Laboratory Manual

Advance Java
(IT-505)

For

Third Year Students
Department: Information Technology

Department of Information Technology Engineering

Vision of IT Department

The Department of Information Technology envisions preparing technically competent problem solvers, researchers, innovators, entrepreneurs, and skilled IT professionals for the development of rural and backward areas of the country for the modern computing challenges.

Mission of the CSE Department:

- To offer valuable education through an effective pedagogical teaching-learning process.
- To shape technologically strong students for industry, research & higher studies.
- To stimulate the young brain entrenched with ethical values and professional behaviors for the progress of society.

Program Educational Objectives

Graduates will be able to

- Our graduates will show management skills and teamwork to attain employers' objectives in their careers.
- Our graduates will explore the opportunities to succeed in research and/or higher studies.
- Our graduates will apply technical knowledge of Information Technology for innovation and entrepreneurship.
- Our graduates will evolve ethical and professional practices for the betterment of society.

Program Outcomes (POs)

Engineering Graduates will be able to:

1. **Engineering knowledge**: Apply the knowledge of mathematics, science, engineering Fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis**: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. **Design/development of solutions**: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, andenvironmental considerations.
4. **Conduct investigations of complex problems**: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data,and synthesis of the information to provide valid conclusions.
5. **Modern tool usage**: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
6. **The engineer and society**: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and sustainability**: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. **Ethics**: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work**: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication**: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, andgive and receive clear instructions.
11. **Project management and finance**: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. **Life-long learning**: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change

Course Outcomes

Adv. Java (IT-505)

CO1	Ability to access database through Java programs, using Java Data Base Connectivity (JDBC).
CO2	Design the dynamic web pages, using Servlets and JSP.
CO3	Access the remote methods in an application using Remote Method Invocation (RMI)
CO4	Demonstrate the multi-tier architecture of web-based enterprise applications using Enterprise Java Beans (EJB).
CO5	Develop Stateful, Stateless and Entity Beans and use Struts frameworks, which gives the opportunity to reuse the codes for quick development.

Course	Course Outcomes	CO Attainment	PO1	PO2	PO3	PO4	PO5	PO 6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
CO1	Understand basic concepts and identify various data models (E-R modelling concepts) and apply these concepts for designing databases.		**2**	**1**	**1**	**1**	**1**	0	0	0	0	0	0	0	0	0	0
CO2	Apply relational database theory by SQL and describe relational algebra expression, tuple and domain relational expression for writing queries in relational algebra.		**1**	**1**	0	**1**	**1**	0	0	0	0	0	0	0	0	**2**	0
CO3	Understand and implement various Relational Database Management Systems through Oracle/SQL/PL SQL.		**2**	**2**	0	**1**	**1**	0	0	0	0	0	0	0	0	**1**	0
CO4	Identify and improve the database design by normalization.		**2**	**1**	**1**	**1**	0	0	0	0	**1**	**2**	0	0	0	0	0
CO5	Evaluate optimize queries and transaction processes for solving real world problems.		**1**	**2**	0	**2**	**1**	0	0	0	**1**	**1**	**1**	0	0	0	**1**

List of Program

1. PLAY TWO AUDIOS IN A SEQUENCE CONTINUOUSLY USING AUDIOCLIP INTERFACE

Aim:

To write a java applet program to play two sound notes simultaneously using the play() method in Audio Clip interface.

Algorithm:

Step 1:
Start the program.

Step 2:
Import java packages such as java.applet.*, java.awt.*and java.awt.event.*.

Step 3:
Define class with name ‘pgm3’ and extends it from the class ‘Applet’ and also implements theinterface ‘ActionListener’.

Step 4:
Define the init() method and create two button objects labeled as ‘play audio 1’,’paly audio 2’
respectively.

Step 5:
Add the buttons to panel and add Action listener for each button to handle Action Event.

Step 6:
Define action Performed () method for handling click events of buttons.

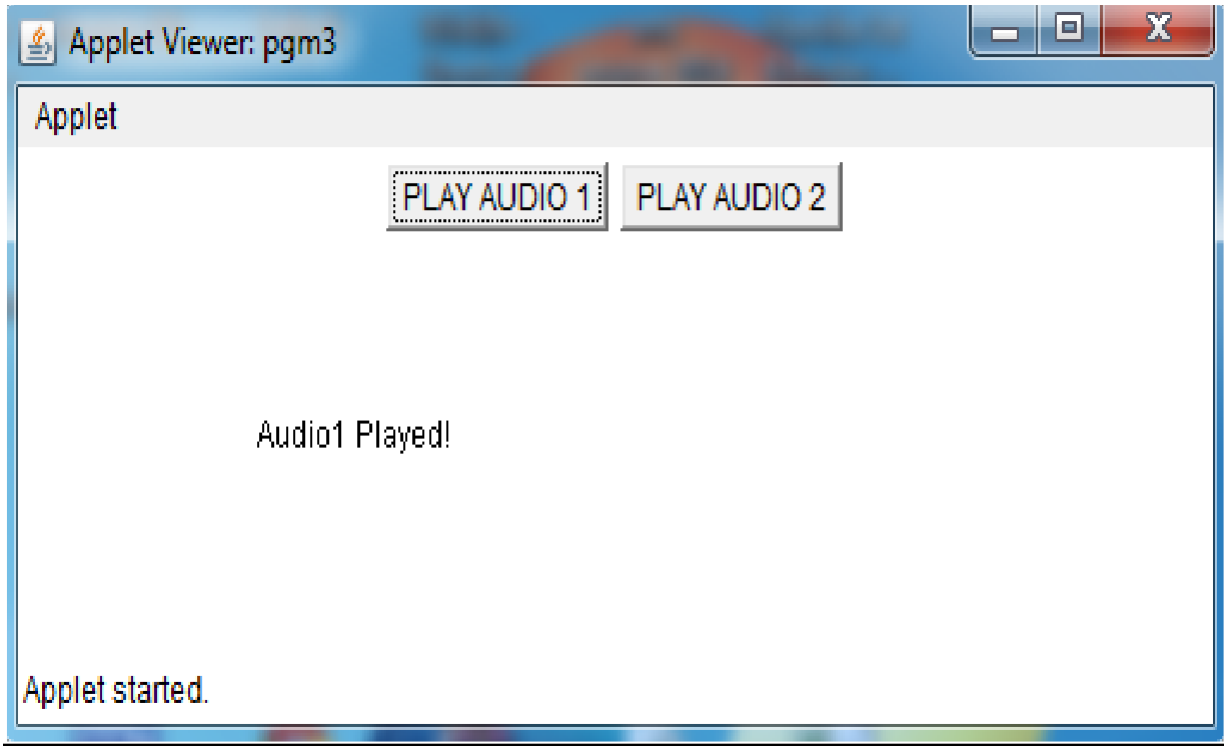
Step 7:
Stop the program.

SOURCE CODE:

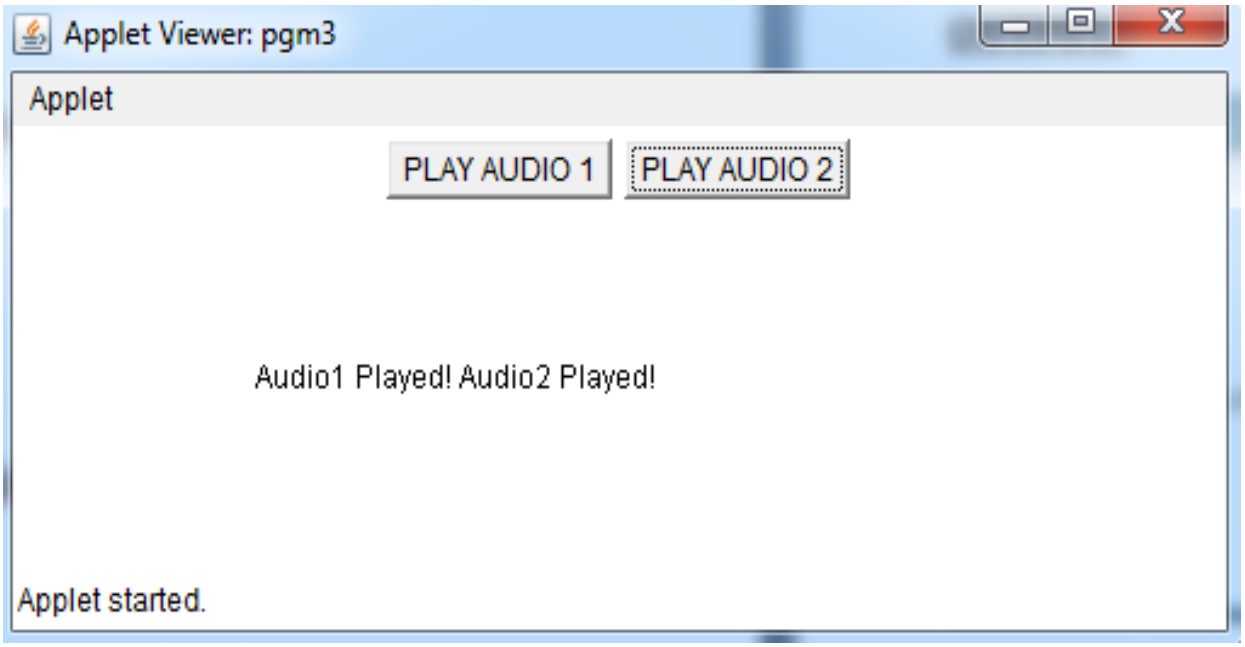
```
// Play Two Audios in a Sequence Continuously Using AudioClip Interfaceimport

java.applet.*;
import java.awt.*;
import java.awt.event.*;
/*<applet code="pgm3" height=500 width=500></applet>*/public class pgm3 extends
Applet implements ActionListener
{
Button b1,b2; AudioClip ac; String str=""; public void init()
{
b1=new Button("PLAY AUDIO 1");b2=new Button("PLAY AUDIO 2");add(b1);
add(b2); b1.addActionListener(this);b2.addActionListener(this);
}
public void actionPerformed(ActionEvent ae)
{
if(ae.getSource()==b1)
{
ac=getAudioClip(getCodeBase(),"x1.wav");str +="Audio1";
}
else if(ae.getSource()==b2)
{
ac=getAudioClip(getCodeBase(),"x2.wav");str +=" Audio2";
}
ac.play();
str +=" Played!";repaint();
}
public void paint(Graphics g)
{
g.drawString(str,100,100);
}
```

OUTPUT:

C:\Program Files \Java\jdk1.7.0\bin>javac pgm3.java C:\Program Files \Java\jdk1.7.0\bin>appletviewer pgm3.java



RESULT: The above program has been executed successfully and the output was verified.


Applet Viewer: pgm3
Applet
PLAY AUDIO 1
PLAY AUDIO 2
Audio1 Played! Audio2 Played!
Applet started.

2. CREATE SAMPLE APPLICATION FORM USING JAPPLET

Aim: To write a java program to create sample application form in JApplet using swing control.

Algorithm:

Step 1:
Define class pgm8 which extends from JApplet and implement the interface ActionListener.

Step 2:
Create object for JLabel, JTextfield, JButton, JCheckbox, JRadiobutton as needed.

Step 3:
Add all the components into the container.

Step 4:
Add Action listener to the buttons for handling events.

Step 5:
Define actionPerformed() method, write a source code for button "ok" and "cancel", Printgiven information while clicking "ok" button. If we click "cancel" button reset the form.

Step 6:
Stop the program.

