PROGRAM – 1

1.1 Write a program to implement event handling in Java AWT.

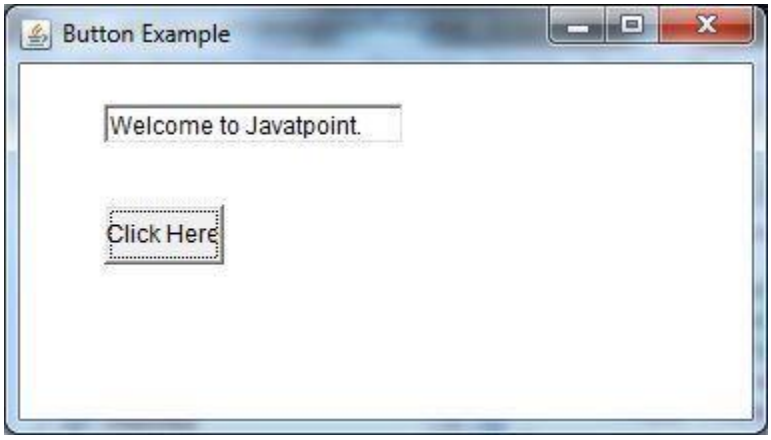
```
import java.awt.*;
import java.awt.event.*;
class AEvent extends Frame implements ActionListener{
TextField tf;
AEvent(){

//create components
tf=new TextField();
tf.setBounds(60,50,170,20);
Button b=new Button("click me");
b.setBounds(100,120,80,30);

//register listener
b.addActionListener(this);//passing current instance

//add components and set size, layout and visibility
add(b);add(tf);
setSize(300,300);
setLayout(null);
setVisible(true);
}
public void actionPerformed(ActionEvent e){
tf.setText("Welcome");
}
public static void main(String args[]){
new AEvent();
}
}
```

OUTPUT – 1.1

1.2 Implementation of Action Listener Class (On Button Click).

```
import java.awt.*;
import java.awt.event.*;
//1st step
public class ActionListenerExample implements ActionListener{
public static void main(String[] args) {
    Frame f=new Frame("ActionListener Example");
    final TextField tf=new TextField();
    tf.setBounds(50,50, 150,20);
    Button b=new Button("Click Here");
    b.setBounds(50,100,60,30);
    //2nd step
    b.addActionListener(this);
    f.add(b);f.add(tf);
    f.setSize(400,400);
    f.setLayout(null);
    f.setVisible(true);
}
//3rd step
public void actionPerformed(ActionEvent e){
        tf.setText("Welcome to Javatpoint.");
}
}
```

OUTPUT – 1.2

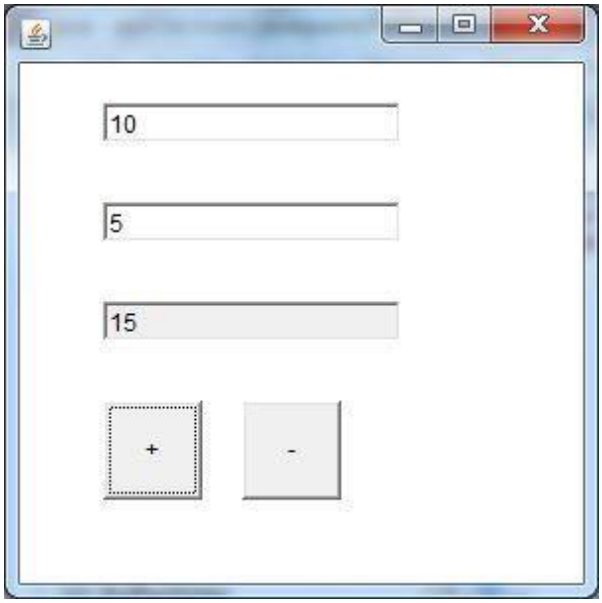


1.3 Action listener in Text Field (Addition Substraction).

```
import java.awt.*;
import java.awt.event.*;
public class TextFieldExample extends Frame implements ActionListener{
   TextField tf1,tf2,tf3;
   Button b1,b2;
   TextFieldExample(){
      tf1=new TextField();
      tf1.setBounds(50,50,150,20);
      tf2=new TextField();
      tf2.setBounds(50,100,150,20);
      tf3=new TextField();
      tf3.setBounds(50,150,150,20);
      tf3.setEditable(false);
      b1=new Button("+");
      b1.setBounds(50,200,50,50);
      b2=new Button("-");
      b2.setBounds(120,200,50,50);
      b1.addActionListener(this);
      b2.addActionListener(this);
      add(tf1);add(tf2);add(tf3);add(b1);add(b2);
      setSize(300,300);
      setLayout(null);
      setVisible(true);
   }
   public void actionPerformed(ActionEvent e) {
      String s1=tf1.getText();
      String s2=tf2.getText();
      int a=Integer.parseInt(s1);
      int b=Integer.parseInt(s2);
      int c=0;
      if(e.getSource()==b1){
         c=a+b;
      }else if(e.getSource()==b2){
         c=a-b;
      }
```

```
        String result=String.valueOf(c);
        tf3.setText(result);
        }
public static void main(String[] args) {
    new TextFieldExample();
}
}
```

OUTPUT – 1.3



Sr. no	VIVA QUESTIONS (1)
1.	**What Are Awt Peers?**
	A component is associated with a standard AWT button object, a peer object and an interfacing button object constructed per the native GUI.
2.	**What Are The Different Types Of Controls In Awt?**
	The AWT supports the following types of controls: o Labels, Pushbuttons, Checkboxes, Choice lists, Lists, Scroll bars, Text components o These controls are subclasses of component.
3.	**What Is The Use Of The Window Class?**
	The window class can be used to create a plain, bare bones window that does not have a border or menu. The window class can also be used to display introduction or welcome screens.
4.	**What Interface Is Extended By Awt Event Listener?**
	The java.util.EventListener interface is extended by all the AWT event listeners.
5.	**Which Container May Contain A Menu Bar?**
	Frame
6.	**What is AWT?**
	AWT stands for Abstract Window Toolkit and infact it is a package – **java.awt**. The classes of this package give the power of creating a GUI environment to the programmer in Java.

7.	**What is the method used to change the background color of components like text field?**
	setBackground(Color clr) of **java.awt.Component** class.
8.	**What is the purpose of using Event Handlers?**
	Any UI based application would have to respond to user actions. For ex: If you click on the Save button in a web page, you would expect the system to save the values you entered in the screen. Similarly there might be numerous actions that might need to be captured and handled by the system. Event handlers help us with that. Capturing user button clicks, opening of new pages etc.
9.	**What class is the top of the AWT event hierarchy?**
	The java.awt.AWTEvent class is the highest-level class in the AWT event-class hierarchy.
10.	**In which package are most of the AWT events that support the event-delegation model defined?**
	Most of the AWT-related events of the event-delegation model are defined in the java.awt.event package. The AWTEvent class is defined in the java.awt package.

