JAVA PROGRAMMING

LABORATORY MANUAL

B.TECH (R-17 Regulation)
(II YEAR – II SEM)
(2018-19)

DEPARTMENT OF
COMPUTER SCIENCE AND ENGINEERING

MALLA REDDY COLLEGE OF ENGINEERING & TECHNOLOGY

(Autonomous Institution – UGC, Govt. of India)

Recognized under 2(f) and 12 (B) of UGC ACT 1956

(Affiliated to JNTUH, Hyderabad, Approved by AICTE - Accredited by NBA & NAAC – 'A' Grade - ISO 9001:2015 Certified)

Maisammaguda, Dhulapally (Post Via. Hakimpet), Secunderabad – 500100, Telangana State, India

Department of Computer Science and Engineering

Vision

- To acknowledge quality education and instill high patterns of discipline making the students technologically superior and ethically strong which involves the improvement in the quality of life in human race.

Mission

- To achieve and impart holistic technical education using the best of infrastructure, outstanding technical and teaching expertise to establish the students into competent and confident engineers.
- Evolving the center of excellence through creative and innovative teaching learning practices for promoting academic achievement to produce internationally accepted competitive and world class professionals.

PROGRAMME EDUCATIONAL OBJECTIVES (PEOs)

PEO1 – ANALYTICAL SKILLS

1. To facilitate the graduates with the ability to visualize, gather information, articulate, analyze, solve complex problems, and make decisions. These are essential to address the challenges of complex and computation intensive problems increasing their productivity.

PEO2 – TECHNICAL SKILLS

2. To facilitate the graduates with the technical skills that prepare them for immediate employment and pursue certification providing a deeper understanding of the technology in advanced areas of computer science and related fields, thus encouraging to pursue higher education and research based on their interest.

PEO3 – SOFT SKILLS

3. To facilitate the graduates with the soft skills that include fulfilling the mission, setting goals, showing self-confidence by communicating effectively, having a positive attitude, get involved in team-work, being a leader, managing their career and their life.

PEO4 – PROFESSIONAL ETHICS

To facilitate the graduates with the knowledge of professional and ethical responsibilities by paying attention to grooming, being conservative with style, following dress codes, safety codes,and adapting themselves to technological advancements.

PROGRAM SPECIFIC OUTCOMES (PSOs)

After the completion of the course, B. Tech Computer Science and Engineering, the graduates will have the following Program Specific Outcomes:

1. **Fundamentals and critical knowledge of the Computer System:-** Able to Understand the working principles of the computer System and its components , Apply the knowledge to build, asses, and analyze the software and hardware aspects of it .

2. **The comprehensive and Applicative knowledge of Software Development:** Comprehensive skills of Programming Languages, Software process models, methodologies, and able to plan, develop, test, analyze, and manage the software and hardware intensive systems in heterogeneous platforms individually or working in teams.

3. **Applications of Computing Domain & Research:** Able to use the professional, managerial, interdisciplinary skill set, and domain specific tools in development processes, identify the research gaps, and provide innovative solutions to them.

PROGRAM OUTCOMES (POs)

Engineering Graduates will be able to:

1. **Engineering knowledge**: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis**: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. **Design / development of solutions**: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. **Conduct investigations of complex problems**: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5. **Modern tool usage**: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
6. **The engineer and society**: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and sustainability**: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. **Ethics**: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work**: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication**: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. **Project management and finance**: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multi disciplinary environments.
12. **Life- long learning**: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

1.Lab Objectives:

- To prepare students to become familiar with the Standard Java technologies of J2SE
- To prepare students to excel in Object Oriented programming and to succeed as a Java Developer through global rigorous education.
- To provide Students with a solid foundation in OOP fundamentals required to solve programming problems and also to learn Advanced Java topics like J2ME, J2EE, JSP, JavaScript
- To train Students with good OOP programming breadth so as to comprehend, analyze, design and create novel products and solutions for the real life problems.
- To inculcate in students professional and ethical attitude, multidisciplinary approach and an ability to relate java programming issues to broader application context.
- To provide student with an academic environment aware of excellence, written ethical codes and guidelines and lifelong learning needed for a successful professional career.

2.Lab Outcomes:

Upon successful completion of this course, the students will be able to:

- Able to analyze the necessity for Object Oriented Programming paradigm and over structured programming and become familiar with the fundamental concepts in OOP.
- Demonstrate an ability to design and develop java programs, analyze, and interpret object oriented data and report results.
- Demonstrate an ability to design an object oriented system, AWT components or multithreaded process as per needs and specifications.
- Demonstrate an ability to visualize and work on laboratory and multidisciplinary tasks like console and windows applications both for standalone and Applets programs

3. Introduction about lab

There are 65 systems installed in this Lab. Their configurations are as follows:

- **Hardware / Software's installed: I**ntel® CORE™ i3-3240 CPU@3.40GHZ RAM:4GB / C, C++ Compiler ,EditPlus.
- Systems are provided for students in the **1:1 ratio.**
- Systems are assigned numbers and same system is allotted for students when they do the lab.
- All systems are configured in **DUAL BOOT** mode i.e., Students can boot from Windows XP or Linux as per their lab requirement. This is very useful for students because they are familiar with different Operating Systems so that they can execute their programs in different programming environments.

