Lab Manual

Java Programming

Computer Science & Engineering | Information Technology

(II- B. Tech. – II– Semester)

Regulation R18

1. Syllabus

CS408PC: JAVA PROGRAMMING LAB

B.Tech. II Year II Sem. **L T P C**
0 0 2 1

1. Use Eclipse or Net bean platform and acquaint 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 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.
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 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 interthread communication.
13. Write a Java program to list all the files in a directory including the files present in all its subdirectories.
14. Write a Java program that implements Quick sort algorithm for sorting a list of names in ascending Order.
15. Write a Java program that implements Bubble sort algorithm for sorting in descending order and also shows the number of interchanges occurred for the given set of integers.

PROGRAM OUTCOMES (PO's)

PO No.	**Program Outcomes (PO's)**
PO1	An ability to apply knowledge of computing, mathematics, science and engineering fundamentals appropriate to the discipline.
PO2	An ability to analyze a problem, and identify and formulate the computing requirements appropriate to its solution.
PO3	An ability to design, implement, and evaluate a computer-based system, process, component, or program to meet desired needs with appropriate consideration for public health and safety, cultural, societal and environmental considerations.
PO4	An ability to design and conduct experiments, as well as to analyze and interpret data.
PO5	An ability to use current techniques, skills, and modern tools necessary for computing practice.
PO6	An ability to analyze the local and global impact of computing on individuals, organizations, and society.
PO7	Knowledge of contemporary issues.
PO8	An understanding of professional, ethical, legal, security and social issues and responsibilities.
PO9	An ability to function effectively individually and on teams, including diverse and multidisciplinary, to accomplish a common goal.
PO10	An ability to communicate effectively with a range of audiences.
PO11	An understanding of engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects.
PO12	Recognition of the need for and an ability to engage in continuing professional development.

3.Lesson/Course Plan

Week No.	Name of the Program	No. of Hours required	Text Books	**Mode of Assessment**
1	Use Eclipse or Net bean platform and acquaint 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.	1	1	Viva&Execution
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.	1	T1	Viva&Execution
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.	1	T1	Viva&Execution
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.	1	T1	Viva&Execution
5	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.	1	T1	Viva&Execution
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.	1	T1	Viva&Execution
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 selected color. Initially, there is no message shown.	1	T1	Viva&Execution
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	1	T1	Viva&Execution

	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.	1	T1	Viva&Execution
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).	1	T1	Viva&Execution
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).	1	T1	Viva&Execution
12	Write a Java program that correctly implements the producer – consumer problem using the concept of interthread communication.	1	T1	Viva&Execution
13	Write a Java program to list all the files in a directory including the files present in all its subdirectories.	1	T1	Viva&Execution
14	Write a Java program that implements Quick sort algorithm for sorting a list of names in ascending Order	1	T1	Viva&Execution
15	Write a Java program that implements Bubble sort algorithm for sorting in descending order and also shows the number of interchanges occurred for the given set of integers.	1	T1	Viva&Execution
16	LEAD Experments	1	T1	Viva&Execution
	Total no of HOURS required to complete syllabus	**16**		

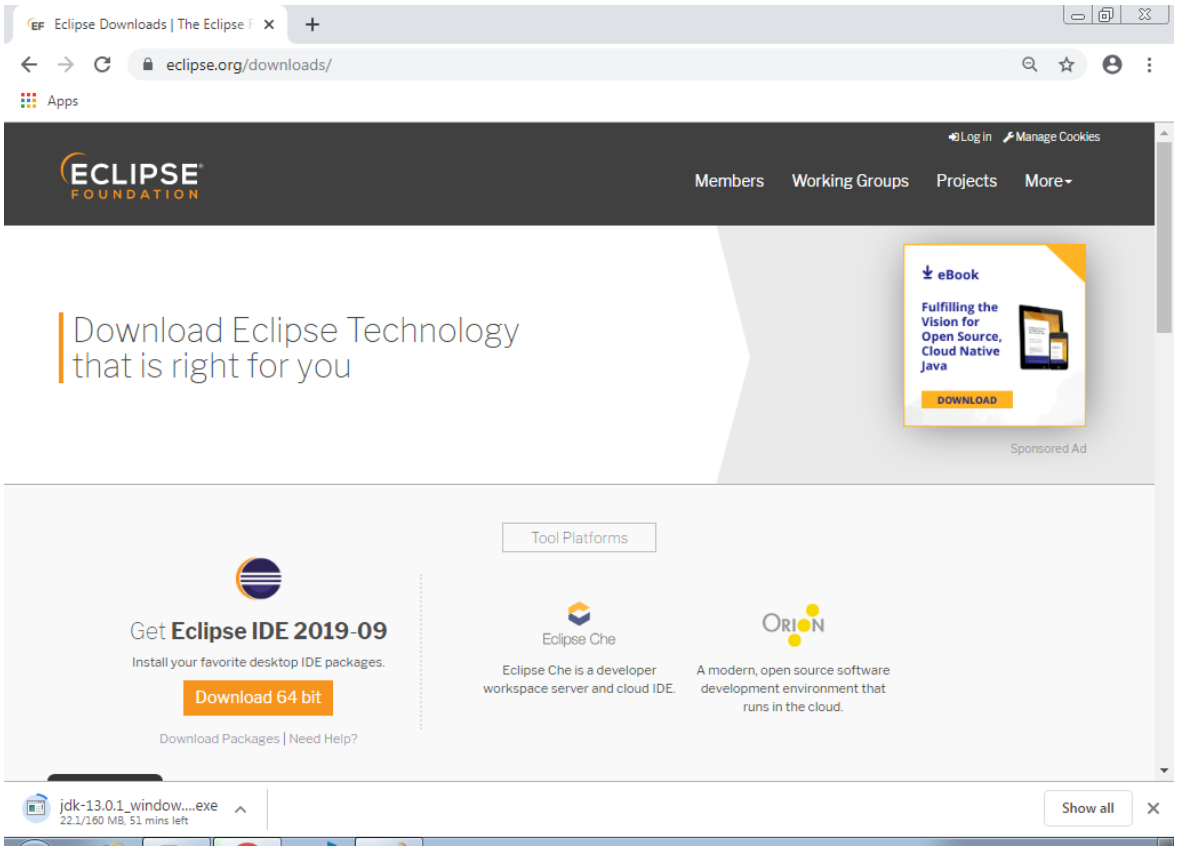
PROGRAMS

Week 1.

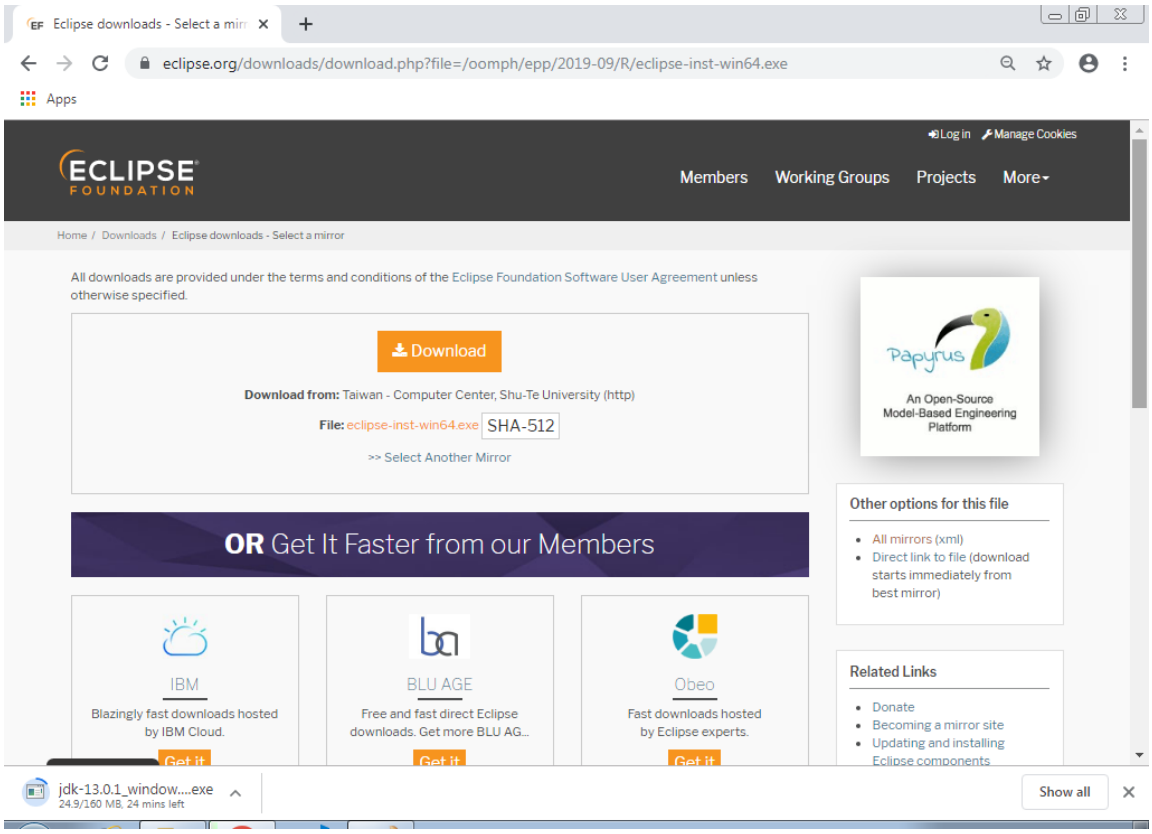
Aim: Use Eclipse or Net bean platform and acquaint with the various menus. Create a test project, add a test class, and run it. See how you can use auto suggestions, auto fill. Try code formatter and code refactoring like renaming variables, methods, and classes. Try debug step by step with a small program of about 10 to 15 lines which contains at least one if else condition and a for loop.

Solution:

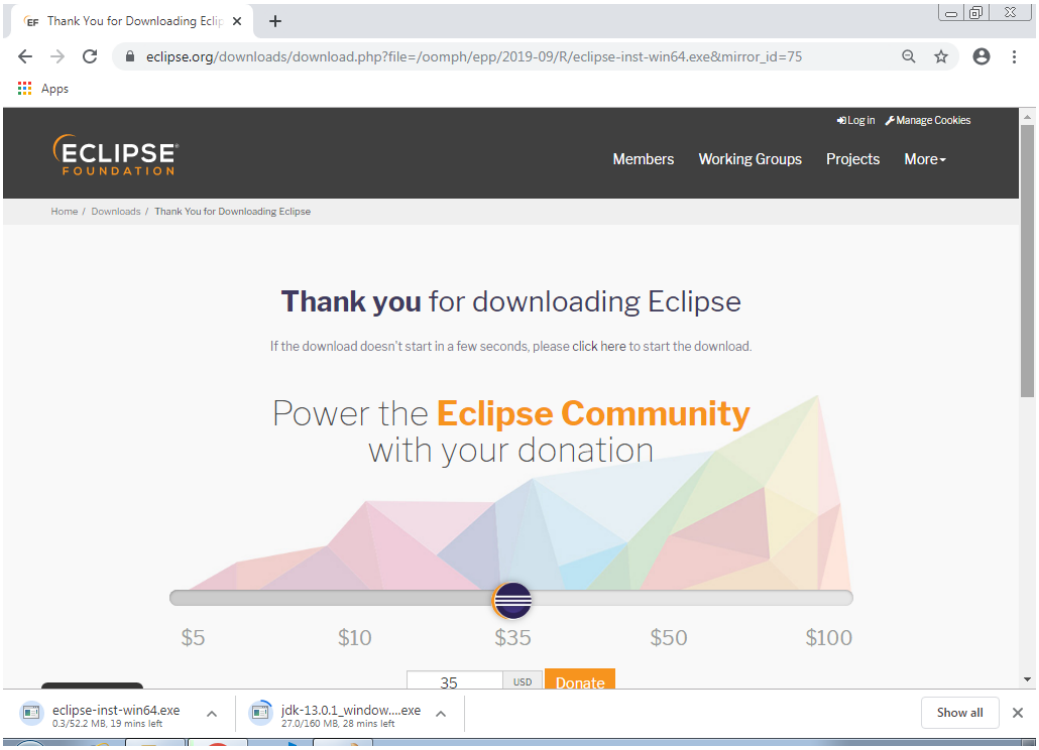
- **Step 1 -** Install JDK in the computer.
- **Step 2 -** Set the path in the Environment Variables from Advanced Setting of computer
- **Step 3 -** Download Eclipse from Eclipse website
- **Step 4 -** Install the Eclipse (follow the screen to install eclipse)



