Department of CSE

Laboratory Manual	
Course:	B.Tech.
Year & Semester:	II – I
Class:	CSE
Subject:	**OOP Through JAVA Lab Manual**
Regulation:	R22

BALAJI INSTITUTE OF TECHNOLOGY AND SCIENCE (AUTONOMOUS)

B.Tech(Department of Computer Science & Engineering)

OBJECT ORIENTED PROGRAMMING THROUGH JAVA LAB

Course Outcomes:

- Able to write programs for solving real world problems using the java collection framework.
- Able to write programs using abstract classes.
- Able to write multithreaded programs.
- Able to write Applets programs.
- Able to write GUI programs using swing controls in Java.

List of Experiments:

1. Use Eclipse or Net bean platform and acquaint yourself with the various menus. Create a test project, add a test class, and run it. See how you can use auto suggestions, auto fill. Try code formatter and code refactoring like renaming variables, methods, and classes. Try debug step by step with a small program of about 10 to 15 lines which contains at least one if else condition and a for loop.
2. Write a Java program that works as a simple calculator. Use a grid layout to arrange buttons for the digits and for the +, -,*, % operations. Add a text field to display the result. Handle any possible exceptions like divided by zero.
3. A) Develop an applet in Java that displays a simple message.
   B) Develop an applet in Java that receives an integer in one text field, and computes its factorial Value and returns it in another text field, when the button named "Compute" is clicked.
4. Write a Java program that creates a user interface to perform integer divisions. The user enters two numbers in the text fields, Num1 and Num2. The division of Num1 and Num 2 is displayed in the Result field when the Divide button is clicked. If Num1 or Num2 were not an integer, the program would throw a Number Format Exception. If Num2 were Zero, the program would throw an Arithmetic Exception. Display the exception in a message dialog box.
5. Write a Java program that implements a multi-thread application that has three threads. First thread generates a random integer every 1 second and if the value is even, the second thread computes the square of the number and prints. If the value is odd, the third thread will print the value of the cube of the number.
6. Write a Java program for the following: Create a doubly linked list of elements. Delete a given element from the above list. Display the contents of the list after deletion.
7. Write a Java program that simulates a traffic light. The program lets the user select one of three lights: red, yellow, or green with radio buttons. On selecting a button, an appropriate message with "Stop" or "Ready" or "Go" should appear above the buttons in the selected color. Initially, there is no message shown.

8. Write a Java program to create an abstract class named Shape that contains two integers and an empty method named print Area (). Provide three classes named Rectangle, Triangle, and Circle such that each one of the classes extends the class Shape. Each one of the classes contains only the method print Area () that prints the area of the given shape.
9. Suppose that a table named Table.txt is stored in a text file. The first line in the file is the header, and the remaining lines correspond to rows in the table. The elements are separated by commas. Write a java program to display the table using Labels in Grid Layout.
10. Write a Java program that handles all mouse events and shows the event name at the center of the window when a mouse event is fired (Use Adapter classes).
11. Write a Java program that loads names and phone numbers from a text file where the data is organized as one line per record and each field in a record are separated by a tab (\t). It takes a name or phone number as input and prints the corresponding other value from the hash table (hint: use hash tables).
12. Write a Java program that correctly implements the producer – consumer problem using the concept of inter thread communication.
13. Write a Java program to list all the files in a directory including the files present in all its subdirectories.

Week 1

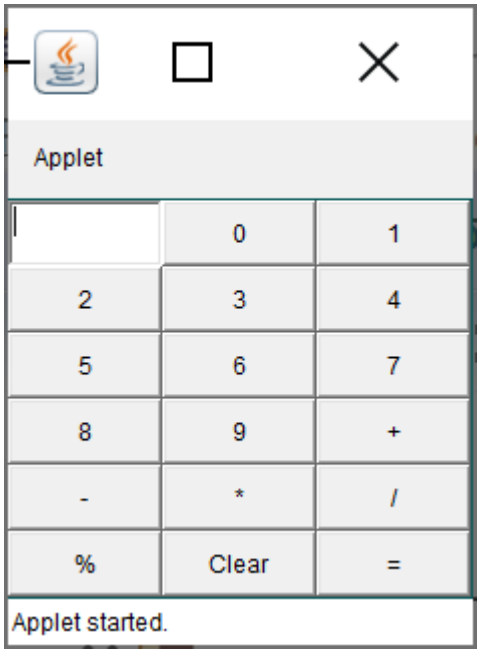
Write a Java program that works as a simple calculator. Use a grid layout to arrange buttons for thedigits and for the +, -,*, % operations. Add a text field to display the result. Handle any possible exceptions like divided by zero.

Source Code:

```
import java.awt.*;
import java.awt.event.*;
import java.applet.*;

/*
 * <applet code="Calculator" width=500 height=500></applet>
 * */

class Calculator extends Applet implements ActionListener
{
	String msg=" ";
	int v1,v2,result;
	TextField t1;
	Button b[]=new Button[10];
	Button add,sub,mul,div,clear,mod,EQ;
	char OP;
	public void init()
	{
		Color k=new Color(10,89,90);
		setBackground(k);
		t1=new TextField(50);
		GridLayout gl=new GridLayout(6,3);
		setLayout(gl);
		for(int i=0;i<10;i++)
			b[i]=new Button(""+i);
		add=new Button("+");
		sub=new Button("-");
		mul=new Button("*");
		div=new Button("/");
		mod=new Button("%");
		clear=new Button("Clear");
		EQ=new Button("=");
		t1.addActionListener(this);
		add(t1);
		for(int i=0;i<10;i++)
			add(b[i]);
		add(add);
		add(sub);
		add(mul);
		add(div);
		add(mod);

		add(clear);
		add(EQ);
		for(int i=0;i<10;i++)
			b[i].addActionListener(this);

		add.addActionListener(this);
		sub.addActionListener(this);
		mul.addActionListener(this);
		div.addActionListener(this);
		mod.addActionListener(this);
		clear.addActionListener(this);
		EQ.addActionListener(this);
	}
```

```
	public void actionPerformed(ActionEvent ae)
	{
		String str=ae.getActionCommand();
		char ch=str.charAt(0);

		if ( Character.isDigit(ch))
			t1.setText(t1.getText()+str);
		else
			if(str.equals("+"))
			{
				v1=Integer.parseInt(t1.getText());
				OP='+';
				t1.setText("");
			}
			else if(str.equals("-"))
			{
				v1=Integer.parseInt(t1.getText()); OP='-';
				t1.setText("");
			}
			else if(str.equals("*"))
			{
				v1=Integer.parseInt(t1.getText());
				OP='*';
				t1.setText("");
			}
			else if(str.equals("/"))
			{
				v1=Integer.parseInt(t1.getText());
				OP='/';
				t1.setText("");
			}
			else if(str.equals("%"))
			{
			   v1=Integer.parseInt(t1.getText()); OP='%';
			   t1.setText("");
			}
		if(str.equals("=")){
			v2=Integer.parseInt(t1.getText());
			if(OP=='+')
				result=v1+v2;
			else if(OP=='-')
				result=v1-v2;
			else if(OP=='*')
				result=v1*v2;
			else if(OP=='/')
				result=v1/v2;
			else if(OP=='%')
				result=v1%v2;
			t1.setText(""+result);
		}
		if(str.equals("Clear"))
		{
			t1.setText("");
		}
	}
}
```