4. Guidelines to students

A. Standard operating procedure

a) Explanation on today's experiment by the concerned faculty using PPT covering the following aspects:

1) Name of the experiment
2) Aim
3) Software/Hardware requirements

b) Writing the java programs by the students

c) Commands for executing programs 100 mins.

Writing of the experiment in the Observation Book

The students will write the today's experiment in the Observation book as per the following format:

a) Name of the experiment
b) Aim
c) Software/Hardware required
d) Writing the program
e) Viva-Voce Questions and Answers
f) Errors observed (if any) during compilation/execution
g) Signature of the Faculty

h) Viva-Voce Questions and Answers

i) Errors observed (if any) during compilation/execution

j) Signature of the Faculty

B. Guide Lines to Students in Lab

Disciplinary to be maintained by the students in the Lab

- Students are required to carry their lab observation book and record book with completed experiments while entering the lab.
- Students must use the equipment with care. Any damage is caused student is punishable
- Students are not allowed to use their cell phones/pen drives/ CDs in labs.
- Students need to be maintain proper dress code along with ID Card
- Students are supposed to occupy the computers allotted to them and are not supposed to talk or make noise in the lab.
- Students, after completion of each experiment they need to be updated in observation notes and same to be updated in the record.
- Lab records need to be submitted after completion of experiment and get it corrected with the concerned lab faculty.
- If a student is absent for any lab, they need to be completed the same experiment in the free time before attending next lab.

Steps to perform experiments in the lab by the student

Step1: Students have to write the date, aim, Software & Hardware requirements for that experiment in the observation book.

Step2: Students have to listen and understand the experiment explained by the faculty and note down the important points in the observation book.

Step3: Students need to write procedure/algorithm in the observation book.

Step4: Analyze and Develop/implement the logic of the program by the student in respective platform

Step5: after approval of logic of the experiment by the faculty then the

experiment has to be executed on the system.

Step6: After successful execution the results are to be shown to the faculty and noted the same in the observation book.

Step7: Students need to attend the Viva-Voce on that experiment and write the same in the observation book.

Step8: Update the completed experiment in the record and submit to the concerned faculty in-charge.

Instructions to maintain the record

- Before start of the first lab they have to buy the record and bring the record to the lab.
- Regularly (Weekly) update the record after completion of the experiment and get it corrected with concerned lab in-charge for continuous evaluation.
- In case the record is lost inform the same day to the faculty in charge and get the new record within 2 days the record has to be submitted and get it corrected by the faculty.
- If record is not submitted in time or record is not written properly, the evaluation marks (5M) will be deducted.

Awarding the marks for day to day evaluation

Total marks for day to day evaluation is 15 Marks as per Autonomous (JNTUH).

These 15 Marks are distributed as:

Record	5 Marks
Exp setup/program written	5 Marks
Result and Viva-Voce	5 Marks

Allocation of Marks for Lab Internal

Total marks for lab internal are 25 Marks as per Autonomous (JNTUH.)

These 25 Marks are distributed as:

Average of day to day evaluation marks: 15 Marks

Lab Mid exam: 10 Marks

Allocation of Marks for Lab External

Total marks for lab Internal and External are 75 Marks as per Autonomous / (JNTUH.)

These 50 External Lab Marks are distributed as:

Program Written	20 Marks
Program Execution and Result	15 Marks
Viva-Voce	10 Marks
Record	5 Marks

C. General laboratory instructions

1. Students are advised to come to the laboratory at least 5 minutes before (to the starting time), those who come after 5 minutes will not be allowed into the lab.
2. Plan your task properly much before to the commencement, come prepared to the lab with the synopsis / program / experiment details.
3. Student should enter into the laboratory with:
    a. Laboratory observation notes with all the details (Problem statement, Aim, Algorithm, Procedure, Program, Expected Output, etc.,) filled in for the lab session.
    b. Laboratory Record updated up to the last session experiments and other utensils (if any) needed in the lab.
    c. Proper Dress code and Identity card.

4. Sign in the laboratory login register, write the TIME-IN, and occupy the computer system allotted to you by the faculty.
5. Execute your task in the laboratory, and record the results / output in the lab observation note book, and get certified by the concerned faculty.
6. All the students should be polite and cooperative with the laboratory staff, must maintain the discipline and decency in the laboratory.
7. Computer labs are established with sophisticated and high end branded systems, which should be utilized properly.
8. Students / Faculty must keep their mobile phones in SWITCHED OFF mode during the lab sessions. Misuse of the equipment, misbehaviors with the staff and systems etc., will attract severe punishment.
9. Students must take the permission of the faculty in case of any urgency to go out ; if anybody found loitering outside the lab / class without permission during working hours will be treated seriously and punished appropriately.
10. Students should LOG OFF/ SHUT DOWN the computer system before he/she leaves the lab after completing the task (experiment) in all aspects. He/she must ensure the system / seat is kept properly.