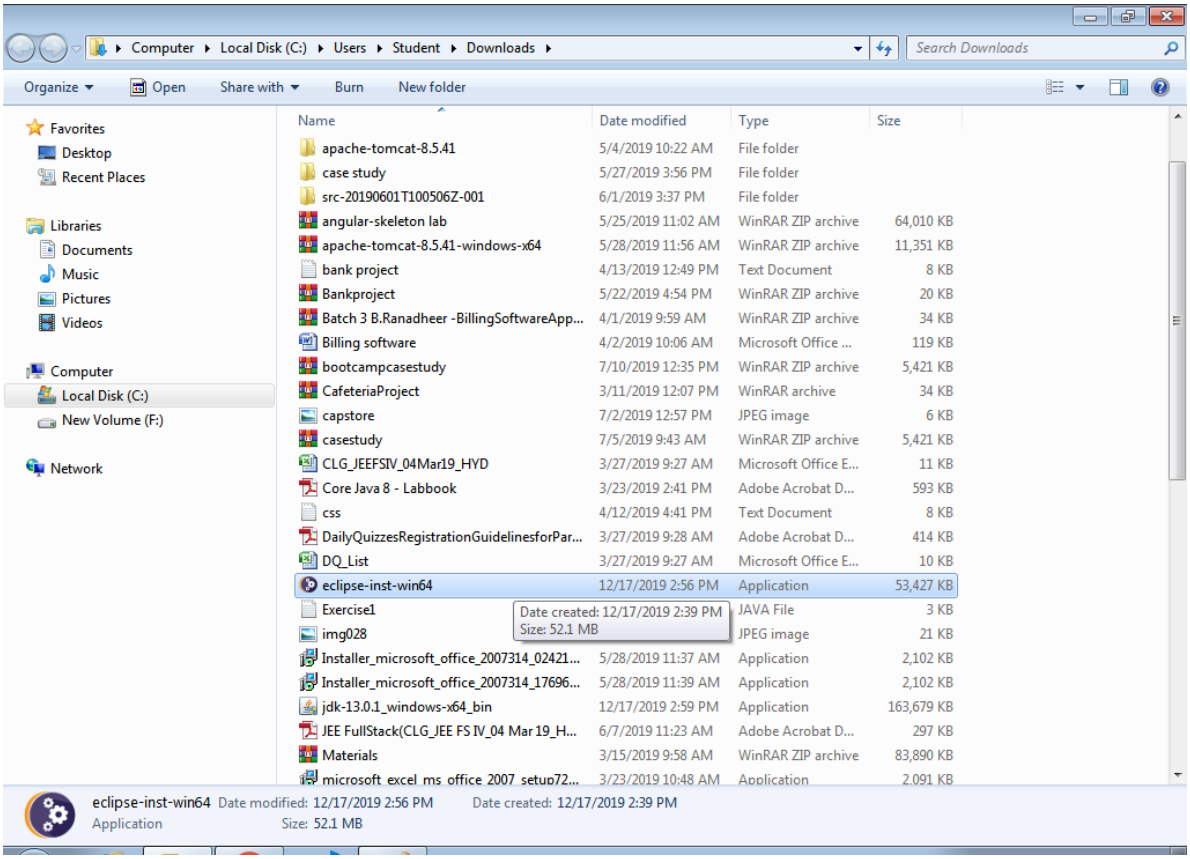
Select the sultable version based on your OS.



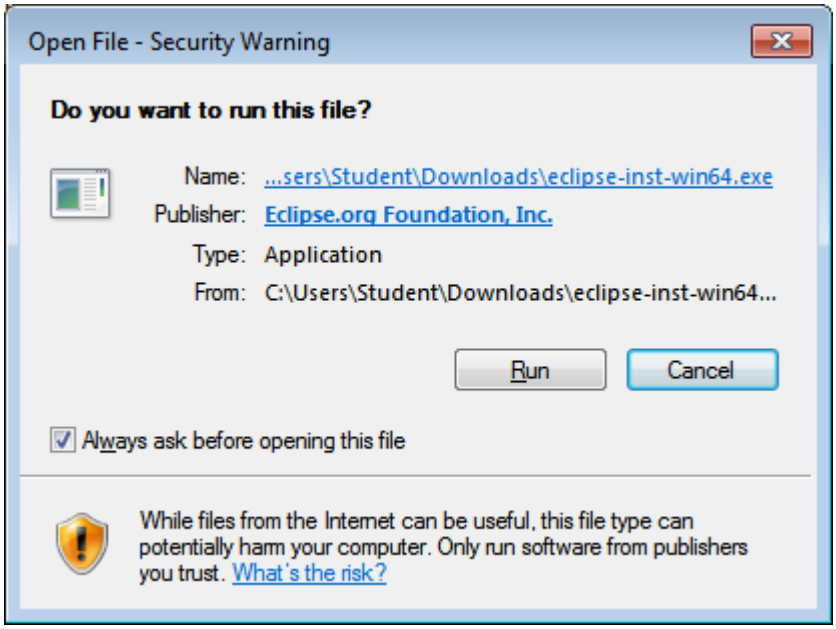
Then download get starts.



Double click on the Eclipse Application.



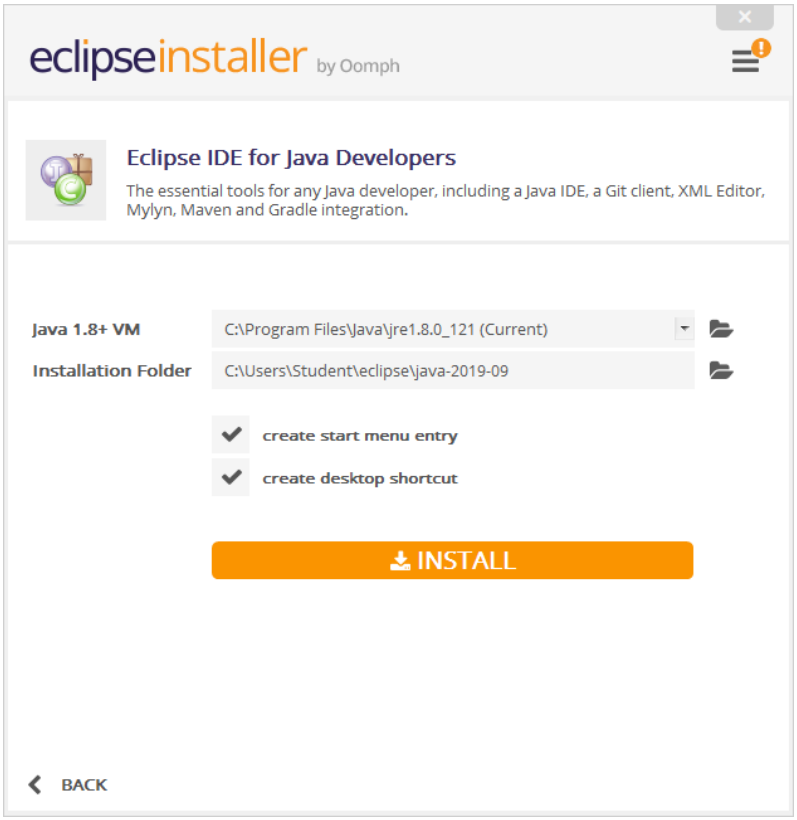
Click on Run in the Security Warning box.



Then, the installation process begins.

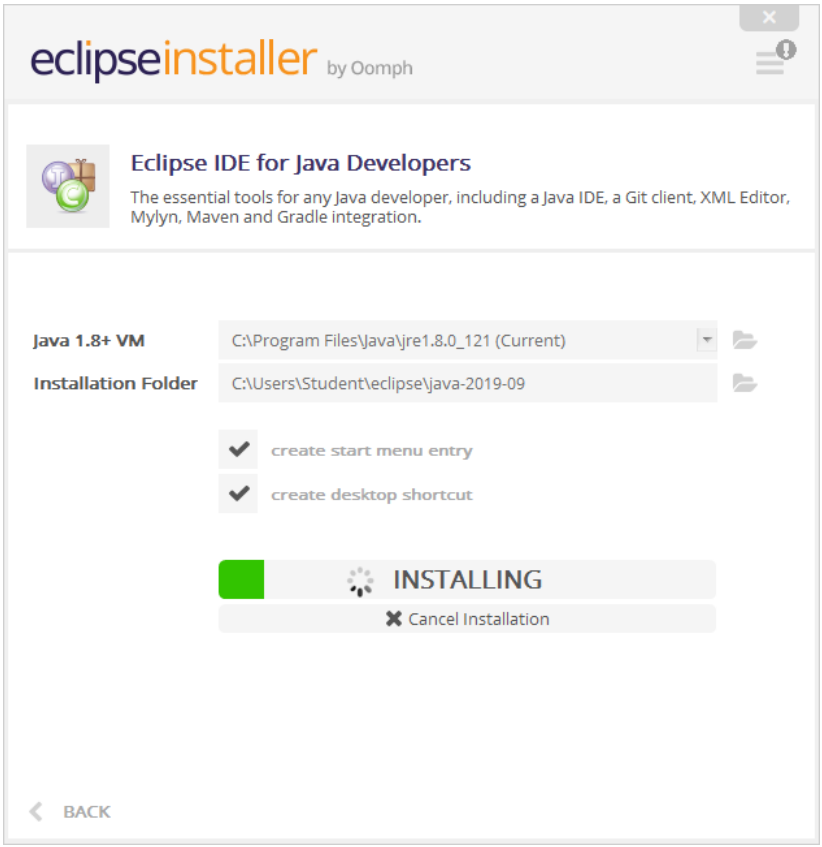
Click on Eclipse IDE for Java Developers.

Click on Install button.