Output:



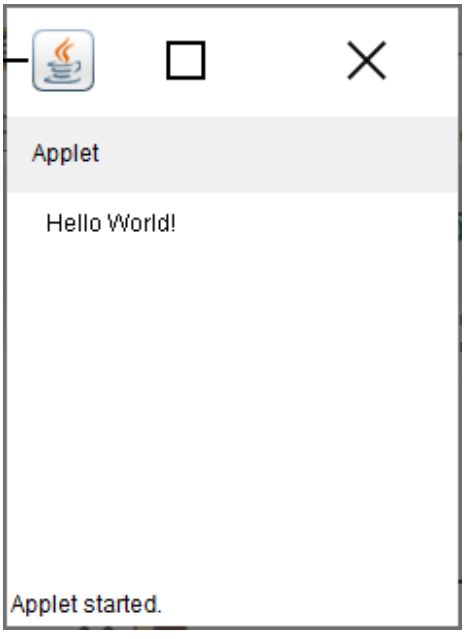
Week2

a) Develop an applet in Java that displays a simple message.
b) Develop an applet in Java that receives an integer in one text field, and computes its factorial Value and returns it in another text field, when the button named "Compute" is clicked.

a) Develop an applet in Java that displays a simple message.

```
// Import the packages to access the classes and methods in awt and applet classes.
import java.awt.*;
import java.applet.*;

/* <applet code="Applet1" width=200 height=300></applet>*/

public class Applet1 extends Applet
{
	// Paint method to display the message.
	public void paint(Graphics g)
	{
		g.drawString("Hello World!",20,20);
	}
}
```

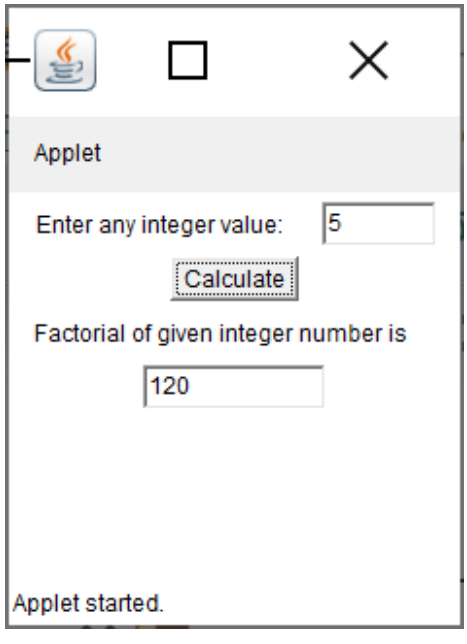
Output:



b) Develop an applet in Java that receives an integer in one text field, and computes its factorial Value and returns it in another text field, when the button named "Compute" is clicked.

```
import java.awt.*;
import java.awt.event.*;
import java.applet.Applet;

/*<applet code="Fact.class" height=300 width=300></applet>*/

public class Factorial extends Applet implements ActionListener
{
	Label l1,l2;
	TextField t1,t2;
	Button b1;
	public void init()
	{
		l1=new Label("Enter any integer value: ");
		add(l1);
		t1=new TextField(5);
		add(t1);
		b1=new Button("Calculate");
		add(b1);
		b1.addActionListener(this);
		l2=new Label("Factorial of given integer number is ");
		add(l2);
		t2=new TextField(10);
		add(t2);
	}
	public void actionPerformed(ActionEvent e)
	{
		if(e.getSource()==b1)
		{
			int fact=fact(Integer.parseInt(t1.getText()));
			t2.setText(String.valueOf(fact));
		}
	}
	int fact(int f)
	{
		return((f==0)?1:f*fact(f-1));
	 }
    }
```

Output:



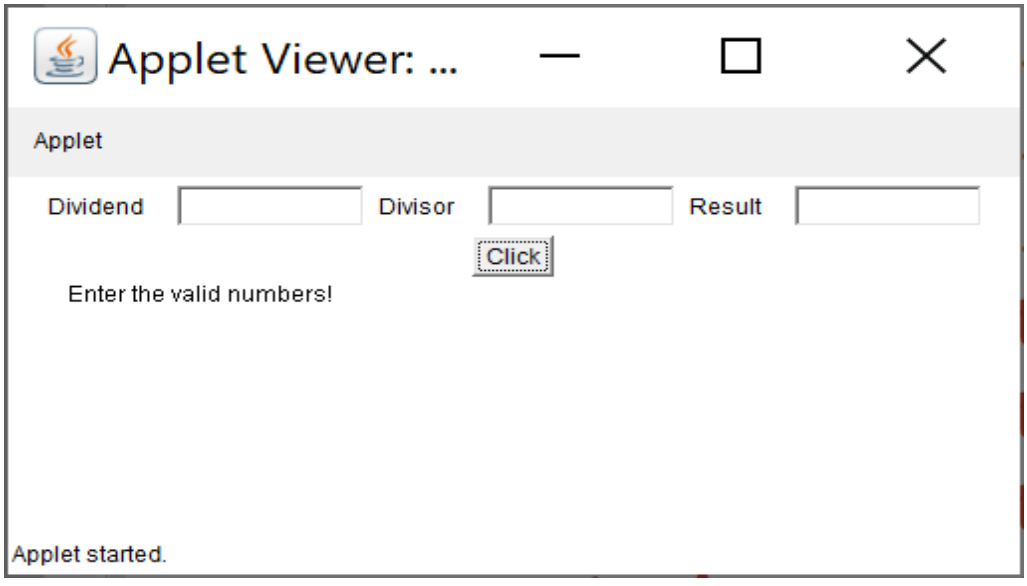
Week 3:

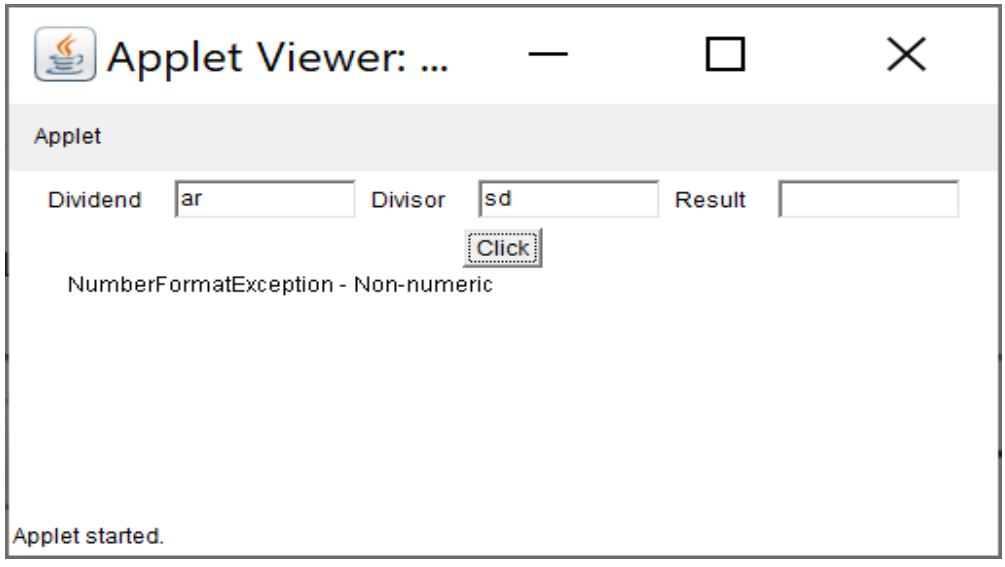
Write a Java program that creates a user interface to perform integer divisions. The user enters two numbers in the text fields, Num1 and Num2. The division of Num1 and Num 2 is displayed in the Result field when the Divide button is clicked. If Num1 or Num2 were not an integer, the program would throw a Number Format Exception. If Num2 were Zero, the program would throw an Arithmetic Exception. Display the exception in a message dialog box.

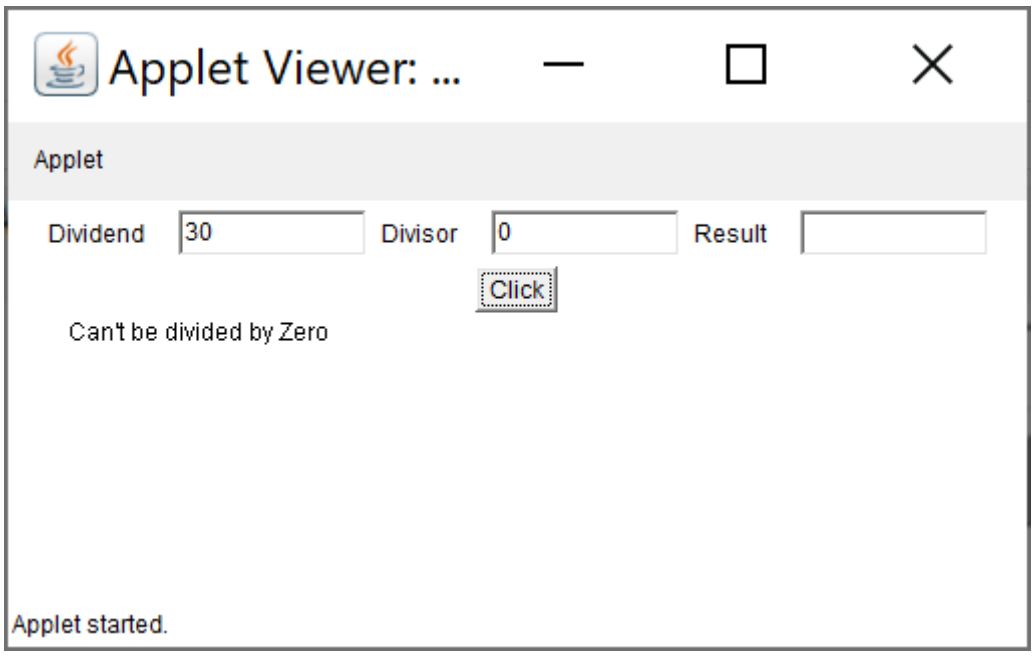
```
import java.awt.*;
import java.awt.event.*;
import java.applet.*;

/*<applet code="DivisionExample"width=230 height=250></applet>*/

public class DivisionExample extends Applet implements ActionListener
{      String msg;
       TextField num1, num2, res;
       Label l1, l2, l3;
       Button div;

       public void init()
       {
              l1 = new Label("Dividend");
              l2 = new Label("Divisor");
              l3 = new Label("Result");
              num1 = new TextField(10);
              num2 = new TextField(10);
              res  = new TextField(10);
              div = new Button("Click");
              div.addActionListener(this);
              add(l1);
              add(num1);
              add(l2);
              add(num2);
              add(l3);
              add(res);
              add(div);
       }

       public void actionPerformed(ActionEvent ae)
       {
            String arg = ae.getActionCommand();
            int num1 = 0, num2 = 0;
              if (arg.equals("Click"))
              {
                     if (this.num1.getText().isEmpty() | this.num2.getText().isEmpty())
                     {
                            msg = "Enter the valid numbers!";
                            repaint();
                     } else {
                            try {
                                   num1 = Integer.parseInt(this.num1.getText());
                                   num2 = Integer.parseInt(this.num2.getText());

                                   int num3 = num1 / num2;
```

```
                        res.setText(String.valueOf(num3));
                        msg = "Operation Succesfull!!!";
                        repaint();
                    } catch (NumberFormatException ex) {
                        System.out.println(ex);
                        res.setText("");
                        msg = "NumberFormatException - Non-numeric";
                        repaint();
                    } catch (ArithmeticException e) {
                        System.out.println("Can't be divided by Zero" + e);
                        res.setText("");
                        msg = "Can't be divided by Zero";
                        repaint();
                    }
                }
            }
        }

        public void paint(Graphics g) {
            g.drawString(msg, 30, 70);
        }
}
```

Output:






Applet Viewer: ...
Applet
Dividend
30
Divisor
0
Result
Click
Can't be divided by Zero
Applet started.


Applet Viewer: ...
Applet
Dividend
30
Divisor
10
Result
3
Click
Operation Succesfull!!!
Applet started.