Head of the Department **Principal**

INDEX

PROGRAM -1 Date:

Aim: Write a java program to find the Fibonacci series using recursive and non recursive functions

Program:

```
//Class to write the recursive and non recursive functions.
class fib
{
int a,b,c;
//        Non recursive function to find the Fibonacci series.
void nonrecursive(int n)
{
a=0;
b=1;
c=a+b;
System.out.print(b);
while(c<=n)
{
System.out.print(c);
a=b;
b=c;
c=a+b;
}
}
// Recursive function to find the Fibonacci series.
int recursive(int n)
{
if(n==0)
        return (0);
         if(n==1)
                return (1);
        else
                return(recursive(n-1)+recursive(n-2));
}
}
// Class that calls recursive and non recursive functions .
 class fib1
{
public static void main(String args[])
{
int n;
// Accepting the value of n at run time.
 n=Integer.parseInt(args[0]);
System.out.println("the recursion using non recursive is"); // Creating object for the fib
class.fib f=new fib();
// Calling non recursive function of fib
 class. f.nonrecursive(n);
System.out.println("the recursion using recursive is"); ffor(int i=0;i<=n;i++)
{
// Calling recursive function of fib class. int F1=f.recursive(i);
System.out.print(F1);
}
}
}
```

Three Test Outputs:

Signature of the faculty

EXERCISE:

1. Write a java program to print the multiplication table .
2. Write a java program to find the Factorial of a given integer using recursive and non recursive functions

PROGRAM -2 Date:

Aim: Write a java program to multiply two given matrices.

```
// Class to find multiplication of matrices.
class matri
{
public static void main(String args[])
{
// Accept the number of rows and columns at run time.
int m=Integer.parseInt(args[0]);
int n=Integer.parseInt(args[1]);
// Initialize the arrays.
int a[][]=new int[m][n]; int b[][]=new int[m][n]; int c[][]=new int[m][n]; int i=2;
// Loop to accept the values into a matrix.
for(int j=0;j<m;j++)
{for(int k=0;k<n;k++)
{
a[j][k]=Integer.parseInt(args[i]);
i++;
}
}
// Loop to accept the values into b matrix.
for(int j=0;j<m;j++)
{
for(int k=0;k<n;k++)
{
        b[j][k]=Integer.parseInt(args[i]);
        i++;
}
}
// Loop to multiply two matrices .
for(int j=0;j<m;j++)
{
for(int k=0;k<n;k++)
  {
c[j][k]=0;
for(int l=0;l<m;l++)
{
        c[j][k]=c[j][k]+(a[j][l]*b[l][k]);
}
}
}
// Loop to display the result .
for(int j=0;j<m;j++)
{
for(int k=0;k<n;k++)
{
System.out.print(c[j][k]);
}
System.out.println();
}
}
}
```

Three test outputs:

Signature of the faculty

PROGRAM -3 Date:

Aim: Write a java program that prompts the user for an integer and then printouts all prime numbers up to that integer

Program:

```
import java.lang.*;
class Prime
{
public static void main(String arg[])
{
int n,c,i,j;
n=Integer.parseInt(arg[0]);
System.out.println("prime numbers are");
for(i=1;i<=n;i++)
{
c=0;
for(j=1;j<=i;j++)
{
if(i%j==0)
c++;
}
if(c==2)
System.out.println(" "+i);
}
}
}
```

Three test outputs:

Signature of the faculty

EXERCISE:

1. Write a java program to find all even and odd integers up to a given integer.
2. Write a java program to add and subtract two given matrices.
3. Write a java program that reads a line of integers and displays each integers and the product of all integers use String Tokenizer.

PROGRAM -4 Date:

Aim: Write a java program that checks whether a given string is palindrome or not

Program:

```
// Class to find whether string is palindrome or not.
class palindrome
{
public static void main(String args[])
{
// Accepting the string at run time.
 String s=args[0];
String s1=""; int l,j;
// Finding the length of the string.
l=s.length();
// Loop to find the reverse of the string.
for(j=l-1;j>=0;j--)
{
s1=s1+s.charAt(j);
}
// Condition to find whether two strings are equal // and display the message.
if(s.equals(s1))
System.out.println("String "+s+" is palindrome");
else
System.out.println("String "+s+" is not palindrome");
}

}
```

Three test outputs:

Signature of the faculty

EXERCISE:

1. Write a java program to sort the given integers in ascending/descending order.
2. Write a java program to display characters in a string in sorted order.
3. write a program that uses a sequence inputstream to output the contents of two files.
4. Write a java program that reads a file and displays the file on the screen, with an asterisk mark before each line.
5. Write a java program that displays the number of characters, lines, words, white spaces in a text file.