PROGRAM - 2:

2.1 Write a program in Java to implement Calculator using Swing Technology.

```
import java.awt.event.*;
import javax.swing.*;
import java.awt.*;
class calculator extends JFrame implements ActionListener {

        // create a frame
        static JFrame f;


        // create a textfield
        static JTextField l;


        // store oprerator and operands
        String s0, s1, s2;


        // default constrcutor
        calculator()
        {
                s0 = s1 = s2 = "";
        }

        // main function

        public static void main(String args[])
        {
                // create a frame
                f = new JFrame("calculator");

                try {
                        // set look and feel
                        UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());
```

```
}

catch (Exception e) {

        System.err.println(e.getMessage());

}

// create a object of class
calculator c = new calculator();


// create a textfield
l = new JTextField(16);


// set the textfield to non editable
l.setEditable(false);


// create number buttons and some operators
JButton b0, b1, b2, b3, b4, b5, b6, b7, b8, b9, ba, bs, bd, bm, be, beq, beq1;


// create number buttons
b0 = new JButton("0");
b1 = new JButton("1");
b2 = new JButton("2");
b3 = new JButton("3");
b4 = new JButton("4");
b5 = new JButton("5");
b6 = new JButton("6");
b7 = new JButton("7");
b8 = new JButton("8");
b9 = new JButton("9");
```

```
// equals button
beq1 = new JButton("=");

// create operator buttons
ba = new JButton("+");
bs = new JButton("-");
bd = new JButton("/");
bm = new JButton("*");
beq = new JButton("C");


// create . button
be = new JButton(".");


// create a panel
JPanel p = new JPanel();


// add action listeners
bm.addActionListener(c);
bd.addActionListener(c);
bs.addActionListener(c);
ba.addActionListener(c);
b9.addActionListener(c);
b8.addActionListener(c);
b7.addActionListener(c);
b6.addActionListener(c);
b5.addActionListener(c);
b4.addActionListener(c);
b3.addActionListener(c);
b2.addActionListener(c);
b1.addActionListener(c);
b0.addActionListener(c);
be.addActionListener(c);
beq.addActionListener(c);
beq1.addActionListener(c);
```

```
        // add elements to panel
        p.add(l);
        p.add(ba);
        p.add(b1);
        p.add(b2);
        p.add(b3);
        p.add(bs);
        p.add(b4);
        p.add(b5);
        p.add(b6);
        p.add(bm);
        p.add(b7);
        p.add(b8);
        p.add(b9);
        p.add(bd);
        p.add(be);
        p.add(b0);
        p.add(beq);
        p.add(beq1);

        // set Background of panel
        p.setBackground(Color.blue);

        // add panel to frame
        f.add(p);

        f.setSize(200, 220);
        f.show();
    }

    public void actionPerformed(ActionEvent e)

    {

    String s = e.getActionCommand();
```

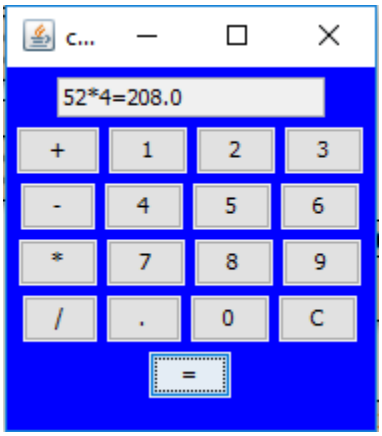
```
// if the value is a number

if ((s.charAt(0) >= '0' && s.charAt(0) <= '9') || s.charAt(0) == '.') {

        // if operand is present then add to second no
        if (!s1.equals(""))
                s2 = s2 + s;

        else

                s0 = s0 + s;



        // set the value of text
        l.setText(s0 + s1 + s2);
}

else if (s.charAt(0) == 'C') {

        // clear the one letter
        s0 = s1 = s2 = "";


        // set the value of text
        l.setText(s0 + s1 + s2);
}

else if (s.charAt(0) == '=') {


        double te;

        // store the value in 1st
        if (s1.equals("+"))
                te = (Double.parseDouble(s0) + Double.parseDouble(s2));
        else if (s1.equals("-"))
```

```
                te = (Double.parseDouble(s0) - Double.parseDouble(s2));
        else if (s1.equals("/"))
                te = (Double.parseDouble(s0) / Double.parseDouble(s2));

        else

                te = (Double.parseDouble(s0) * Double.parseDouble(s2));

        // set the value of text
        l.setText(s0 + s1 + s2 + "=" + te);


        // convert it to string
        s0 = Double.toString(te);

        s1 = s2 = "";

}

else {
        // if there was no operand

        if (s1.equals("") || s2.equals(""))
                s1 = s;
        // else evaluate

        else {
```

```
                double te;

                // store the value in 1st
                if (s1.equals("+"))
                        te = (Double.parseDouble(s0) + Double.parseDouble(s2));
                else if (s1.equals("-"))
                        te = (Double.parseDouble(s0) - Double.parseDouble(s2));
                else if (s1.equals("/"))
                        te = (Double.parseDouble(s0) / Double.parseDouble(s2));

                else

                        te = (Double.parseDouble(s0) * Double.parseDouble(s2));



                // convert it to string

                s0 = Double.toString(te);



                // place the operator
                s1 = s;


                // make the operand blank
                s2 = "";
        }


        // set the value of text
        l.setText(s0 + s1 + s2);
}
```

```
        }
}
```

OUTPUT – 2



Sr. no	VIVA QUESTIONS (2)
1.	**What is Event-Dispatcher-Thread (EDT) in Swing?**
	The Event Dispatcher Thread or EDT is a special thread in Swing and AWT. The Event-Driven Thread is used to render graphics and listen for events in Swing.
2.	**Does Swing is thread-safe? What do you mean by swing is not thread-safe?**
	Since Swing components are not thread-safe it means you can not update these components in any thread other than Event-Dispatcher-Thread. If you do so you will see unexpected behavior e.g. freezing GUI due to the deadlock of incorrect values etc.
3.	**What are differences between Swing and AWT?**
	· The AWT component are considered to be **heavyweight** while Swing component are considered **lightweights** · The Swing GUI has pluggable look and feel.
4.	**What's Java Swing?**
	Swing is a GUI toolkit for Java. It is one part of the Java Foundation Classes (JFC). Swing includes graphical user interface (GUI) widgets such as text boxes, buttons, split-panes, and tables.
5.	**What Is Jfc?**
	JFC stands for Java Foundation Classes. The Java Foundation Classes (JFC) are a set of Java class libraries provided as part of Java 2 Platform, Standard Edition (J2SE) to support building graphics user interface (GUI) and graphics functionality for client applications that will run on popular platforms such as Microsoft Windows, Linux, and Mac OSX.
6.	**What Are Heavyweight Components?**

	A heavyweight component is one that is associated with its own native screen
7.	**What Is Lightweight Component?**
	A lightweight component is one that "borrows" the screen resource of an ancestor (which means it has no native resource of its own -- so it's "lighter").
8.	**What Is Double Buffering?**
	Double buffering is the process of use of two buffers rather than one to temporarily hold data being moved to and from an I/O device. Double buffering increases data transfer speed because one buffer can be filled while the other is being emptied.
9.	**What Is An Event In Swing?**
	Changing the state of an object is called an event.
10.	**What Is An Event Handler In Swing?**
	An event handler is a part of a computer program created to tell the program how to act in response to a specific event.

