**Slip 1**

1.Write a Java program using Multithreading to display all the alphabets between ‘A’ to ‘Z’ after every 2 seconds.

public class Slip1A extends Thread

{

char c;

public void run()

{

for(c = 'A'; c<='Z';c++)

{

System.out.println(""+c);

try

{

Thread.sleep(2000);

}

catch(Exception e)

{

e.printStackTrace();

}

}

}

public static void main(String args[])

{

Slip1A t = new Slip1A ();

t.start();

}

}

2. Write a Java program to accept the details of Employee (Eno, EName, Designation, Salary) from a user and store it into the database. (Use Swing)

import java.awt.\*;  
import javax.swing.\*;  
import java.awt.event.\*;  
import java.sql.\*;  
public class Ass1 extends Frame implements ActionListener  
{  
JLabel l1,l2,l3;  
JTextField t1,t2,t3;  
JButton b;  
Connection cn;  
Statement st;  
ResultSet rs;  
public Ass1()  
{  
setLayout(null);  
l1=new JLabel(“Eno”);

l2=new JLabel(“EName”);  
l3=new JLabel(“Salary”);  
t1=new JTextField();  
t2=new JTextField();  
t3=new JTextField();  
b=new Button(“Save”);  
l1.setBounds(50,50,100,30);  
t1.setBounds(160,50,100,30);  
l2.setBounds(50,90,100,30);  
t2.setBounds(160,90,100,30);  
l3.setBounds(50,130,100,30);  
t3.setBounds(160,130,100,30);  
b.setBounds(50,170,100,30);  
add(l1);  
add(t1);  
add(l2);  
add(t2);

add(t3);  
add(b);  
b.addActionListener(this);  
setSize(500,500);  
setVisible(true);  
addWindowListener(new WindowAdapter()  
{  
public void windowClosing(WindowEvent e)  
{  
System.exit(0);  
}  
});  
}  
public void actionPerformed(ActionEvent oe)  
{  
String str=oe.getActionCommand();  
if(str.equals(“Save”))  
{  
try

Class.forName(“org.postgresql.Driver”);  
cn=DriverManager.getConnection(“jdbc:postgresql://localhost/mydb”,”root”,””);  
st =cn.createStatement();  
int en=Integer.parseInt(t1.getText());  
String enn=t2.getText();  
int sal=Integer.parseInt(t3.getText());  
String strr=”insert into emp values(” + en + ” ,'” + enn + “‘,” + sal + “)”;  
int k=st.executeUpdate(strr);  
if(k>0)  
{

JOptionPane.showMessageDialog(null,”Record Is Added”);  
}  
}  
catch(Exception er)

{  
System.out.println(“Error”);  
}  
}  
}  
public static void main(String args[])  
{  
new Ass1().show();  
}  
}

**Slip 3**

1.Write a JSP program to display the details of Patient (PNo, PName, Address, age, disease) in tabular form on browser.

<html>

<body>

<%@ page import="java.sql.\*;" %>

<%! inthno;

String hname,address; %>

<%

try{

Connection cn

Class.forName(“org.postgresql.Driver”);  
cn=DriverManager.getConnection(“jdbc:postgresql://localhost/hospital”,”root”,””);  
Statement st=cn.createStatement();

ResultSetrs=st.executeQuery("select \* from patient");

%>

<table border="1" width="40%">

<tr>

<td>Patient No</td> <td>Name</td>

<td>Address</td> </tr>

<% while(rs.next()) { %>

<tr><td><%= rs.getInt("pno") %></td>

<td><%= rs.getString("pname") %></td>

<td><%= rs.getString("address") %> </tr>

<td><%= rs.getString(“age") %> </tr>

<td><%= rs.getString("disease ") %> </tr>

<%

}

cn.close();

}catch(Exception e)

{

out.println(e);

}

%>

</body>

</html>

2.Write a Java program to create LinkedList of String objects and perform the following: i. Add element at the end of the list ii. Delete first element of the list iii. Display the contents of list in reverse order

**Slip 4**

1.Write a Java program using Runnable interface to blink Text on the JFrame (Use

Swing)

import java.awt.\*;

import java.awt.event.\*;

import java.swing.\*;

public class BlinkText extends JFrame implements Runnable

{

Thread t;