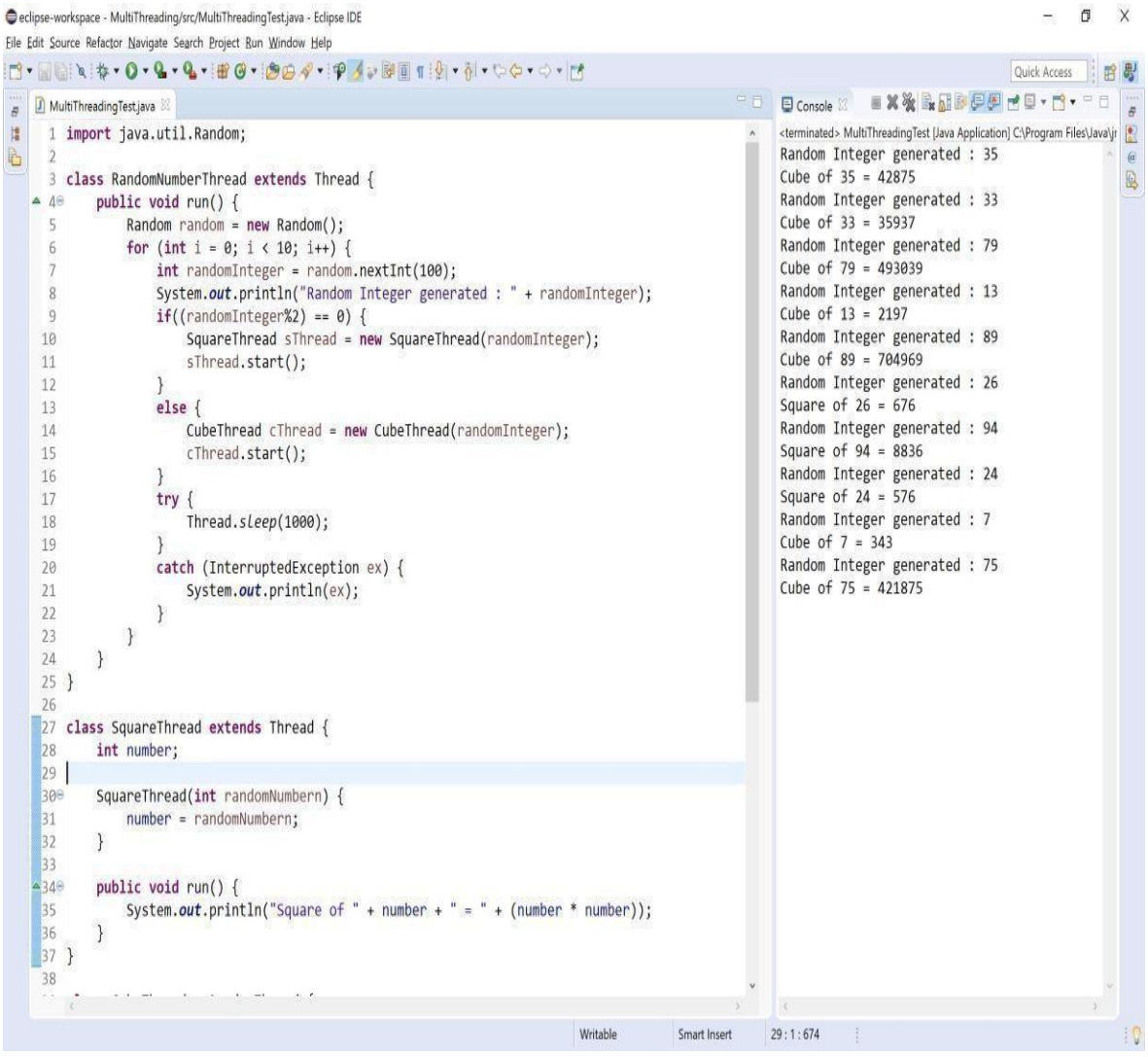
Week 4:

Write a Java program that implements a multi-thread application that has three threads. First thread generates random integer every 1 second and if the value is even, second thread computes the square of the number and prints. If the value is odd, the third thread will print the value of cube of the number.

```
import java.util.Random;
class RandomNumberThread 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);
	    if((randomInteger%2) == 0)
	    {
	          SquareThread sThread = new SquareThread(randomInteger);
	          sThread.start();
	    }
	    else
	            {     CubeThread cThread = new CubeThread(randomInteger);
	                  cThread.start();

	            }
	            try   Thread.sleep(1000);
	            {

	            }
```

```
                    catch (InterruptedException ex)
                    {
                    System.out.println(ex);
                    }
              }
        }
  }
  class SquareThread extends Thread {
        int number;
        SquareThread(int
              randomNumbern)  {number
              = randomNumbern;
        }

        public void run() {
              System.out.println("Square of " + number + " = " + (number *number));
        }
  }
  class CubeThread extends Thread {
        int number;

        CubeThread(int
              randomNumber) {
              number =
              randomNumber;
        }

        public void run() {
              System.out.println("Cube of " + number + " = " + number * number
              *
number);
        }

        }
  public class MultiThreadingTest {
        public static void main(String args[]) {
              RandomNumberThread rnThread = new
              RandomNumberThread();rnThread.start();
        }
  }
```

Output:


eclipse-workspace - MultiThreading/src/MultiThreadingTest.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
MultiThreadingTest.java
1 import java.util.Random;
2
3 class RandomNumberThread extends Thread {
4 public void run() {
5 Random random = new Random();
6 for (int i = 0; i < 10; i++) {
7 int randomInteger = random.nextInt(100);
8 System.out.println("Random Integer generated : " + randomInteger);
9 if((randomInteger%2) == 0) {
10 SquareThread sThread = new SquareThread(randomInteger);
11 sThread.start();
12 }
13 else {
14 CubeThread cThread = new CubeThread(randomInteger);
15 cThread.start();
16 }
17 try {
18 Thread.sleep(1000);
19 }
20 catch (InterruptedException ex) {
21 System.out.println(ex);
22 }
23 }
24 }
25 }
26
27 class SquareThread extends Thread {
28 int number;
29
30 SquareThread(int randomNumbern) {
31 number = randomNumbern;
32 }
33
34 public void run() {
35 System.out.println("Square of " + number + " = " + (number * number));
36 }
37 }
38
Console
<terminated> MultiThreadingTest [Java Application] C:\Program Files\Java\jr
Random Integer generated : 35
Cube of 35 = 42875
Random Integer generated : 33
Cube of 33 = 35937
Random Integer generated : 79
Cube of 79 = 493039
Random Integer generated : 13
Cube of 13 = 2197
Random Integer generated : 89
Cube of 89 = 704969
Random Integer generated : 26
Square of 26 = 676
Random Integer generated : 94
Square of 94 = 8836
Random Integer generated : 24
Square of 24 = 576
Random Integer generated : 7
Cube of 7 = 343
Random Integer generated : 75
Cube of 75 = 421875
Writable
Smart Insert
29 : 1 : 674