PROGRAM – 3

3.1 Write a Java program that makes a Connection with database using JDBC and prints metadata of this connection

```
import java.sql.*;
class MysqlCon{
public static void main(String args[]){
try{
Class.forName("com.mysql.jdbc.Driver");
Connection con=DriverManager.getConnection(
"jdbc:mysql://localhost:3306/sonoo","root","root");
//here sonoo is database name, root is username and password
Statement stmt=con.createStatement();
ResultSet rs=stmt.executeQuery("select * from emp");
while(rs.next())
System.out.println(rs.getInt(1)+" "+rs.getString(2)+" "+rs.getString(3));
con.close();
}
catch(Exception e){ System.out.println(e);}
}
}
```

3.2 Include the database Connectivity in the program to insert, update, delete and display of information.

```
package com.devdaily.sqlprocessortests;

import java.sql.*;

public class BasicJDBCDemo

{

 Connection conn;

 public static void main(String[] args)

 {

  new BasicJDBCDemo();

 }

 public BasicJDBCDemo()

 {

  try{

    Class.forName("com.mysql.jdbc.Driver").newInstance();

    String url = "jdbc:mysql://localhost/coffeebreak";

    conn = DriverManager.getConnection(url, "username", "password");

    doTests();

    conn.close();}

  catch (ClassNotFoundException ex) {System.err.println(ex.getMessage());}

  catch (IllegalAccessException ex) {System.err.println(ex.getMessage());}

  catch (InstantiationException ex) {System.err.println(ex.getMessage());}

  catch (SQLException ex)          {System.err.println(ex.getMessage());}

 }

 private void doTests()
```

```
{
 doSelectTest();
 doInsertTest(); doSelectTest();
 doUpdateTest(); doSelectTest();
 doDeleteTest(); doSelectTest();
}
private void doSelectTest()
{
 System.out.println("[OUTPUT FROM SELECT]");
 String query = "SELECT COF_NAME, PRICE FROM COFFEES";
 try
 {
  Statement st = conn.createStatement();
  ResultSet rs = st.executeQuery(query);
  while (rs.next())
  {
   String s = rs.getString("COF_NAME");
   float n = rs.getFloat("PRICE");
   System.out.println(s + " " + n);
  }
 }
 catch (SQLException ex)
 {
  System.err.println(ex.getMessage());
 }}
```

```
 private void doInsertTest()
 {
  System.out.print("\n[Performing INSERT] ... ");
  try
  {
   Statement st = conn.createStatement();
   st.executeUpdate("INSERT INTO COFFEES " +
                    "VALUES ('BREAKFAST BLEND', 200, 7.99, 0, 0)");
  }
  catch (SQLException ex)
  {
   System.err.println(ex.getMessage());
  }
 }
 private void doUpdateTest()
 {
  System.out.print("\n[Performing UPDATE] ... ");
  try
  {
   Statement st = conn.createStatement();
   st.executeUpdate("UPDATE COFFEES SET PRICE=4.99 WHERE
COF_NAME='BREAKFAST BLEND'");
  }
  catch (SQLException ex){
   System.err.println(ex.getMessage());}
```

```
  }
  private void doDeleteTest()
  {
    System.out.print("\n[Performing DELETE] ... ");
    try
    {
      Statement st = conn.createStatement();
     st.executeUpdate("DELETE FROM COFFEES WHERE COF_NAME='BREAKFAST
BLEND'");
    }
    catch (SQLException ex)
    {
      System.err.println(ex.getMessage());
    }
  }
}
```

Sr. no	VIVA QUESTIONS (3)
1.	**What is JDBC?**
	JDBC is a Java API that is used to connect and execute the query to the database. JDBC API uses JDBC drivers to connect to the database. JDBC API can be used to access tabular data stored into any relational database.
2.	**What is JDBC Driver?**
	JDBC Driver is a software component that enables Java application to interact with the database.
3.	**What is the return type of Class.forName() method?**
	The Class.forName() method returns the object of java.lang.Class object.
4.	**How can we set null value in JDBC PreparedStatement?**
	By using setNull() method of PreparedStatement interface, we can set the null value to an index. The syntax of the method is given below. **void** setNull(**int** parameterIndex, **int** sqlType) **throws** SQLException
5.	**What is the role of the JDBC DriverManager class?**
	The DriverManager class acts as an interface between user and drivers. It keeps track of the drivers that are available and handles establishing a connection between a database and the appropriate driver. The DriverManager class maintains a list of Driver classes that have registered themselves by calling the method DriverManager.registerDriver().
6.	**What is the major difference between java.util.Date and java.sql.Date data type?**
	The major difference between java.util.Date and java.sql.Date is that, java.sql.Date represents date without time information whereas, java.util.Date represents both date and time information.

7.	**What does JDBC setMaxRows method do?**
	The setMaxRows(int i) method limits the number of rows the database can return by using the query. This can also be done within the query as we can use the limit cause in MySQL.
8.	**What is connection pooling?**
	This is also one of the most popular question asked during JDBC Interviews. Connection pooling is the mechanism by which we reuse the recourse like connection objects which are needed to make connection with database .In this mechanism client are not required every time make new connection and then interact with database instead of that connection objects are stored in connection pool and client will get it from there. so it's a best way to share a server resources among the client and enhance the application performance.
9.	**Does the JDBC-ODBC Bridge support multiple concurrent open statements per connection?**
	No, we can open only one statement object when using JDBC-ODBC Bridge.
10.	**What Does The Connection Object Represents?**
	The connection object represents communication context, i.e., all communication with database is through connection object only.

PROGRAM - 4:

4.1 Write a java program for two way TCP communication for server and client. It should look like a simple chat application.

Client program: GossipClient.java

```
import java.io.*;
import java.net.*;
public class GossipClient
{
 public static void main(String[] args) throws Exception
 {
   Socket sock = new Socket("127.0.0.1", 3000);
                    // reading from keyboard (keyRead object)
   BufferedReader keyRead = new BufferedReader(new InputStreamReader(System.in));
                    // sending to client (pwrite object)
   OutputStream ostream = sock.getOutputStream();
   PrintWriter pwrite = new PrintWriter(ostream, true);


                    // receiving from server ( receiveRead object)
   InputStream istream = sock.getInputStream();
   BufferedReader receiveRead = new BufferedReader(new InputStreamReader(istream));
   System.out.println("Start the chitchat, type and press Enter key");

   String receiveMessage, sendMessage;
   while(true)
```

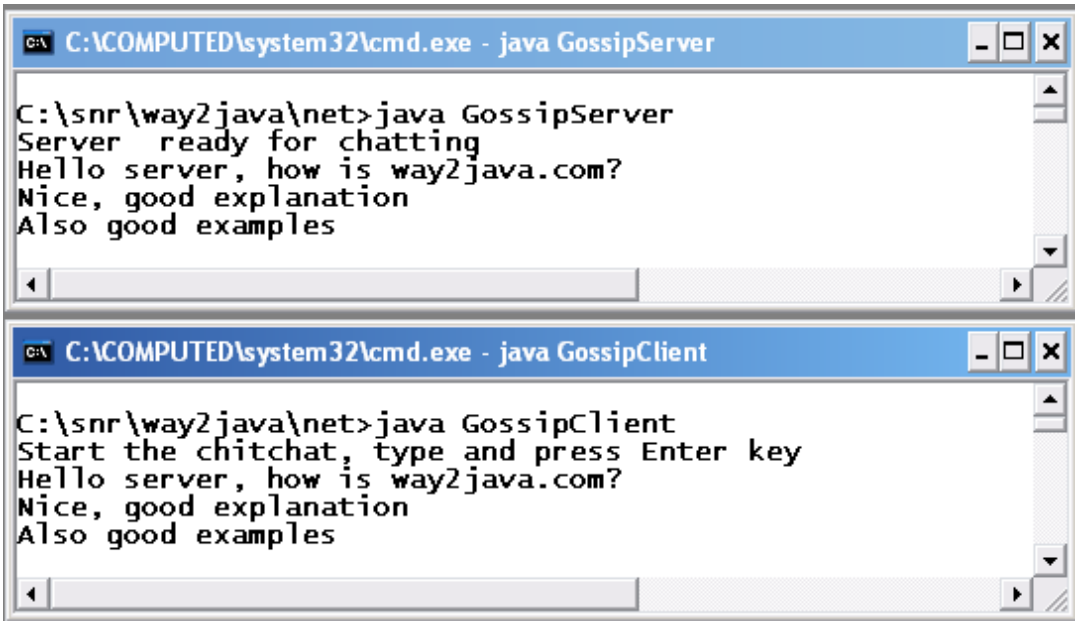
```
{ sendMessage = keyRead.readLine(); // keyboard
reading pwrite.println(sendMessage); // sending to
server pwrite.flush();          // flush the data
if((receiveMessage = receiveRead.readLine()) != null) //receive from server
{
   System.out.println(receiveMessage); // displaying at DOS prompt
}}
}}
```