JLabel l1;

int f;

public BlinkText()

{

t=new Thread(this);

t.start();

setLayout(null);

l1=new JLabel("Hello JAVA");

l1.setBounds(100,100,100,40);

add(l1);

setSize(300,300);

setVisible(true);

f=0;

}

public void run()

{

try

{

if(f==0)

{

t.sleep(200);

l1.setText("");

f=1;

}

if(f==1)

{

t.sleep(200);

l1.setText("Hello Java");

f=0;

}

}catch(Exception e)

{

System.out.println(e);

}

run();

}

public static void main(String args[])

{

new BlinkText();

}

}

2. Write a Java program to store city names and their STD codes using an appropriate

collection and perform following operations:

i. Add a new city and its code (No duplicates)

ii. Remove a city from the collection

iii. Search for a city name and display the code

import java.awt.\*;

import java.awt.event.\*;

import javax.swing.\*;

import java.util.\*;

class Slip16\_2 extends JFrame implements ActionListener

{

JTextField t1,t2,t3;

JButton b1,b2,b3;

JTextArea t;

JPanel p1,p2;

Hashtable ts;

Slip16\_2()

{

ts=new Hashtable();

t1=new JTextField(10);

t2=new JTextField(10);

t3=new JTextField(10);

b1=new JButton("Add");

b2=new JButton("Search");

b3=new JButton("Remove");

t=new JTextArea(20,20);

p1=new JPanel();

p1.add(t);

p2= new JPanel();

p2.setLayout(new GridLayout(2,3));

p2.add(t1);

p2.add(t2);

p2.add(b1);

p2.add(t3);

p2.add(b2);

p2.add(b3);

add(p1);

add(p2);

b1.addActionListener(this);

b2.addActionListener(this);

b3.addActionListener(this);

setLayout(new FlowLayout());

setSize(500,500);

setVisible(true);

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

}

public void actionPerformed(ActionEvent e)

{

if(b1==e.getSource())

{

String name = t1.getText();

int code = Integer.parseInt(t2.getText());

ts.put(name,code);

Enumeration k=ts.keys();

Enumeration v=ts.elements();

String msg="";

while(k.hasMoreElements())

{

msg=msg+k.nextElement()+" = "+v.nextElement()+"\n";

}

t.setText(msg);

t1.setText("");

t2.setText("");

}

else if(b2==e.getSource())

{

String name = t3.getText();

if(ts.containsKey(name))

{

t.setText(ts.get(name).toString());

}

else

JOptionPane.showMessageDialog(null,"City not found ...");

}

else if(b3==e.getSource())

{

String name = t3.getText();

if(ts.containsKey(name))

{

ts.remove(name);

JOptionPane.showMessageDialog(null,"City Deleted ...");

}

else

JOptionPane.showMessageDialog(null,"City not found ...");

}

}

public static void main(String a[])

{

new Slip16\_2();

}

}

**Slip 6**

1.Write a Java program to accept ‘n’ integers from the user and store them in a Collection. Display them in the sorted order. The collection should not accept duplicate elements. (Use a suitable collection). Search for a particular element using predefined search method in the Collection framework.

import java.util.\*;

import java.io.\*;

class Slip19\_2

{

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

{

int no,element,i;

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

TreeSet ts=new TreeSet();

System.out.println("Enter the of elements :");

no=Integer.parseInt(br.readLine());

for(i=0;i<no;i++)

{

System.out.println("Enter the element : ");

element=Integer.parseInt(br.readLine());

ts.add(element);

}

System.out.println("The elements in sorted order :"+ts);

System.out.println("Enter element to be serach : ");

element = Integer.parseInt(br.readLine());

if(ts.contains(element))

System.out.println("Element is found");

else

System.out.println("Element is NOT found");

}

}

2. Write a java program using multithreading to simulate traffic signal (Use Swing).

import java.awt.\*;

import java.awt.event.\*;

import java.util.\*;

import javax.swing.\*;

// Main class

// Extending JFrame class and

// Implementing ItemListener interface

public class Traffic\_Signal

extends JFrame implements ItemListener {

// Setting the buttons for the layout

JRadioButton jr1;

JRadioButton jr2;

JRadioButton jr3;