PROGRAM -5 A) Date:

Aim: Write an applet program that displays a simple message
Program:
Applet1.java:

```
// Import the packages to access the classes and methods in awt and applet classes.
import java.awt.*; import java.applet.*;
public class Applet1 extends Applet
{
// Paint method to display the message.
public void paint(Graphics g)
{
g.drawString("HELLO           WORLD",20,20);
}
}
```

Applet1.html:

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

Three test Outputs:

Signature of the faculty

EXERCISE:1.Write an applet program that accepts an integer and display the factorial of a given integer. 2Write an applet program that accepts an integer and display the prime numbers up to that given integer.

PROGRAM -5 B Date:

Aim: Write a Java program compute factorial value using Applet

```
import java.awt.*;
import java.awt.event.*;
import java.applet.Applet;
public class FactorialApplet extends Applet implements ActionListener
{
	/*<applet code="FactorialApplet" height=300 width=300>
	</applet>*/
	Label l1,l2;
	TextField t1,t2;
	Button b1;
	public void  init()
	{
		setLayout(new FlowLayout(FlowLayout.LEFT));
		l1=new Label("Enter the value:");
		add(l1);
		t1=new TextField(10);
		add(t1);
		l2=new Label("Factorial value is:");
		add(l2);
		t2=new TextField(10);
		add(t2);
		b1=new Button("Compute");
		add(b1);
		b1.addActionListener(this);
	}
	public void actionPerformed(ActionEvent e)
	{
		if((e.getSource())==b1)
		{
			int value=Integer.parseInt(t1.getText());
			int fact=factorial(value);
			t2.setText(String.valueOf(fact));
		}
	}
	int factorial(int n)
	{
		if(n==0)
			return 1;
		else
			return n*factorial(n-1);
	}
}
```

Three Test Outputs:

Signature of the faculty

Exercise: write an applet program for displaying the circle in green color.

PROGRAM -6 Date:

Aim: Write a java program that works as a simple calculator. Use a Grid Layout to arrange Buttons for digits and for the + - * % operations. Add a text field to display the result.

Program:

```
import javax.swing.*;
import javax.swing.JOptionPane; import java.awt.*;
import java.awt.event.*;

// Class that initialize the applet and create calculator.
public class Calculator extends JApplet
{
public void init()
{
CalculatorPanel calc=new CalculatorPanel(); getContentPane().add(calc);
}
}
// Class that creates the calculator panel .
class CalculatorPanel extends JPanel implements ActionListener
{
// Creation of JButton.
JButton n1,n2,n3,n4,n5,n6,n7,n8,n9,n0,plus,minus,mul,div,dot,equal;
static JTextField result=new JTextField("0",45); static String lastCommand=null;
// Create the JObjectPane.
JOptionPane p=new JOptionPane(); double preRes=0,secVal=0,res;
private static void assign(String no)
{
if((result.getText()).equals("0")) result.setText(no);
else      if(lastCommand=="=")
{
result.setText(no); lastCommand=null; }
else
result.setText(result.getText()+no);
}

 // Creation of control panel of calculator and adding buttons using GridLayout.
 public CalculatorPanel()
{
setLayout(new GridLayout());
 result.setEditable(false);
 result.setSize(300,200);
 add(result);
JPanel panel=new JPanel();
panel.setLayout(new GridLayout(5,5));
n7=new JButton("7");
panel.add(n7);
n7.addActionListener(this);
n8=new JButton("8");
panel.add(n8);
n8.addActionListener(this);
n9=new JButton("9");
```

```
panel.add(n9);
n9.addActionListener(this);
div=new JButton("/");
panel.add(div);
div.addActionListener(this);
n4=new JButton("4");
panel.add(n4);
n4.addActionListener(this);
n5=new JButton("5");
panel.add(n5);
n5.addActionListener(this);
n6=new JButton("6");
panel.add(n6);
n6.addActionListener(this);
mul=new JButton("*");
panel.add(mul);
mul.addActionListener(this);
n1=new JButton("1");
panel.add(n1);
n1.addActionListener(this);
n2=new JButton("2");
panel.add(n2);
n2.addActionListener(this);
n3=new JButton("3");
panel.add(n3);
n3.addActionListener(this);
minus=new JButton("-");
panel.add(minus);
minus.addActionListener(this);
dot=new JButton(".");
panel.add(dot);
dot.addActionListener(this);
n0=new JButton("0");
 panel.add(n0); n0.addActionListener(this);
 equal=new JButton("=");
panel.add(equal);
equal.addActionListener(this);
plus=new JButton("+");
panel.add(plus);
plus.addActionListener(this);
add(panel);

}
// Implementing method in ActionListener.
public void actionPerformed(ActionEvent ae)
{
if(ae.getSource()==n1)
            assign("1");
else if(ae.getSource()==n2)
            assign("2");
 else if(ae.getSource()==n3)
             assign("3");
else if(ae.getSource()==n4)
```

```
            assign("4");
 else if(ae.getSource()==n5)
            assign("5");
else if(ae.getSource()==n6)
            assign("6");
 else if(ae.getSource()==n7)
            assign("7");
else if(ae.getSource()==n8)
             assign("8");
 else if(ae.getSource()==n9)
            assign("9");
else if(ae.getSource()==n0)
            assign("0");
else      if(ae.getSource()==dot)
{
if(((result.getText()).indexOf("."))==-1) result.setText(result.getText()+"."); }
else      if(ae.getSource()==minus)
{
preRes=Double.parseDouble(result.getText()); lastCommand="-";
result.setText("0");
}
else      if(ae.getSource()==div)
{
preRes=Double.parseDouble(result.getText());
lastCommand="/";
result.setText("0");
 }
else if(ae.getSource()==equal)
{
 secVal=Double.parseDouble(result.getText());
 if(lastCommand.equals("/"))
            res=preRes/secVal;
else if(lastCommand.equals("*"))
            res=preRes*secVal;
else if(lastCommand.equals("-"))
            res=preRes-secVal;
else if(lastCommand.equals("+"))
            res=preRes+secVal;
result.setText(" "+res); lastCommand="=";
}
else      if(ae.getSource()==mul)
{
preRes=Double.parseDouble(result.getText());
lastCommand="*";
result.setText("0");
}
else      if(ae.getSource()==plus)
{
preRes=Double.parseDouble(result.getText());
lastCommand="+";
result.setText("0");
}
}
```