SOURCE CODE:

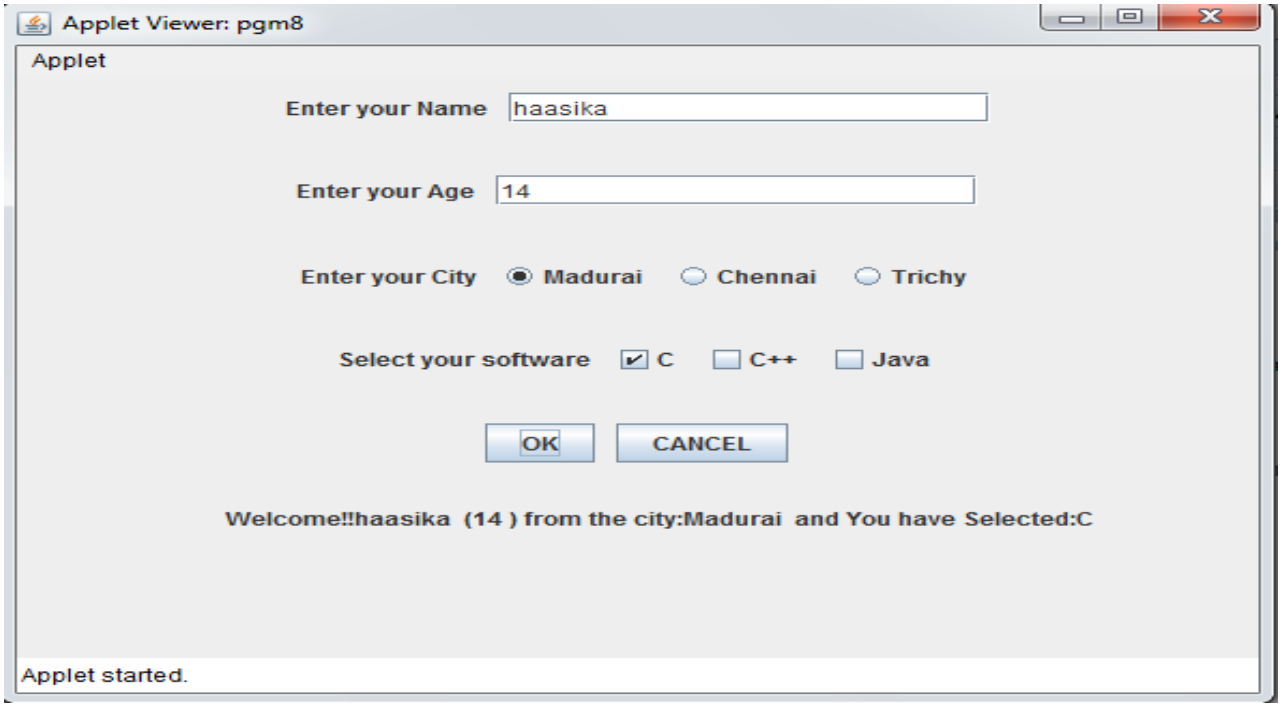
```
//Create Sample Application Form Using JAppletimport javax.swing.*;
import java.awt.*;
import java.awt.event.*;
/*<applet code="pgm8" height=500 width=700></applet>*/ public class pgm8 extends
JApplet implements ActionListener
{
JButton b1,b2; JTextField  t1,t2; JLabel l1,l2,l3,l4,msg; Container cp; JRadioButton
r1,r2,r3;
JCheckBox ch1,ch2,ch3; String str,x1,x2; ButtonGroup bg,bg1; JPanel
p1,p2,p3,p4,p5,p6;public void init()
{
cp=getContentPane(); cp.setLayout(new GridLayout(7,1));p1=new JPanel();
p1.setLayout(new FlowLayout(FlowLayout.CENTER,10,10));p2=new JPanel();
p2.setLayout(new FlowLayout(FlowLayout.CENTER,10,10));p3=new JPanel();
p3.setLayout(new FlowLayout(FlowLayout.CENTER,10,10));p4=new JPanel();
p4.setLayout(new FlowLayout(FlowLayout.CENTER,10,10));p5=new JPanel();
p5.setLayout(new FlowLayout(FlowLayout.CENTER,10,10));p6=new JPanel();
p6.setLayout(new FlowLayout(FlowLayout.CENTER,10,10));cp.add(p1);
cp.add(p2);
cp.add(p3);
cp.add(p4);
cp.add(p5);
cp.add(p6);
l1=new JLabel("Enter your Name");t1=new JTextField(20);
l2=new JLabel("Enter your Age");t2=new JTextField(20);
bg=new ButtonGroup();
l3=new JLabel("Enter your City"); r1=new JRadioButton("Madurai");r2=new
JRadioButton("Chennai");r3=new JRadioButton("Trichy");

                    bg.add(r1);
bg.add(r2);
bg.add(r3);
l4=new JLabel("Select your software");ch1=new JCheckBox("C");
ch2=new JCheckBox("C++");ch3=new JCheckBox("Java");bg1=new ButtonGroup();
bg1.add(ch1);
bg1.add(ch2);
bg1.add(ch3);
b1=new JButton("OK"); b2=new JButton("CANCEL");b1.addActionListener(this);
b2.addActionListener(this); r1.addActionListener(this); r2.addActionListener(this);
r3.addActionListener(this); ch1.addActionListener(this);ch2.addActionListener(this);
ch3.addActionListener(this);msg=new JLabel(""); p6.add(msg);
p1.add(l1);
p1.add(t1);
p2.add(l2);
p2.add(t2);
```

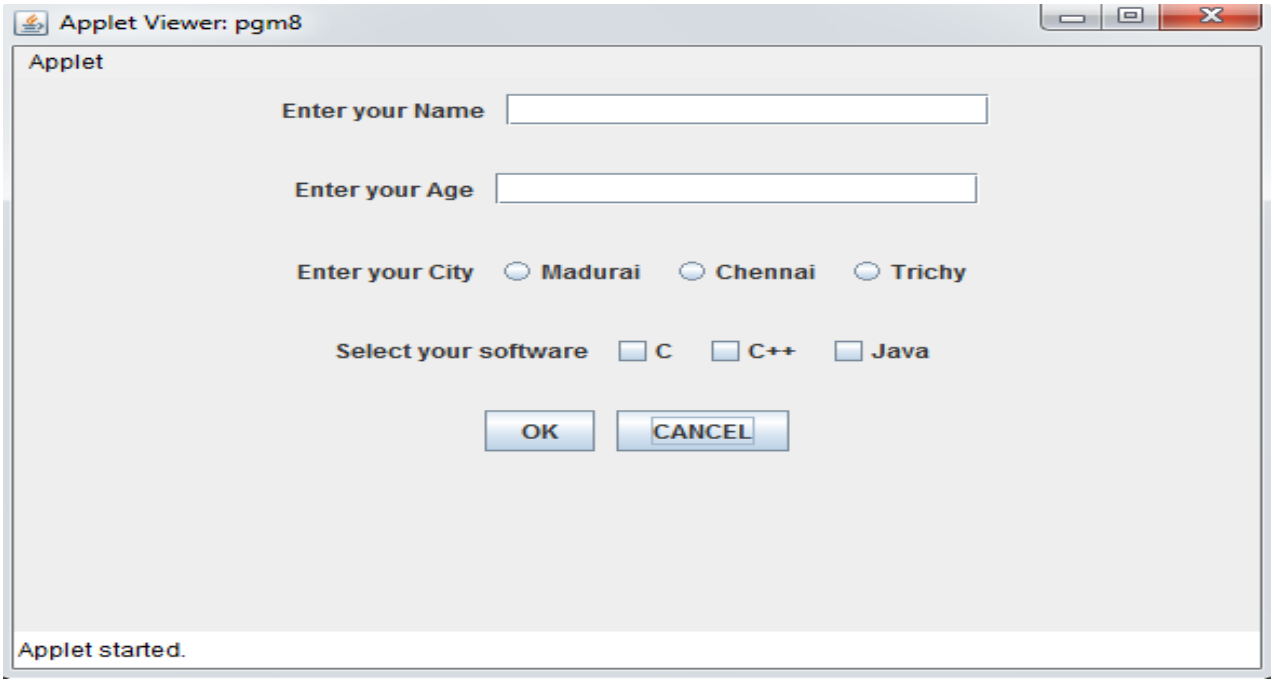
```
p3.add(l3);
p3.add(r1);
p3.add(r2);
p3.add(r3);
p4.add(l4);
p4.add(ch1);
p4.add(ch2);
p4.add(ch3);
p5.add(b1);
p5.add(b2); msg=new JLabel("");p6.add(msg);
}
public  void actionPerformed(ActionEvent ae)
{
if(ae.getSource()==b1)
{
if(r1.isSelected()==true)
{


        x1=r1.getText();
}
else if(r2.isSelected()==true)
{

x1=r2.getText()


              }
else if(r3.isSelected()==true)
{
x1=r3.getText();
}
str=" from the city:"+x1; if(ch1.isSelected()==true)
{
x2=ch1.getText();
}
if(ch2.isSelected()==true)
{
x2+=","+ch2.getText();
}
if(ch3.isSelected()==true)
{
x2+=" ,"+ch3.getText();
}
str+=" and You have Selected:"+x2; msg.setText("Welcome!!"+t1.getText()+"
("+t2.getText()+" )"+str);
}
```

```
if(ae.getSource()==b2)
{
t1.setText("");
t2.setText(""); ch1.setSelected(false);ch2.setSelected(false);ch3.setSelected(false);
r1.setSelected(false); r2.setSelected(false); r3.setSelected(false);
msg.setText("Your Registration is Cancelled");
}
}
}
```

OUTPUT:

C:\Program Files \Java\jdk1.7.0\bin>javac pgm8.java C:\Program Files \Java\jdk1.7.0\bin>appletviewer pgm8.java





RESULT: The above program has been executed successfully and the output was verified.

3. USE JDBC CONNECTIVITY AND CREATE TABLE, INSERT, DELETE AND UPDATE DATA

Aim:

To write a java program using JDBC Connection with ODBC technique to create table and
perform insert, update and delete data using MS-Access.

Way to Procedure:

Database Creation:

Step 1:
Go to Start □Programs □Microsoft Access 2007 □Select the option BLANK DATABASE

Step 2:
Choose Path of database and give filename "student" with file format Microsoft access 2002-2003 format (.mdb).

Step 3:
Save table as "student_tab".

Step 4:
Select design view of table by clicking right button and give fields of table such as id, namewith respective data type number, text respectively and set unique key in id.

Step 5:
Give sample records and save the database and table data.

Step 6:
Close the package Microsoft access.

Data Source name creation:

Step 1:
Go to □ start □control panel □administrative tools □data source (ODBC) and get thewizard.

Step 2:
In DSN wizard, click add button and choose driver as "Microsoft diver do access (*.mdb).

Step 3:
Give data source name as "stud" and select path of database then click OK.

Step 4:
Exit from ODBC wizard.

Algorithm:

Step1:
Start the program.

Step 2:
Include packages java.io and java.sql.

Step 3:
Define class with name “jdbc” and define the main function.

Step 4:
Declare objects for Connection, Statement, ResultSet and also declare the object forBufferedReader class.

Step 5:
Declare local variables ch,rno,n as integer and na as String.

Step 6:
Register the JdbcOdbcDriver and make a connection using getConnection() by giving Data
Source Name “Stud”.

Step 7:
Define switch case 1 for insert records,case 2 for delete records,3 for update records and 4for Display records.

Step 8:
Do Step 7 until will give choice > 4.

Step 9:
Close Statement object and Connection object.

Step 10:
Stop the Program.

SOURCE CODE:

```
// Use JDBC connectivity and create table, insert, update and delete dataimport java.io.*;
import java.sql.*;
class jdbc
{
public static void main(String ar[])throws Exception
{
Connection con;
Statement st;
ResultSet rs;
BufferedReader br=new BufferedReader (new InputStreamReader(System.in));int
ch,rno,n;
String na; Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
con=DriverManager.getConnection("jdbc:odbc:stud"); st=con.createStatement();
do
{
System.out.println("DATABASE MANIPULATION USING JDBC");
System.out.println("1.Insert\n2.Delete\n3.Update\n4.Display"); System.out.println("Enter
the choice"); ch=Integer.parseInt(br.readLine());
switch(ch)
{
case 1:
System.out.println("Enter Id to Insert:"); rno=Integer.parseInt(br.readLine());
System.out.println("Enter name to Insert:");na=br.readLine();
try
{
n=st.executeUpdate("insert into student_tab values("+rno+",'"+na+"')");
System.out.println(n+" row Inserted!!");
}
catch(SQLException e)   {       }break;
case 2:
System.out.println("Enter Id to Delete:");rno=Integer.parseInt(br.readLine());
try
{
n=st.executeUpdate("delete * from student_tab where id="+rno);System.out.println(n+"
row Deleted!!");
}
catch(SQLException e){}break;
case 3:
System.out.println("Enter Id to Edit:");


        rno=Integer.parseInt(br.readLine()); System.out.println("Enter name to Edit:");
na=br.readLine();
try
{
                        n=st.executeUpdate("update student_tab set name='"+na+"' where
                                                                id="+rno);
System.out.println(n+" row Updated!!");
```

```
}
catch(SQLException e){}break;
case 4:
try
{
rs=st.executeQuery("select * from student_tab");
System.out.println("ID\tNAME\n**********"); while(rs.next())
{
System.out.println(rs.getInt(1)+"\t"+rs.getString(2));
}
}
catch(SQLException e) { }break;
default:
System.out.println("Invalid Choice");
}
}while(ch<=4);st.close();
con.close();
}
}
```

OUTPUT:

C:\Program Files \Java\jdk1.7.0\bin>javac jdbc.javaC:\Program Files \Java\jdk1.7.0\bin>java jdbc

DATABASE MANIPULATION USING JDBC
1..Insert 2.Delete 3.Update 4.Display Enter Choice:
1
Enter Id to Insert:
111
Enter name to Insert:
haafi
1 row Inserted!!
DATABASE MANIPULATION USING JDBC
1.Insert 2.Delete 3.Update 4.Display Enter Choice:4ID NAME
****
111 haafi
222 sita
DATABASE MANIPULATION USING JDBC
1.Insert 2.Delete 3.Update 4.Display Enter Choice:2
Enter Id to Delete:
222
1 row Deleted!!
DATABASE MANIPULATION USING JDBC
1.Insert 2.Delete 3.Update 4.Display Enter Choice:
4
ID NAME
****
111 haafi
DATABASE MANIPULATION USING JDBC
1.Insert 2.Delete 3.Update
4.Display Enter Choice:3
Enter Id to Edit:
111
Enter name to Edit:
haasika
1 row Updated!!
DATABASE MANIPULATION USING JDBC
1.Insert 2.Delete 3.Update 4.Display Enter Choice:
4
ID NAME
****
111 haasika
DATABASE MANIPULATION USING JDBC
1.Insert 2.Delete 3.Update 4.Display Enter Choice: 5Invalid Choice

RESULT: The above program has been executed successfully and the output was verified.