// Setting the field area

JTextField j1 = new JTextField(10);

ButtonGroup b = new ButtonGroup();

String msg = " ";

// Initially setting the co-ordinates to 0,0,0

int x = 0, y = 0, z = 0;

public Traffic\_Signal(String msg)

{

super(msg);

setLayout(new FlowLayout());

// Assigning name to the button declared above

// with help of JRadioButton class

jr1 = new JRadioButton("Red");

jr2 = new JRadioButton("Yellow");

jr3 = new JRadioButton("Green");

jr1.addItemListener(this);

jr2.addItemListener(this);

jr3.addItemListener(this);

add(jr1);

add(jr2);

add(jr3);

b.add(jr1);

b.add(jr2);

b.add(jr3);

add(j1);

// Method 1

// To add a window

addWindowListener(new WindowAdapter() {

public void windowClosing(WindowEvent e)

{

// It haults here itself

System.exit(0);

}

});

}

// Method 2

// To change colors of traffic signal

public void itemStateChanged(ItemEvent ie)

{

// If it is red

if (ie.getSource() == jr1) {

if (ie.getStateChange() == 1) {

// Then display message- Stop

msg = "Stop!";

x = 1;

// Repainting the box with original one

// Practically black

repaint();

}

else {

msg = "";

}

}

// If state is Orange or technically jr2

if (ie.getSource() == jr2) {

if (ie.getStateChange() == 1) {

// Then display message-

// Get ready in waiting state

msg = "Get Ready to go!";

y = 1;

// Again repainting the button

repaint();

}

else {

msg = "";

}

}

// If state is Green

if (ie.getSource() == jr3) {

if (ie.getStateChange() == 1) {

// Then display message- Go

msg = "Go!!";

z = 1;

repaint();

}

else {

msg = "";

}

}

j1.setText(msg);

}

// Method 3

// handling the paint graphics and

// dimensions of the buttons via

// setting co-ordinates

public void paint(Graphics g)

{

g.drawRect(100, 105, 110, 270);

g.drawOval(120, 150, 60, 60);

g.drawOval(120, 230, 60, 60);

g.drawOval(120, 300, 60, 60);

// Case: Red

if (x == 1) {

g.setColor(Color.RED);

g.fillOval(120, 150, 60, 60);

g.setColor(Color.WHITE);

g.fillOval(120, 230, 60, 60);

g.setColor(Color.WHITE);

g.fillOval(120, 300, 60, 60);

x = 0;

}

// Case: Orange

if (y == 1) {

g.setColor(Color.WHITE);

g.fillOval(120, 150, 60, 60);

g.setColor(Color.YELLOW);

g.fillOval(120, 230, 60, 60);

g.setColor(Color.WHITE);

g.fillOval(120, 300, 60, 60);

y = 0;

}

// Case: Green

if (z == 1) {

g.setColor(Color.WHITE);

g.fillOval(120, 150, 60, 60);

g.setColor(Color.WHITE);

g.fillOval(120, 230, 60, 60);

g.setColor(Color.GREEN);

g.fillOval(120, 300, 60, 60);

z = 0;

}

}

// Method 4

// Main driver method

public static void main(String args[])

{

// Creating object of Jframe class inside main()

// method

JFrame jf = new Traffic\_Signal("Traffic Light");

// Setting size and making traffic signal

// operational using setVisible() method

jf.setSize(500, 500);

jf.setVisible(true);

}

}

**Slip 7**

1.Write a java program that implements a multi-thread application that has three threads. First thread generates random integer number after every one second, if the number is even; second thread computes the square of that number and prints it. If the number is odd, the third thread computes the cube of that number and prints it.

import java.util\*;

int x;

Square(int n)

{

x = n;

}

public void run()

{

int sqr = x \* x;

System.out.println("Square of " + x + " = " + sqr );

}

}

class Cube extends Thread

{

int x;

Cube(int n)

{

x = n;

}

public void run()

{

int cub = x \* x \* x;

System.out.println("Cube of " + x + " = " + cub );

}

}

class Number extends Thread

{

public void run()

{

Random random = new Random();

for(int i =0; i<10; i++)

{

int randomInteger = random.nextInt(100);

System.out.println("Random Integer generated : " + randomInteger);

Square s = new Square(randomInteger);

s.start();

Cube c = new Cube(randomInteger);

c.start();

try {

Thread.sleep(1000);

This thread generates random number 10 times

between 1 to 100 for every 1 second. The generated

random number is then passed as argument to

Square and Cube threads.

