```



```
$fl = 0;
for($i=0;$i<sizeof($address);$i++){
    if($address[$i] == $val){
        $fl = 1;
        break;
    }
}
if($fl==0){
    echo 'INValid';
}
else{
    echo 'Valid';
}
```

?>

### OUTPUT:



### RESULT:

Thus, the program is executed and output is obtained.

# ***XML, DTD, XSLT***

**Date- 22-02-19**

**Experiment-8**

### **PROGRAM 1**

#### **AIM:**

To design an XML document that stores information about patients in a hospital. Information about patients must include their name (in three parts), Social Security Number, age, room number, primary insurance company – including member identification number, group number, phone number, and address – secondary insurance company (with same parts as the primary insurance company has), known medical problems, and known drug allergies. Both attributes and nested tags must be included.

Make up a sample data for at least four patients.

#### **PROGRAM:**

```
<hospital>
<patient>
<name firstname="A" middlename="B" lastname="C" />
<sex>Male</sex>
<room-number>1</room-number>
</primary-insurance-company>
<secondary-insurance-company>
<id>21</id>
<group-id>41</group-id>
<phone>222222222</phone>
<address>Madurai</address>
</secondary-insurance-company>
</patient>
<patient>
<name firstname="x" middlename="Y" lastname="Z" />
```

```
<sex>Male</sex>
<room-number>2</room-number>
<id>22</id>
<group-id>51</group-id>
<phone>2222222222</phone>
<address>Delhi</address>
</secondary-insurance-company>
</patient>
</hospital>
```

**RESULT:**

Thus, the program is executed and output is obtained.

**PROGRAM 2****AIM:**

To write a DTD for the document described in Program 1, but with the following restrictions: name, Social Security number, age, room number, and primary insurance company are required. All the other elements are optional, as are middle names.

**PROGRAM:**

```
<!ELEMENT hospital (patient)>
<!ELEMENT patient (name,sex,room-number,age,social-security-number,primary-
insurance-company,secondary-insurance-company)>
<!ELEMENT name (#PCDATA)>
<!ATTLIST firstname #REQUIRED>
<!ATTLIST middlename >
<!ATTLIST lastname #REQUIRED>
<!ELEMENT age(#PCDATA) #REQUIRED>
```

2016503543

```
<!ELEMENT sex(#PCDATA) #REQUIRED>
<!ELEMENT room-number (#PCDATA) #REQUIRED>
<!ELEMENT social-security-number(#PCDATA) #REQUIRED>
<!ELEMENT primary-insurance-company (id,group-id,phone,address) #REQUIRED>
<!ELEMENT id (#PCDATA)>
<!ELEMENT group-id (#PCDATA)>
<!ELEMENT phone (#PCDATA)>
<!ELEMENT address (#PCDATA)>
<!ELEMENT secondary-insurance-company (id,group-id,phone,address)>
```

**RESULT:**

Thus, the program is executed and output is obtained.

**PROGRAM 3****AIM:**

To create a CSS style sheet for the XML document of program 1 and use it to create a display of the document.

**PROGRAM:**

XML

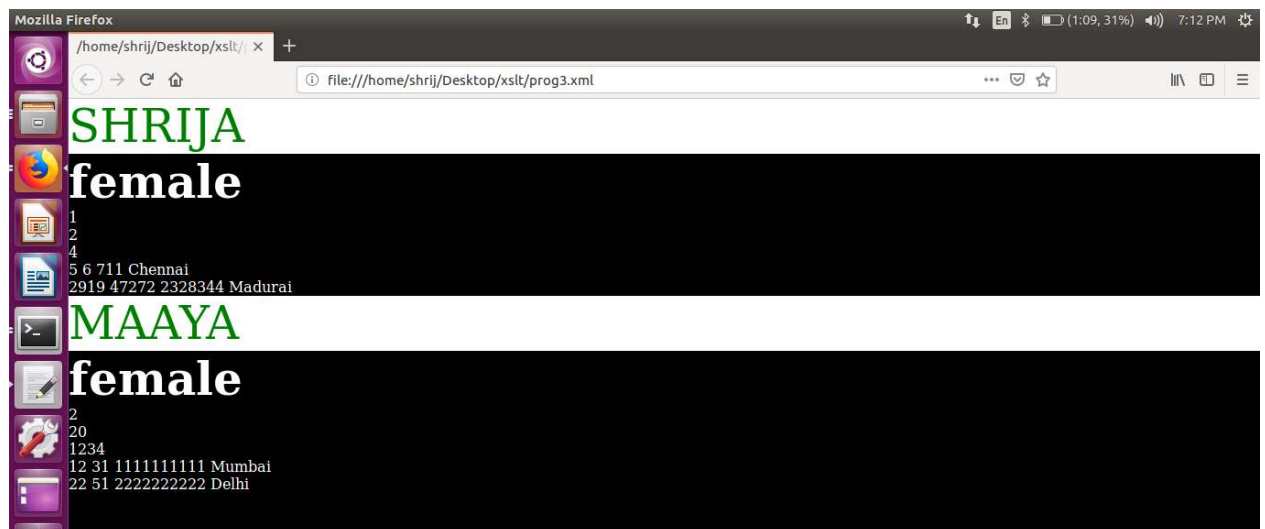
```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/css" href="prog3.css" ?>
<hospital>
<patient>
<name firstname="kiki" middlename="gigi" lastname="jiji">SHRIJA</name>
<sex>female</sex>
<room-number>1</room-number>
<age>2</age>
<social-security-number>4</social-security-number>
<primary-insurance-company>
<id>5</id>
```

2016503543