Week 5:

Write a Java program for the following: Create a doubly linked list of elements. Display the contents of the list after insertion.

```
public class DoubleLinkedList
{
	class Node
	{	int data; Node previous;Node next;
		public Node(int data)
		{
			this.data = data;
		}
	}
	Node head, tail = null;
	public void addNode(int data)
	{
		Node newNode = new Node(data);
		if (head == null)
		{
			head = tail = newNode;
			head.previous = null;
			tail.next = null;
		}
		else
		{
			tail.next = newNode;
			newNode.previous = tail;
			tail = newNode;
			tail.next = null;
		}
	}
	public void display()
	{
		if (head == null)
		{
			System.out.println("List is empty");
			return;
		}
		System.out.println("Nodes of doubly linked list:  ");
		 for(Node t=head; t!= null; t=t.next)
			System.out.print(t.data + " ");
	}
	public static void main(String[] args)
	{
		DoubleLinkedList dList = new DoubleLinkedList();
		 dList.addNode(1);
		 dList.addNode(2);
		 dList.addNode(3);
		 dList.addNode(4);
		 dList.addNode(5);
		 dList.display();
	}
}
```

OUTPUT:

Nodes of doubly linked list: 1 2 3 4 5

Week 6:

Write a Java program that simulates a traffic light. The program lets the user select one of three lights: red, yellow, or green with radio buttons. On selecting a button, an appropriate message with "Stop" or "Ready" or "Go" should appear above the buttons in selected color. Initially, there is no message shown.

```
import java.applet.Applet;
import java.awt.*;
import java.awt.event.*;

   //<applet code = "TrafficLights" width = 1000 height = 500></applet>

public class TrafficLights extends Applet implements ItemListener
{
        CheckboxGroup grp = new CheckboxGroup();
        Checkbox redLight, yellowLight, greenLight;
        Label msg;

        public void init()
        {
                redLight = new Checkbox("Red", grp, false);
                yellowLight = new Checkbox("Yellow", grp, false);
                greenLight = new Checkbox("Green", grp, false);
                msg = new Label("                    ");

                redLight.addItemListener(this);
                yellowLight.addItemListener(this);
                greenLight.addItemListener(this);

                add(redLight);
                add(yellowLight);
                add(greenLight);
                add(msg);
                msg.setFont(new Font("Serif", Font.BOLD, 20));
        }
        public void itemStateChanged(ItemEvent ie)
        {
                redLight.setForeground(Color.BLACK);
                yellowLight.setForeground(Color.BLACK);
                greenLight.setForeground(Color.BLACK);

                if(redLight.getState() == true) {
                        redLight.setForeground(Color.RED);
                        msg.setForeground(Color.RED);
                        msg.setText("STOP");
                }
```

```
            else if(yellowLight.getState() == true)
            {
                    yellowLight.setForeground(Color.YELLOW);
                    msg.setForeground(Color.YELLOW);
                    msg.setText("READY");
            }
            else
            {

                   greenLight.setForeground(Color.GREEN);
                    msg.setForeground(Color.GREEN);
                    msg.setText("GO");

              }

        }

}
```

Output:

WEEK 7:

Write a Java program to create an abstract class named Shape that contains two integers and an empty method named print Area (). Provide three classes named Rectangle, Triangle, and Circle such that each one of the classes extends the class Shape. Each one of the classes contains only the method print Area () that prints the area of the given shape.

Source code:

```
import java.util.*;

abstract class Shape
{
	int length, breadth, radius;
	Scanner input = new Scanner(System.in);
	abstract void printArea();
}

class Rectangle extends Shape {
	 void printArea()
	 {
		System.out.println("*** Finding the Area of Rectangle ***");
		System.out.print("Enter length and breadth: ");
		length = input.nextInt();
		breadth = input.nextInt();
		System.out.println("The area of Rectangle is: " + length * breadth);
	}
}

class Triangle extends Shape {
	void printArea() {
		System.out.println("\n*** Finding the Area of Triangle ***");
		System.out.print("Enter Base And Height: ");
		length = input.nextInt();
		breadth = input.nextInt();
		System.out.println("The area of Triangle is: " + (length * breadth)/2);
	}
}
class Cricle extends Shape {
	void printArea() {
		System.out.println("\n*** Finding the Area of Cricle ***");
		System.out.print("Enter Radius: ");
		radius = input.nextInt();
		System.out.println("The area of Cricle is: " + 3.14f * radius * radius);
	}
}
public class AbstractClass {
	public static void main(String[] args) {
		Rectangle rec = new Rectangle();
		rec.printArea();

		Triangle tri = new Triangle();
		tri.printArea();

		Cricle cri = new Cricle();
		cri.printArea();
	}
}
```

Output:

➔ javac AbstractClass.java

➔java AbstractClass

*** Finding the Area of Rectangle ***
Enter length and breadth: 2 3
The area of Rectangle is: 6

*** Finding the Area of Triangle ***
Enter Base And Height: 3 3
The area of Triangle is: 4

*** Finding the Area of Cricle ***
Enter Radius: 3
The area of Cricle is: 28.26

WEEK 8:

Suppose that a table named Table.txt is stored in a text file. The first line in the file header and the remaining lines correspond to row in the table. The elements are separated by commas. Write a Java program to display the table using labels in grid layout.