Server program: GossipServer.JAVAimport JAVA.io.*;

```
import java.io.*;
import java.net.*;
public class GossipServer{
  public static void main(String[] args) throws Exception{
    ServerSocket sersock = new ServerSocket(3000);
    System.out.println("Server ready for chatting");
    Socket sock = sersock.accept( );
                    // reading from keyboard (keyRead object)
    BufferedReader keyRead = new BufferedReader(new InputStreamReader(System.in));
                      // sending to client (pwrite object)
                      OutputStream ostream =
    sock.getOutputStream(); PrintWriter pwrite = new
      PrintWriter(ostream, true);
```

```
                    // receiving from server ( receiveRead object)
    InputStream istream = sock.getInputStream();
    BufferedReader receiveRead = new BufferedReader(new InputStreamReader(istream));


    String receiveMessage, sendMessage;
    while(true)
    {
     if((receiveMessage = receiveRead.readLine()) != null)
     {
       System.out.println(receiveMessage);
     }
     sendMessage = keyRead.readLine();
     pwrite.println(sendMessage);
     pwrite.flush();
    }
  }
}
```

OUTPUT – 4


C:\COMPUTED\system32\cmd.exe - java GossipServer
C:\snr\way2java\net>java GossipServer
Server ready for chatting
Hello server, how is way2java.com?
Nice, good explanation
Also good examples
C:\COMPUTED\system32\cmd.exe - java GossipClient
C:\snr\way2java\net>java GossipClient
Start the chitchat, type and press Enter key
Hello server, how is way2java.com?
Nice, good explanation
Also good examples

Sr. no	VIVA QUESTIONS (4)
1.	**What Is Socket?**
	A socket is one end-point of a two-way communication link between two programs running on the network. Socket classes are used to represent the connection between a client program and a server program. The java.net package provides two classes--Socket and ServerSocket--that implement the client side of the connection and the server side of the connection, respectively.
2.	**What Is The Difference Between Tcp And Udp?**
	○ TCP and UDP are both transport-level protocols. TCP is designed to provide reliable communication across a variety of reliable and unreliable networks and internets. ○ UDP provides a connectionless service for application-level procedures. Thus, UDP is basically an unreliable service; delivery and duplicate protection are not guaranteed.
3.	**What Does A Socket Consists Of?**
	The combination of an IP address and a port number.
4.	**What Are Some Advantages Of Java Sockets?**
	○ Sockets are flexible and sufficient. Efficient socket based programming can be easily implemented for general communications. ○ Sockets cause low network traffic. Unlike HTML forms and CGI scripts that generate and transfer whole web pages for each new request, Java applets can send only necessary updated information.
5.	**What Are Some Disadvantages Of Java Sockets?**

	○ Security restrictions are sometimes overbearing because a Java applet running in a Web browser is only able to establish connections to the machine where it came from, and to nowhere else on the network ○ Despite all of the useful and helpful Java features, Socket based communications allows only to send packets of raw data between applications. Both the client-side and server-side have to provide mechanisms to make the data useful in any way.
6.	**What is multiprogramming?**
	Multiprogramming is a rapid switching of the CPU back and forth between processes.
7.	**What happens if ServerSocket is not able to listen on the specified port?**
	public ServerSocket() creates an unbound server socket.It throws IOException if specified port is busy when opening the socket.
8.	**What does bind() method of ServerSocket offer?**
	bind() binds the server socket to a specific address (IP Address and port). If address is null, the system will pick an ephemeral port and valid local address to bind socket.
9.	**Which class represents an Internet Protocol address?**
	InetAddress represents an Internet Protocol address. It provides static methods like getByAddress(), getByName() and other instance methods like getHostName(), getHostAddress(), getLocalHost().
10.	**What happens if IP Address of host cannot be determined?**
	UnknownHostException is thrown when IP Address of host cannot be determined. It is an extension of IOException.

PROGRAM - 5:

Create a simple calculator application that demonstrates the use of RMI. You are not required to create GUI.

We have to run these four classes to make calculator using RMI.

- Calculator.java
- CalculatorImpl.java
- CalculatorClient.java
- CalculatorServer.java

Calculator.java

```
public interface Calculator
        extends java.rmi.Remote {
   public long add(long a, long b)
      throws java.rmi.RemoteException;

   public long sub(long a, long b)
      throws java.rmi.RemoteException;

   public long mul(long a, long b)
      throws java.rmi.RemoteException;

   public long div(long a, long b)
      throws java.rmi.RemoteException;
}
```

CalculatorImpl.java

```
public class CalculatorImpl
    extends
      java.rmi.server.UnicastRemoteObject
    implements Calculator {

    // Implementations must have an
    //explicit constructor
    // in order to declare the
    //RemoteException exception
    public CalculatorImpl()
        throws java.rmi.RemoteException {
        super();
    }

    public long add(long a, long b)
        throws java.rmi.RemoteException {
        return a + b;
    }

    public long sub(long a, long b)
        throws java.rmi.RemoteException {
        return a - b;
    }

    public long mul(long a, long b)
        throws java.rmi.RemoteException {
```

```
    return a * b;
  }

  public long div(long a, long b)
    throws java.rmi.RemoteException {
    return a / b;
  }}
```

CalculatorClient.java

```
import java.rmi.Naming;
import java.rmi.RemoteException;
import java.net.MalformedURLException;
import java.rmi.NotBoundException;

public class CalculatorClient {

  public static void main(String[] args) {
    try {
      Calculator c = (Calculator)
                Naming.lookup(
        "rmi://localhost/CalculatorService");
      System.out.println( c.sub(4, 3) );
      System.out.println( c.add(4, 5) );
      System.out.println( c.mul(3, 6) );
      System.out.println( c.div(9, 3) );
```

```
}
catch (MalformedURLException murle) {
   System.out.println();
   System.out.println(
   "MalformedURLException");
   System.out.println(murle);
}
catch (RemoteException re) {
   System.out.println();
   System.out.println(
            "RemoteException");
   System.out.println(re);
}
catch (NotBoundException nbe) {
   System.out.println();
   System.out.println(
           "NotBoundException");
   System.out.println(nbe);
}
catch (
   java.lang.ArithmeticException
                       ae) {
   System.out.println();
   System.out.println(
   "java.lang.ArithmeticException");
   System.out.println(ae);}}
```

CalculatorServer.java

```
import java.rmi.Naming;

public class CalculatorServer {

  public CalculatorServer() {
    try {
      Calculator c = new CalculatorImpl();
      Naming.rebind("rmi://localhost:1099/CalculatorService", c);
    } catch (Exception e) {
      System.out.println("Trouble: " + e);
    }
  }

  public static void main(String args[]) {
    new CalculatorServer();
  }
}

//Now use rmic to create the stub and skeleton class files.
 > rmic CalculatorImpl
```

Running the RMI System

You are now ready to run the system! You need to start three consoles, one for the server, one for the client, and one for the RMIRegistry.