```
<group-id>6</group-id>
<phone>711</phone>
<address>Chennai</address>
</primary-insurance-company>
<secondary-insurance-company>
<id>2919</id>
<group-id>47272</group-id>
<phone>2328344</phone>
<address>Madurai</address>
</secondary-insurance-company>
</patient>
<patient>
<name firstname="shrija" middlename="ja" lastname="ja">MAAYA</name>
<sex>female</sex>
<room-number>2</room-number>
<age>20</age>
<social-security-number>1234</social-security-number>
<primary-insurance-company>
<id>12</id>
<group-id>31</group-id>
<phone>1111111111</phone>
<address>Mumbai</address>
</primary-insurance-company>
<secondary-insurance-company>
<id>22</id>
<group-id>51</group-id>
<phone>2222222222</phone>
<address>Delhi</address>
</secondary-insurance-company>
</patient>
```

```
</hospital>
CSS
hospital {
    color: white;
    background-color : black;
    width: 300%;
}
```

OUTPUT:



RESULT:

Thus, the program is executed and output is obtained.

#### PROGRAM 4

##### AIM:

To create an XSLT stylesheet for one patient element of the XML document of program 1 and use it to create a display of that element.

##### PROGRAM:

**XML**

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="prog4.xsl" ?>
<hospital>
<patient>
<name firstname="shrija" middlename="ja" lastname="ja">ABC</name>
<sex>female</sex>
<room-number>1</room-number>
<age>10</age>
<social-security-number>1</social-security-number>
<primary-insurance-company>
<id>2</id>
<group-id>3</group-id>
<phone>4</phone>
<address>Chennai</address>
</primary-insurance-company>
<secondary-insurance-company>
<id>5</id>
<group-id>81</group-id>
<phone>2171721727</phone>
<address>Madurai</address>
</secondary-insurance-company>
</patient>
<patient>
<name firstname="kiki" middlename="gigi" lastname="bebe">XYZ</name>
<sex>Male</sex>
<room-number>2</room-number>
<age>20</age>
<social-security-number>1226</social-security-number>
<primary-insurance-company>
```



```

<id>12</id>
<group-id>31</group-id>
<phone>1111111111</phone>
<address>Mumbai</address>
</primary-insurance-company>
<secondary-insurance-company>
<id>22</id>
<group-id>51</group-id>
<phone>2222222222</phone>
<address>Delhi</address>
</secondary-insurance-company>
</patient>
</hospital>

