AJP

Exp1-

import javax.swing.JApplet;

import java.awt.Graphics;

import java.awt.event.KeyEvent;

import java.awt.event.KeyListener;

public class Expt1 extends JApplet implements KeyListener {

private String status;

public void init() {

status = "No key pressed";

// Add the KeyListener to the applet

addKeyListener(this);

setFocusable(true);

}

public void paint(Graphics g) {

super.paint(g);

g.drawString("Key Status: " + status, 20, 20);

}

public void keyPressed(KeyEvent e) {

status = "Key Pressed: " + KeyEvent.getKeyText(e.getKeyCode());

repaint();

}

public void keyReleased(KeyEvent e) {

status = "Key Released: " + KeyEvent.getKeyText(e.getKeyCode());

repaint();

}

public void keyTyped(KeyEvent e) {

status = "Key Typed: " + KeyEvent.getKeyText(e.getKeyCode());

repaint();

}

}

Procedure-

Exp-2

import java.awt.\*;

import java.awt.event.MouseEvent;

import java.awt.event.MouseListener;

import javax.swing.JFrame;

public class Expt2 implements MouseListener {

Label l;

JFrame frame;

Expt2() {

frame = new JFrame();

l = new Label();

l.setBounds(25, 60, 280, 30);

l.setAlignment(Label.CENTER);

frame.add(l);

frame.setSize(300, 300);

frame.setLayout(null);

frame.setVisible(true);

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

frame.addMouseListener(this);

}

public static void main(String args[]) {

new Expt2();

}

@Override

public void mouseClicked(MouseEvent var1) {

l.setText("Mouse Clicked");

}

@Override

public void mouseEntered(MouseEvent var1) {

l.setText("Mouse Entered");

}

@Override

public void mouseExited(MouseEvent var1) {

l.setText("Mouse Exited");

}

@Override

public void mousePressed(MouseEvent var1) {

// TODO Auto-generated method stub`

}

@Override

public void mouseReleased(MouseEvent var1) {

// TODO Auto-generated method stub

}

}

Exp-3

package Expt3;

import java.awt.Font;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import javax.swing.JButton;

import javax.swing.JFrame;

import javax.swing.JLabel;

import javax.swing.JTextField;

public class Marksheet extends JFrame implements ActionListener {

JButton button;

JButton button1;

JTextField tf;

JTextField tf1;

JFrame frame1;

JFrame frame;

Marksheet() {

Font font = new Font("Arial", Font.BOLD, 24);

frame = new JFrame();

frame.setLayout(null);

frame.setSize(500, 500);

frame1 = new JFrame();

frame1.setLayout(null);

frame1.setSize(500, 500);

JLabel label = new JLabel();

JLabel label1 = new JLabel();

JLabel label2 = new JLabel();

JLabel label3 = new JLabel();

label3.setText("-----------Report----------");

label3.setBounds(200, 0, 450, 50);

label3.setFont(font);

label3.setAlignmentX(JLabel.CENTER);

tf = new JTextField();

tf1 = new JTextField();

button = new JButton("Print");

button1 = new JButton("Reset");

button.setBounds(100, 220, 100, 30);

button.setFocusable(false);

button1.setBounds(200, 220, 100, 30);

button1.setFocusable(false);

label1.setText("Please Enter the marks of the students");

label1.setBounds(200, 0, 450, 50);

label1.setFont(font);

label1.setAlignmentX(JLabel.CENTER);

label.setBounds(20, 50, 190, 50);

label.setText("Advanced Java Programming");

label2.setBounds(20, 80, 190, 50);

label2.setText("Cellular Networks");

frame.add(label);

frame.add(label1);

frame.add(label2);

frame.add(tf);

frame.add(tf1);

frame.add(button);

frame.add(button1);

frame1.add(label3);

button.addActionListener(this);

button1.addActionListener(this);

tf.setBounds(250, 65, 150, 20);

tf1.setBounds(250, 95, 150, 20);

frame.setVisible(true);

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

frame1.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

}

@Override

public void actionPerformed(ActionEvent e) {

if (e.getSource() == button) {

frame.dispose();

frame1.setVisible(true);

String ajpMarks = tf.getText();

String cnMarks = tf1.getText();

JLabel label4 = new JLabel("<html>AJP Marks: " + ajpMarks + "<br>Cellular Networks Marks: " + cnMarks + "</html>");

label4.setBounds(50, 100, 250, 200);

frame1.add(label4);

} else if (e.getSource() == button1) {

tf.setText("");

tf1.setText("");

}

}

}

EXP-4

package Expt4;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

public class JDBC {

public static void main(String[] args) {

try {

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

String URL="jdbc:mysql://localhost:3306/VRD";

String USER="root";

String PASS="qwerty2486";

Connection con=DriverManager.getConnection(URL, USER, PASS);

String q="insert into table1(tName,tCity) values(?,?)";

PreparedStatement psmt=con.prepareStatement(q);

psmt.setString(1, "JOHN SNOW");

psmt.setString(2, "WINTERFELL");

psmt.executeUpdate();

System.out.println("Inserted.......");

con.close();

} catch (Exception e) {

e.printStackTrace();

}

}

}

package Expt4;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

public class retrive {

public static void main(String[] args) {

try {

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

String URL = "jdbc:mysql://localhost:3306/VRD";

String USER = "root";

String PASS = "qwerty2486";

Connection con = DriverManager.getConnection(URL, USER, PASS);

String q = "select \* from table1";

java.sql.Statement stmt = con.createStatement();

ResultSet rs = stmt.executeQuery(q);

while (rs.next()) {

System.out.println("Name:- " + rs.getString("tName") + ", City:- " + rs.getString("tCity"));