- *Start with the Registry. You must be in the directory that contains the classes you have written. From there, enter the following:*

  > rmiregistry

- *If all goes well, the registry will start running and you can switch to the next console.*

In the second console start the server hosting the CalculatorService, and enter the following:

> java CalculatorServer

- *It will start, load the implementation into memory and wait for a client connection.*

In the last console, start the client program.

> java CalculatorClient//

OUTPUT – 5

1

9

18

3

Sr. no	VIVA QUESTIONS (5)
1.	**What is Java Remote Method Invocation (RMI)?**
	The Remote Method Invocation (Java RMI) is an API that provides a mechanism to create distributed application in java. The RMI allows an object to invoke methods on an object running in another JVM.The RMI provides remote communication between the applications usingtwo objects **stub** and **skeleton**.
2.	**What is the principle of RMI architecture**
	The principle of RMI states that "definition of the behavior and the implementation of that behavior as separate concepts, RMI allows the code that defines the behavior and the code that implements the behavior to remain separate and to run on separate JVMs".
3.	**What is RMI remote object?**
	A remote object is a Java object whose method can be invoked from another JVM. RMI uses stub and skeleton object for communication with the remote object.
4.	**How does RMI communicate with the remote object?**
	RMI uses stub and skeleton object for communication with the remote object. The stub is an object, acts as a gateway to the client. It resides the client and acts as a proxy for the remote object. The skeleton is an object, acts as a gateway to the server. All the client incoming requests are routed through the skeleton.

5.	**What are different layers of RMI architecture.**
	Stub and skeleton layer is responsible for intercepting method calls on the client interface and redirect it to remote RMI service method. **Remote reference layer** manages and interprets the references from client to remote objects.
6.	**What is remote interface in RMI.**
	Any Java interface that extends java.rmi.Remote is a remote interface. Remote interface methods can be invoked from another JVM and the object implementing this remote interface becomes remote object.
7.	**Which package is used for remote method invocation?**
	java.rmi
8.	**How many types of protocol implementations does RMI have?**
	3
9.	**In RMI Distributed object applications need to do?**
	Remote objects, Communicate with remote objects, load class definitions for objects that are passed around
10.	**In RMI applications which program obtains a remote reference to one or more remote objects on a server and then invokes methods on them?**
	Client

Program-6

6.1 Create Servlet that prints “Hello World”

```
// Import required java libraries
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

 // Extend HttpServlet classpublic class HelloWorld extends HttpServlet {
  private String message;
  public void init() throws ServletException {

// Do required initialization
    message = "Hello World";
  }
  public void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {

    // Set response content type
    response.setContentType("text/html");


    // Actual logic goes here.
    PrintWriter out = response.getWriter();
    out.println("<h1>" + message + "</h1>");
  }
  public void destroy() {
    // do nothing.}}
```

6.2 Create Servlet that prints Today's Date.

```
import java.io.*;
import javax.servlet.*;

public class DateSrv extends GenericServlet
{
    //implement service()
    public void service(ServletRequest req, ServletResponse res) throws IOException,
ServletException
    {
        //set response content type
        res.setContentType("text/html");
        //get stream obj
        PrintWriter pw = res.getWriter();
        //write req processing logic
        java.util.Date date = new java.util.Date();
        pw.println("<h2>"+"Current Date & Time: " +date.toString()+"</h2>");
        //close stream object
        pw.close();
    }
}
```

OUTPUT – 6.2

Current Date & Time: Mon Dec 12 18:01:39 IST 2016

6.3 Create Servlet for login page, if the username and password is correct then print message "Hello Username" else a message "Login Failed"

index.html

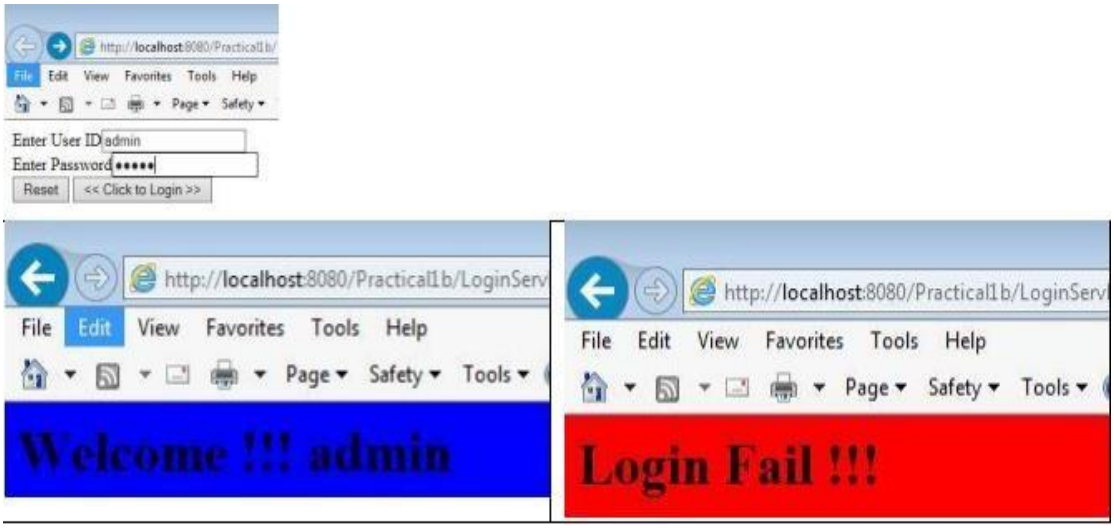
```
<html><head><title>Login Form</title></head>
<form action="LoginServlet" >
Enter User ID<input type="text" name="txtId"><br>
Enter Password<input type="password" name="txtPass"><br>
 <input type="reset"><input type="submit" value=" Click to Login " ></form></html>
```

LoginServlet.java

```
package mypack;
import java.io.*;
import javax.servlet.ServletException;
import javax.servlet.http.*;
public class LoginServlet extends HttpServlet {
public void doGet(HttpServletRequest request, HttpServletResponse response)
                    throws ServletException, IOException {
                response.setContentType("text/html;charset=UTF-8");
                 PrintWriter out = response.getWriter();
                 out.println("<html><head><title>Servlet LoginServlet</title></head>");
                 String uname = request.getParameter("txtId");
```

```
String upass = request.getParameter("txtPass");
if(uname.equals("admin") && upass.equals("12345")){
out.println("<body bgcolor=blue>");
out.println("<h1> Welcome !!!"+uname+"</h1>");
}
else
{
out.println("<body bgcolor=red>");
out.println("<h1> Login Fail !!! </h1>");
}
out.println("</body></html>");
}}
```