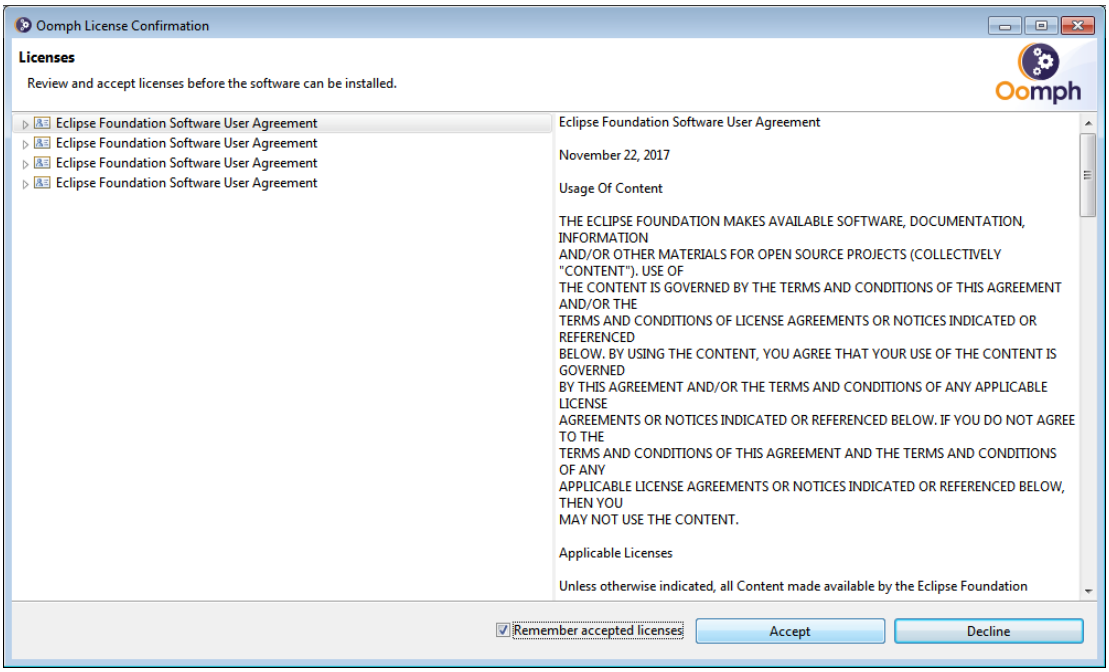


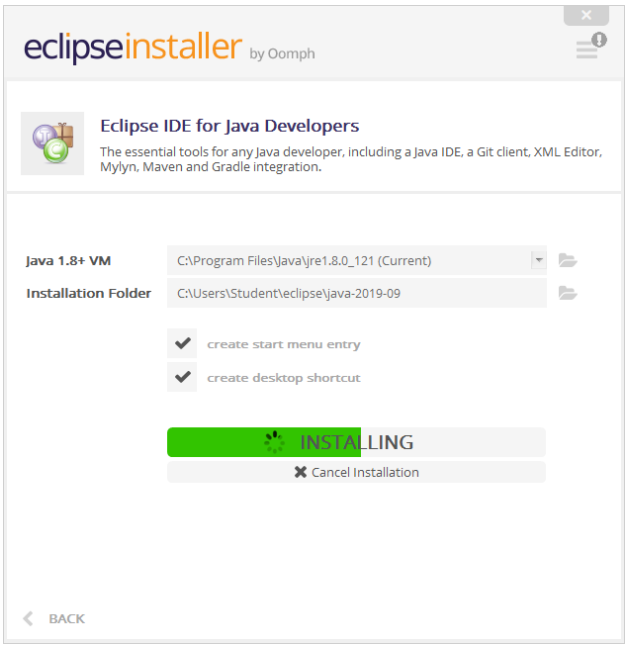
Click on Accept Now.

Then the Eclipse installation begins.

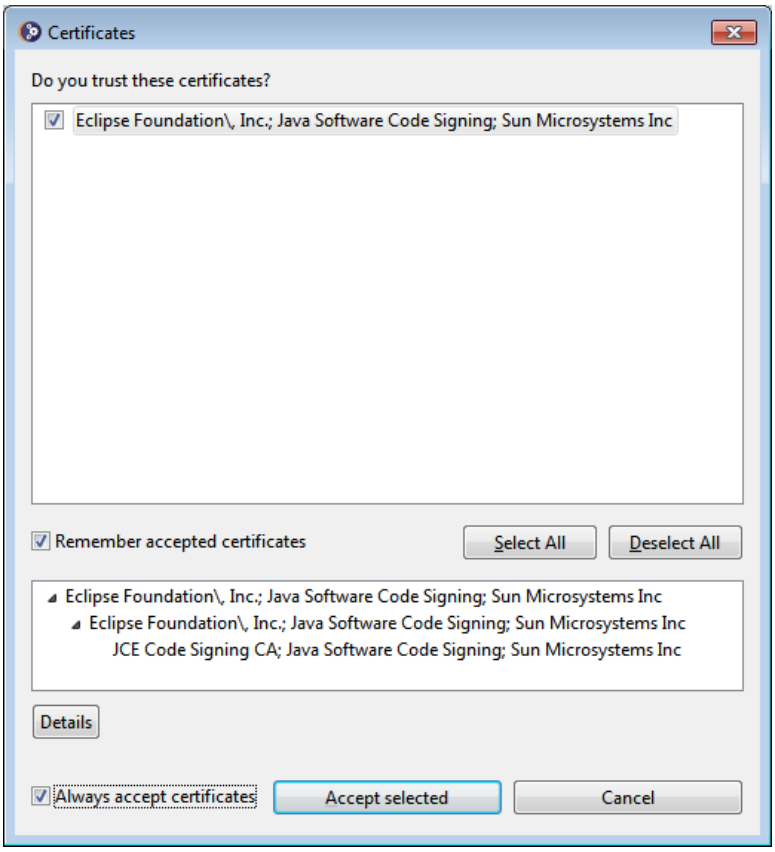


Click on Accept





Click on Select All and Accept Selected.

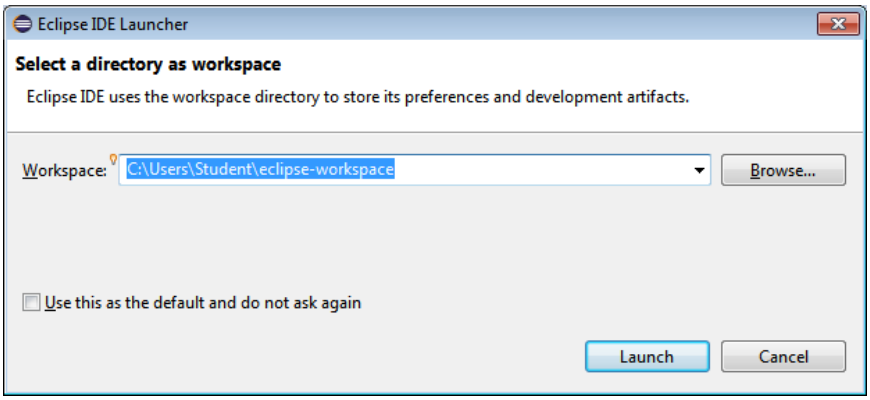


After completing, click on Launch to start the Eclipse IDE.

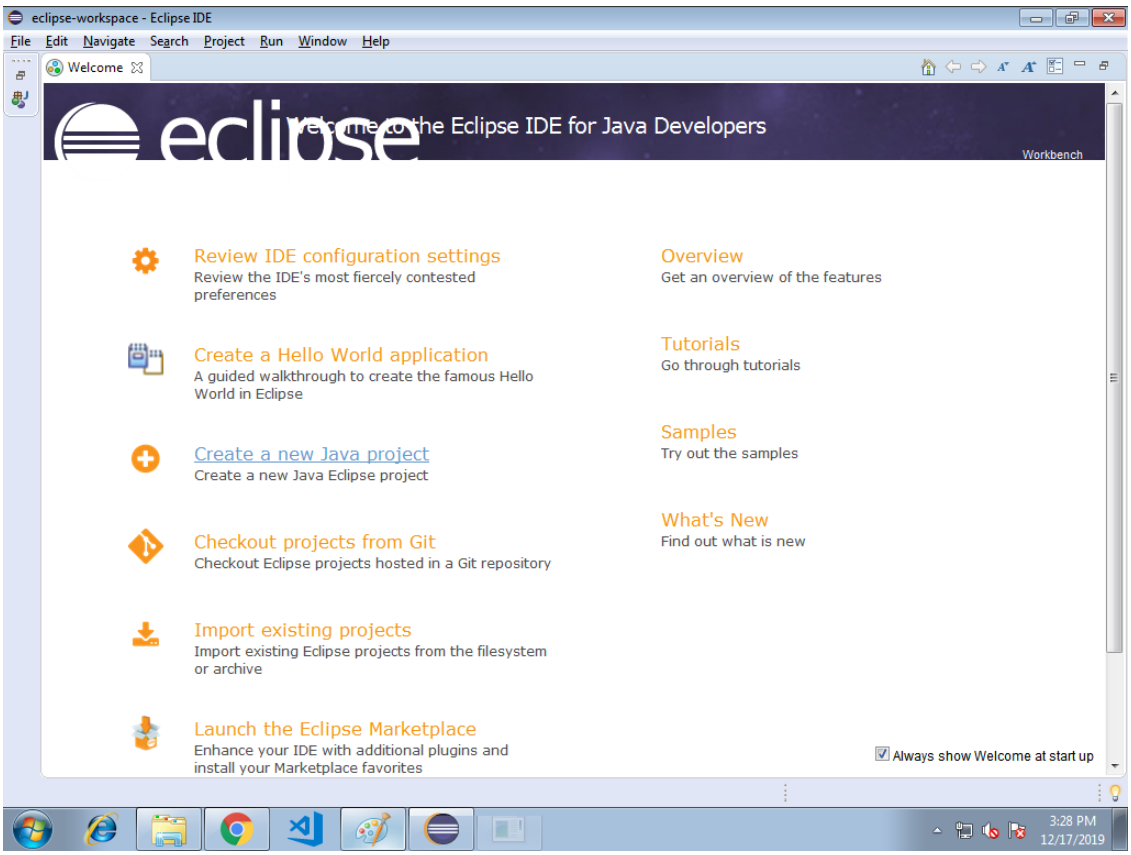

eclipseinstaller by Oomph
Eclipse IDE for Java Developers
The essential tools for any Java developer, including a Java IDE, a Git client, XML Editor, Mylyn, Maven and Gradle integration.
Java 1.8+ VM
C:\Program Files\Java\jre1.8.0_121 (Current)
Installation Folder
C:\Users\Student\eclipse\java-2019-09
create start menu entry
create desktop shortcut
LAUNCH
show readme file
open in system explorer
keep installer
BACK

CREATING PROJECT AND CLASSES IN ECLIPSE IDE

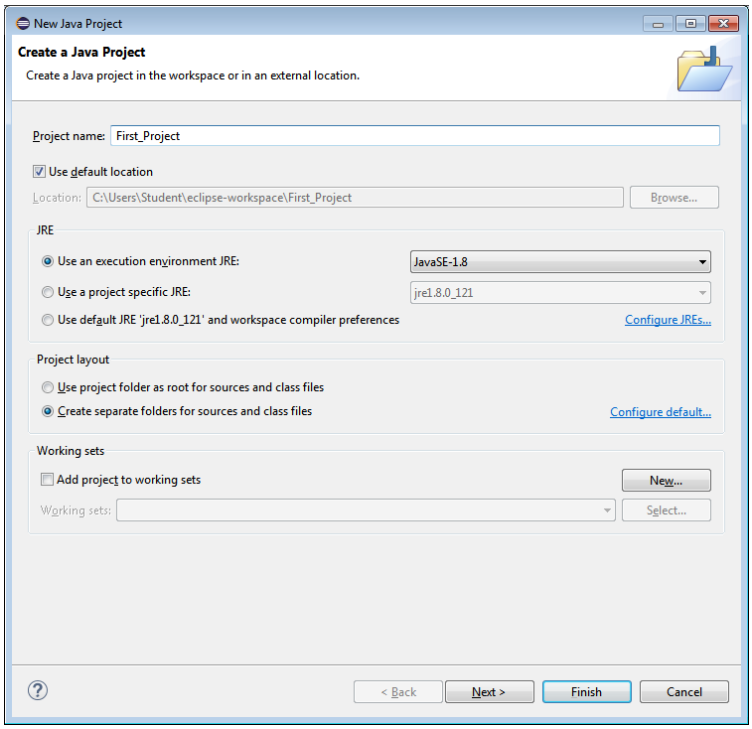
Browse the Workspace for storing the java project and click on Launch.