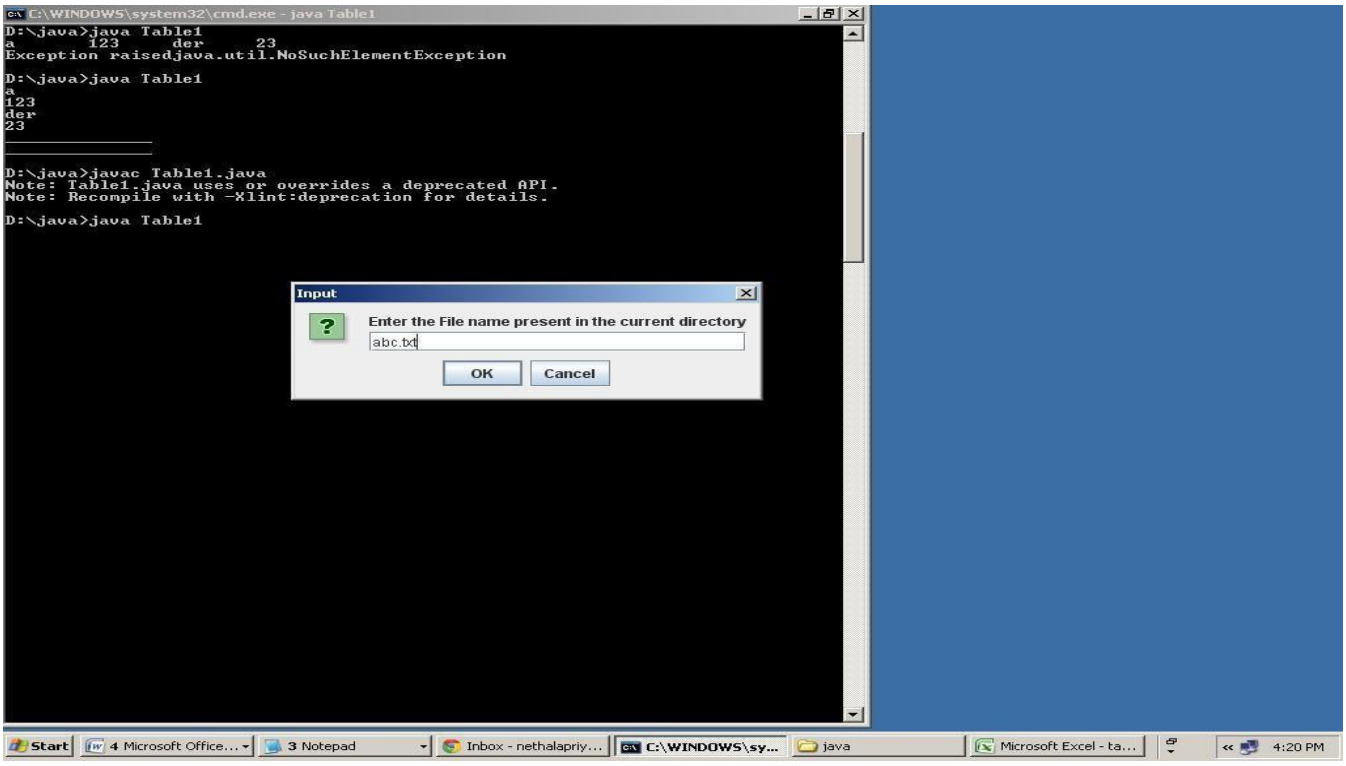
```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import java.util.*;
import java.io.*;
public class Table1 extends JFrame
{
        int i=0, j=0,k=0;
        Object data[][]=new Object[5][4];
        Object list[][]=new Object[5][4];
        JButton save;
        JTable table1;
        FileInputStream fis;
        DataInputStream dis;
        public Table1()
        {
                String d= " ";
                Container con=getContentPane();
                con.setLayout(new BorderLayout());
                final String[] colHeads={"Name","Roll Number","Department","Percentage"};
                try
                {
                   String s=JOptionPane.showInputDialog("Enter a File name");
                   FileInputStream fis=new FileInputStream(s);
                   DataInputStream dis = new DataInputStream(fis);
                   while ((d=dis.readLine())!=null)
                   {
                        StringTokenizer st1=new StringTokenizer(d,",");
                        while (st1.hasMoreTokens())
                           {
                           for (j=0;j<4;j++)
                           {
```

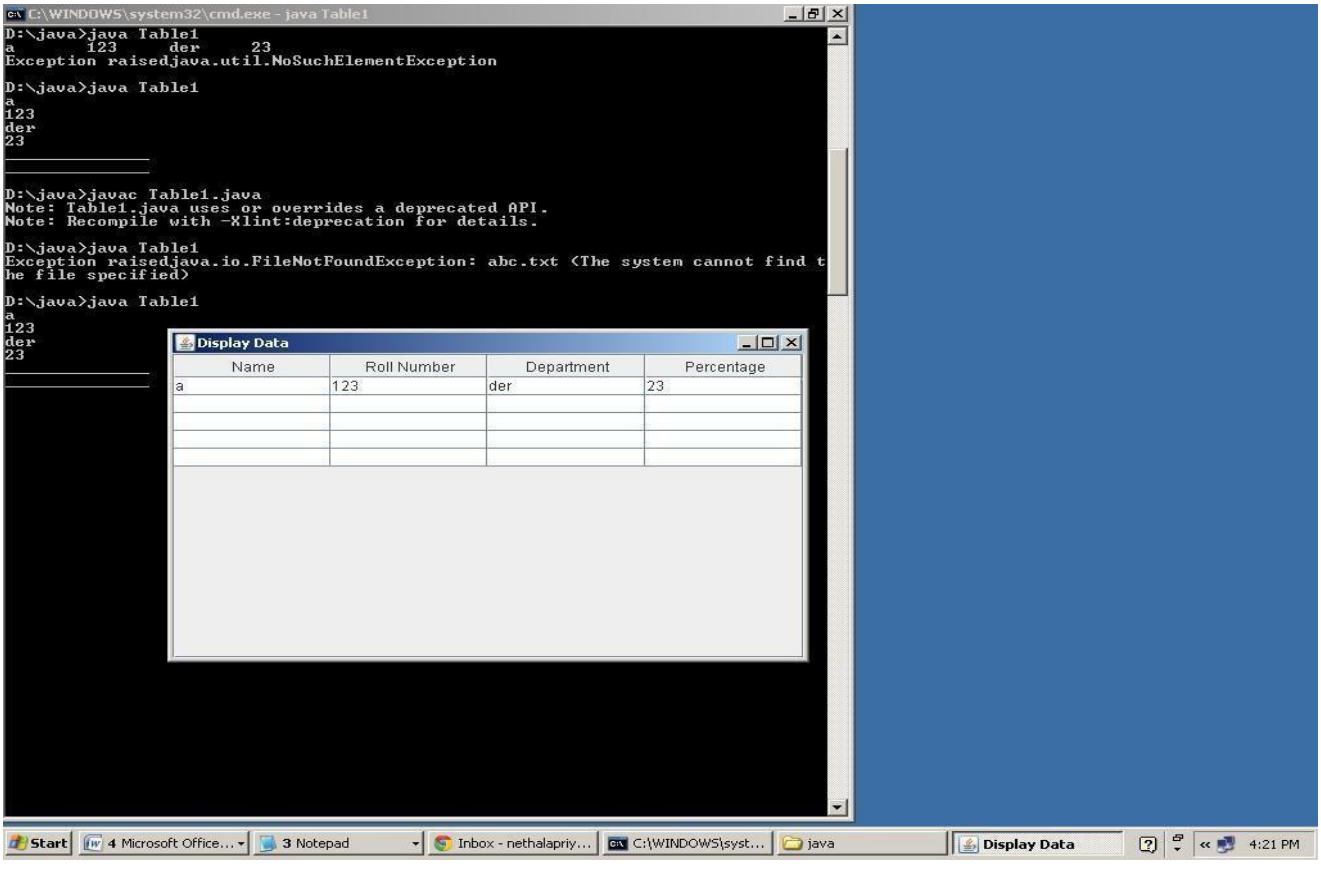
```
                                        data[i][j]=st1.nextToken();
                                        System.out.println(data[i][j]);
                                    }
                                    i++;
                                }
                                 System.out.println ("_");
                            }
                    } catch (Exception e)
                      {
                            System.out.println ("Exception raised" +e.toString());
                      }
                      table1=new JTable(data,colHeads);
                      int  v=ScrollPaneConstants.VERTICAL_SCROLLBAR_AS_NEEDED;
                      int h=ScrollPaneConstants.HORIZONTAL_SCROLLBAR_AS_NEEDED;
                      JScrollPane scroll=new JScrollPane(table1,v,h);
                      con.add(scroll,BorderLayout.CENTER);
                }
        public static void main(String args[])
          {

                Table1 t=new Table1();

                t.setBackground(Color.green);

                t.setTitle("Display Data");

                t.setSize(500,300);

                t.setVisible(true);

                t.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        }
}
```