}

Calculator.html:

<applet code="Calculator" width=200 height=300> </applet>

Three Test Outputs:

Signature of the faculty

EXERCISE:

Write a java program that use a Grid Layout to arrange Buttons for alphabets. Add a text field to display the words.

PROGRAM -7 Date:

Aim: Write a Java program for display the exception in a message dialogbox

```
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
public class NumOperations extends JApplet implements ActionListener
{
	/*<applet code="NumOperations" width=300 height=300>
</applet>*/
	JLabel l1,l2,l3;
	JTextField t1,t2,t3;
	JButton b1;
	public void init()
	{
		Container contentPane=getContentPane();
		contentPane.setLayout(new FlowLayout());
		l1=new JLabel("Enter num1:");
		contentPane.add(l1);
		t1=new JTextField(15);
		contentPane.add(t1);
		l2=new JLabel("Enter num2:");
		contentPane.add(l2);
		t2=new JTextField(15);
		contentPane.add(t2);
		l3=new JLabel("The Result");
		contentPane.add(l3);
		t3=new JTextField(15);
		contentPane.add(t3);
		b1=new JButton("Divide");
		contentPane.add(b1);
		b1.addActionListener(this);
	}
	public void actionPerformed(ActionEvent e)
	{
		if(e.getSource()==b1)
		{
			try
			{
				int a=Integer.parseInt(t1.getText());
				int b=Integer.parseInt(t1.getText());
				Float c=Float.valueOf(a/b);
				t3.setText(String.valueOf(c));
			}
			catch(NumberFormatException e1)
			{
				JOptionPane.showMessageDialog(this,"Not a valid number");
			}
			catch(ArithmeticException e2)
			{
				JOptionPane.showMessageDialog(this,e2.getMessage());
			}
		}
```

```
        }
}
```

Three test outputs:

Signature of the faculty

Exercise:write a java program that illustrate the use of GridBaglayout.

PROGRAM -8 Date:

Aim: Write a Java program that implements a multi-thread application that has three threads

Program:

```
// Class that create the thread.
class NewThread implements Runnable
{ String name; Thread t;
// NewThread constructor that takes the thread name as parameter.
 NewThread(String threadname)
{
 name=threadname; t=new Thread(this,name);
System.out.println("new thread"+t); t.start();
}

//        Method to run the thread.
public void run()
{
//        The code that may generate the exception. try
{
//        Loop to display the thread name and the value.
for(int i=0;i<5;i++)
{
 System.out.println(name+""+i); Thread.sleep(1000);
}
}
// The block that catches the exception.
 catch(Exception e)
{         System.out.println("child interrupted");

}
System.out.println(name+""+"exiting");
}
}

// Class that takes the thread name and run the main thread.
class multithread
{
 public static void main(String args[ ])
 { // Creating child threads.
new NewThread("one"); new NewThread("two");
new NewThread("three");
// Block that may generate the exception.
try
{
for(int i=5;i>0;i--)

{
 System.out.println("main thread"+i);
Thread.sleep(10000);
}
}
```

```
// Block that catch the exception.
 catch(Exception e)
{
System.out.println("main thread interrupted");
}
System.out.println("main thread exiting");
}
}
```

Three test outputs:

Signature of the faculty

Exercise: Write a java program that correctly implements producer consumer problem using the concept of inter thread communication.