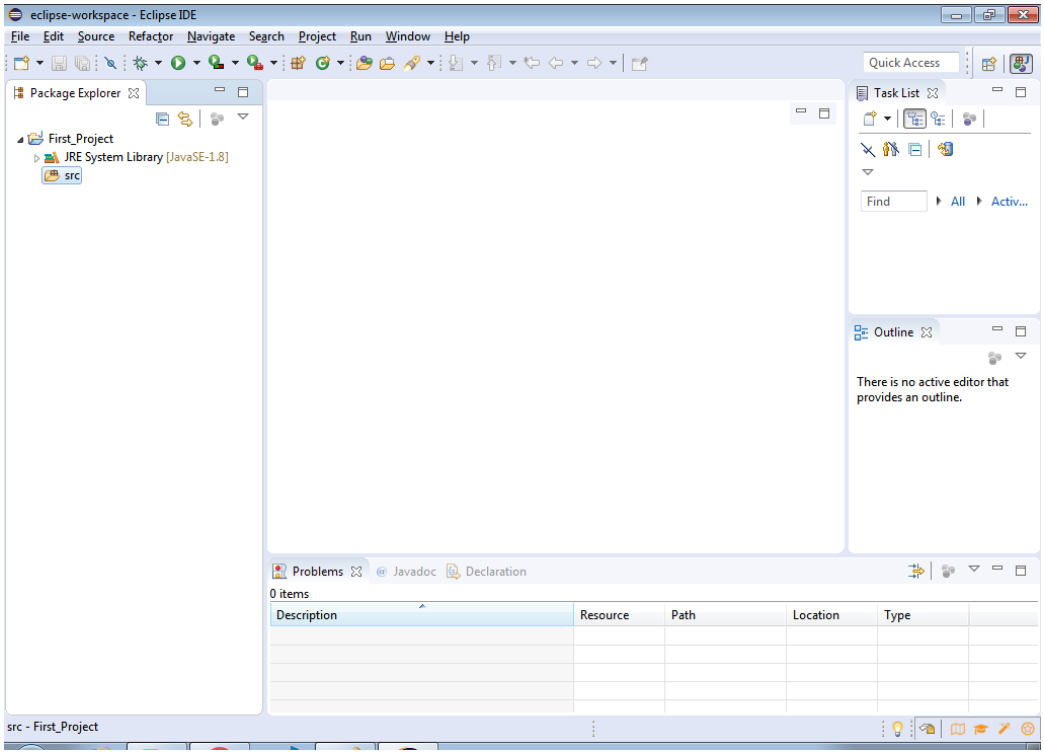
Select "Create a new Java project".

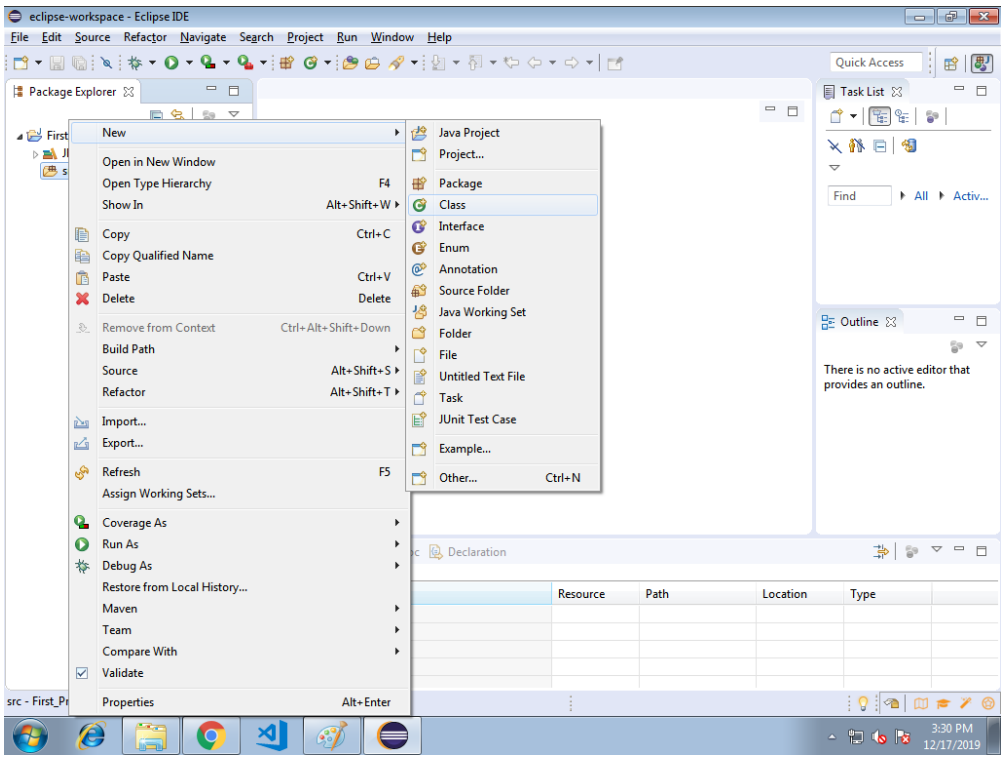


Type the project name and click on Finish.

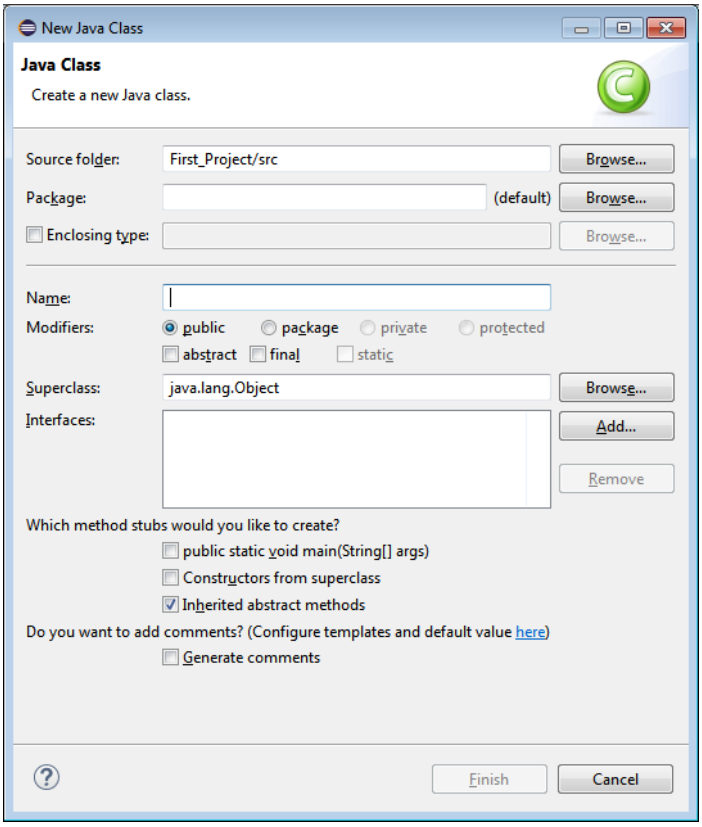


Now, create the class in src directory from Package Explorer window.