4. IMPLEMENT A CLIENT/SERVER APPLICATION USING RMI

Aim: To write a java program to implement a Client/Server application using RM1.

Algorithm:

Program 1: Define the Remote Interface

Step 1:
Start the program.
Step 2:
Import the package java.rmi.*.
Step 3:
Define interface "AddServerIntf" by extends from Remote.
Step 4:
Declare the methods to perform arithmetic operation add, sub, mul, div, modulo and throwsRemoteException.
Step 5:
Stop the program.

Program 2: Implement Remote Interface

Step 1:
Start the program.
Step 2:
Import the packages java.rmi.* and java.rmi.server.*.

Step 3:
Define class AddServerImpl by extends from "UnicastRemoteObject" and implements"AddServerIntf".
Step 4:
Define the procedure for interface methods by throwing RemoteException.
Step 5:
Stop the program.

Program 3: Implementation of Server Machine

Step 1:
Start the program.
Step 2:
Import the packages java.rmi.* and java.net.*
Step 3:
Define class server with main function.
Step 4:
Create object for the class AddServerImpl.
Step 5:
Using naming.rebind() method add the interface to the server .
Step 6:
Stop the program.

Program 4: Implementation of Client Machine

Step 1:
Start the program.
Step 2:
Define class "client" with main() function
Step 3:
Create object for server interface with proper URL definition using naming.lookup();
Step 4:
Using this object call required methods and handling exception.
Step 5:
Stop the program.

SOURCE CODE:

// 1. Define the Remote Interface

```
import java.rmi.*;
public interface AddServerIntf extends Remote
{
int add(int a,int b) throws RemoteException; int sub(int a,int b) throws RemoteException;
int mul(int a,int b) throws RemoteException; int div(int a,int b) throws RemoteException;
int mod(int a,int b) throws RemoteException;
}
```

// 2. Implement Remote Interface

```
import java.rmi.*; import java.rmi.server.*;
public class AddServerImpl extends UnicastRemoteObject implements AddServerIntf
{
public AddServerImpl() throws RemoteException
{}
public int add(int a,int b)throws RemoteException
{
return (a+b);
```

```
}
public int sub(int a,int b)throws RemoteException
{
return (a-b);
}
public int mul(int a,int b)throws RemoteException
{
return (a*b);
}
public int div(int a,int b)throws RemoteException
{
return (a/b);
}
public int mod(int a,int b)throws RemoteException
{
return (a%b);
}
}
```

// 3. Implementation of Server Machine

```
import java.rmi.*; import java.net.*; public class AddServer
{
public static void main(String args[])

{
try
{

AddServerImpl obj = new AddServerImpl();Naming.rebind("addserver", obj);
System.out.println("server started");

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

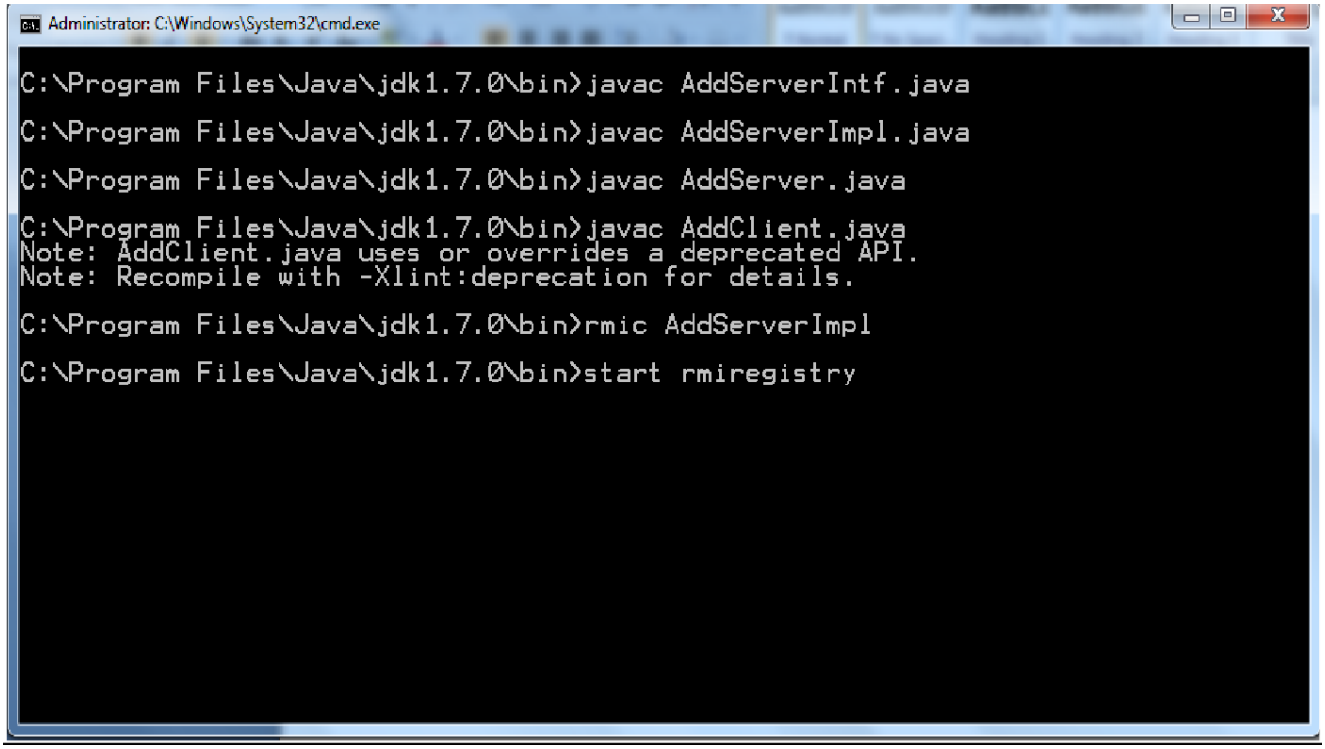
// 4. Implementation of Client Machine

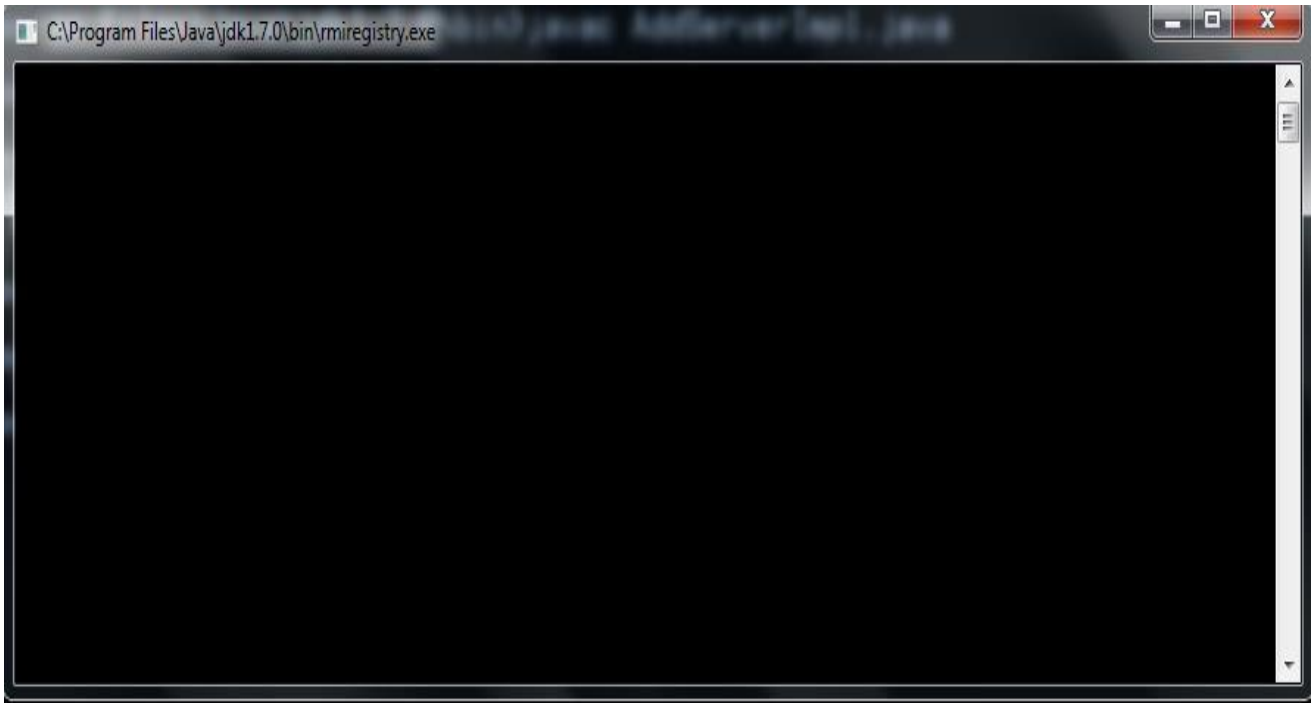
```
import java.rmi.*; import java.io.*; public class AddClient