Output varies each time a program is executed.

} catch (InterruptedException ex) {

System.out.println(ex);

}

}

}

}

public class LAB3B {

public static void main(String args[])

{

Number n = new Number();

n.start();

}

}

2. Write a java program for the following:

i. To create a Product (Pid, Pname, Price) table.

ii. Insert at least five records into the Product table.

iii. Display all the records from a Product table.

Assume Database is already created.

import java.sql.\*;

import java.awt.\*;

import java.awt.event.\*;

import javax.swing.\*;

import java.util.\*;

class Slip13\_2 extends JFrame implements ActionListener

{

JLabel l1,l2,l3;

JTextField t1,t2,t3;

JButton b1,b2,b3;

String sql;

JPanel p,p1;

Connection con;

PreparedStatement ps;

JTable t;

JScrollPane js;

Statement stmt ;

ResultSet rs ;

ResultSetMetaData rsmd ;

int columns;

Vector columnNames = new Vector();

Vector data = new Vector();

Slip13\_2()

{

l1 = new JLabel("Enter pid :");

l2 = new JLabel("Enter pname :");

l3 = new JLabel("price :");

t1 = new JTextField(20);

t2 = new JTextField(20);

t3 = new JTextField(20);

b1 = new JButton("Save");

b2 = new JButton("Display");

b3 = new JButton("Clear");

b1.addActionListener(this);

b2.addActionListener(this);

b3.addActionListener(this);

p=new JPanel();

p1=new JPanel();

p.add(l1);

p.add(t1);

p.add(l2);

p.add(t2);

p.add(l3);

p.add(t3);

p.add(b1);

p.add(b2);

p.add(b3);

add(p);

setLayout(new GridLayout(2,1));

setSize(600,800);

setVisible(true);

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

}

public void actionPerformed(ActionEvent e)

{

if((JButton)b1==e.getSource())

{

int no = Integer.parseInt(t1.getText());

String name = t2.getText();

int p = Integer.parseInt(t3.getText());

System.out.println("Accept Values");

try

{

Class.forName(“org.postgresql.Driver”);

con=DriverManager.getConnection(“jdbc:postgresql://192.168.100.254/product”,”oracle”,”oracle”);

sql = "insert into proj values(?,?,?)";

ps = con.prepareStatement(sql);

ps.setInt(1,pid);

ps.setString(2,p name);

ps.setInt(3,price);

System.out.println("values set");

int n=ps.executeUpdate();

if(n!=0)

{

JOptionPane.showMessageDialog(null,"Record insered ...");

}

else

JOptionPane.showMessageDialog(null,"Record NOT inserted ");

}//end of try

catch(Exception ex)

{

System.out.println(ex);

//ex.printStackTrace();

}

}//end of if

else if((JButton)b2==e.getSource())

{

try

{

Class.forName(“org.postgresql.Driver”);

con=DriverManager.getConnection(“jdbc:postgresql://192.168.100.254/product”,”oracle”,”oracle”);

System.out.println("Connected");

stmt=con.createStatement();

rs = stmt.executeQuery("select \* from prod");

rsmd = rs.getMetaData();

columns = rsmd.getColumnCount();

//Get Columns name

for(int i = 1; i <= columns; i++)

{

columnNames.addElement(rsmd.getColumnName(i));

}

//Get row data

while(rs.next())

{

Vector row = new Vector(columns);

for(int i = 1; i <= columns; i++)

{

row.addElement(rs.getObject(i));

}

data.addElement(row);

}

t = new JTable(data, columnNames);

js = new JScrollPane(t);

p1.add(js);

add(p1);

setSize(600, 600);

setVisible(true);

}

catch(Exception e1)

{

System.out.println(e1);

}

}

else

{

t1.setText(" ");

t2.setText(" ");

t3.setText(" ");

}

}//end of method

public static void main(String a[])

{

Slip13\_2 ob = new Slip13\_2();