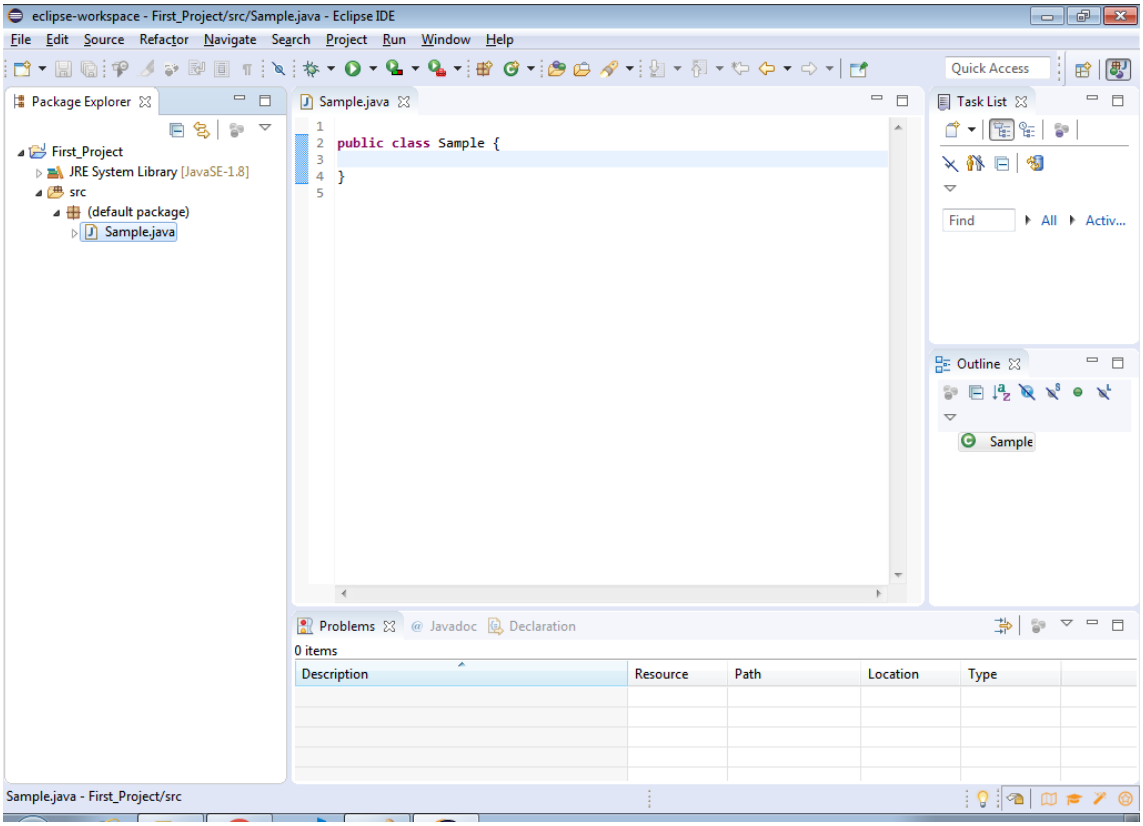




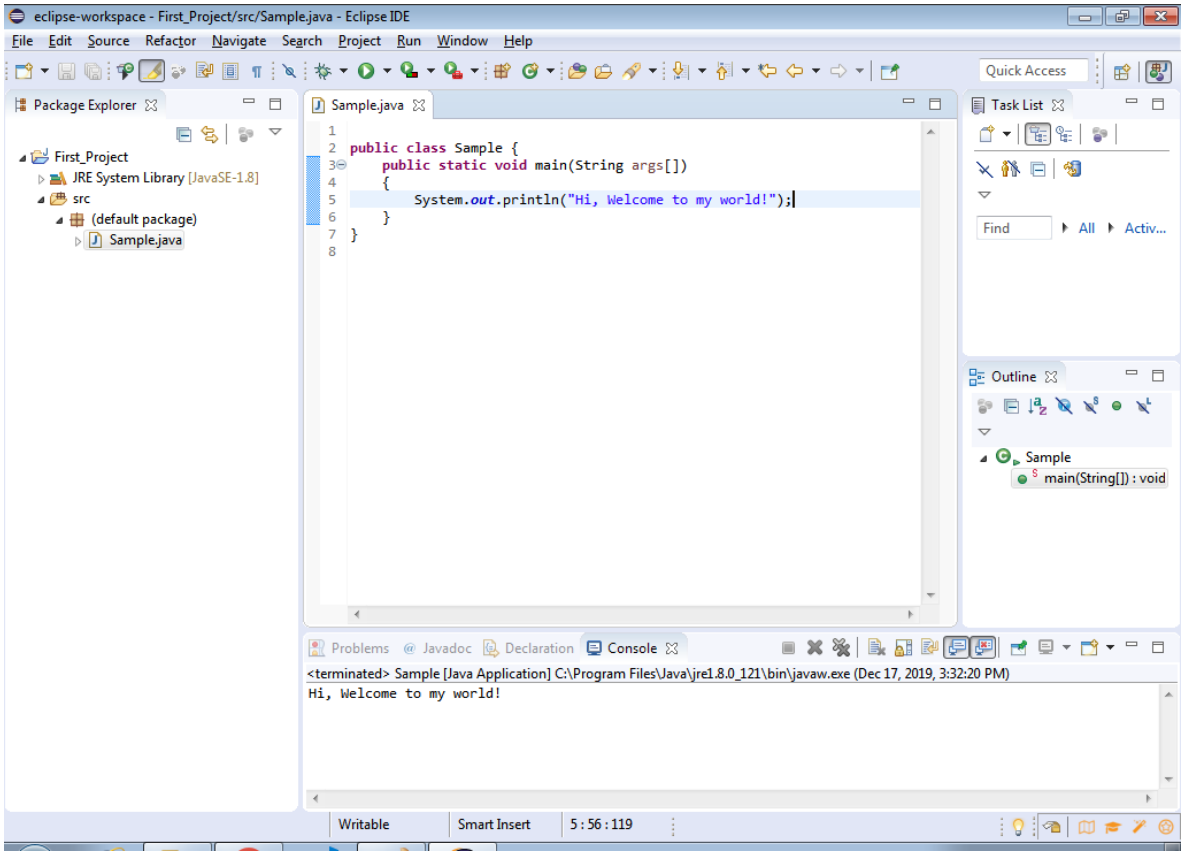
Type the class name and click on Finish.



Type the java code.



Click on Play button to run or execute the java code.



Week 2:

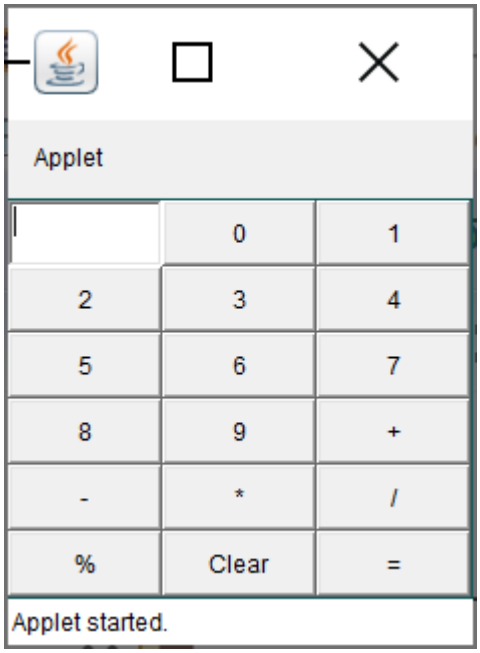
Write a Java program that works as a simple calculator. Use a grid layout to arrange buttons for the digits and for the +, -,*, % operations. Add a text field to display the result. Handle any possible exceptions like divided by zero.

Source Code:

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

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

public 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:



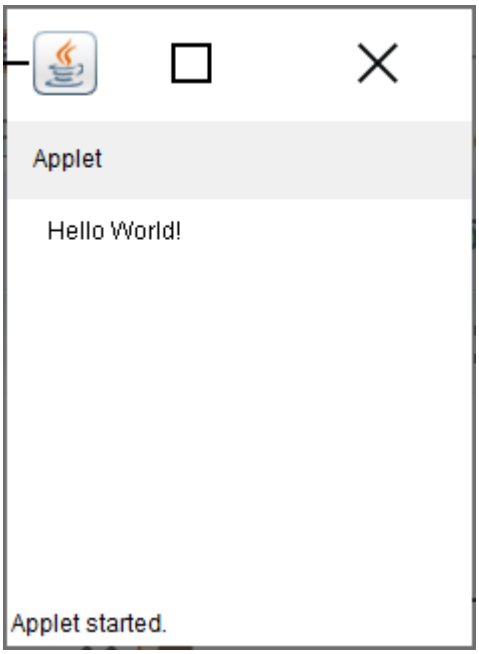
Week 3:

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.

Source code for question a:

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

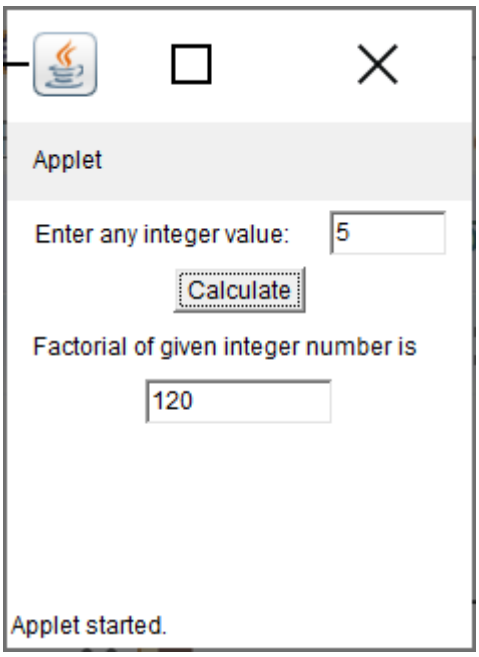
Output:



Source code for question b:

```
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)	{
		int s=0; if(f==0)
			return 1;
		else
			return f*fact(f-1);
	}
}
```

Output:



Week 4:

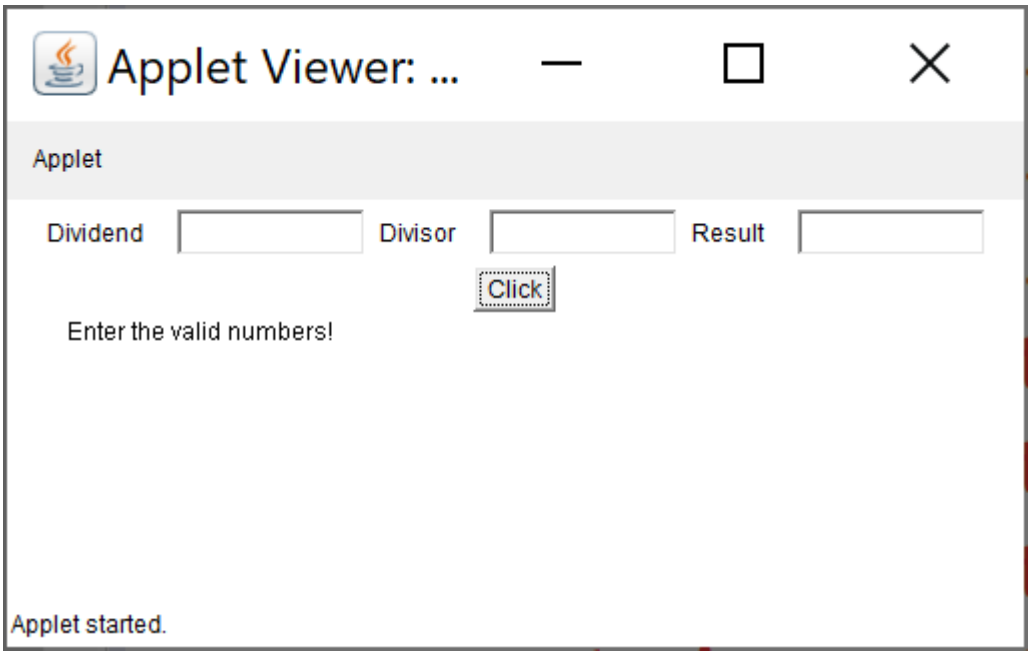
Write a Java program that creates a user interface to perform integer divisions. The user enters two numbers in the text fields, Num1 and Num2. The division of Num1 and Num 2 is displayed in the Result field when the Divide button is clicked. If Num1 or Num2 were not an integer, the program would throw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

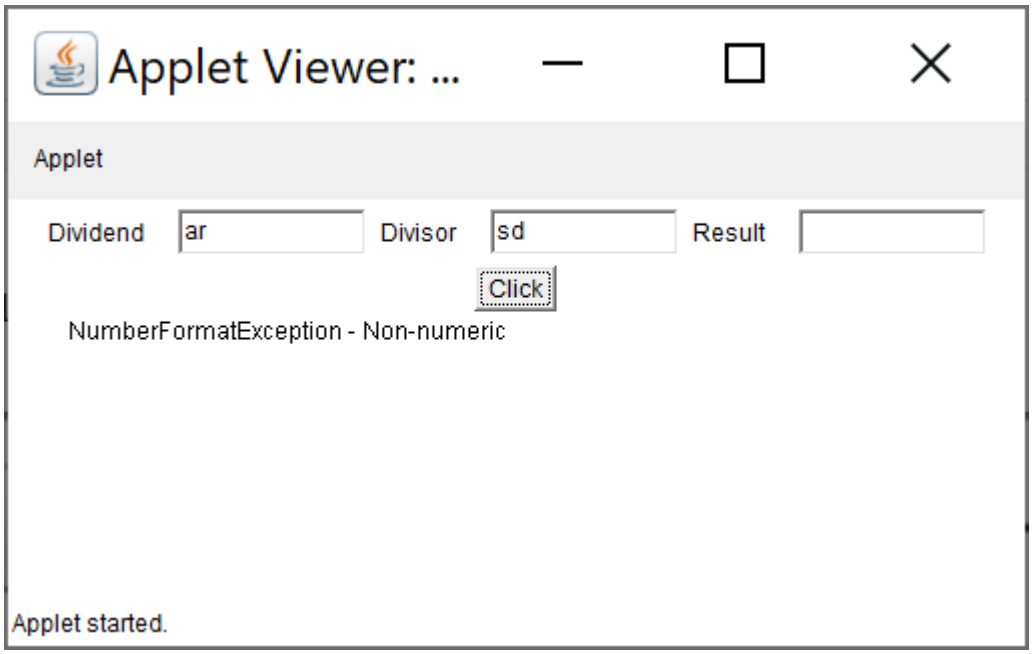
Source code:

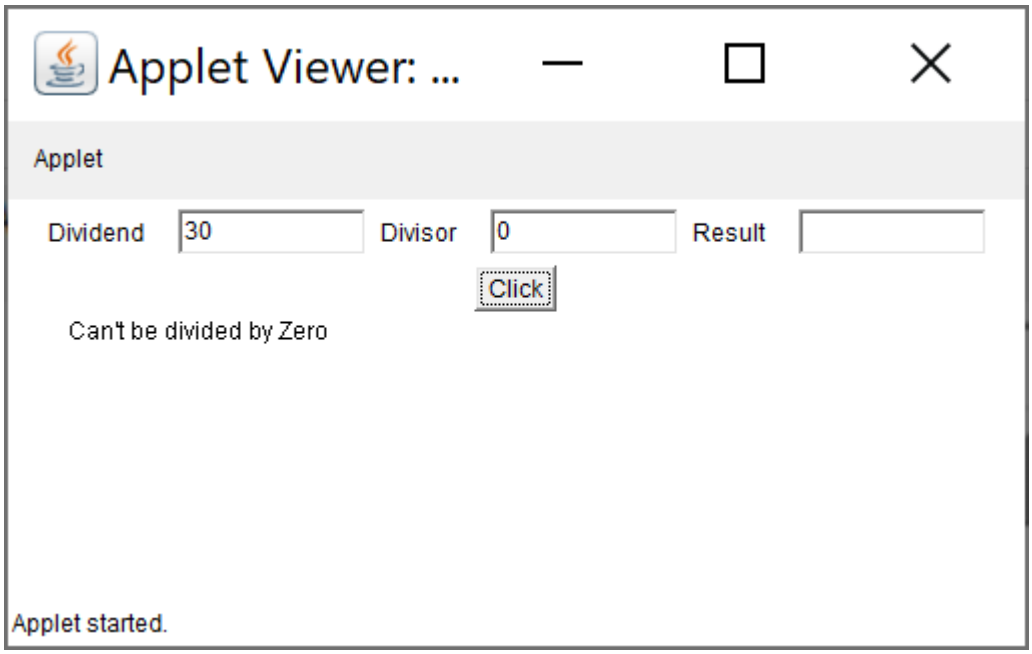
```
import java.awt.*;
import java.awt.event.*;
import java.applet.*;

/*<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";
```

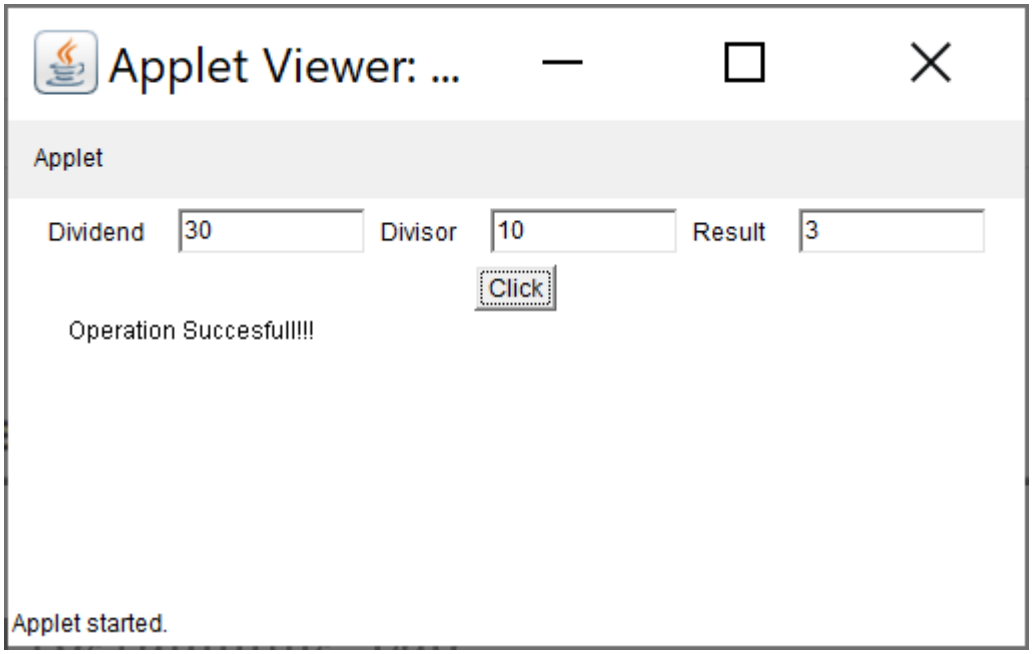
```
                        repaint();
                    } catch (ArithmeticException e) {
                        System.out.println("Can't be divided by Zero" + e);
                        res.setText("");
                        msg = "Can't be divided by Zero";
                        repaint();
                    }
                }
            }
        }

        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.

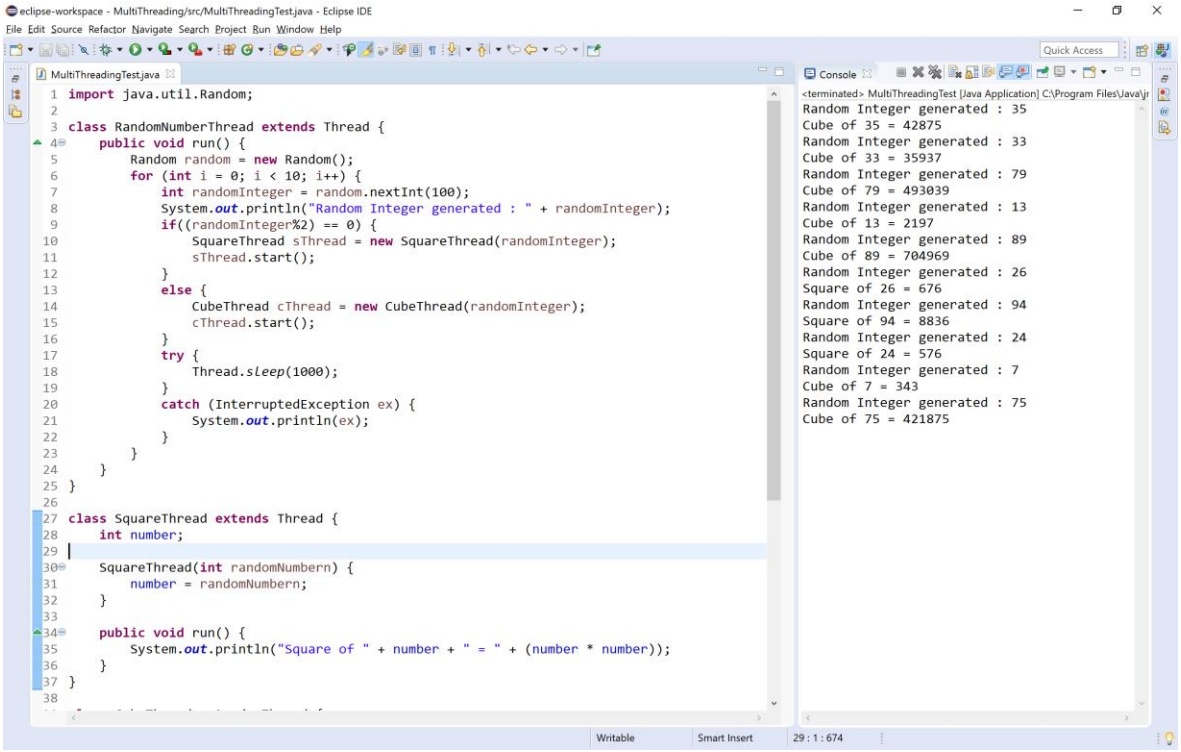
Week 5:

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.

Source code:

```
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
MultiThreadingTest.java
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));
}
}
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

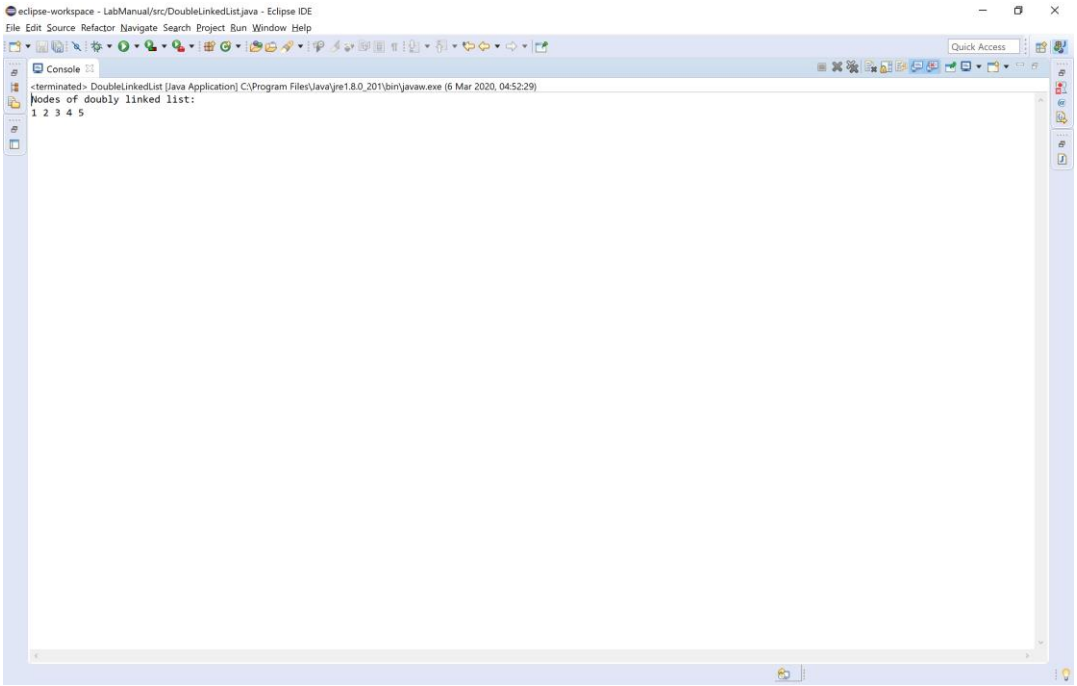
Week6:

Write a C++ to illustrate the concepts of console I/O operations.

Source code:

```
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() {
		Node current = head;
		if (head == null) {
			System.out.println("List is empty");
			return;
		}
		System.out.println("Nodes of doubly linked list: ");
		while (current != null) {

			System.out.print(current.data + " ");
			current = current.next;
		}
	}
	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:


eclipse-workspace - LabManual/src/DoubleLinkedList.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
Console
<terminated> DoubleLinkedList [Java Application] C:\Program Files\Java\jre1.8.0_201\bin\javaw.exe (6 Mar 2020, 04:52:29)
Nodes of doubly linked list:
1 2 3 4 5

Week 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 selected color. Initially, there is no message shown.

Source code:

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

/*
 * <applet code = "TrafficLightsExample" width = 1000 height = 500>
 * </applet>
 * */

public class TrafficLightsExample 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:


Applet Viewer: Traffi...
Applet
Red
Yellow
Green
STOP
Applet started.

Week 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.

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 AbstractClassExample {
	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:

```
eclipse-workspace - AbstractClass/src/AbstractClassExample.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
AbstractClassExample.java
    }
}

class Triangle extends Shape {
    void printArea() {
        System.out.println("\n*** Finding the Area
        System.out.print("Enter Base And Height: "
        length = input.nextInt();
        breadth = input.nextInt();
        System.out.println("The area of Triangle i
    }
}

class Cricle extends Shape {
    void printArea() {
        System.out.println("\n*** Finding the Area
        System.out.print("Enter Radius: ");
        radius = input.nextInt();
        System.out.println("The area of Cricle is
    }
}

public class AbstractClassExample {
    public static void main(String[] args) {
        Rectangle rec = new Rectangle();
        rec.printArea();
```

```
Console
<terminated> AbstractClassExample [Java Application] C:\Program Files\Java\jre1.8.0_201\bin\javaw.exe (2 F
*** 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: 4 5
The area of Triangle is: 10