PROGRAM -9 A) Date:

Aim: Write a java program that connects to a database using JDBC

Program:

```
import java.sql.Connection;
import java.sql.DriverManager;
public class PostgreSQLJDBC
 {
  public static void main(String args[])
{
     Connection c = null;
     try {
       Class.forName("org.postgresql.Driver");
       c = DriverManager .getConnection("jdbc:postgresql://localhost:5432/testdb",
          "postgres", "123");
     } catch (Exception e) {
       e.printStackTrace();
       System.err.println(e.getClass().getName()+": "+e.getMessage());
       System.exit(0);
     }
     System.out.println("Opened database successfully");
  }
}
```

Three test outputs:

Signature of the faculty

Program
B): Write a java program to connect to a database using JDBC and insert values into it

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.Statement;
public class PostgreSQLJDBC
{
  public static void main(String args[])
{
    Connection c = null;
    Statement stmt = null;
    try {
      Class.forName("org.postgresql.Driver");
      c = DriverManager
        .getConnection("jdbc:postgresql://localhost:5432/testdb",
        "manisha", "123");
      c.setAutoCommit(false);
      System.out.println("Opened database successfully");
      stmt = c.createStatement();
      String sql = "INSERT INTO COMPANY (ID,NAME,AGE,ADDRESS,SALARY) "
          + "VALUES (1, 'Paul', 32, 'California', 20000.00 );";
     stmt.executeUpdate(sql);

      sql = "INSERT INTO COMPANY (ID,NAME,AGE,ADDRESS,SALARY) "
          + "VALUES (2, 'Allen', 25, 'Texas', 15000.00 );";
      stmt.executeUpdate(sql);

      sql = "INSERT INTO COMPANY (ID,NAME,AGE,ADDRESS,SALARY) "
          + "VALUES (3, 'Teddy', 23, 'Norway', 20000.00 );";
      stmt.executeUpdate(sql);

      sql = "INSERT INTO COMPANY (ID,NAME,AGE,ADDRESS,SALARY) "
          + "VALUES (4, 'Mark', 25, 'Rich-Mond ', 65000.00 );";
      stmt.executeUpdate(sql);

      stmt.close();
      c.commit();
      c.close();
    } catch (Exception e) {
      System.err.println( e.getClass().getName()+": "+ e.getMessage() );
      System.exit(0);
    }
    System.out.println("Records created successfully");
  }
}
```

Three test outputs:

Signature of the faculty

Program

C) : Write a java program to connect to a database using JDBC and delete values from it

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;


public class PostgreSQLJDBC6 {
  public static void main( String args[] )
    {
      Connection c = null;
      Statement stmt = null;
      try {
      Class.forName("org.postgresql.Driver");
        c = DriverManager
           .getConnection("jdbc:postgresql://localhost:5432/testdb",
           "manisha", "123");
        c.setAutoCommit(false);
        System.out.println("Opened database successfully");

        stmt = c.createStatement();
        String sql = "DELETE from COMPANY where ID=2;";
        stmt.executeUpdate(sql);
        c.commit();
        ResultSet rs = stmt.executeQuery( "SELECT * FROM COMPANY;" );
        while ( rs.next() ) {
```

```
            int id = rs.getInt("id");
            String name = rs.getString("name");
            int age = rs.getInt("age");
            String address = rs.getString("address");
            float salary = rs.getFloat("salary");
            System.out.println( "ID = " + id );
            System.out.println( "NAME = " + name );
            System.out.println( "AGE = " + age );
            System.out.println( "ADDRESS = " + address );
            System.out.println( "SALARY = " + salary );
            System.out.println();
          }
          rs.close();
          stmt.close();
          c.close();
        } catch ( Exception e ) {
          System.err.println( e.getClass().getName()+": "+ e.getMessage() );
          System.exit(0);
        }
        System.out.println("Operation done successfully");
      }
}
```

Three test outputs:

Signature of the faculty

PROGRAM -10 Date:

Aim: Write a java program to simulate a traffic light

Program:

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
// Class that allows user to select the traffic lights.
public class Trafficlight extends JFrame implements ItemListener
{
JRadioButton redbut,yellowbut,greenbut;
 public Trafficlight()
{
Container c = getContentPane();
c.setLayout(new FlowLayout());
// Create the button group.
ButtonGroup group= new ButtonGroup();
redbut = new JRadioButton("Red");
yellowbut = new JRadioButton("Yellow");
greenbut = new JRadioButton("Green");
group.add(redbut);
 group.add(yellowbut);
 group.add(greenbut);
// Add the buttons to the container.
c.add(redbut);
c.add(yellowbut);
 c.add(greenbut);
// Add listeners to perform action
 redbut.addItemListener(this);
 yellowbut.addItemListener(this);
 greenbut.addItemListener(this);
 addWindowListener(new WindowAdapter()
{
// Implement methods in Window Event class.
public void windowClosing(WindowEvent e)
{
System.exit(0);
}
} );
setTitle("Traffic Light ");
setSize(250,200);
setVisible(true);
}
// Implement methods in Item Event class.
public void itemStateChanged(ItemEvent e)
{
String name= " ",color=" ";
if(redbut.isSelected() )
          name = "Red";
 else if(yellowbut.isSelected() )
           name = "Yellow";
else if(greenbut.isSelected() )
          name = "Green";
```

```
JOptionPane.showMessageDialog(null,"The "+name+" light is simulated, "MessgeBox",
JOptionPane.INFORMATION_MESSAGE);
}
public static void main(String args[] )
{
new trafficlight();
}
}
```

Three Test Outputs:

Signature of the faculty

EXERCISE:
Write a java program that lets the user select one the three options: IT, CSE or ECE. When a radio button is selected, the radio button is turned on and only one option can be on at a time no option is on when program starts.

PROGRAM -11 Date:

Aim: Write a java program to create an abstract class named shape that contains an empty method named number of sides (). Provide three classes named trapezoid, triangle and Hexagon such that each one of the classes extends the class shape. Each one of the class contains only the method number of sides () that shows the number of sides in the given geometrical figures.

Program:

```
// Abstract class that contains abstract method.
abstract class Shape
{
abstract void numberOfSides();
}
// Classes that illustrates the abstract method.
class Trapezoid
{
void     numberOfSides()
{
System.out.println("The no. of side's in trapezoidal are6");
}
}
class    Triangle
{
void     numberOfSides()
{
System.out.println("The no. of side's in triangle are:3 ");
}
}
class    Hexogon
{
void numberOfSides()
{        System.out.println("The no. of side's in hexogon are:6  ");

}
}
// Class that create objects and call the method.
class ShapeDemo
{
public static void main(String args[])
{
Trapezoid obj1 = new Trapezoid();
Triangle obj2 = new Triangle();
Hexogon obj3 = new Hexogon();
obj1.numberOfSides();
obj2.numberOfSides();
obj3.numberOfSides(); }
}
```

Three test outputs:

Signature of the faculty

Exercise:write a program to compute area of different shapes using abstract class.

PROGRAM -12 Date:

Aim: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 TableDemo extends JFrame
{
	int i=0;
	int j=0;
	Object TabData[][]=new Object[5][2];
	JTable mytable;
	FileInputStream fr;
	DataInputStream in;
	public TableDemo()
	{
		String str=" ";
		Container contentpane=getContentPane();
		contentpane.setLayout(new BorderLayout());
		final String[] Column={","};
		try
		{
			FileInputStream fr=new FileInputStream("table.txt");
			DataInputStream in=new DataInputStream(fr);
			if((str=in.readLine())!=null)
			{
				StringTokenizer s=new StringTokenizer(str,",");
				while(s.hasMoreTokens())
				{
					for(int k=0;k<2;k++)
					{
						Column[k]=s.nextToken();
					}
				}
			}
			while((str=in.readLine())!=null)
			{
				StringTokenizer s=new StringTokenizer(str,",");
				while(s.hasMoreTokens())
				{
					for(j=0;j<2;j++)
					{
						TabData[i][j]=s.nextToken();
					}
					i++;
				}
			}
		}catch(Exception e)
		 {
			System.out.println(e.getMessage());
		}
```

```
                mytable=new JTable(TabData,Column);
                int v=ScrollPaneConstants.VERTICAL_SCROLLBAR_AS_NEEDED;
                int h=ScrollPaneConstants.HORIZONTAL_SCROLLBAR_AS_NEEDED;
                JScrollPane scroll=new JScrollPane(mytable,v,h);
                contentpane.add(scroll,BorderLayout.CENTER);
        }
        public static void main(String args[])
        {
                TableDemo t=new TableDemo();
                t.setSize(300,300);
                t.setVisible(true);
                t.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        }
}
```

Three test outputs:

Signature of the faculty

PROGRAM -13 Date:

Aim:Write a java program for handling mouse events

Program: mouseevent.java import java.awt.*;
import java.awt.event.*; import java.applet.*;

```
// Class that handles mouse events.
public class mouseevent extends Applet implements MouseListener,MouseMotionListener
{
 String msg="";
int mousex=0,mousey=0;
//        Method to initialize the applet.
public void init()
{
        addMouseListener(this);
        addMouseMotionListener(this);
}

//        Method to handle mouse clicked event .

public void mouseClicked(MouseEvent me)
{
mousex=0;
mousey=10; msg="mouse clicked"; repaint();
}

// Method to handle mouse entered event . public void mouseEntered(MouseEvent me)
{
mousex=0;
mousey=10; msg="mouse Entered"; repaint();
}

// Method to handle mouse entered event .
public void mouseExited(MouseEvent me)
{
 mousex=0; mousey=10; msg="mouse exited";
repaint();
}
// Method to handle mouse pressed event .
public void mousePressed(MouseEvent me)
{
 mousex=me.getX(); mousey=me.getY();
msg="down";
repaint();
}

//        Method to handle mouse relesed event .
 public void mouseReleased(MouseEvent me)
{
mousex=me.getX();
mousey=me.getY();
msg="Up";
```

```
repaint();
}


//          Method to handle mouse dragged event .
public void mouseDragged(MouseEvent me)
{
mousex=me.getX();
mousey=me.getY();
msg="";
showStatus("Dragged mouse at"+mousex+""+mousey); repaint();
}

// Method to handle mouse moved event .
public void mouseMoved(MouseEvent me)
{
showStatus("Moving mouseat"+me.getX()+""+me.getY());
}

// Method to display the message .
public void paint(Graphics g)
{
g.drawString(msg,mousex,mousey);
}
}
```

mouseevent.html:

```
/* <applet code=”mouseevent” width=200 height=200> </applet>
*/
```

Three Test Outputs:

Signature of the faculty

EXERCISE:
1.Write a java program for handling KEY BOARD events.