}

}

**Slip 8**

1.Write a java program to define a thread for printing text on output screen for ‘n’ number of times. Create 3 threads and run them. Pass the text ‘n’ parameters to the thread constructor. Example: i. First thread prints “COVID19” 10 times. ii. Second thread prints “LOCKDOWN2020” 20 times iii. Third thread prints “VACCINATED2021” 30 times

import java.io.\*;

import java.lang.String.\*;

class Ass\_seta3 extends Thread

{

String msg="";

int n;

Ass\_seta3(String msg,int n)

{

this.msg=msg;

this.n=n;

}

public void run()

{

try

{ for(int i=1;i<=n;i++)

{

System.out.println(msg+" "+i+" times");

}

}

catch(Exception e){}

}

}

public class seta3

{

public static void main(String a[])

{

int n=Integer.parseInt(a[0]);

Ass\_seta3 t1=new Ass\_seta3("COVID 19",n);

t1.start();

Ass\_seta3 t2=new Ass\_seta3("LOCKDOWN2020",n+10);

t2.start();

Ass\_seta3 t3=new Ass\_seta3("VACCINATED2021",n+20);

t3.start();

}

}

2.Write a JSP program to check whether a given number is prime or not. Display the result in red color.

source file name: Primeno.html

<html>

<head>

<title>Prime no JSP program</title>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width">

</head>

<body>

<form action="http://localhost:8080/JspPrograms/PrimeNumber.jsp" method="post">

enter any no:

<input type="text" name="t1" >

<br>

<input type="submit" >

</form>

</body>

</html>

source file name: PrimeNumber.jsp

<%

int n=Integer.parseInt(request.getParameter("t1"));

out.println(" given number is: "+n);

int d=2;

while(d<n)

{

if(n%d==0)

{

out.println("<br> "+n+" is not Prime no.");

break;

}

else

d++;

}

if(n==d)

out.println("<br>"+n+" is Prime no.");

%>

**Slip 12**

1.Write a JSP program to check whether given number is Perfect or not. (Use Include directive).

Index.html file:

<!DOCTYPE html>

<html>

<head>

<title>PERFECT NUMBER</title>

</head>

<body>

<form action="perfect.jsp" method="post">

Enter Number :<input type="text" name="num">

<input type="submit" value="Submit" name="s1">

</form>

</body>

</html>

Perfect.jsp file:

<%@ page import="java.util.\*" %>

<%

if(request.getParameter("s1")!=null)

{

Integer num,a,i,sum = 0;

num = Integer.parseInt(request.getParameter("num"));

a = num;

for(i=1;i<a;i++)

{

if(a%i==0)

{

sum=sum + i;

}

}

if(sum==a)

{

out.println(+num+ "is a perfect number");

}

else

{

out.println(+num+ "is not a perfect number");

}

}

%>

2.Write a Java Program to create a PROJECT table with field’s project\_id, Project\_name, Project\_description, Project\_Status. Insert values in the table. Display all the details of the PROJECT table in a tabular format on the screen.(using swing).

import java.sql.\*;

import java.awt.\*;

import java.awt.event.\*;

import javax.swing.\*;

import java.util.\*;

class Slip13\_2 extends JFrame implements ActionListener

{

JLabel l1,l2,l3;

JTextField t1,t2,t3;

JButton b1,b2,b3;

String sql;

JPanel p,p1;

Connection con;

PreparedStatement ps;

JTable t;

JScrollPane js;

Statement stmt ;

ResultSet rs ;

ResultSetMetaData rsmd ;

int columns;

Vector columnNames = new Vector();

Vector data = new Vector();

Slip13\_2()

{

l1 = new JLabel("Enter pid :");

l2 = new JLabel("Enter pname :");

l3 = new JLabel("price :");

t1 = new JTextField(20);

t2 = new JTextField(20);

t3 = new JTextField(20);

b1 = new JButton("Save");

b2 = new JButton("Display");

b3 = new JButton("Clear");

b1.addActionListener(this);

b2.addActionListener(this);

b3.addActionListener(this);

p=new JPanel();

p1=new JPanel();

p.add(l1);

p.add(t1);

p.add(l2);

p.add(t2);

p.add(l3);

p.add(t3);

p.add(b1);

p.add(b2);

p.add(b3);

add(p);

setLayout(new GridLayout(2,1));

setSize(600,800);

setVisible(true);