}

con.close();

} catch (Exception e) {

e.printStackTrace();

}

}

}

Exp-5

import java.rmi.Remote;

import java.rmi.RemoteException;

public interface PalindromeChecker extends Remote {

boolean isPalindrome(String word) throws RemoteException;

}

//5th first

import java.rmi.RemoteException;

import java.rmi.server.UnicastRemoteObject;

public class PalindromeCheckerImpl extends UnicastRemoteObject implements PalindromeChecker {

protected PalindromeCheckerImpl() throws RemoteException {

super();

}

@Override

public boolean isPalindrome(String word) throws RemoteException {

String reversed = new StringBuilder(word).reverse().toString();

return word.equalsIgnoreCase(reversed);

}

}

//5 -2

import java.rmi.Naming;

import java.util.Scanner;

public class PalindromeClient {

public static void main(String[] args) {

try {

PalindromeChecker palindromeChecker = (PalindromeChecker) Naming.lookup("rmi://localhost/PalindromeChecker");

Scanner sc = new Scanner(System.in);

System.out.println("write a word you want to find weather is palindrome or not: ");

String word = sc.nextLine();

boolean isPalindrome = palindromeChecker.isPalindrome(word);

System.out.println("Is \"" + word + "\" a palindrome? " + isPalindrome);

} catch (Exception e) {

e.printStackTrace();

}

}

}

//5-3

import java.rmi.Naming;

import java.rmi.registry.LocateRegistry;

public class PalindromeServer {

public static void main(String[] args) {

try {

PalindromeChecker palindromeChecker = new PalindromeCheckerImpl();

LocateRegistry.createRegistry(1099); // Start the RMI registry on port 1099

Naming.rebind("PalindromeChecker", palindromeChecker);

System.out.println("Palindrome Server is running...");

} catch (Exception e) {

e.printStackTrace();

}

}

}

//5th fourth

Exp-6

package AJP;

import java.io.\*;

import java.net;

import java.util.\*;

class Expt6 {

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

// To get and print InetAddress of Local Host

InetAddress address1 = InetAddress.getLocalHost();

System.out.printin("InetAddress of Local Host : " + address1);

// To get and print InetAddress of Named Host

InetAddress address2 = InetAddress.getByName("127.0.0.1");

System.out.println("InetAddress of Named Host : " + address2) ;

// To get and print ALL InetAddresses of Named Host

InetAddress address3 [] = InetAddress.getAllByName ("127.0.0.1") ;

for (int i = 0; i ‹ address3.length; i++) {

System.out.printin("ALL InetAddresses of Named Host : "

address3[i]);

?

// To get and print InetAddresses of

/ Host with specified IP Address

byte IPAddress [] = { 125, 0, 0, 1 };

InetAddress address4 = InetAddress.getByAddress (IPAddress);

System.out. println("InetAddresses of Host with specified IP Address

: " + address4); // To get and print

InetAddresses of Host

}

}

Exp-7

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"><input type="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();

}

}

EXP-8

package AJP;

import java.sql.\*;

public class Expt8 {

static final String DB\_URL = "jdbc:mysql://localhost:3386/";

static String USER = "root"; //

static String PASS = "gupta$007"; //

public static void main(String[] args) {

7 Open a connection

try (Connection conn = DriverManager. getConnection (DB\_URL, USER,

PASS);

Statement stmt = conn. createStatement () ;) {

String sql = "CREATE DATABASE STUDENTS";

stmt.executeUpdate (sq1);

System.out.println("Database created successfully...");

} catch (SOLException e) {

e.printStackTrace ();

}

}

}

Exp-9

<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>

<!DOCTYPE html>

<html>

<head>

<meta charset="UTF-8">

<title>Java Exp 9</title>

</head>

<body>  
%-- JSP comments --%>  
<%@page import="java.util.Date"%> <%!

Date date; %>

<%

date = new Date();

%>  
<b>System date and time: </b> <%= date %> </body>  
</html>

EXP-10

package calci;

import java.io.IO Exception;

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 CalculatorServlet extends HttpServlet {

protected void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

int num1 = Integer.parseInt(request.getParameter("num1"));

int num2 = Integer.parseInt(request.getParameter("num2"));

String operation = request.getParameter("opr");

int result = 0;

String operator = "";

switch (operation) {

case "+":

result = num1 + num2;

operator = "Addition";

break;

case "-":

result = num1 - num2;

operator = "Subtraction";

break;

case "\*":

result = num1 \* num2;

operator = "Multiplication";

break;

case "/":

result = num1 / num2;

operator = "Division";

break;

}

request.setAttribute("result", result);

request.setAttribute("operator", operator);

request.getRequestDispatcher("calculator.html").forward(request, response);

}

}

<html>

<head>

<title>Calculator App</title>

</head>

<body>

<form action="CalculatorServlet" method="post">

Enter First Number: <input type="text" name="num1"><br>

Enter Second Number: <input type="text" name="num2"><br>

Select an Operation:<br>

<input type="radio" name="opr" value="+"> Addition<br>

<input type="radio" name="opr" value="-"> Subtraction<br>

<input type="radio" name="opr" value="\*"> Multiply<br>

<input type="radio" name="opr" value="/"> Divide<br>

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

</form>

<%

int result = (Integer) request.getAttribute("result");

String operator = (String) request.getAttribute("operator");

if (operator != null) {

%>

<h2>Calculator Result:</h2>

<p>Operation: <%= operator %></p>

<p>Result: <%= result %></p>

<%

}

%>

</body>

</html>