{
public static void main(String args[])
{
try
{
DataInputStream ds=new DataInputStream(System.in); String s="rmi://MY-
PC/addserver";
```

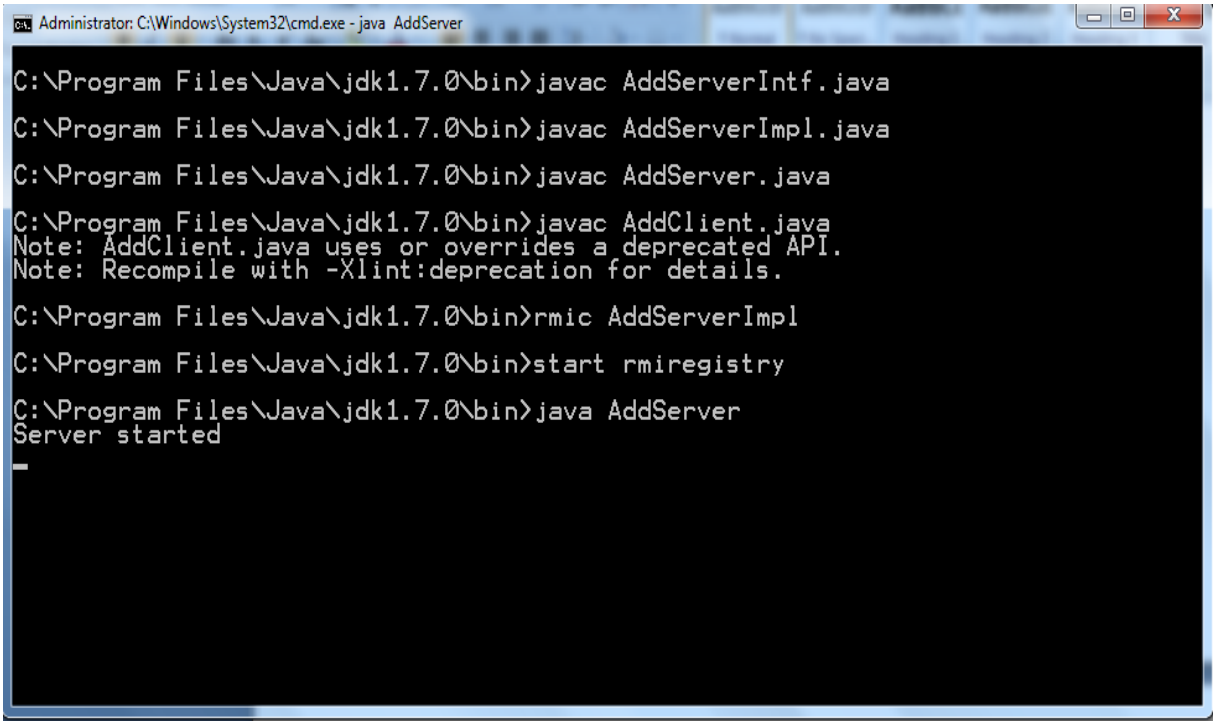
```
AddServerIntf obj = (AddServerIntf)Naming.lookup(s);System.out.println("ENTER THE
VALUES FOR a & b:");int a=Integer.parseInt(ds.readLine());
int b=Integer.parseInt(ds.readLine()); System.out.println("ADDITION="+obj.add(a,b));
System.out.println("SUBTRACTION="+obj.sub(a,b));
System.out.println("MULTIPLICATION="+obj.mul(a,b));
System.out.println("DIVISION="+obj.div(a,b));
System.out.println("MODULODIVISION="+obj.mod(a,b));
}
catch (Exception e)
{
System.out.println("Exception: " + e);
}
}
}
```

OUTPUT:

LOCAL HOST SERVER SIDE COMMAND WINDOW:

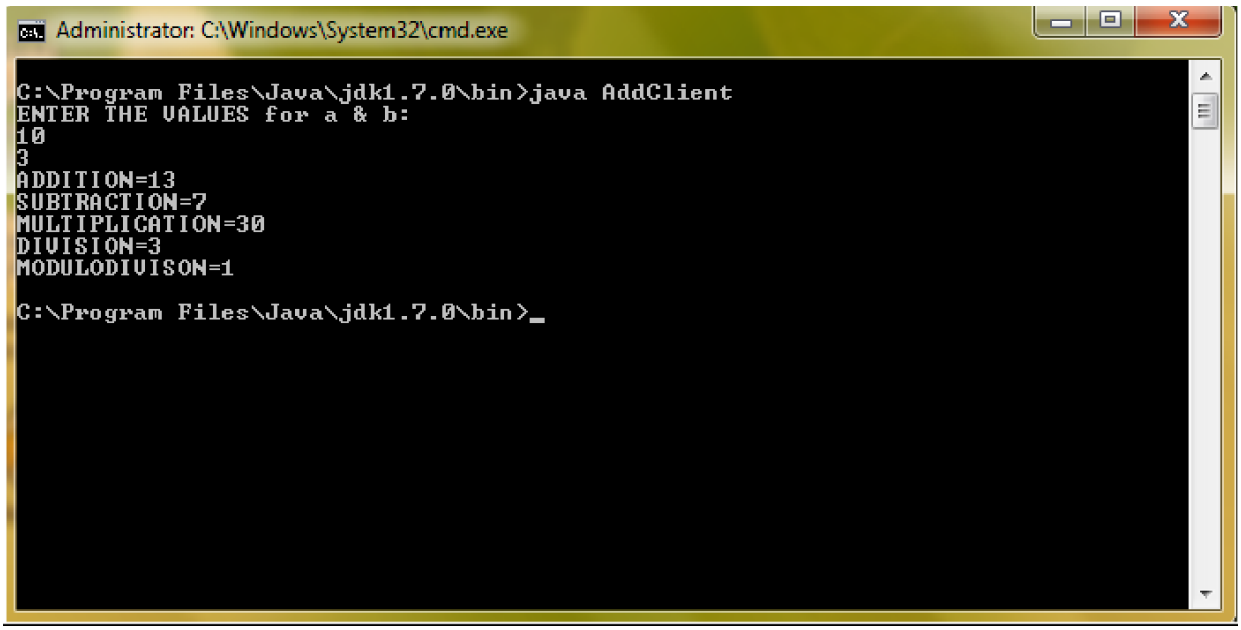






LOCAL HOST CLIENT SIDE COMMAND WINDOW:

RESULT: The above program has been executed successfully and the output was verified.



5. CREATE A COOKIE AND SET THE EXPIRY TIME

Aim: To write a java program to create a cookie and set the expiry time of the same.

Algorithm:

Step 1:
Start the program.

Step 2:
Create html file which contains the text fields for first and last name.

Step 3:
Set submit button also.

Step 4:
Create java program and import needed package.

Step 5:
Define class cook by extending Httpservlet.

Step 6:
Define objects for "cookie" class within Doget () method.

Step 7:
Set expiry time for 2 cookies using SetMaxAge ()

Step 8:
Add cookies to response object.

Step 9:
Set content type of page.

Step 10:
Create object for printWriter for printing firstname and lastname which is from html file.

Step 11:
Stop the program.

SOURCE CODE:

// Create a Cookie and Set the Expiry Time

Hello.html

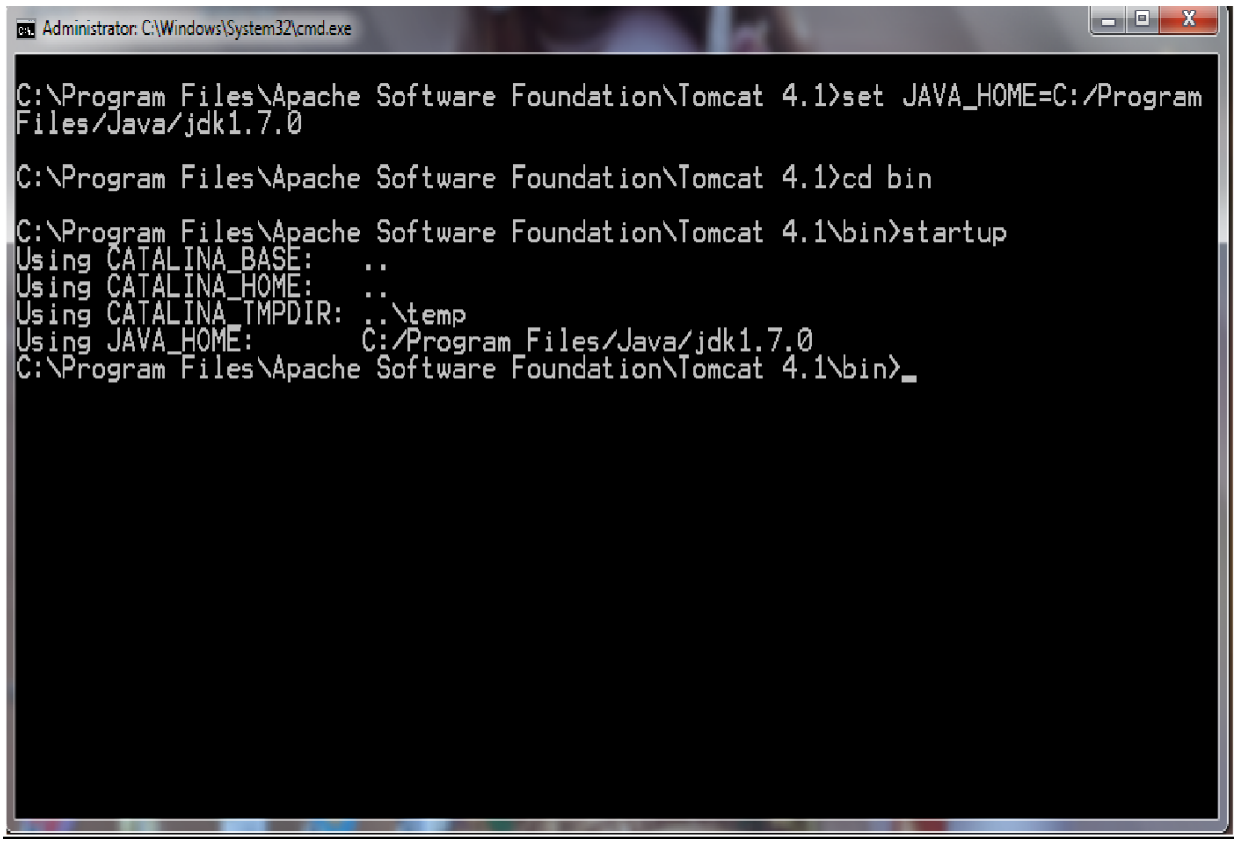
```
<html>
<head><title>cookies</title></head>
<body>
<form action="http://local host:8080/examples/servlet/cook" method="get">FIRST
NAME:<input type="text" name="fn"/></br>
LAST NAME:<input type="text" name="ln"/>
</br><input type="submit" value="SUBMIT"/>
</form></body></html>
```

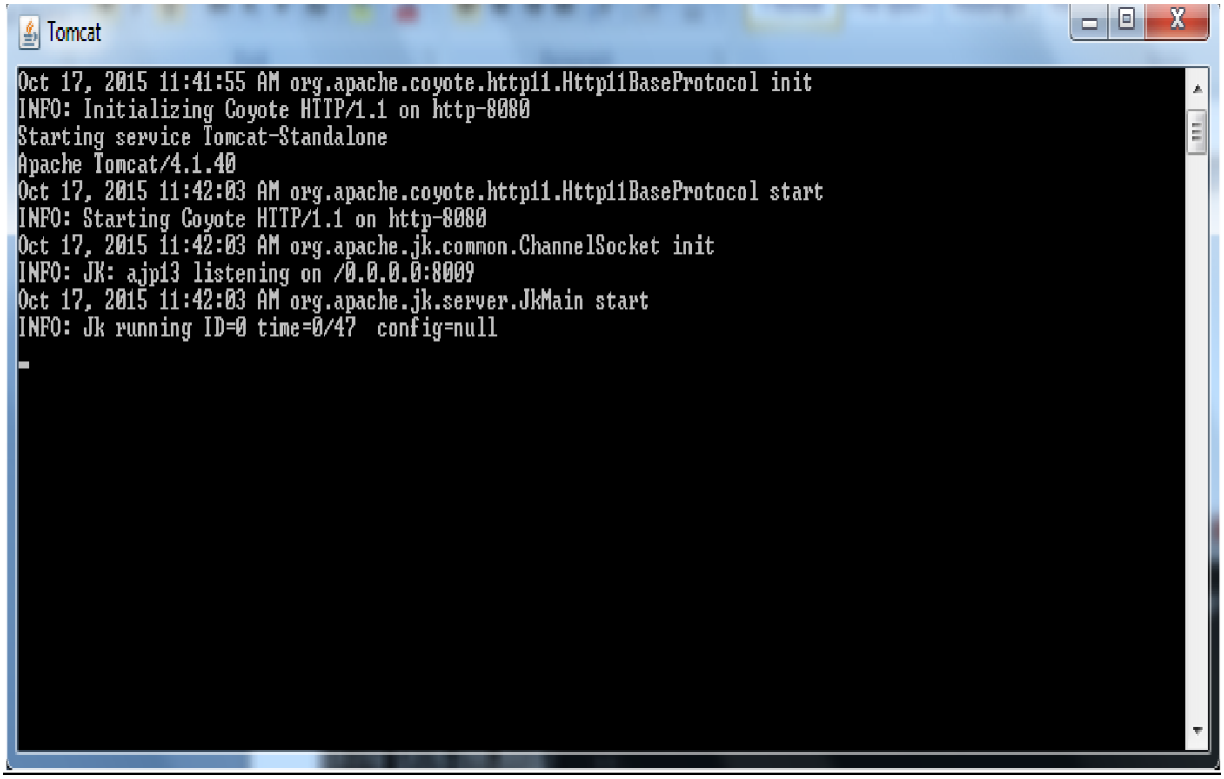
Cook.java