OUTPUT 6.3



Sr. no	VIVA QUESTIONS (6)
1.	**What is a servlet?**
	Java Servlet is server side technologies to extend the capability of web servers by providing support for dynamic response and data persistence. The javax.servlet and javax.servlet.http packages provide interfaces and classes for writing our own servlets.
2.	**How constructor can be used for a servlet?**
	We cannot declare constructors for interface in Java. This means we cannot enforce this requirement to any class which implements Servlet interface. Also, Servlet requires ServletConfig object for initialization which is created by container.
3.	**Can servlet class declare constructor with ServletConfig object as an argument?**
	ServletConfig object is created after the constructor is called and before init() is called. So, servlet init parameters cannot be accessed in the constructor.
4.	**What is the difference between servlets and applets?**
	Servlets execute on Server and doesn't have GUI. Applets execute on browser and has GUI.
5.	**Which code is used to get an attribute in a HTTP Session object in servlets?**
	session.getAttribute(String name)
6.	**What are the functions of Servlet container?**
	Lifecycle management Communication support Multithreading support

7.	**Can we get PrintWriter and ServletOutputStream both in a servlet?**
	We can't get instances of both PrintWriter and ServletOutputStream in a single servlet method, if we invoke both the methods; getWriter() and getOutputStream() on response; we will get java.lang.IllegalStateException at runtime with message as other method has already been called for this response.
8.	**When servlet object is created?**
	At the time of first request.
9.	**What is difference between GenericServlet and HttpServlet?**
	The GenericServlet is protocol independent whereas HttpServlet is HTTP protocol specific. HttpServlet provides additional functionalities such as state management etc.
10.	**What is difference between ServletConfig and ServletContext?**
	The container creates object of ServletConfig for each servlet whereas object of ServletContext is created for each web application.

Program 7

7.1 Create a servlet that implements ServletContextAttributeListener interface such that a message dialog is displayed whenever an attribute is added or removed or replaced.

Step 1: Create a dynamic web project

Create a new dynamic web project .

Step 2: Create ServletRequestAttributeListener and configure in web.xml

Create AttributeListener class in your web application as follows.

```
package com.thejavageek;
import javax.servlet.ServletRequestAttributeEvent;
import javax.servlet.ServletRequestAttributeListener;
public class AttributeListener implements ServletRequestAttributeListener {

public void attributeAdded(ServletRequestAttributeEvent event) {
System.out.println("Attribute added : " + event.getName() + " = "
+ event.getValue());
}

public void attributeRemoved(ServletRequestAttributeEvent event) {
System.out.println("Attribute removed : " + event.getName() + " = "
+ event.getValue());
}
```

```
public void attributeReplaced(ServletRequestAttributeEvent event) {
System.out.println("Attribute replaced : " + event.getName() + " = "
+ event.getValue());
}
}
```

//Configure in web.xml as follows:

```
<listener>
    <listener-class>com.thejavageek.AttributeListener</listener-class>
</listener>
```

Step 3: Create HelloServlet.

To listen to attribute change events, we must add, remove and replace them somewhere. Create a HelloServlet which will perform operations on attributes.

```
package com.thejavageek;
import java.io.IOException;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
public class HelloServlet extends HttpServlet {
private static final long serialVersionUID = 1L;
protected void doGet(HttpServletRequest request,
```

```
HttpServletResponse response) throws ServletException, IOException {
request.setAttribute("firstName", "prasad");
request.setAttribute("firstName", "pranil");
request.removeAttribute("firstName");
}
}
```

Configure HelloServlet in web.xml

```
<servlet>
  <description></description>
  <display-name>HelloServlet</display-name>
  <servlet-name>HelloServlet</servlet-name>
  <servlet-class>com.thejavageek.HelloServlet</servlet-class>
</servlet>
<servlet-mapping>
  <servlet-name>HelloServlet</servlet-name>
  <url-pattern>/HelloServlet</url-pattern>
</servlet-mapping>
```

Step 4: Run application.

ow simply deploy the application under tomcat server and start tomcat. Hit url http://localhost:8080/ServletListenerExamples/HelloServlet.

OUTPUT 7

Attribute added : firstName = prasad

Attribute replaced : firstName = prasad

Attribute removed : firstName = pranil

Sr. no	VIVA QUESTIONS (7)
1.	**What is a ServletContextAttributeListener ?**
	The listener **ServletContextAttributeListener** is an interface extending from base interface java.util.EventListener.
2.	**What is servlet attributes and their scope?**
	Servlet attributes are used for inter-servlet communication, we can set, get and remove attributes in web application. There are three scopes for servlet attributes – request scope, session scope and application scope.
3.	**How do we call one servlet from another servlet?**
	We can use RequestDispatcher forward() method to forward the processing of a request to another servlet. If we want to include the another servlet output to the response, we can use RequestDispatcher include() method.
4.	**Why do we have servlet filters?**
	Servlet Filters are pluggable java components that we can use to intercept
5.	**How to get the actual path of servlet in server?**
	We can use following code snippet to get the actual path of the servlet in file system. getServletContext().getRealPath(request.getServletPath())
6.	**How to get the server information in a servlet?**
	We can use below code snippet to get the servlet information in a servlet through servlet context object. getServletContext().getServerInfo()
7.	**What are different ways for servlet authentication?**
	Servlet Container provides different ways of login based servlet authentication: **HTTP Basic Authentication** **HTTP Digest Authentication**

	HTTPS Authentication **Form Based Login**:
8.	**What Is A War File?**
	WAR stands for Web Archive. It is a compressed version of your web application. You can use this WAR file to deploy your web application.
9.	**How Would You Create Deadlock On Your Servlet?**
	Calling a doPost() method inside doGet() and doGet()method inside doPost() wouleate a deadlock for a servlet.
10.	**Why Is A Constructor Needed In A Servlet Even If We Use The Init Method?**
	1. Although the init method of the servlet initializes it, a constructor instantiates it. 2. A developer might never explicitly call the servlet's constructor but a container uses it to create an instance of the servlet.

Program 8

Create a JSP that prints "Hello World".

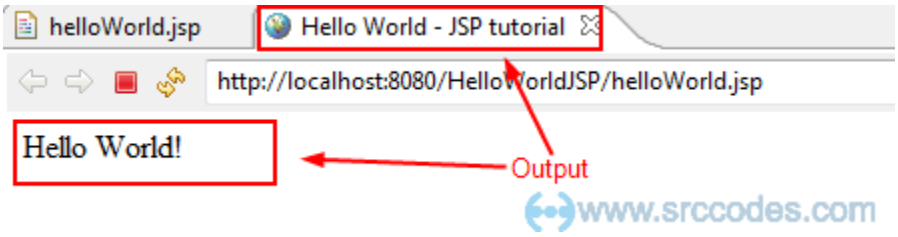
Create a JSP that implements Decision –Making Statement.

Create a JSP that add and subtract two numbers.