Abc.txt:-

a,123,der,23b,456,frg,45

Output:-


C:\WINDOWS\system32\cmd.exe - java Table1
D:\java>java Table1
a 123 der 23
Exception raisedjava.util.NoSuchElementException
D:\java>java Table1
a
123
der
23
D:\java>javac Table1.java
Note: Table1.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
D:\java>java Table1
Input
Enter the File name present in the current directory
abc.txt
OK
Cancel
Start
4 Microsoft Office...
3 Notepad
Inbox - nethalapriy...
C:\WINDOWS\sy...
java
Microsoft Excel - ta...
4:20 PM


C:\WINDOWS\system32\cmd.exe - java Table1
D:\java>java Table1
a 123 der 23
Exception raisedjava.util.NoSuchElementException
D:\java>java Table1
a
123
der
23
D:\java>javac Table1.java
Note: Table1.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
D:\java>java Table1
Exception raisedjava.io.FileNotFoundException: abc.txt (The system cannot find t
he file specified)
D:\java>java Table1
a
123
der
23
Display Data
Name
Roll Number
Department
Percentage
a
123
der
23
Start
4 Microsoft Office...
3 Notepad
Inbox - nethalapriy...
C:\WINDOWS\syst...
java
Display Data
4:21 PM

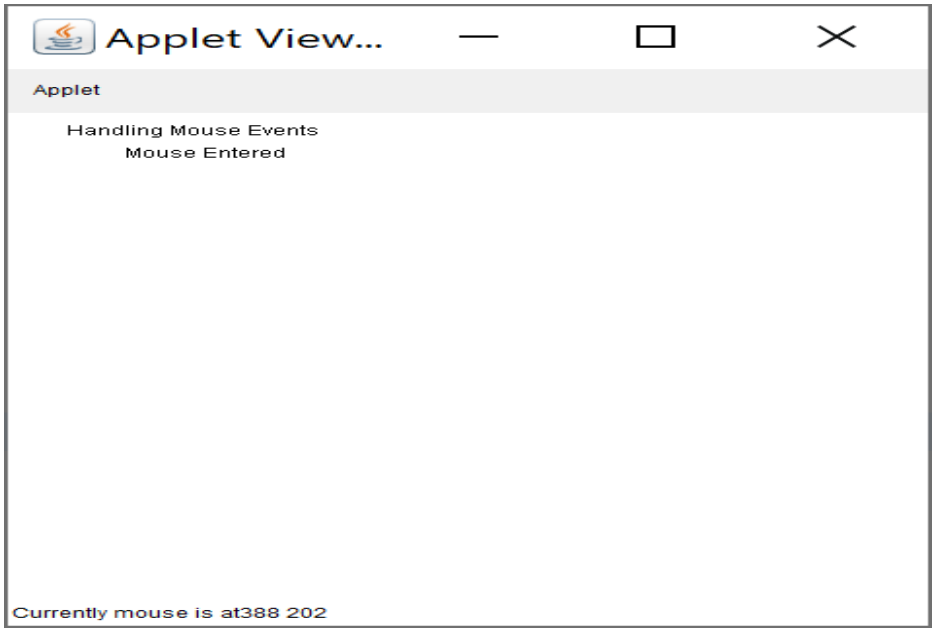
Week 9:

Write a Java program that handles all mouse events and shows the event name at the center of the window when a mouse event is fired (Use Adapter classes).

```
import java.awt.*;
import java.applet.*;
import java.awt.event.*;

/*<applet code="MouseDemo" width=300 height=300> </applet>*/

public class MouseDemo extends Applet implements MouseListener, MouseMotionListener
{
	int mx = 0,my = 0;
	String msg = "";

	public void init()
	 {
		addMouseListener(this);
		addMouseMotionListener(this);
	}

	public void mouseClicked(MouseEvent me)
	 {
		mx = 20;
		my = 40;
		msg = "Mouse Clicked";
		repaint();
	}

	public void mousePressed(MouseEvent me)
	{
		mx = 30;
		my = 60;
		msg = "Mouse Pressed";
		repaint();
	}

	public void mouseReleased(MouseEvent me)
	{
		mx = 30;
		my = 60;
		msg = "Mouse Released";
		repaint();
	}

	public void mouseEntered(MouseEvent me)
	{
		mx = 40;
		my = 80;
		msg = "Mouse Entered";
		repaint();
	}
```

```
    public void mouseExited(MouseEvent me)
    {
        mx = 40;
        my = 80;
        msg = "Mouse Exited";
        repaint();
    }

    public void mouseDragged(MouseEvent me)
    {
        mx = me.getX();
        my = me.getY();
        showStatus("Currently mouse dragged" + mx + " " + my);
        repaint();
    }

    public void mouseMoved(MouseEvent me)
    {
        mx = me.getX();
        my = me.getY();
        showStatus("Currently mouse is at" + mx + " " + my);
        repaint();
    }

    public void paint(Graphics g)
    {
        g.drawString("HandlingMouse Events", 30, 20);
        g.drawString(msg, 60, 40);
    }
}
```