PROGRAM -14 Date:

Aim:Write a Java program loads phone no,name from a text file using hash table

Program:

```
// Demonstrate a Hashtable
import java.util.*;
class HTDemo {
public static void main(String args[]) {
Hashtable balance = new Hashtable();
Enumeration names;
String str;
double bal;
balance.put("John Doe", new Double(3434.34));
balance.put("Tom Smith", new Double(123.22));
balance.put("Jane Baker", new Double(1378.00));
balance.put("Todd Hall", new Double(99.22));
balance.put("Ralph Smith", new Double(-19.08));
// Show all balances in hash table.
names = balance.keys();
while(names.hasMoreElements()) {
str = (String) names.nextElement();
System.out.println(str + ": " +
balance.get(str));
}
System.out.println();
// Deposit 1,000 into John Doe's account
bal = ((Double)balance.get("John Doe")).doubleValue();
balance.put("John Doe", new Double(bal+1000));
System.out.println("John Doe's new balance: " +
balance.get("John Doe"));
}
}
```

Three test outputs:

Signature of the faculty

Exercise:

Write a Java program loads list of student names and roll numbers from a text file

PROGRAM -15 Date:

Aim: Implement the above program to load phone no, name from database instead of text file

```
import java.sql.*;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.Statement;
public class PostgreSQLJDBC  {
  public static void main( String args[] )
    {
      Connection c = null;
      Statement stmt = null;
      try {
        Class.forName("org.postgresql.Driver");
        c = DriverManager
           .getConnection("jdbc:postgresql://localhost:5432/testdb",
           "manisha", "123");
        System.out.println("Opened database successfully");

        stmt = c.createStatement();
        String sql = "CREATE TABLE COMPANY " +
                     "(ID INT  PRIMARY KEY     NOT NULL," +
                     " NAME           TEXT    NOT NULL, " +
                     " AGE            INT     NOT NULL, " +
                     " ADDRESS        CHAR(50), " +
                     " SALARY         REAL)";
        stmt.executeUpdate(sql);
        stmt.close();
        c.close();
      } catch ( Exception e ) {
        System.err.println( e.getClass().getName()+": "+ e.getMessage() );
        System.exit(0);
      }
      System.out.println("Table created successfully");
    }
}
```

Three test outputs:

Signature of the faculty

Exercise: Implement the above program to load emp details name,salary,address, from database .

PROGRAM -16 Date:

Aim:Write a Java program that takes tab separated data from a text file and inserts them into a database.

Program:

```
import java.io.BufferedReader;
import java.io.FileReader;

public class TabSeparatedFileReader {

    public static void main(String args[]) throws Exception {
      /**
       * Source file to read data from.
       */
      String dataFileName = "C:/temp/myTabSeparatedFile.txt";

      /**
       * Creating a buffered reader to read the file
       */
      BufferedReader bReader = new BufferedReader(
            new FileReader(dataFileName));
       String line;
       /**
       * Looping the read block until all lines in the file are read.
       */
      while ((line = bReader.readLine()) != null) {

         /**
          * Splitting the content of tabbed separated line
          */
         String datavalue[] = line.split("\t");
         String value1 = datavalue[0];
         String value2 = datavalue[1];
         int value3 = Integer.parseInt(datavalue[2]);
         double value4 = Double.parseDouble(datavalue[3]);

         /**
          * Printing the value read from file to the console
          */
         System.out.println(value1 + "\t" + value2 + "\t" + value3 + "\t"
               + value4);
      }
      bReader.close();
   }
```

Three test outputs:

Signature of the faculty

Exercise:
Write a program to reverse the specified n number of characters from the given text file and insert the data into database.

PROGRAM -17 Date:

Aim: Write a Java program that prints the meta-data of a given table

Program:

```
import java.sql.Connection;
import java.sql.DatabaseMetaData;
import java.sql.DriverManager;
import java.sql.SQLException;
public class JDBCDriverInformation {
	static String userid="scott", password = "tiger";
	static String url = "jdbc:odbc:bob";
	static Connection con = null;
	public static void main(String[] args) throws Exception {
	   Connection con = getOracleJDBCConnection();
	   if(con!= null){
	     System.out.println("Got Connection.");
	     DatabaseMetaData meta = con.getMetaData();
	     System.out.println("Driver Name : "+meta.getDriverName());
	     System.out.println("Driver Version : "+meta.getDriverVersion());

	   }else{
	          System.out.println("Could not Get Connection");
	   }
	}

	public static Connection getOracleJDBCConnection(){

		try {
			Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
		} catch(java.lang.ClassNotFoundException e) {
			System.err.print("ClassNotFoundException: ");
			System.err.println(e.getMessage());
		}

		try {
		  con = DriverManager.getConnection(url, userid, password);
		} catch(SQLException ex) {
			System.err.println("SQLException: " + ex.getMessage());
		}

		return con;
	}


}
```

Three test outputs:

Signature of the faculty

Exercise: Write a Java program that prints the meta-data of a given hash table.