8.1 Create a JSP that prints "Hello World".

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Hello World - JSP tutorial</title>
</head>
<body>
    <%= "Hello World!" %>
</body>
</html>
```

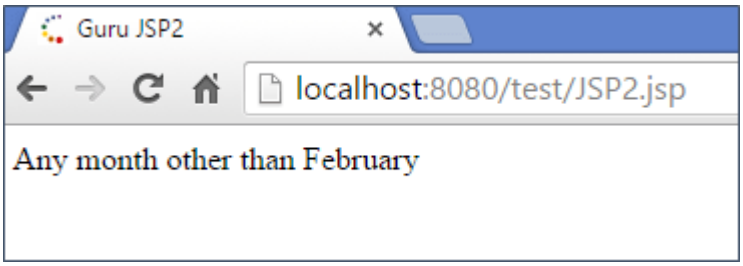
OUTPUT 8.1


helloWorld.jsp
Hello World - JSP tutorial
http://localhost:8080/HelloWorldJSP/helloWorld.jsp
Hello World!
Output
www.srccodes.com

8.2 Create a JSP that implements Decision –Making Statement.

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Guru JSP2</title>
</head>
<body>
<%! int month=5; %>
<% if(month==2){ %>
<a>Its February</a>
<% }else{ %>
<a>Any month other than February</a>
<%} %>
</body>
</html>
```

OUTPUT 8.2


Guru JSP2
localhost:8080/test/JSP2.jsp
Any month other than February

8.3 Create a JSP that add and subtract two numbers.

index.html

```
<html>
   <head>
      <title>Enter two numbers to add up</title>
   </head>

   <body>
      <form action="./add.jsp">
         First number: <input type="text" name="t1"/>
         Second number: <input type="text" name="t2"/>
         <input type="submit" value="SUBMIT" />
      </form>
   </body>
</html>
```

add.jsp

```
<html>
   <head>
      <title>Enter two numbers to add up</title>
   </head>

   <body>
   <%= "<h1> The sum is
"+(Integer.parseInt(request.getParameter("t1"))+Integer.parseInt(request.getParameter("t2")))+"
</h1>"%>
   </body>
</html>
```

web.xml

```
<web-app></web-app>
```

OUTPUT 8.3

java-demos.blogspot.com

java-demos.blogspot.com

Sr. no	VIVA QUESTIONS (8)
1.	**Which page directive should be used in JSP to generate a PDF page?**
	<%page contentType="application/pdf"> tag is used in JSP to generate PDF.
2.	**Is_jspService() method of HttpJspPage class should not be overridden.**
	_jspService() method is created by JSP container. Hence, it should not be overridden.
3.	**Which is the correct order of phases in JSP life cycle?**
	The correct order is Compilation, Initialization, Execution, Cleanup.
4.	**Application is instance of which class?**
	Application object is wrapper around the ServletContext object and it is an instance of a javax.servlet.ServletContext object.
5.	**In JSP page how can we handle runtime exception?**
	This is another popular JSP interview question which has asked to check how candidate used to handle Error and Exception in JSP. We can use the errorPage attribute of the page directive to have uncaught run-time exceptions automatically forwarded to an error processing page.
6.	**Which JSP lifecycle methods can be overridden?**
	We can override jspInit() and jspDestroy() methods using JSP declaration scripting element. We should override jspInit() methods to create common resources that we would like to use in JSP service method and override jspDestroy() method to release the common resources.
7.	**Explain JSP Technology.**
	JSP is a standard extension of Java and is defined on top of Servlet extensions. Its goal is to simplify management and creation of dynamic web pages. It is platform-independent, secure, and it makes use of Java as a server side scripting language.

8.	**How to include static files in a JSP page?**
	Static pages are always included using JSP include directive. This way the inclusion is performed in the translation phase once. Note that a relative URL must be supplied for file attribute. Although static resources may be included, it is not preferred as each request requires inclusion.
9.	**How can a thread safe JSP page be implemented?**
	It can be done by having them implemented by the SingleThreadModel Interface. Add <%@page isThreadSafe=”false” %> directive in the JSP page.
10.	**Define JSP Scriptlet.**
	It a JSP tag that encloses Java code in JSP pages. Their syntax is <% %>. Code written in scriptlet executes every time the program is run..

Program 9

Create a JSP for login module.

Create a web page that prints 1 to 10 using JSTL.

Create a JSP for login module.

/*FOR LOGIN*/

```
package demotest;

import java.io.IOException;

import javax.servlet.RequestDispatcher;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

/*** Servlet implementation class guru_login*/

public class guru_login extends HttpServlet {

public guru_login() {

super();

// TODO Auto-generated constructor stub

}

protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {

// TODO Auto-generated method stub

String username = request.getParameter("username");

String password = request.getParameter("password");
```

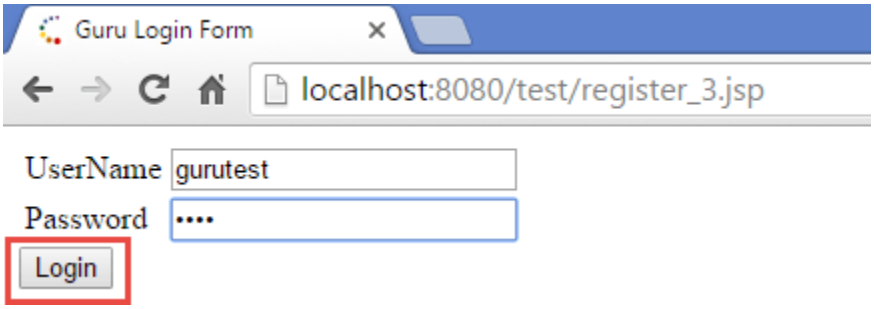
```
if(username.isEmpty() || password.isEmpty() )

{
RequestDispatcher req = request.getRequestDispatcher("register_3.jsp");
req.include(request, response);
}
else
{
RequestDispatcher req = request.getRequestDispatcher("register_4.jsp");
req.forward(request, response);
}
}
}
```

/*FOR LOGOUT*/

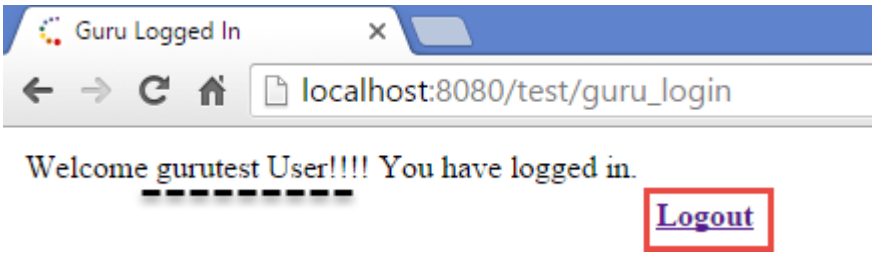
```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

pageEncoding="ISO-8859-1"%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">

<title>Guru Logged In</title>

</head>

<body>

<table style="with: 50%">

<tr><td>

<% String username = request.getParameter("username"); %>

<a>Welcome <% out.println(username); %> User!!!! You have logged in.</a></td></tr>

<tr></tr><tr><td></td><td></td><td><a href="register_3.jsp"><b>Logout</b></a></td></tr>

</table>

</body>

</html>
```

OUTPUT 9.1

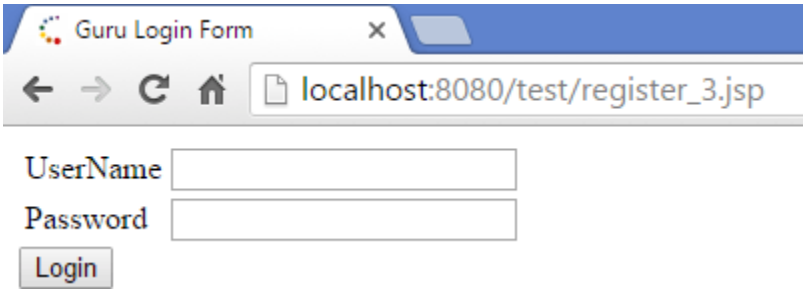
Here when we click on register_3.jsp we get two fields"username" and "password" with a login button.