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

}

public void actionPerformed(ActionEvent e)

{

if((JButton)b1==e.getSource())

{

int no = Integer.parseInt(t1.getText());

String name = t2.getText();

int p = Integer.parseInt(t3.getText());

System.out.println("Accept Values");

try

{

Class.forName(“org.postgresql.Driver”);

con=DriverManager.getConnection(“jdbc:postgresql://192.168.100.254/project”,”oracle”,”oracle”);

sql = "insert into proj values(?,?,?)";

ps = con.prepareStatement(sql);

ps.setInt(1,pid);

ps.setString(2,p name);

ps.setInt(3,price);

System.out.println("values set");

int n=ps.executeUpdate();

if(n!=0)

{

JOptionPane.showMessageDialog(null,"Record insered ...");

}

else

JOptionPane.showMessageDialog(null,"Record NOT inserted ");

}//end of try

catch(Exception ex)

{

System.out.println(ex);

//ex.printStackTrace();

}

}//end of if

else if((JButton)b2==e.getSource())

{

try

{

Class.forName(“org.postgresql.Driver”);

con=DriverManager.getConnection(“jdbc:postgresql://192.168.100.254/project”,”oracle”,”oracle”);

System.out.println("Connected");

stmt=con.createStatement();

rs = stmt.executeQuery("select \* from proj");

rsmd = rs.getMetaData();

columns = rsmd.getColumnCount();

//Get Columns name

for(int i = 1; i <= columns; i++)

{

columnNames.addElement(rsmd.getColumnName(i));

}

//Get row data

while(rs.next())

{

Vector row = new Vector(columns);

for(int i = 1; i <= columns; i++)

{

row.addElement(rs.getObject(i));

}

data.addElement(row);

}

t = new JTable(data, columnNames);

js = new JScrollPane(t);

p1.add(js);

add(p1);

setSize(600, 600);

setVisible(true);

}

catch(Exception e1)

{

System.out.println(e1);

}

}

else

{

t1.setText(" ");

t2.setText(" ");

t3.setText(" ");

}

}//end of method

public static void main(String a[])

{

Slip13\_2 ob = new Slip13\_2();

}

}

**Slip 13**

1. Write a Java program to display information about the database and list all the tables in the database. (Use DatabaseMetaData).

import java.sql.\*;

import java.io.\*;

public class DBMetaData

{

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

{

ResultSet rs = null;

Class.forName("org.postgresql.Driver");

Connection conn = DriverManager.getConnection("jdbc:postgresql://localhost/dbtry","postgres","redhat");

DatabaseMetaData dbmd = conn.getMetaData();

System.out.println("Database Product name = " + dbmd.getDatabaseProductName());

System.out.println("User name = " + dbmd.getUserName());

System.out.println("Database driver name= " + dbmd.getDriverName());

System.out.println("Database driver version = "+ dbmd.getDriverVersion());

System.out.println("Database product name = " + dbmd.getDatabaseProductName());

System.out.println("Database Version = " + dbmd.getDriverMajorVersion());

rs = dbmd.getTables(null,null,null, new String[]{"TABLE"});

System.out.println("List of tables...");

while(rs.next())

{

String tblName = rs.getString("TABLE\_NAME");

System.out.println("Table : "+ tblName);

}

conn.close();

}

}

1. Write a Java program to show lifecycle (creation, sleep, and dead) of a thread. Program should print randomly the name of thread and value of sleep time. The name of the thread should be hard coded through constructor. The sleep time of a thread will be a random integer in the range 0 to 4999.

Class MyThread extends Thread

{ public MyThread(String s)

{

super(s);

}

public void run()

{

System.out.println(getName()+"thread created.");

while(true)

{

System.out.println(this);

int s=(int)(math.random()\*5000);

System.out.println(getName()+"is sleeping for :+s+"msec");

try{

Thread.sleep(s);

}

catch(Exception e)

{

}

}

}

Class ThreadLifeCycle

{

public static void main(String args[])

{

MyThread t1=new MyThread("shradha"),t2=new MyThread("pooja");

t1.start();

t2.start();

try

{

t1.join();

t2.join();

}

catch(Exception e)

{

}

System.out.println(t1.getName()+"thread dead.");