*** Finding the Area of Cricle ***
Enter Radius: 5
The area of Cricle is: 78.5
```

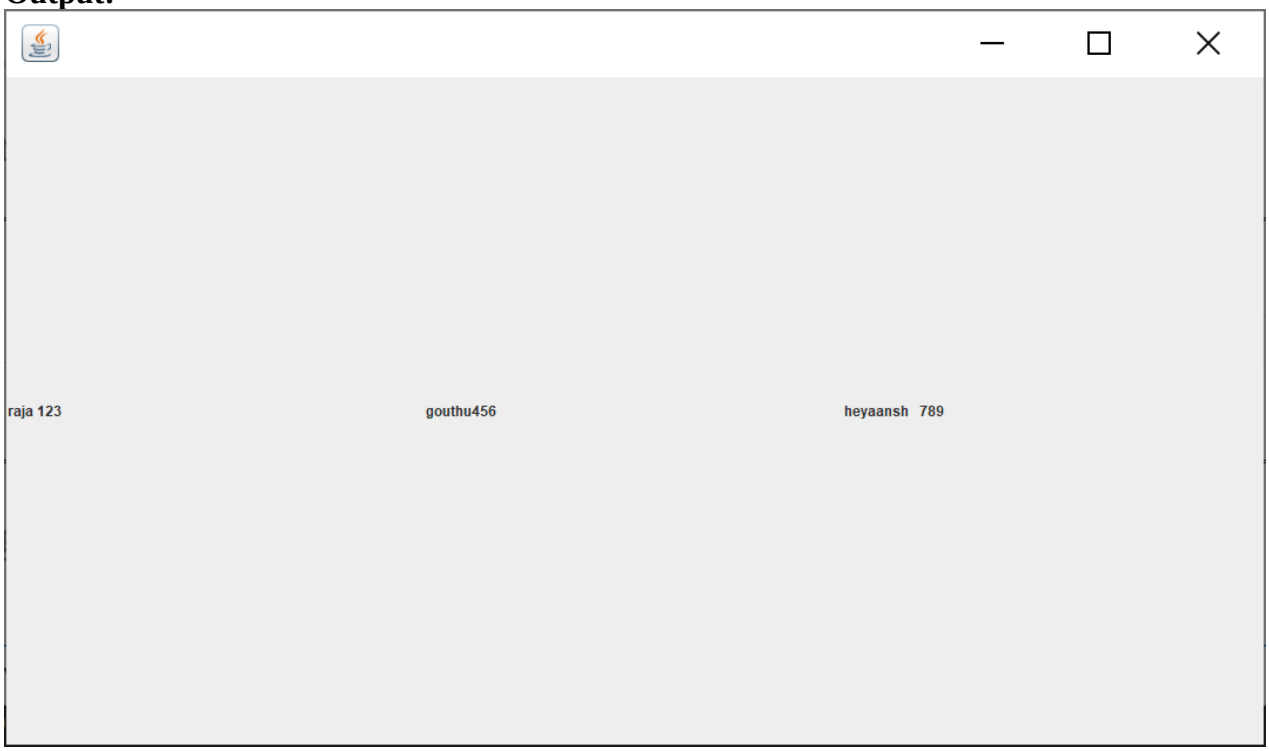
Week 9:

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.

Source code:

```
import java.io.*;
import java.util.*;
import java.awt.*;
import javax.swing.*;

class A extends JFrame {
	public A() {
		setSize(400, 400);
		setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
		GridLayout g = new GridLayout(0, 3);
		setLayout(g);
		try {
			FileInputStream fin = new
FileInputStream("C:\\Users\\User\\eclipse-workspace\\LabManual\\src\\HashTab.txt");
			Scanner sc = new Scanner(fin).useDelimiter(",");
			String[] arrayList;
			String a;
			while (sc.hasNextLine()) {
				a = sc.nextLine();
				arrayList = a.split(",");
				for (String i : arrayList) {
					add(new JLabel(i));
				}
			}
		} catch (Exception ex) {
		}
		setDefaultLookAndFeelDecorated(true);
		pack();
		setVisible(true);
	}
}

public class TableTest {

	public static void main(String[] args) {
		A a = new A();
	}
}
```

Output:



Week 10:

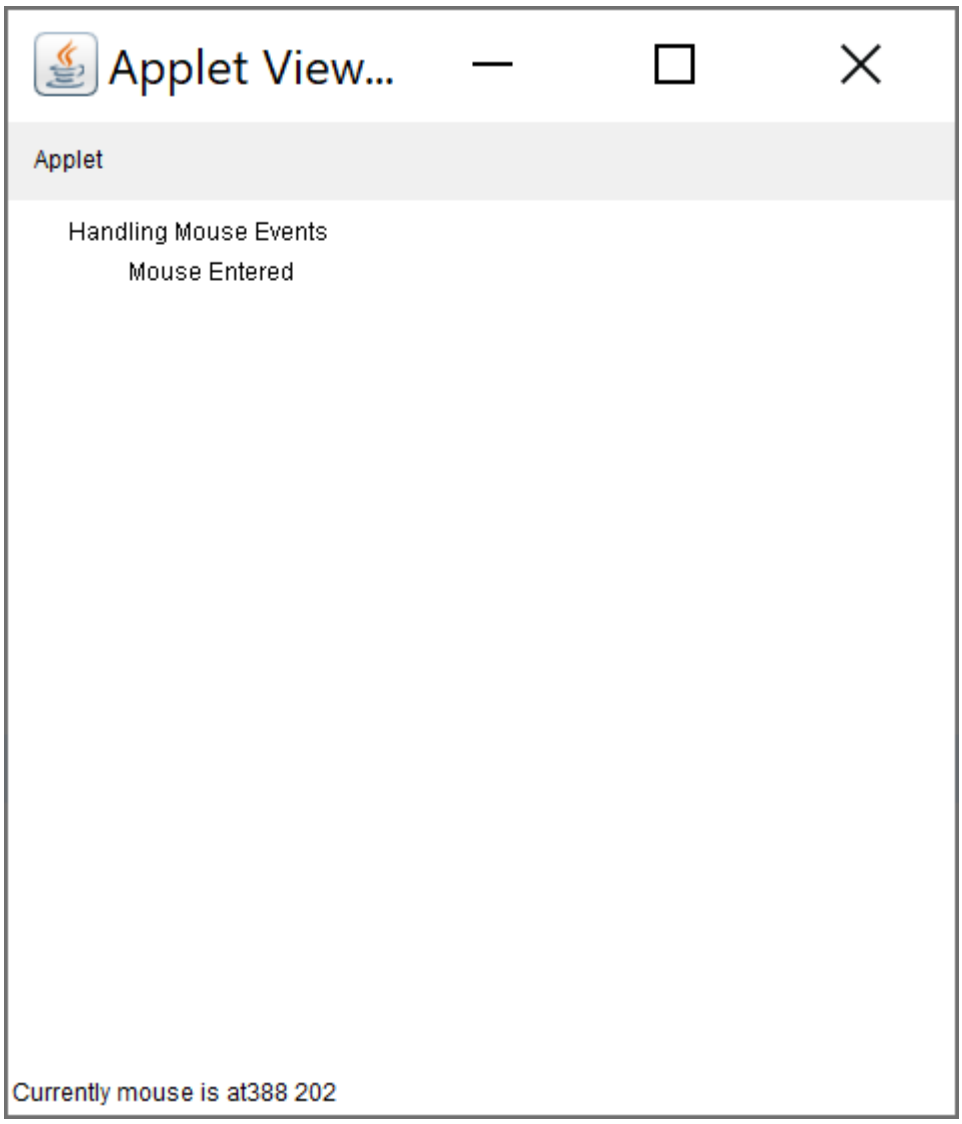
Write a Java program that handles all mouse events and shows the event name at the center of the window when a mouse event is fired (Use Adapter classes).

Source code:

```
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;
	int 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("Handling Mouse Events", 30, 20);
            g.drawString(msg, 60, 40);
        }
}
```

Output:



Week 11:

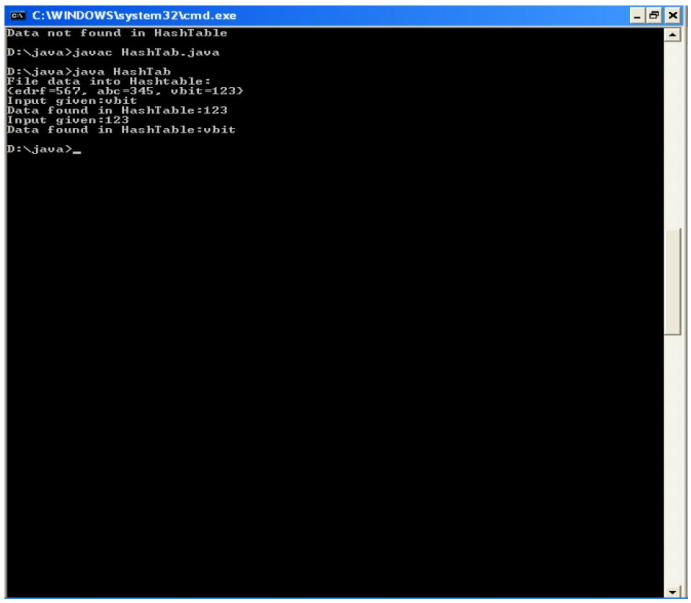
Write a java program that loads names and phone numbers from a text file where the data is organized as one line per record and each field in a record are separated by a tab (\t).it takes a name or phone number as input and prints the corresponding other value from the hash table(hint: use hash tables)

Source code:

```
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("D:\\java\\" + 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;
        }
}
```

Output:



Week – 12

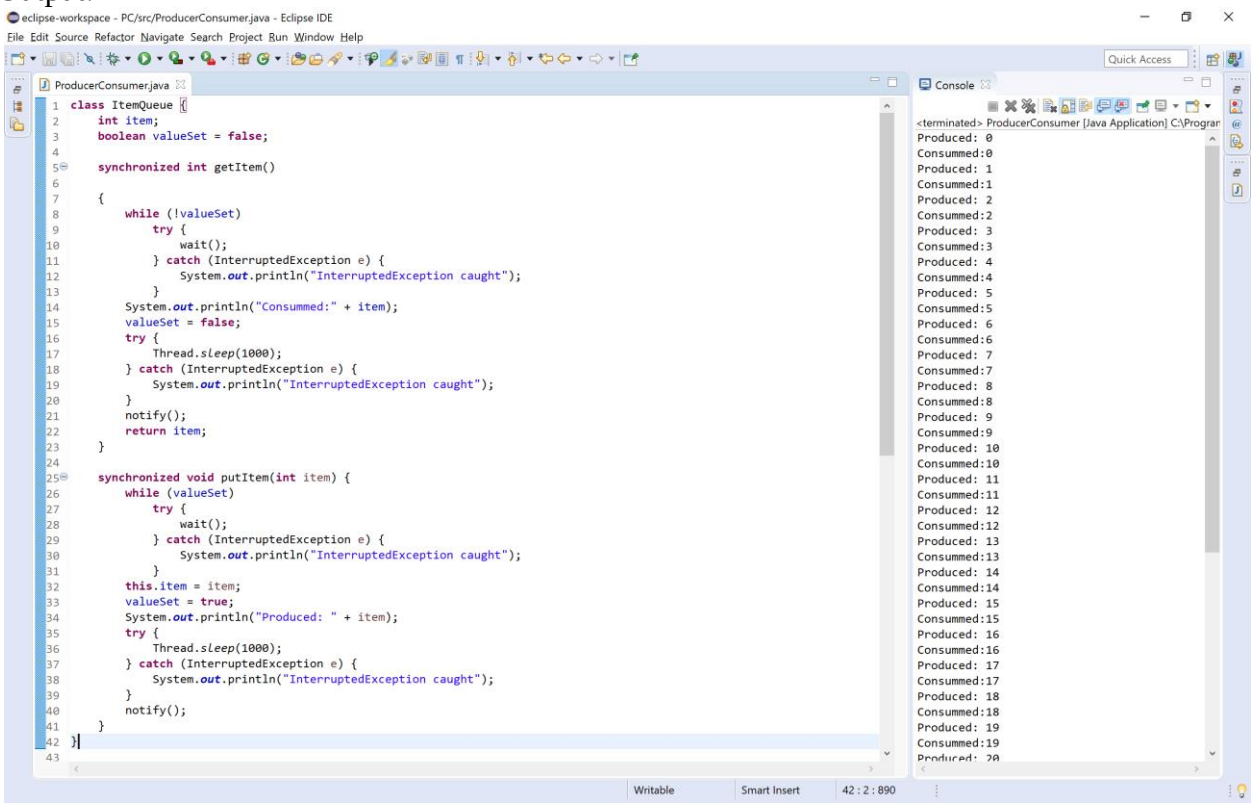
Write a Java program that correctly implements the producer – consumer problem using the concept of interthread communication.

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++);
		}
	}
}
```

```java
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);

	}
}
```