```

## XSL

```

<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">
<html>
<body bgcolor="yellow">
<h2 align="center" style="color:red font-family:helvetica">Hospital Patient
History</h2>
<table border="1" align="center">
<tr bgcolor="green">
<th style="text-align:left font-size:300%">FName</th>
<th style="text-align:left font-size:300%">MName</th>
<th style="text-align:left font-size:300%">LName</th>
<th style="text-align:left font-size:300%">Sex</th>
<th style="text-align:left font-size:300%">Room-number</th>
<th style="text-align:left font-size:300%">Age</th>

```

2016503543

```

<th style="text-align:left font-size:300%">Social Security Number</th>
<th style="text-align:left font-size:300%">SIC:ID</th>
<th style="text-align:left font-size:300%">SIC:GrpID</th>
<th style="text-align:left font-size:300%">SIC:Phone</th>
<th style="text-align:left font-size:300%">SIC:Address</th>
<th style="text-align:left font-size:300%">SIC:ID</th>
<th style="text-align:left font-size:300%">SIC:GrpID</th>
<th style="text-align:left font-size:300%">SIC:Phone</th>
<th style="text-align:left font-size:300%">SIC:Address</th>
</tr>
<xsl:for-each select="hospital/patient[name='ABC']">
<tr>
<td><xsl:value-of select="name/@firstname"/></td>
<td><xsl:value-of select="name/@middlename"/></td>
<td><xsl:value-of select="name/@lastname"/></td>
<td><xsl:value-of select="sex"/></td>
<td><xsl:value-of select="room-number"/></td>
<td><xsl:value-of select="age"/></td>
<td><xsl:value-of select="social-security-number"/></td>
<td><xsl:value-of select="primary-insurance-company/id"/></td>
<td><xsl:value-of select="primary-insurance-company/group-id"/></td>
<td><xsl:value-of select="primary-insurance-company/phone"/></td>
<td><xsl:value-of select="primary-insurance-company/address"/></td>
<td><xsl:value-of select="secondary-insurance-company/id"/></td>
<td><xsl:value-of select="secondary-insurance-company/group-
id"/></td>
<td><xsl:value-of select="secondary-insurance-company/phone"/></td>
<td><xsl:value-of select="secondary-insurance-
company/address"/></td>
</tr>

```

```

</xsl:for-each>
</table>
</body>
</html>
</xsl:template>
</xsl:stylesheet>

```

**OUTPUT:**

FName	MName	LName	Sex	Room-number	Age	Social Security Number	SIC:ID	SIC:GrpID	SIC:Phone	SIC:Address	SIC:ID	SIC:GrpID	SIC:Phone	SIC:Address
shrija	ja	ja	female	1	10	1	2	3	4	Chennai	5	81	2171721727	Madurai

**RESULT:**

Thus, the program is executed and output is obtained.

**PROGRAM 5****AIM:**

To modify the XSLT stylesheet of Program 4 so that it formats all the patient elements in the XML document of Program 1 and use the stylesheet to create a display of the whole document.

**PROGRAM:****XML**

```

<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="prog5.xsl" ?>
<hospital>
<patient>

```

2016503543

```
<name firstname="shrija" middlename="sathya"
lastname="narayanan">shrijasathayanarayanan</name>
<sex>female</sex>
<room-number>1</room-number>
<age>10</age>
<social-security-number>8181</social-security-number>
<primary-insurance-company>
<id>59</id>
<group-id>10</group-id>
<phone>10</phone>
<address>Chennai</address>
</primary-insurance-company>
<secondary-insurance-company>
<id>2</id>
<group-id>4</group-id>
<phone>2222</phone>
<address>Madurai</address>
</secondary-insurance-company>
</patient>
<patient>
<phone>11</phone>
<address>Mumbai</address>
</primary-insurance-company>
<secondary-insurance-company>
<id>22</id>
<group-id>51</group-id>
<phone>222</phone>
<address>chennai</address>
</secondary-insurance-company>
</patient>
```

```
</hospital>
```