```
import java.io.*; import javax.servlet.*;
import javax.servlet.http.*;
public class cook extends HttpServlet
{
public void doGet(HttpServletRequest req,HttpServletResponse res)throws
ServletException,IOException
{
Cookie c1=new Cookie("cookie1",req.getParameter("fn")); Cookie c2=new
Cookie("cookie2",req.getParameter("ln"));c1.setMaxAge(60*60*24);
c2.setMaxAge(60*60*24); res.addCookie(c1); res.addCookie(c2);
res.setContentType("text/html");PrintWriter out=res.getWriter();
out.println("<center><font color='red'>SAMPLE COOKIES</font></center>");
out.println("<font color='green'>FIRST NAME:"+req.getParameter("fn")+"</font>");
out.println("<font color='green'>LAST NAME:"+req.getParameter("ln")+"</font>");
}
}
```

OUTPUT:

TOMCAT COMMAND WINDOW:



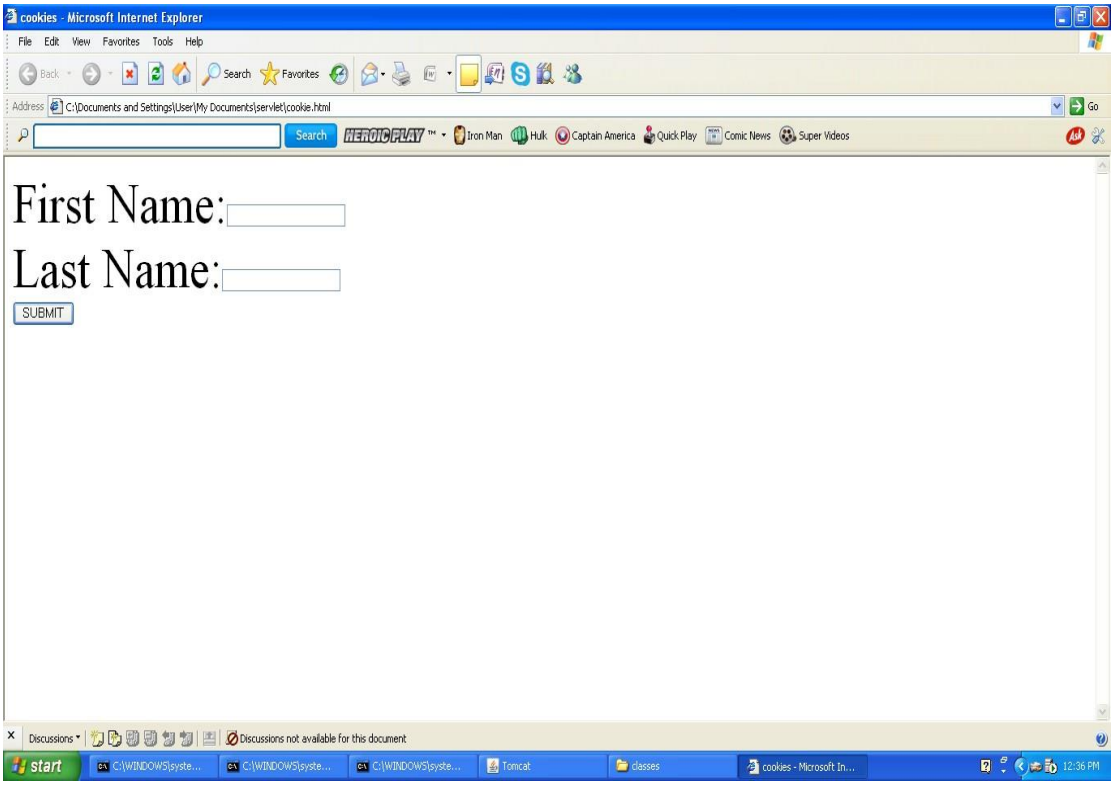


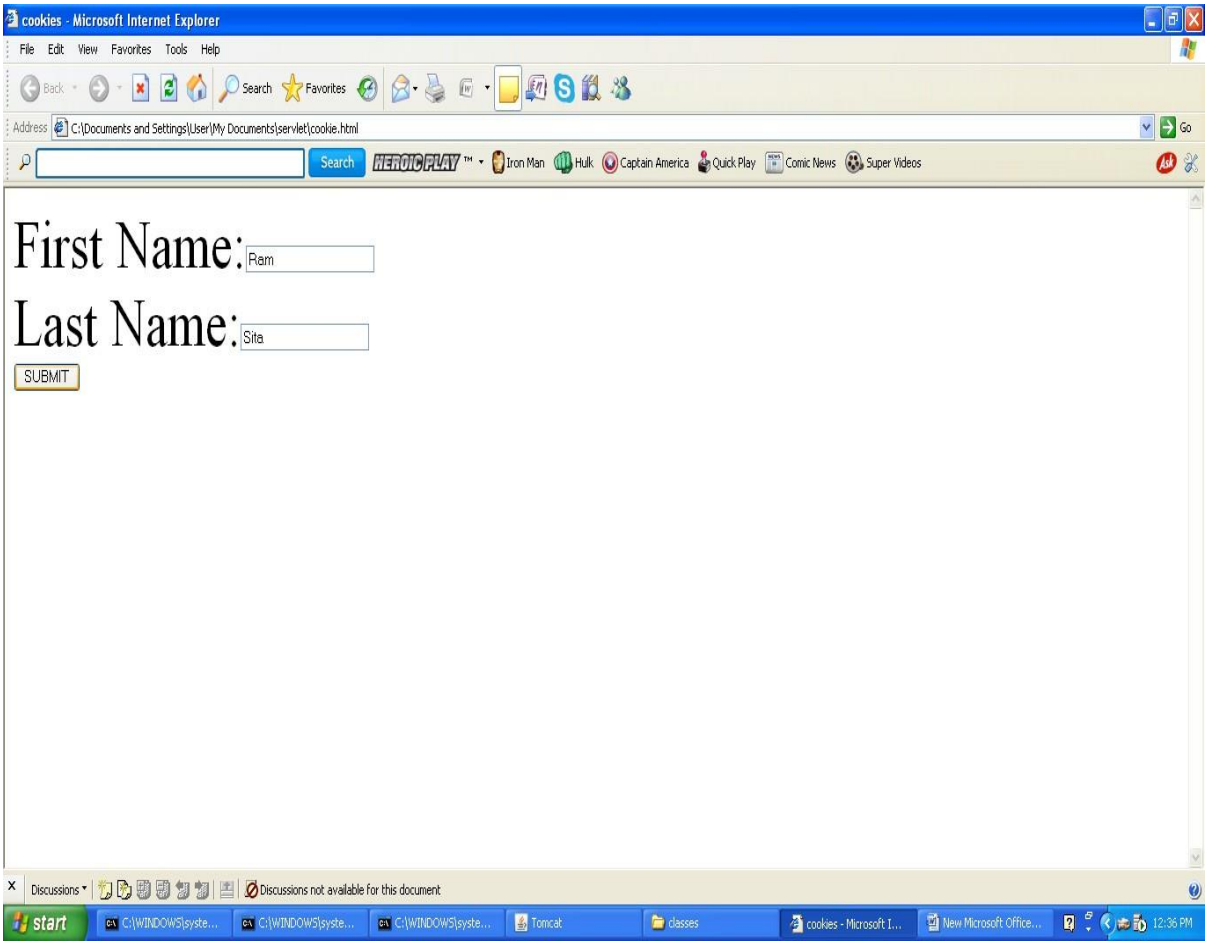
JAVA COMMAND WINDOW:

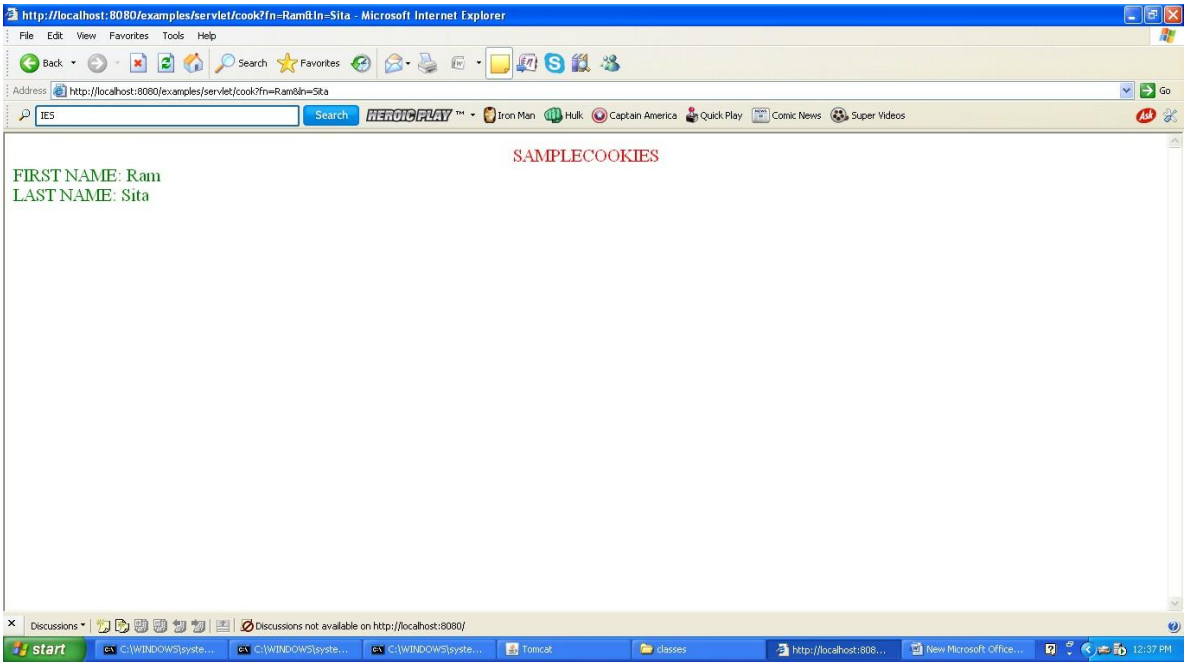
C:\Program Files\Java\jdk1.7.0\bin>javac cook.java -classpath "C:\Program Files \Apache Software Foundation\Tomcat 4.1\common\lib\servlet.jar"

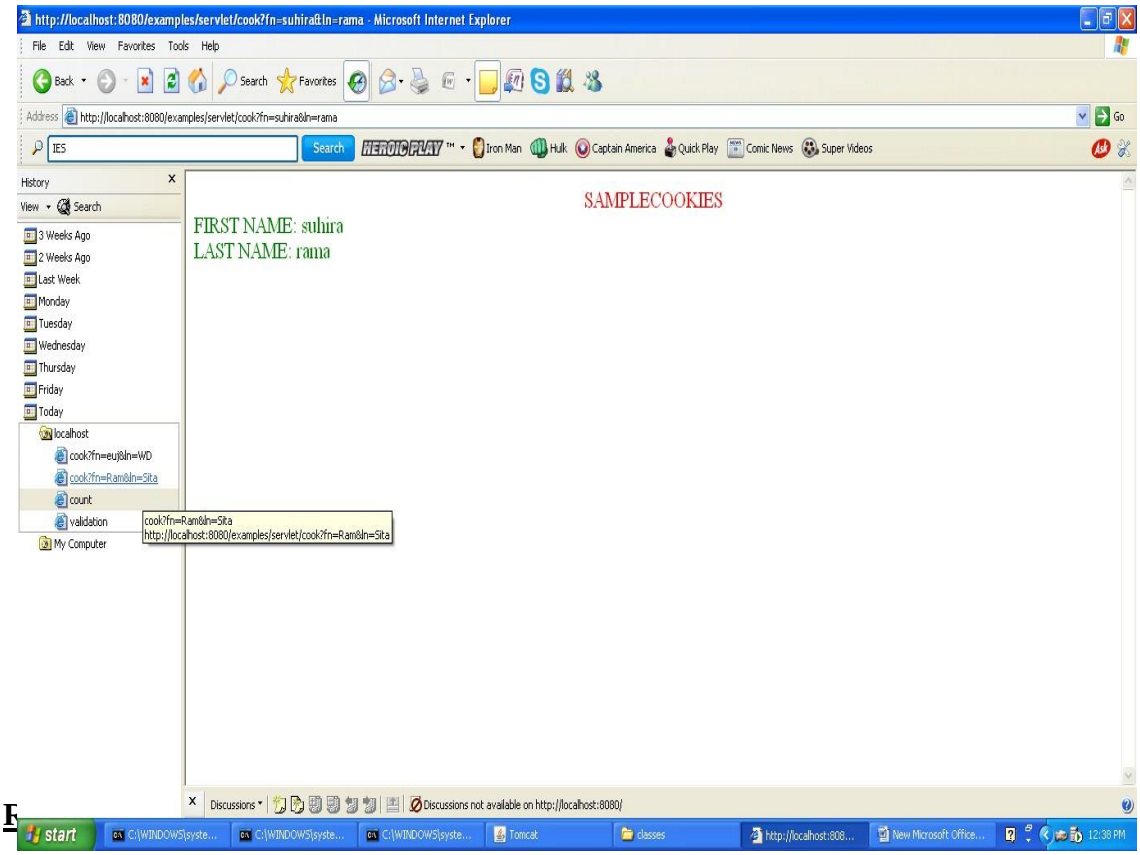
COPY THE FILE:
Local Disc(C:)□Program Files□Java□jdk1.7.0□bin□cook.class.
PASTE THE FILE:
Local Disc(C:) □ Program Files □ Apache Software Foundation □Tomcat 4.1 □ Webapps □
Examples □ WEB_INF □ classes □ paste cook.class

BROWSER WINDOW:




cookies - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address C:\Documents and Settings\User\My Documents\servlet\cookie.html
First Name: Ram
Last Name: Sita
SUBMIT
Discussions not available for this document


http://localhost:8080/examples/servlet/cook?fn=Ram&ln=Sita - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address http://localhost:8080/examples/servlet/cook?fn=Ram&ln=Sita
SAMPLECOOKIES
FIRST NAME: Ram
LAST NAME: Sita
Discussions not available on http://localhost:8080/


http://localhost:8080/examples/servlet/cook?fn=suhira&ln=rama - Microsoft Internet Explorer
SAMPLECOOKIES
FIRST NAME: suhira
LAST NAME: rama
cook?fn=Ram&ln=Sita
http://localhost:8080/examples/servlet/cook?fn=Ram&ln=Sita

F

6. COUNT NUMBER OF ACCESS TIMES OF THE SERVLET PAGE

Aim: To write java program to create Servlet to count the number of access time of that servlet
page.

Algorithm:

Step 1:
Start the program.

Step 2:
Import the packages java.io. javax .servlet, javax.servlet.http .

Step 3:
Define class count extends from Httpservlet .

Step 4:
Set counting variable c as zero.

Step 5:
Define doGet () method. Set content type of page and initialize the object for printWriterclass by calling the method getWriter ()

Step 6:
Increment the variable c by 1 for every access.

Step 7:
Print the value of c which indicates the counting.

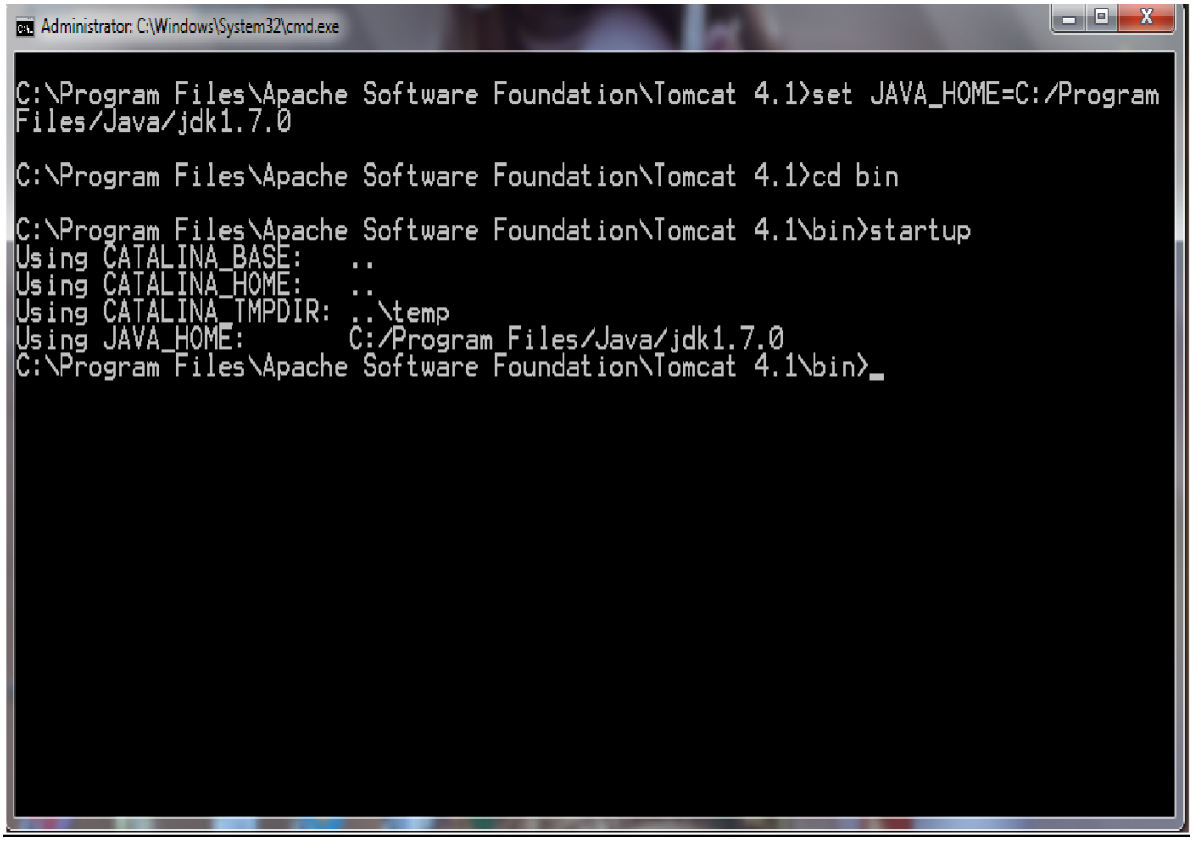
Step 8:
Stop the program.

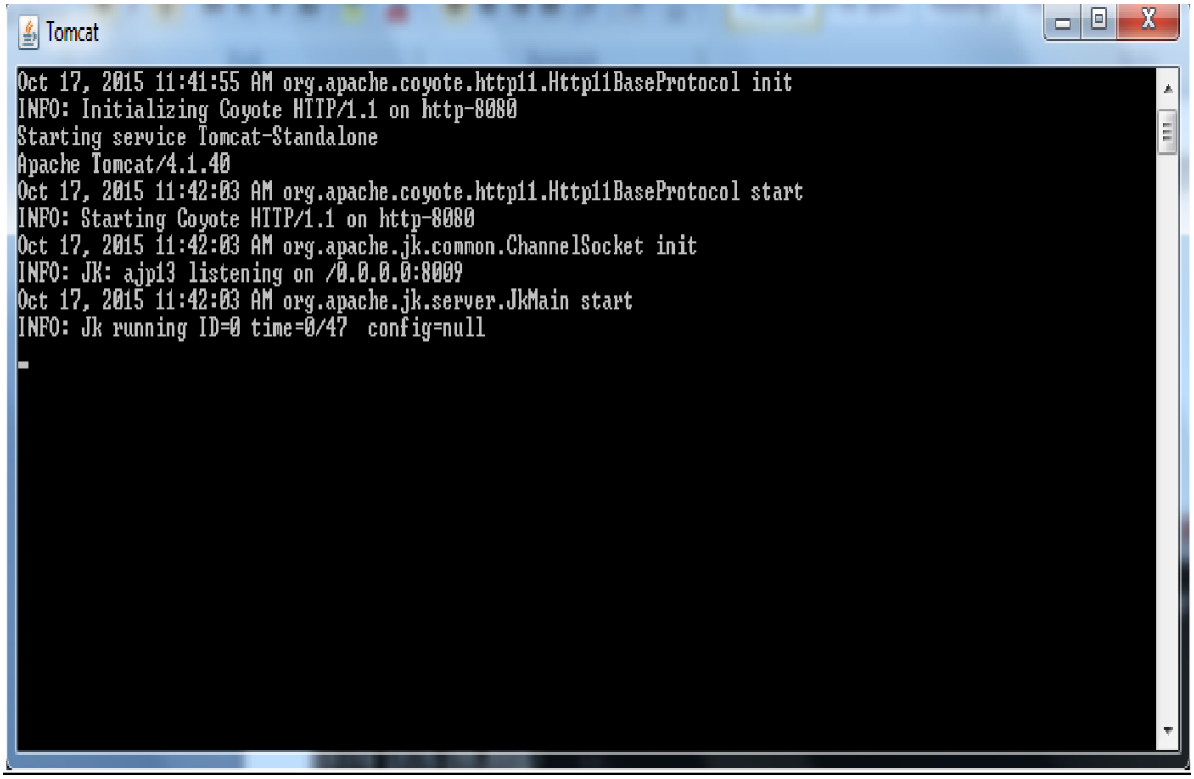
SOURCE CODE:

```
//Count Number of Access Times of the Servlet Pageimport java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class count extends HttpServlet
{
int c=0;
public void doGet(HttpServletRequest req,HttpServletResponse res)throws
ServletException,IOException
{
res.setContentType("text/plain");PrintWriter out=res.getWriter(); c++;
out.println("since loading,this servlet hasbeen accessed"+c+"times");
}
}
```

OUTPUT:

TOMCAT COMMAND WINDOW:



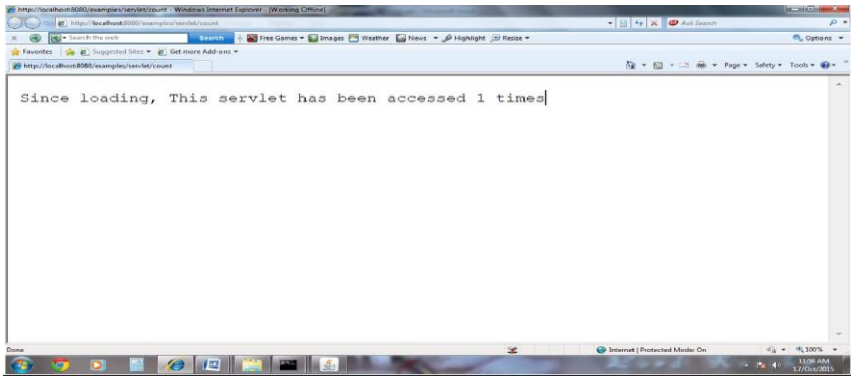


JAVA COMMAND WINDOW:

C:\Program Files\Java\jdk1.7.0\bin>javac count.java -classpath "C:\Program Files \Apache Software Foundation\Tomcat 4.1\common\lib\servlet.jar"

COPY THE FILE:
Local Disc(C:)□Program Files□Java□jdk1.7.0□bin□count.class.
PASTE THE FILE:
Local Disc(C:) □ Program Files □ Apache Software Foundation □Tomcat 4.1 □ Webapps □
Examples □ WEB_INF □ classes □ paste count.class

BROWSER WINDOW:



RESULT:

The above program has been executed successfully and the output was verified.

7. CREATE A FORM AND VALIDATE PASSWORD USING SERVLET

Aim: To write a java program to create a form and validate password using servlet.

Algorithm:

Step 1:
Start the program.
Step 2:
Create html file which contains the textbox for username and password with submit andreset button with alignment made by tables.

Step 3:
Create java program with class "validation"

Step 4:
Import required packages and extends the class from GenericServlet in main class.

Step 5:
Create object for printWriter () and read parameter () which is accessed by Httpservletrequest object.

Step 6:
Check username and password is "admin" if so, print the message "Welcome to thiswebpage", if not so, print the error message.

Step 7:
Close printWriter object.

Step 8:
Stop the program.

SOURCE CODE:

// Create a Form and Validate Password Using Servlet

Login.html:

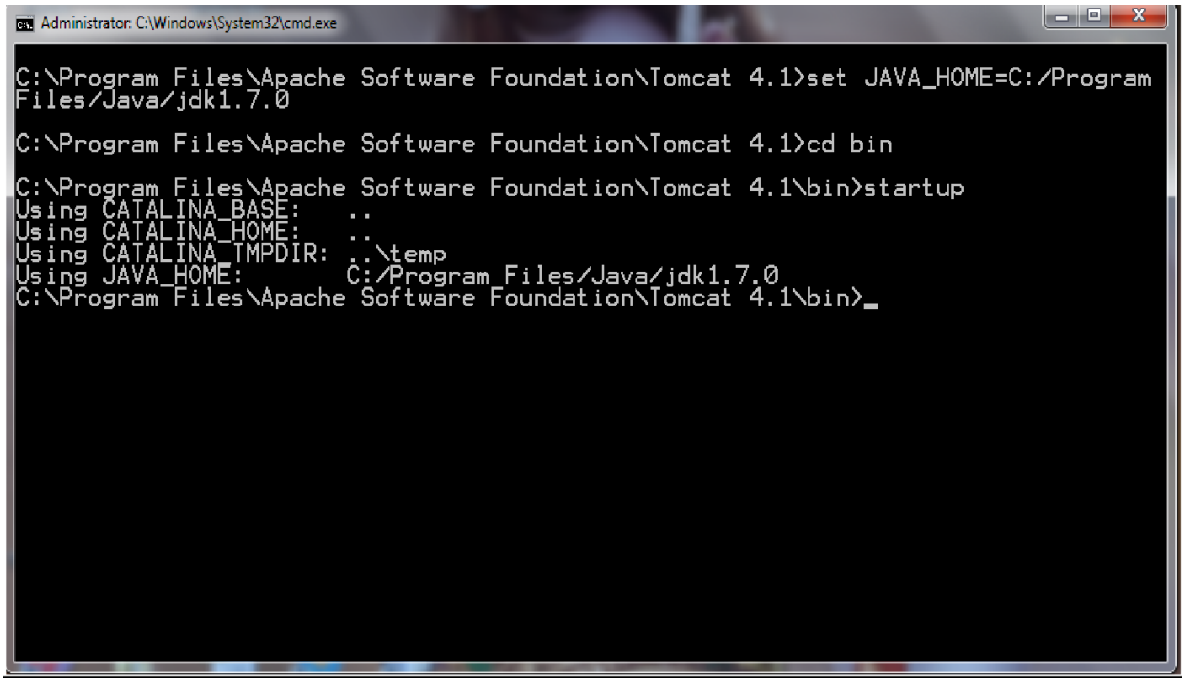
```
<html>
<head><title>login</title></head>
<body>
<form name="login form" method="post"
action="http://localhost:8080/examples/servlet/validation">
<br/><br/><br/><br/><br/>
<table align="center" border="3" border color="blue" cellspacing="0"height="120">
<tr><td align="center"><font color="blue" size="4">LOGIN FORM</font></td></tr>
<tr><td><table><tr><td>UserName</td><td><input type="text"
name="user"/></td></tr>
<tr><td>Password</td><td><input type="password" name="pwd"/></td></tr>
<tr><td align="center"><input type="submit" value="LOGIN"/></td><td
align="center"><inputtype="Reset" value="RESET"/></td></tr>
</table></td></tr></table></form></body>
</html>
```

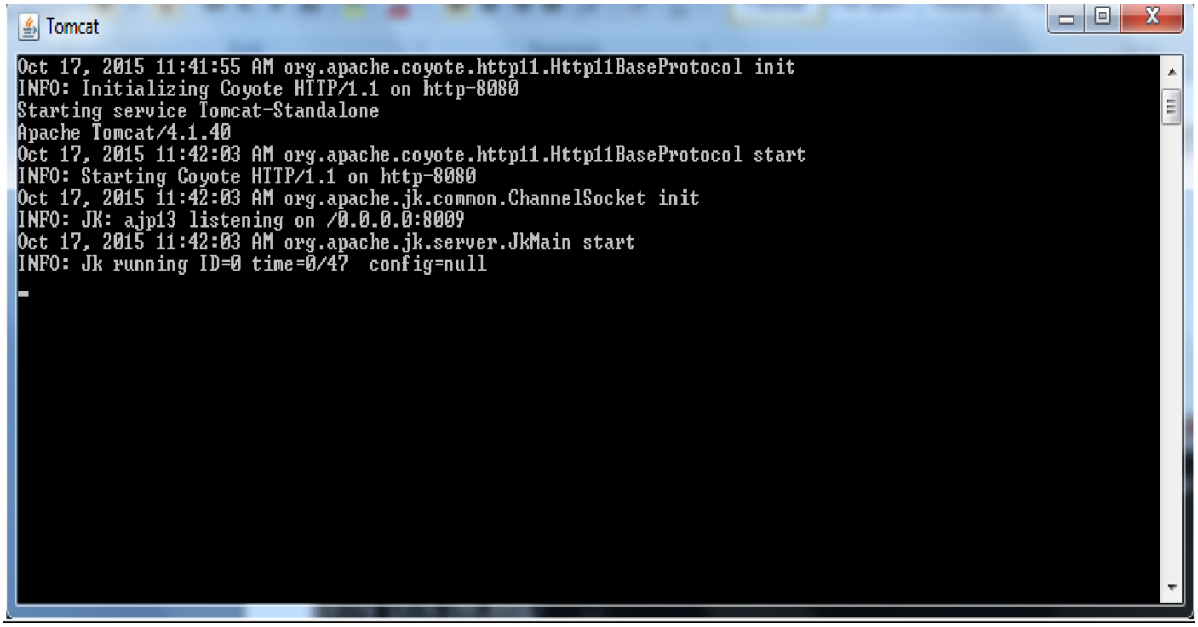
Validation.java:

```
import java.io.*; import java.util.*; import javax.servlet.*;
public class validation extends GenericServlet
{
public void service(ServletRequest req,ServletResponse res)throws
ServletException,IOException
{
PrintWriter pw=res.getWriter(); String x=req.getParameter("user");String
y=req.getParameter("pwd");
if(x.equals("admin")&&y.equals("admin"))
pw.println("<font color='green' size='5'>Welcome to this webpage</font>");else
pw.println("<font color='red' size='5'>Invalid username or password</font>");pw.close();
}
}
```

OUTPUT:

TOMCAT COMMAND WINDOW:



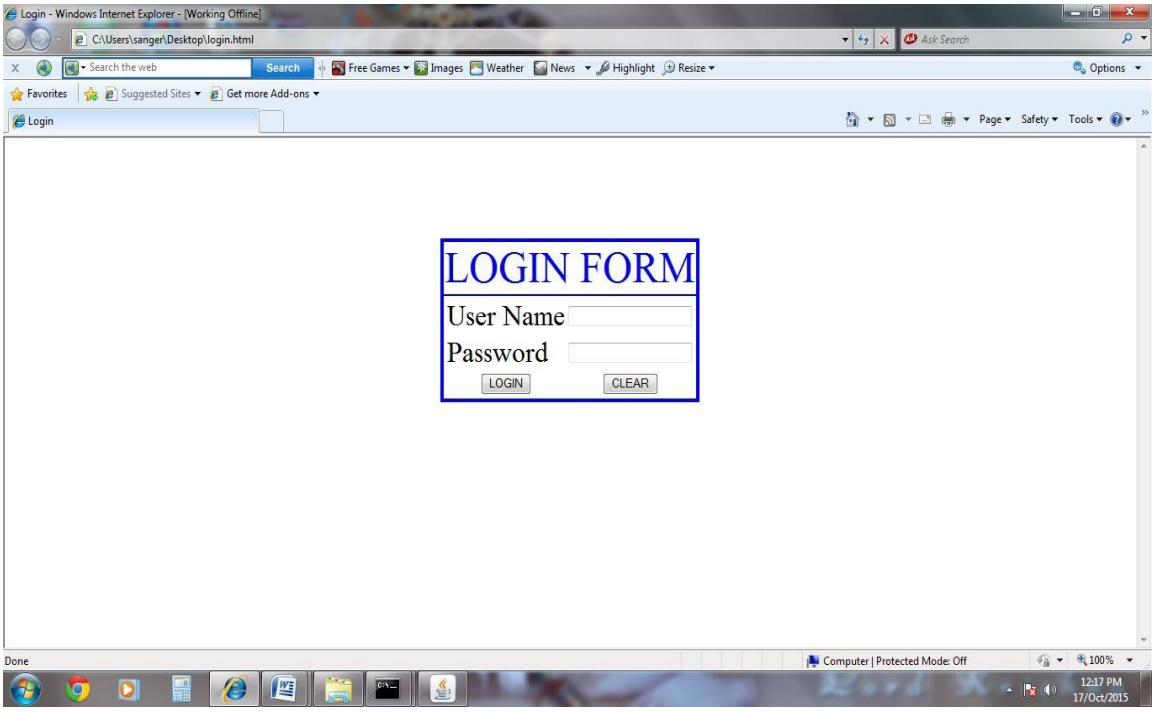


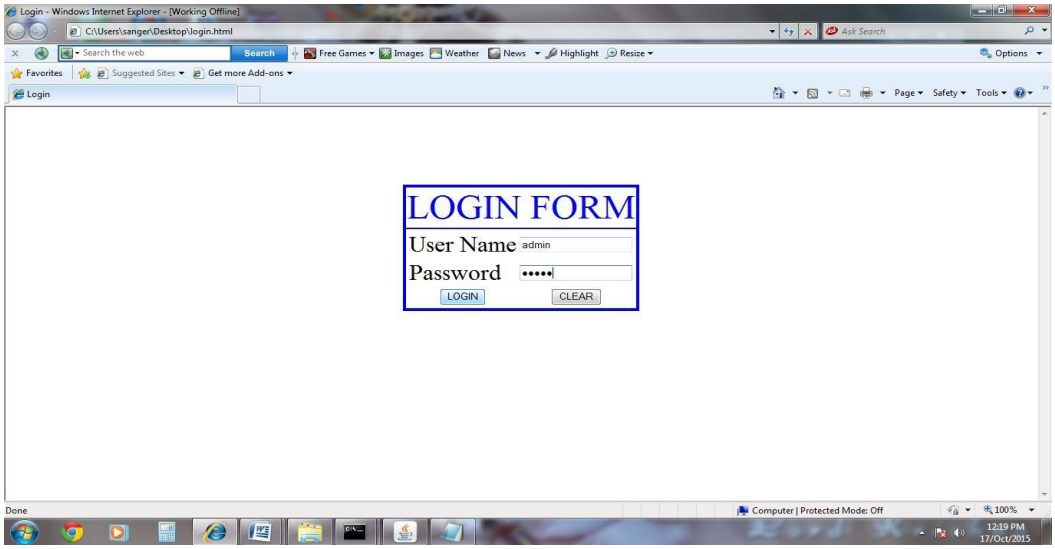
JAVA COMMAND WINDOW:

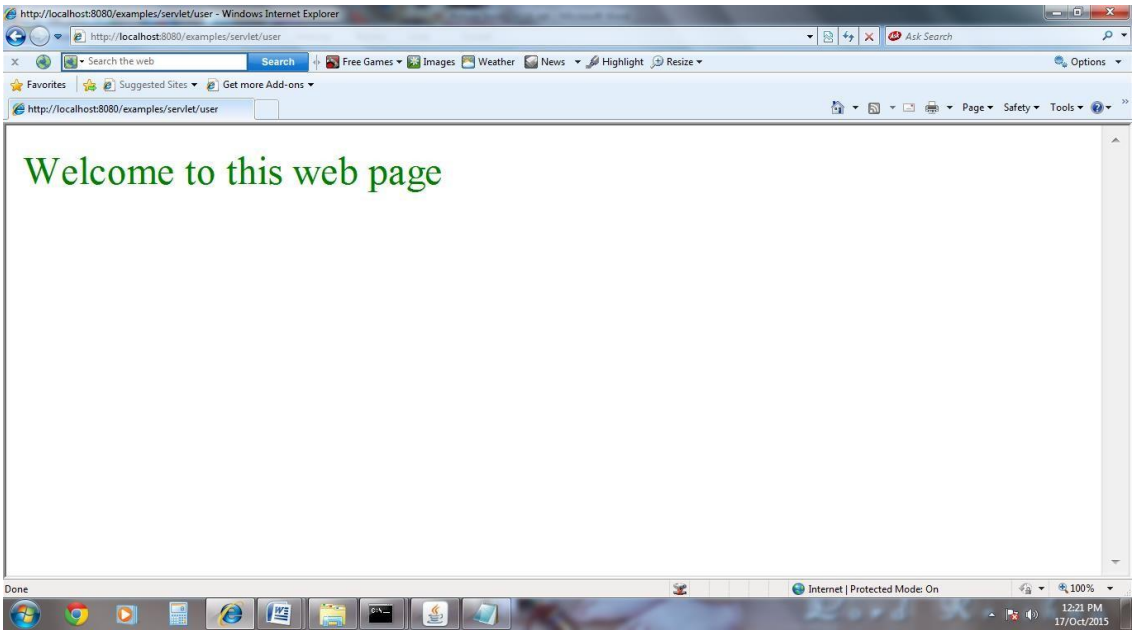
C:\Program Files\Java\jdk1.7.0\bin>javac validation.java -classpath "C:\Program Files \Apache Software Foundation\Tomcat 4.1\common\lib\servlet.jar"

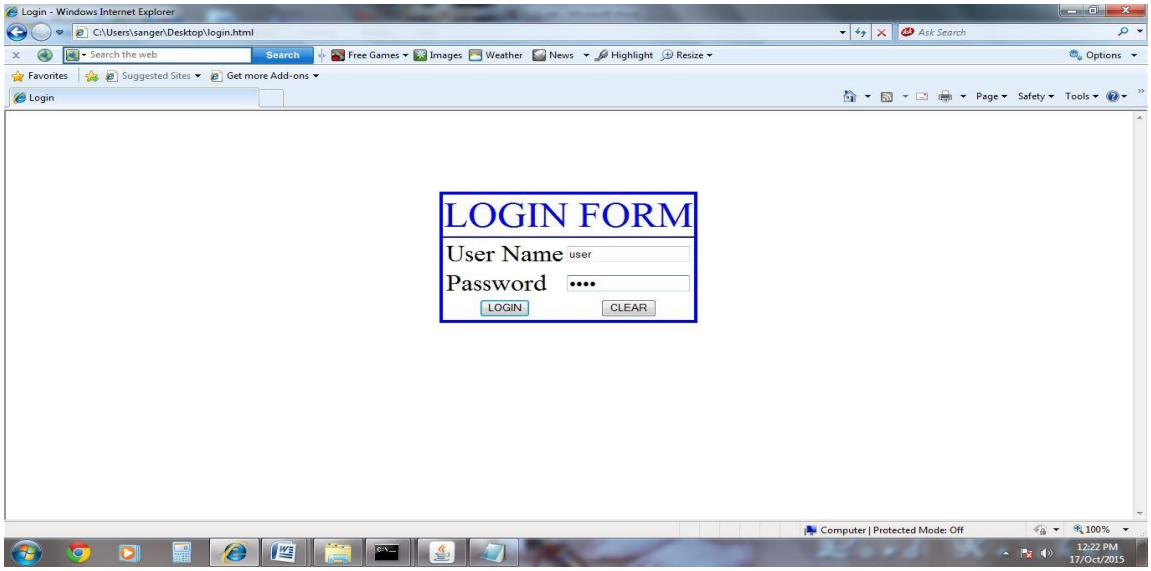
COPY THE FILE:
Local Disc(C:)□Program Files□Java□jdk1.7.0□bin□vaidation.class.
PASTE THE FILE:
Local Disc(C:) □ Program Files □ Apache Software Foundation □Tomcat 4.1 □ Webapps □
Examples □ WEB_INF □ classes □ paste validation.class

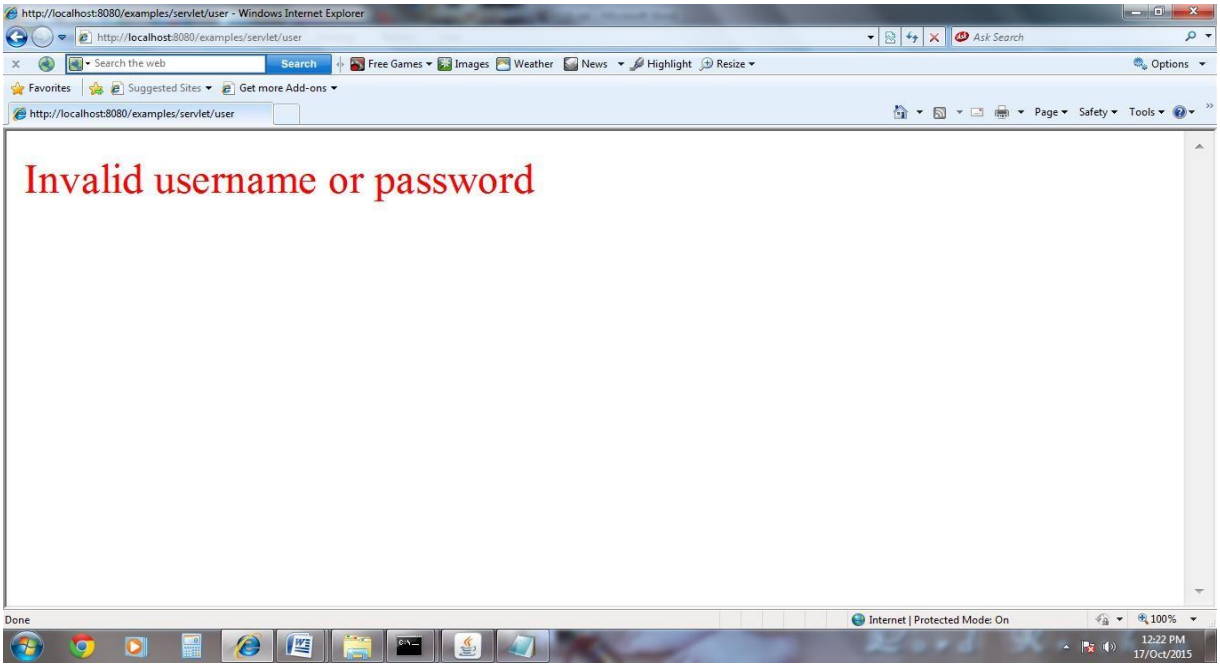
BROWSER WINDOW:




Login - Windows Internet Explorer - [Working Offline]
C:\Users\sanger\Desktop\login.html
LOGIN FORM
User Name
admin
Password
LOGIN
CLEAR
Done


http://localhost:8080/examples/servlet/user - Windows Internet Explorer
http://localhost:8080/examples/servlet/user
Welcome to this web page
Done





RESULT: The above program has been executed successfully and the output was verified.

8. CONVERT AN IMAGE IN RGB TO A GRAYSCALE IMAGE

Aim: Write a java program to convert an image in RGB to a grayscale image.

Algorithm:

Step 1:
Start the program.

Step 2:
Import the packages such as java.awt.*, java.awt.image.*, javax.imageio.ImageIO, java.io.*.

Step 3:
Define the main class rgb with main function and throws IOException.

Step 4:
Create object for DataInputStream, String class.

Step 5:
Create object for BufferedImage and read the input image file namely x1.jpg.

Step 6:
Take individual pixels of image and change the color of pixel by using the methods getRGB(),setRGB() with in for loop.

Step 7:
Get the output filename and stored the converted image within a filename by using ImageIO.write() function with the extension ".jpg".

Step 8:
Stop the program.

SOURCE CODE:

```
//Convert an image in RGB to a Grayscale Image import javax.imageio.ImageIO;
import java.awt.*;
import java.awt.image.*;import java.io.*;

public class rgb
{
public static void main(String args[]) throws IOException
{
DataInputStream in=new DataInputStream(System.in);String s;
int w;
BufferedImage img=ImageIO.read(new File("x1.jpg"));for(w=0;w<img.getWidth();w++)
{
for(int h=0;h<img.getHeight();h++)
{
Color oc=new Color(img.getRGB(w,h));
int avg=((oc.getRed()+oc.getGreen()+oc.getBlue())/3);Color cc=new Color(avg,avg,avg);
img.setRGB(w,h,cc.getRGB());

}
}
System.out.println("Enter the Output FileName:");s=in.readLine();
ImageIO.write(img,".jpg",new File(s+".jpg"));
System.out.println("RGB image x1.jpg was successfully converted to Grayscale
image stored in filename="+s+".jpg");
}
}
```

OUTPUT:

```
C:\Program Files \Java\jdk1.7.0\bin>javac rgb.javaC:\Program Files
\Java\jdk1.7.0\bin>java rgb

Enter the Output FileName:
x2
RGB image x1.jpg was successfully converted to Grayscale image stored in
filename=x2.jpg
```

Notes: Go to Java Home Directory to verify the output.

RESULT: The above program has been executed successfully and the output was verified.

9. DEVELOP CHAT SERVER USING JAVA

Aim: To write java program for chat server using datagram packet and datagram socket.

Algorithm:

Step 1:
Start the program.

Step 2:
Import java.net package for client server application.

Step 3:
Define buffer size, server port, client port, byte array and declare object for datagram socket.

Step 4:
Create method theserver () for sending data to client using datagram packet object and send() method.

Step 5:
Create method theclient() for receiving packets from server using datagram packet object byreceive() method.

Step 6:
Print data in client window by using getdata() and getlength() method.

Step 7:
Define main function and receive command line arguments.

Step 8:
If command line arguments length is 1 then call theserver() otherwise call theclient().

Step 9:
Stop the program.

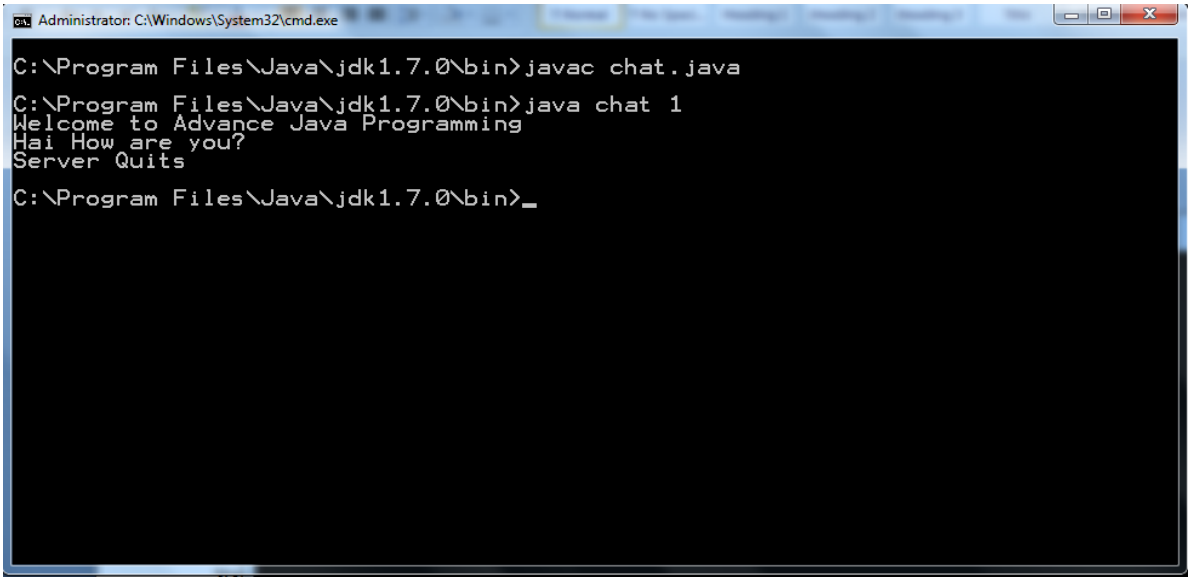
SOURCE CODE:

```
// Develop Chat Server Using Javaimport java.net.*;
class chat
{
public static int BUFSIZE=1024; public static int serverport=1057;public static int
clientport=1058; public static DatagramSocket ds;
public static byte buffer[]=new byte[BUFSIZE];public static void theserver()throws
Exception
{
int pos=0; while(true)
{
int c=System.in.read();switch(c)
{
case -1:
System.out.println("Server quits");return;
case '\n':
        DatagramPacket dp=new  DatagramPacket  (buffer,pos, InetAddress.getLocalHost(),
                                                                        clientport);
ds.send(dp);pos=0; break;
default:
buffer[pos++]=(byte)c;
}
}
}
public static void theclient()throws Exception
{
while(true)
{
DatagramPacket dp=new DatagramPacket(buffer,buffer.length);ds.receive(dp);
String str=new String(dp.getData(),0,dp.getLength());System.out.println(str);
}
}
public static void main(String ar[])throws Exception
{
if(ar.length==1)
{
ds=new DatagramSocket(serverport);theserver();


        }
else
{
                ds=new DatagramSocket(clientport);

                theclient();


}
```

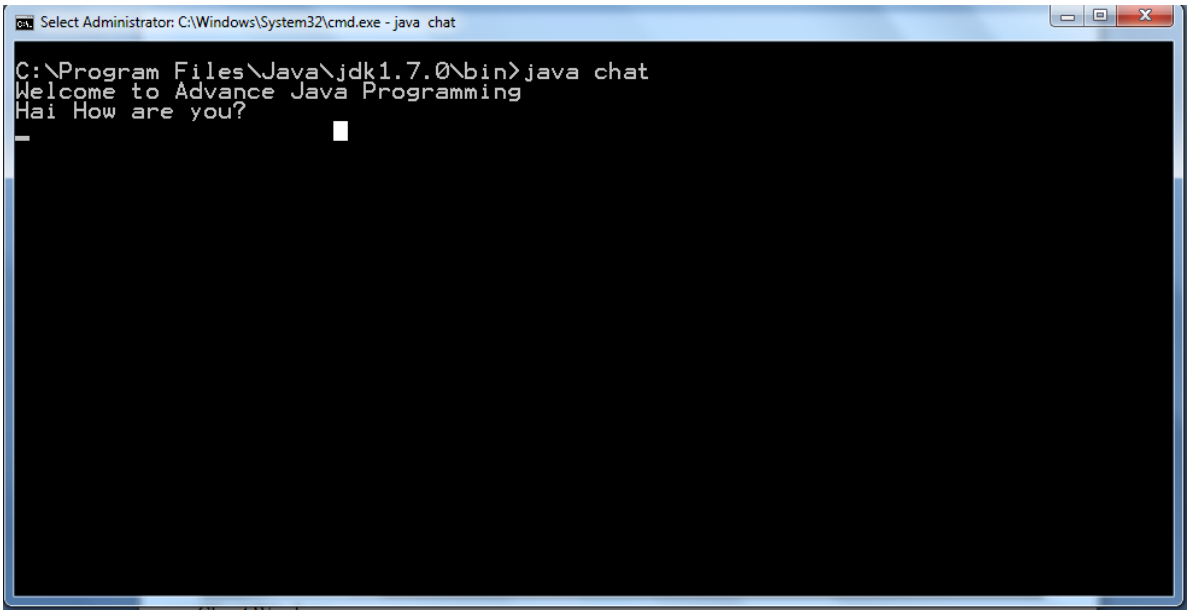
```
}
}
```

OUTPUT:

LOCALHOST SERVER SIDE COMMAND WINDOW



LOCALHOST CLIENT SIDE COMMAND WINDOW



RESULT: The above program has been executed successfully and the output was verified.