Output:



Week 10:

Write a java program that loads names and phone numbers from a text file where the data is organized as one line per record and each field in a record are separated by a tab (\t).it takes a name or phone number as input and prints the corresponding other value from the hash table
(hint: use hash tables)

```
import java.io.BufferedReader;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;
import java.util.Hashtable;
import java.util.Iterator;
import java.util.Set;

public class HashTab {
	public static void main(String[] args) {
		HashTab prog11 = new HashTab();
		Hashtable<String, String> hashData = prog11.readFromFile("HashTab.txt");
		System.out.println("File data into Hashtable:\n" + hashData);
		prog11.printTheData(hashData, "raja");
		prog11.printTheData(hashData, "123");
		prog11.printTheData(hashData, "----");
	}

	private void printTheData(Hashtable<String, String> hashData, String input) {
		String output = null;
		if (hashData != null)
		{
			Set<String> keys = hashData.keySet();
			if (keys.contains(input))
			{
				output = hashData.get(input);
			}
			else
			{
				Iterator<String> iterator = keys.iterator();
				while (iterator.hasNext())
				{
					String key = iterator.next();
					String value = hashData.get(key);
					if (value.equals(input))
					{
						output = key;
						break;
					}
				}
			}
		}

		System.out.println("Input given:" + input);
		if (output != null)
		{
			System.out.println("Data found in HashTable:" + output);
		} else {
			System.out.println("Data not found in HashTable");
		}
	}
```

```
	private Hashtable<String, String> readFromFile(String fileName)
	{
		Hashtable<String, String> hashData = new Hashtable<String, String>();
		try
		{
		     File f = new File("C:\\Users\\sriindu\\Desktop\\sasi\\" + fileName);
		      BufferedReader br = new BufferedReader(new FileReader(f));
		      String line = null;
		      while ((line = br.readLine()) != null)
		      {
				String[] details = line.split("\t");
				hashData.put(details[0], details[1]);
		      }
		}

		catch (FileNotFoundException e)
		{
			e.printStackTrace();
		}
		catch (IOException e)
		{
			e.printStackTrace();
		}
		return hashData;
	}
}
```

HashTab.txt

raja 123
rani 456
anu 789

Output:

C:\Users>javac HashTab.java

C:\Users>java HashTab

File data into Hashtable:
{anu=789, rani=456, raja=123}
Input given : raja
Data found in HashTable : 123
Input given : 123
Data found in HashTable : raja
Input given:
Data not found in HashTable

Week – 11

Write a Java program that correctly implements the producer – consumer problem using the concept of interthread communication.

Source Code:

```
class ItemQueue {
	int item;
	boolean valueSet = false;

	synchronized int getItem()

	{
		while (!valueSet)
			try
			{
				wait();
			}
			catch (InterruptedException e)
			{
				System.out.println("InterruptedException caught");
			}
		   System.out.println("Consummed:" + item);
		   valueSet = false;
		   try
		   {
		      Thread.sleep(1000);
		   }
		   catch (InterruptedException e)
		   {
		       System.out.println("InterruptedException caught");
		   }
		   notify();
		   return item;
	}
	synchronized void putItem(int item)
	{
		while (valueSet)
			try
			{
				wait();
			}
			catch (InterruptedException e)
			{
				System.out.println("InterruptedException caught");
			}
			this.item = item;
			valueSet = true;
			System.out.println("Produced: " + item);
			try
			{
				Thread.sleep(1000);
			}
			catch (InterruptedException e)
			{
				System.out.println("InterruptedException caught");
			}
		notify();
	}
}
```

```
class Producer implements Runnable
{
	ItemQueue itemQueue;
	Producer(ItemQueue itemQueue)
	{
		this.itemQueue = itemQueue;
		new Thread(this, "Producer").start();
	}
	public void run()
	{
		int i = 0;
		while(true)
		{
			itemQueue.putItem(i++);
		}
	}
}

class Consumer implements Runnable
{
	ItemQueue itemQueue;
	Consumer(ItemQueue itemQueue)
	{
		this.itemQueue = itemQueue;
		new Thread(this, "Consumer").start();
	}
	public void run()
	{
		while(true)
		{
			itemQueue.getItem();
		}
	}
}

class ProducerConsumer
{
	public static void main(String args[])
	{
		ItemQueue itemQueue = new ItemQueue();
		new Producer(itemQueue);
		new Consumer(itemQueue);

	}
```

Week – 12

Write a Java program to list all the files in a directory including the files present in all its subdirectories.

```
import java.util.Scanner;
import java.io.*;
public class ListingFiles
{
	public static void main(String[] args)
	{
		String path = null;
		Scanner read = new Scanner(System.in);
		System.out.print("Enter the root directory name: ");
		path = read.next() + ":\\";
		File f_ref = new File(path);
		if (!f_ref.exists())
			System.out.println("Root directory does not exists!");
		else
		{
		     String ch = "y";
		    while (ch.equalsIgnoreCase("y"))
		    {
		    printFiles(path);
		    System.out.print("Do you want to open any sub-directory (Y/N):  ");
		    ch = read.next().toLowerCase();
		    if (ch.equalsIgnoreCase("y"))
		    {
		    System.out.print("Enter the sub-directory name: ");
		    path = path + "\\\\" + read.next();
		    File f_ref_2 = new File(path);
		    if (! f_ref_2.exists())
		    {
		           System.out.println("The sub-directory does not exists!");
		           int lastIndex = path.lastIndexOf("\\");
		            path = path.substring(0, lastIndex);
		    }
		   }
		  }
		 }
	}

	public static void printFiles(String path)
	{
		System.out.println("Current Location: " + path);
		File f_ref = new File(path);
		File[] filesList = f_ref.listFiles();
		for (File file : filesList)
		{
		       if (file.isFile())
		              System.out.println("- " + file.getName());
		        else
		               System.out.println(">" + file.getName());
		 }
	 }
}
```

Output:

Produced: 1
Consumed: 1

Produced: 2
Consumed: 2

Produced: 3
Consumed: 3

Output:


C:\Windows\system32\cmd.exe - java ListingFiles
C:\utility\lab>javac ListingFiles.java
C:\utility\lab>java ListingFiles
Enter the root directory name: c
Current Location: c:\
> $$ AWS Classes
> $$ SLOKAM
> $Recycle.Bin
> $WinREAgent
> android 4.3
- android 4.3.zip
> cse4
> databases