After clicking on the Login button you get the below message with a button of Logout.



When you click on logout button you go back to login page



9.2 Create a web page that prints 1 to 10 using JSTL.

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Core Tag JSP4</title>
</head>
<body>
<c:forEach var="gurucount" begin="1" end="10">
<c:out value="${gurucount}"/>
</c:forEach>
</body>
</html>
```

OUTPUT 9.2

1 2 3 4 5 6 7 8 9 10

Sr. no	VIVA QUESTIONS (9)
1.	**What is JSP?**
	Java Server Pages technology (JSP) is a server-side programming language used to create a dynamic web page in the form of HyperText Markup Language (HTML). It is an extension to the servlet technology.
2.	**Give the syntax for JSP comments.**
	The syntax for JSP comments is: <%-- Comment --%>
3.	**What is the difference between hide comment and output comment?**
	The JSP comment is called hide comment whereas HTML comment is called output comment. If a user views the source of the page, the JSP comment will not be shown whereas HTML comment will be displayed.
4.	**Is JSP technology extensible?**
	Yes
5.	**How can I implement a thread-safe JSP page?**
	You can make your JSPs thread-safe by having them implement the SingleThreadModel interface. This is done by adding the directive <%@ page isThreadSafe="false" %> within your JSP page.
6.	**How can we handle the exceptions in JSP?**
	There are two ways to perform exception handling, one is by the errorPage element of page directive, and second is by the error-page element of the web.xml file.
7.	**What are the two ways to include the result of another page. ?**
	There are two ways to include the result of another page: o By include directive o By include action

8.	**How can we forward the request from JSP page to the servlet?**
	With the help of "forward action" tag, but we need to give the URL-pattern of the servlet.
9.	**Can we use the exception implicit object in any JSP page?**
	No. The exception implicit object can only be used in the error page which defines it with the isErrorPage attribute of page directive.
10.	**How is JSP used in the MVC model?**
	JSP is usually used for presentation in the MVC pattern (Model View Controller), i.e., it plays the role of the view. The controller deals with calling the model and the business classes which in turn get the data, and this data is then presented to the JSP for rendering on to the client.

Program 10

10.1 Create a custom JSP tag that prints current date and time. Use this tag into JSP page.

// Development of Custom Tag :

To develop a custom tag, you need to perform following steps:

i. Develop a tag handler or class file(.java)

ii. Develop the Tag Library Descriptor (TLD) file

iii. Include the Tag Library in a JSP page

iv. Deploy the application //

//Tag Handler File : Build a class that implements the javax.servlet.jsp.tagext tag interface as follows. Compile it and place it under the web-inf/classes directory (in the appropriate package structure).

ShowDateTag.java

```
package foo;

import java.io.*;

import javax.servlet.jsp.*;
import javax.servlet.jsp.tagext.*;

public class ShowDateTag implements Tag {

 private PageContext pageContext;
 private Tag parent;

 public int doStartTag() throws JspException {
  return SKIP_BODY;
 }

 public int doEndTag() throws JspException {
  try {
   pageContext.getOut().write("" + new java.util.Date());
  } catch (IOException ioe) {
   throw new JspException(ioe.getMessage());}
```

```
  return EVAL_PAGE;
 }

 public void release() {
 }

 public void setPageContext(PageContext page) {
  this.pageContext = page;
 }

 public void setParent(Tag tag) {
  this.parent = tag;
 }

 public Tag getParent() {
  return this.parent;
 }

}
```

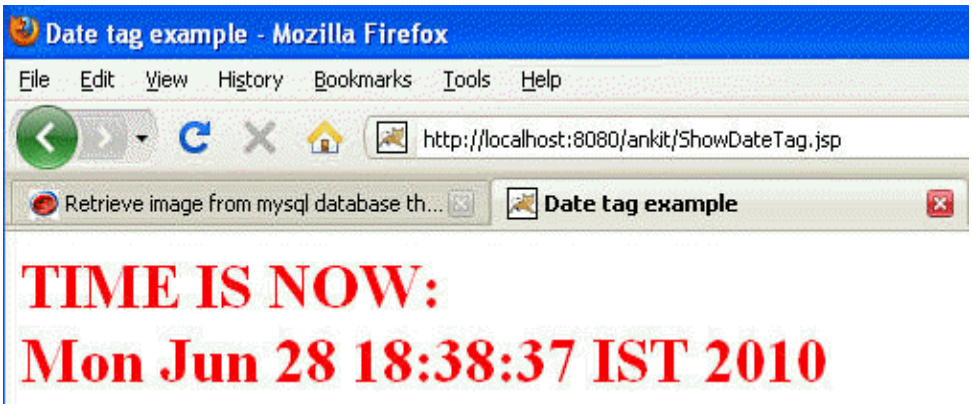
//Tag Library Descriptor (TLD) file : Now we need to describe the tag, so create a file called "taglib.tld" and place it under the web-inf directory.

```
<taglib>
<tlibversion>1.0</tlibversion>
<jspversion>1.1</jspversion>
<shortname>myTag</shortname>
<uri>http://www.roseindia.net/taglib</uri>
<info>My own tag library</info>

<tag>
<name>showDate</name>
<tagclass>foo.ShowDateTag</tagclass>
<info>Show the current date</info>
</tag>

</taglib>
```

JSP file :

```
<html>

<head><title>Date tag example</title></head>

<body>
<h1><font color="red">TIME IS NOW:
<br>
<%@ taglib uri="http://www.roseindia.net/taglib" prefix="myTag" %>

<myTag:showDate/>
</font></h1>
</body>
</html>
```

OUTPUT -10



Sr. no	VIVA QUESTIONS (10)
1.	**What are context initialization parameters?**
	Context initialization parameters are specified by the <context-param> in the web.xml file, and these are initialization parameter for the whole application and not specific to any servlet or JSP.
2.	**What are the different scope values for the <jsp:useBean> tag?**
	There are 4 values: • page • request • session • application
3.	**What do JSP literals consist of?**
	○ Boolean ○ Integer ○ Floating point ○ String ○ Null
4.	**What is the purpose of <jsp:useBean>?**
	Searches for the existence of the object.
5.	**List out the various scope values of JSP action.**
	The possible scope values are: ○ page ○ request ○ session ○ application

6.	**What is the difference in using request.getRequestDispatcher() and context.getRequestDispatcher()?**
	request.getRequestDispatcher(path) is used to create it we need to give the relative path of the resource whereas context.getRequestDispatcher(path)to create it we need to give the absolute path of the resource.
7.	**What is EL in JSP?**
	The Expression Language(EL) is used in JSP to simplify the accessibility of objects. It provides many objects that can be used directly like param, requestScope, sessionScope, applicationScope, request, session, etc.
8.	**Can an interface be implemented in the JSP file?**
	No.
9.	**What is JSTL?**
	JSP Standard Tag Library is a library of predefined tags that ease the development of JSP.
10.	**Which directive is used in JSP custom tag?**
	The JSP taglib directive.