Output:



Week – 13

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

Source Code:

```
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()) {
			printLine();
			System.out.println("Root directory does not exists!");
			printLine();
		} 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()) {
						printLine();
						System.out.println("The sub-directory does not
exists!");
						printLine();
						int lastIndex = path.lastIndexOf("\\");
						path = path.substring(0, lastIndex);
					}
				}
			}
		}
		System.out.println("***** Program Closed *****");
	}
	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());
		}
	}
	public static void printLine() {
		System.out.println("----------------------------------------");
	}
}
```

Output:

```
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() + "\\";
```

```
ListingFiles [Java Application] C:\Program Files\Java\jre1.8.0_201\bin\javaw.exe (3 Feb 2020, 12:18:07)
> Windows
> XML
Do you want to open any sub-directory (Y/N):  y
Enter the sub-directory name: Raja
Current Location: C:\\\Raja
> Balu
> BSC 2018
> BSC Hosted
> CMREC
> Education
```

Week 14

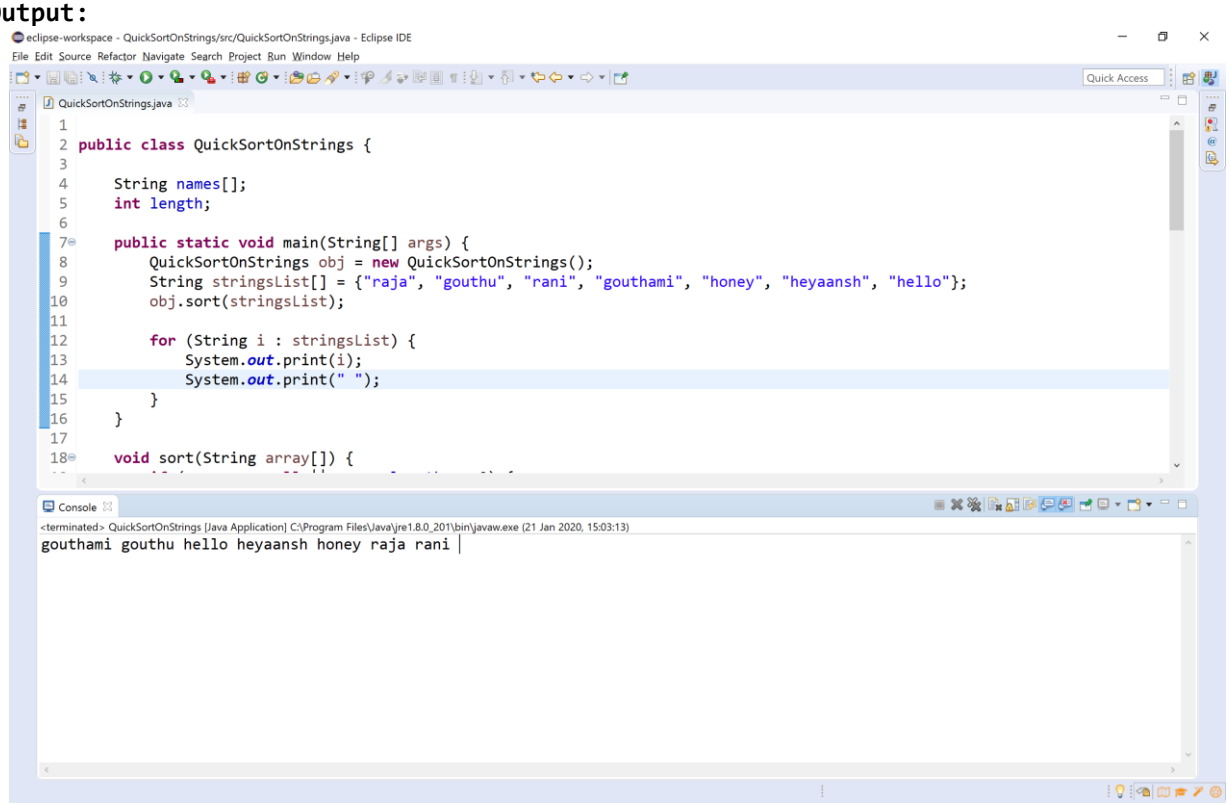
Write a Java program that implements Quick sort algorithm for sorting a list of names in ascending Order.

Source Code:

```
public class QuickSortOnStrings {

    String names[];
    int length;

    public static void main(String[] args) {
        QuickSortOnStrings obj = new QuickSortOnStrings();
        String stringsList[] = {"raja", "gouthu", "rani", "gouthami", "honey",
"heyaansh", "hello"};
        obj.sort(stringsList);

        for (String i : stringsList) {
            System.out.print(i);
            System.out.print(" ");
        }
    }

    void sort(String array[]) {
        if (array == null || array.length == 0) {
            return;
        }
        this.names = array;
        this.length = array.length;
        quickSort(0, length - 1);
    }

    void quickSort(int lowerIndex, int higherIndex) {
        int i = lowerIndex;
        int j = higherIndex;
        String pivot = this.names[lowerIndex + (higherIndex - lowerIndex) / 2];

        while (i <= j) {
            while (this.names[i].compareToIgnoreCase(pivot) < 0) {
                i++;
            }

            while (this.names[j].compareToIgnoreCase(pivot) > 0) {
                j--;
            }

            if (i <= j) {
                exchangeNames(i, j);
                i++;
                j--;
            }
        }
        if (lowerIndex < j) {
            quickSort(lowerIndex, j);
        }
        if (i < higherIndex) {
            quickSort(i, higherIndex);
        }
    }

    void exchangeNames(int i, int j) {
```

```
        String temp = this.names[i];
        this.names[i] = this.names[j];
        this.names[j] = temp;
    }
}
```

Output:



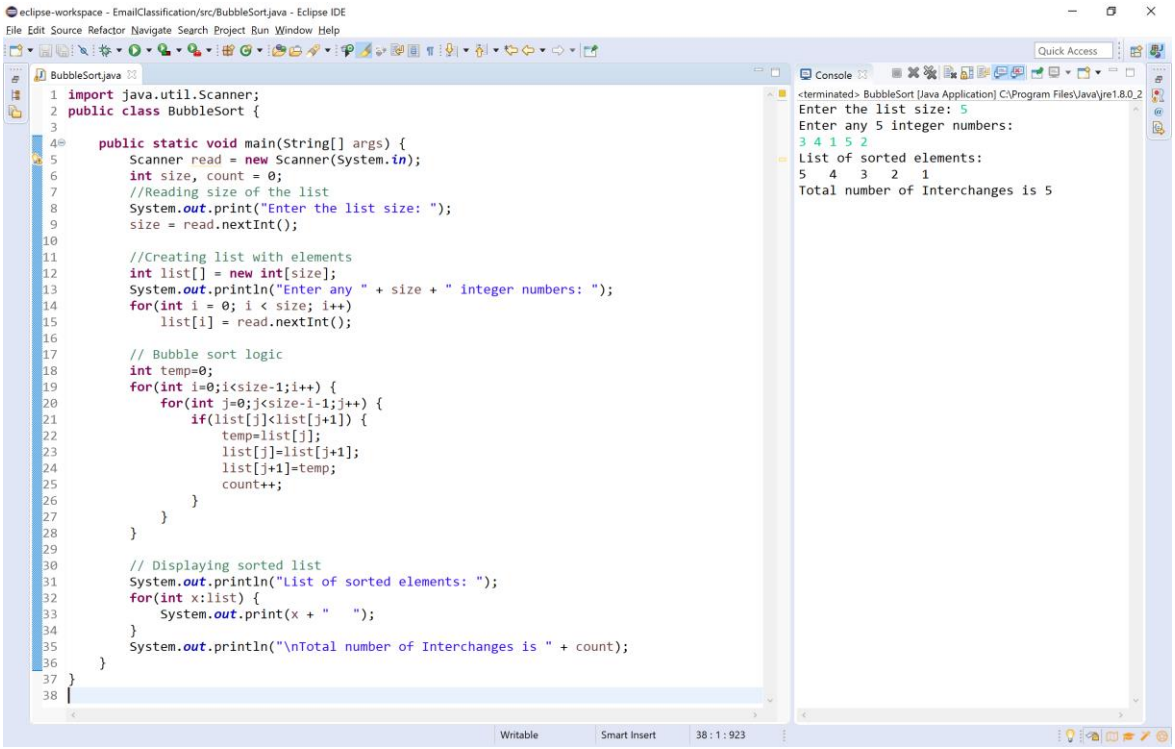
Week 15:

Write a Java program that implements Bubble sort algorithm for sorting in descending order and also shows the number of interchanges occurred for the given set of integers.

Source Code:

```
import java.util.Scanner;
public class BubbleSort {

	public static void main(String[] args) {
		Scanner read = new Scanner(System.in);
		int size, count = 0;
		//Reading size of the list
		System.out.print("Enter the list size: ");
		size = read.nextInt();

		//Creating list with elements
		int list[] = new int[size];
		System.out.println("Enter any " + size + " integer numbers: ");
		for(int i = 0; i < size; i++)
			list[i] = read.nextInt();

		// Bubble sort logic
		int temp=0;
		for(int i=0;i<size-1;i++) {
			for(int j=0;j<size-i-1;j++) {
				if(list[j]<list[j+1]) {
					temp=list[j];
					list[j]=list[j+1];
					list[j+1]=temp;
					count++;
				}
			}
		}

		// Displaying sorted list
		System.out.println("List of sorted elements: ");
		for(int x:list) {
			System.out.print(x + "   ");
		}
		System.out.println("\nTotal number of Interchanges is " + count);
	}
}
```

Output:


eclipse-workspace - EmailClassification/src/BubbleSort.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
BubbleSort.java
import java.util.Scanner;
public class BubbleSort {
public static void main(String[] args) {
Scanner read = new Scanner(System.in);
int size, count = 0;
//Reading size of the list
System.out.print("Enter the list size: ");
size = read.nextInt();
//Creating list with elements
int list[] = new int[size];
System.out.println("Enter any " + size + " integer numbers: ");
for(int i = 0; i < size; i++)
list[i] = read.nextInt();
// Bubble sort logic
int temp=0;
for(int i=0;i<size-1;i++) {
for(int j=0;j<size-i-1;j++) {
if(list[j]<list[j+1]) {
temp=list[j];
list[j]=list[j+1];
list[j+1]=temp;
count++;
}
}
}
// Displaying sorted list
System.out.println("List of sorted elements: ");
for(int x:list) {
System.out.print(x + " ");
}
System.out.println("\nTotal number of Interchanges is " + count);
}
}
Console
<terminated> BubbleSort [Java Application] C:\Program Files\Java\jre1.8.0_2
Enter the list size: 5
Enter any 5 integer numbers:
3 4 1 5 2
List of sorted elements:
5 4 3 2 1
Total number of Interchanges is 5
Writable
Smart Insert
38 : 1 : 923