System.out.println(t2.getName()+"thread dead.");

}

}

**Slip 14**

1.Write a Java program using Multithreading for a simple search engine. Accept a string to be searched. Search the string in all text files in the current folder. Use a separate thread for each file. The result should display the filename and line number where the string is found.

import java.io.\*;

public class SearchThread extends Thread

{

File f1;

String fname;

static String str;

String line;

LineNumberReader reader = null;

SearchThread(String fname)

{

this.fname=fname;

f1=new File(fname);

}

public void run()

{

try

{

FileReader fr=new FileReader(f1);

reader=new LineNumberReader(fr);

while((line=reader.readLine())!=null)

{

if(line.indexOf(str)!=-1)

{

System.out.println("string found in "+fname+"at "+reader.getLineNumber()+"line");

stop();

}

}

}

catch(Exception e)

{

}

}

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

{

Thread t[]=new Thread[20];

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

System.out.println("Enter String to search");

str=br.readLine();

FilenameFilter filter = new FilenameFilter()

{

public boolean accept(File file, String name)

{

if (name.endsWith(".txt"))

{

return true;

}

else

{

return false;

}

}

};

File dir1 = new File(".");

File[] files = dir1.listFiles(filter);

if (files.length == 0)

{

System.out.println("no files available with this extension");

}

else

{

for(int i=0;i<files.length;i++)

{

for (File aFile : files)

{

t[i]=new SearchThread(aFile.getName());

t[i].start();

}

}

}

}

}

2.Write a JSP program to calculate sum of first and last digit of a given number. Display sum in Red Color with font size 18.

HTML FILE

<html>

<body>

<form method=post action="Slip7.jsp">

Enter Any Number : <Input type=text name=num>

<input type=submit value=Display>

</form>

</body>

</html>

JSP FILE:

<%@page contentType="text/html" pageEncoding="UTF-8">

<!DOCTYPE html>

<html>

<body>

<%! intn,rem,r; %>

<% n=Integer.parseInt(request.getParameter("num"));

if(n<10)

{

out.println("Sum of first and last digit is ");

%><font size=18 color=red><%= n %>

<%

}

else

{

rem=n%10;

do

{

r=n%10;

n=n/10;

}while(n>0);

n=rem+r;

out.println("Sum of first and last digit is ");

%><font size=18 color=red><%= n %>

<%

}

%>

</body>

</html>

**Slip 17**

1. Write a java program to accept ‘N’ integers from a user. Store and display integers in sorted order having proper collection class. The collection should not accept duplicate elements.

import java.util.\*;

import java.io.\*;

class Slip19\_2

{

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

{

int no,element,i;

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

TreeSet ts=new TreeSet();

System.out.println("Enter the of elements :");

no=Integer.parseInt(br.readLine());

for(i=0;i<no;i++)

{

System.out.println("Enter the element : ");

element=Integer.parseInt(br.readLine());

ts.add(element);

}

System.out.println("The elements in sorted order :"+ts);

System.out.println("Enter element to be serach : ");

element = Integer.parseInt(br.readLine());

if(ts.contains(element))

System.out.println("Element is found");

else

System.out.println("Element is NOT found");

}

}

1. Write a java program using Multithreading to display the number’s between 1 to 100 continuously in a JTextField by clicking on JButton. (Use Runnable Interface & Swing).

import java.awt.event.\*;

import javax.swing.\*;

class Message implements Runnable

{

JTextField t;

public void run()

{

for(int i =1; i<=100;i++)

{

t.setText(""+i);

try

{

Thread.sleep(50);

}

catch(Exception e)

{

e.printStackTrace();

}

}

}

}

class Slip12\_1 implements ActionListener

{

JFrame f;

JPanel p;

JTextField t;

JButton b;

Thread t1;

Slip12\_1()

{

f = new JFrame();

p = new JPanel();

t = new JTextField(60);

b = new JButton("Start");

t1 = new Thread(this);

b.addActionListener(this);

p.add(t);

p.add(b);

f.add(p);

f.setSize(400, 400);

f.setVisible(true);

}

public void actionPerformed(ActionEvent e)

{

t1.start();

}

}

Email This

BlogThis!

Share to Twitter

Share to Facebook

Share to Pinterest