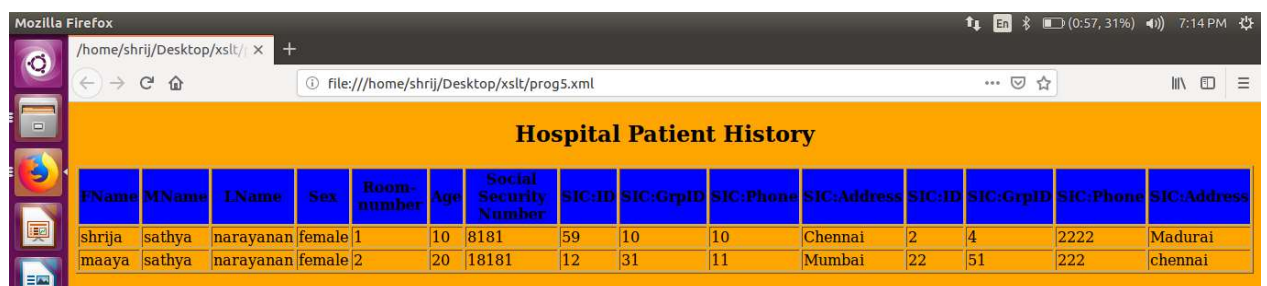
## XSL

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">
<html>
<body bgcolor="orange">
<h2 align="center" style="color:#ffffff font-family:helvetica">Hospital Patient
History</h2>
<table border="1" align="center">
<tr bgcolor="blue">
<th style="text-align:left font-size:600%">FName</th>
<th style="text-align:left font-size:600%">MName</th>
<th style="text-align:left font-size:600%">LName</th>
<th style="text-align:left font-size:600%">Sex</th>
    <th style="text-align:left font-size:600%">Room-number</th>
    <th style="text-align:left font-size:600%">Age</th>
    <th style="text-align:left font-size:600%">Social Security Number</th>
    <th style="text-align:left font-size:600%">SIC:ID</th>
    <th style="text-align:left font-size:600%">SIC:GrpID</th>
    <th style="text-align:left font-size:600%">SIC:Phone</th>
    <th style="text-align:left font-size:600%">SIC:Address</th>
    <th style="text-align:left font-size:600%">SIC:ID</th>
    <th style="text-align:left font-size:600%">SIC:GrpID</th>
    <th style="text-align:left font-size:600%">SIC:Phone</th>
    <th style="text-align:left font-size:600%">SIC:Address</th>
</tr>
<xsl:for-each select="hospital/patient">
<tr>
```

```

<td><xsl:value-of select="name/@firstname"/></td>
<td><xsl:value-of select="name/@middlename"/></td>
    <td><xsl:value-of select="name/@lastname"/></td>
    <td><xsl:value-of select="sex"/></td>
    <td><xsl:value-of select="room-number"/></td>
    <td><xsl:value-of select="primary-insurance-company/address"/></td>
    <td><xsl:value-of select="secondary-insurance-
company/groupid"/></td>
    <td><xsl:value-of select="secondary-insurance-company/phone"/></td>
    <td><xsl:value-of select="secondary-insurance
company/address"/></td>
</tr>
</xsl:for-each>
</table>
</body>
</html>
</xsl:template>
</xsl:stylesheet>

```

**OUTPUT:**


FName	MName	LName	Sex	Room-number	Age	Social Security Number	SIC-ID	SIC:GrpID	SIC:Phone	SIC:Address	SIC-ID	SIC:GrpID	SIC:Phone	SIC:Address
shrja	sathya	narayanan	female	1	10	8181	59	10	10	Chennai	2	4	2222	Madurai
maaya	sathya	narayanan	female	2	20	18181	12	31	11	Mumbai	22	51	222	chennai

**RESULT:**

Thus, the program is executed and output is obtained.