CONTENTS

1. Syllabus

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY
HYDERABAD

II Year B.Tech. CSE II-Semester **T P C**
0 3 2

OBJECT ORIENTED PROGRAMMING LAB

Objectives:

- To make the student learn a object oriented way of solving problems.
- To teach the student to write programs in Java to solve the problems

Recommended Systems/Software Requirements:

- Intel based desktop PC with minimum of 166 MHZ or faster processor with at least 64 MB RAM and 100 MB free disk space
- JDK Kit. Recommended

Week1:
a) Write a Java program that prints all real solutions to the quadratic equation ax2 + bx + c = 0. Read in a, b, c and use the quadratic formula. If the discriminate b2 -4ac is negative, display a message stating that there are no real solutions.
b) The Fibonacci sequence is defined by the following rule:
The fist two values in the sequence are 1 and 1. Every subsequent value is the sum of the two values preceding it. Write a Java program that uses both recursive and non recursive functions to print the nth value in the Fibonacci sequence.
Week 2:
a) Write a Java program that prompts the user for an integer and then prints out all prime numbers up to that integer.
b) Write a Java program to multiply two given matrices.
c) Write a Java Program that reads a line of integers, and then displays each integer, and the sum of all the integers (Use StringTokenizer class of java.util)
Week 3:
a) Write a Java program that checks whether a given string is a palindrome or not. Ex: MADAM is a palindrome.
b) Write a Java program for sorting a given list of names in ascending order.
c) Write a Java program to make frequency count of words in a given text.
Week 4:
a) Write a Java program that reads a file name from the user, then displays information about whether the file exists, whether the file is readable, whether the file is writable, the type of file and the length of the file in bytes.
b) Write a Java program that reads a file and displays the file on the screen, with a line number before each line.
c) Write a Java program that displays the number of characters, lines and words in a text file.
Week 5:
a) Write a Java program that:
i) Implements stack ADT.

ii) Converts infix expression into Postfix form
iii) Evaluates the postfix expression

Week 6:

a) Develop an applet that displays a simple message.

b) Develop an applet 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.

Week 7:

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.

Week 8:

a) Write a Java program for handling mouse events.

Week 9 :

a) Write a Java program that creates three threads. First thread displays "Good Morning" every one second, the second thread displays "Hello" every two seconds and the third thread displays "Welcome" every three seconds.

b) Write a Java program that correctly implements producer consumer problem using the concept of inter thread communication.

Week 10:

Write a program that creates a user interface to perform integer divisions. The user enters two numbers in the textfields, Num1 and Num2. The division of Num1 and Num2 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 NumberFormatException. If Num2 were Zero, the program would throw an ArithmeticException Display the exception in a message dialog box.

Week 11:

a) Write a java program that simulates a traffic light. The program lets the user select one of three lights: red, yellow, or green. When a radio button is selected, the light is turned on, and only one light can be on at a time No light is on when the program starts.

b) Write a Java program that allows the user to draw lines, rectangles and ovals.

Week 12:

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

b) 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 seperated by commas. Write a java program to display the table using JTable component.

TEXT BOOKS:

1. Java How to Program, Sixth Edition, H.M.Dietel and P.J.Dietel, Pearson Education/PHI
2. Introduction to Java programming, Sixth edition, Y.Daniel Liang, Pearson Education
3. Big Java, 2nd edition, Cay Horstmann, Wiley Student Edition, Wiley India Private Limited.

2. Required Work

There will be homework assignments of about 12 programmatic/project scripts/memos, most of which will involve programming. Assignments are due at the beginning of its theoretic class on the date specified. Late assignments will receive 75% of full credit if they are handed in within one week of the specified due date. After one week, no credit will be given or else losing of credits is made in practice. There is no specific attendance policy for the course, although it is expected that absences will leave the student unprepared for tests and assignments. Tests will not be rescheduled except in extreme circumstances. However, the lowest quiz grade will be dropped. Grades will be determined as follows:

Day – to- Day evaluation Marks	15 marks
Internal practical Examination	10 marks
Final Exam (University / External End Exam)	50 marks

At the minimum, traditional grading cutoffs will apply. That is, 90% is guaranteed an A, 87% is guaranteed a B+, etc. Depending on class performance, some shifting of grades (in an upward direction only) may occur as final letter grades are assigned.

3. Course Objectives:

The specific objectives of this course are:

- Understand the principles of abstract data types and encapsulation, and the advantages of using these principles to write programs in JAVA
- To use object-oriented analysis and design methodology to build models of simple applications.
- To explain what Java is, what it is composed of, how it compares to other programming environments, what its advantages are, and how to install and configure the development environment.

4. Course Outcomes

- Upon completion of this course students are able to write Java classes, encapsulating logic, reusing existing code through inheritance / polymorphism and composition, and modeling real-world relationships between objects.

5. Course Assessment:

Course objective	Intended educational outcome	Means of assessment	Criteria of success	**Objectives ABET(a-k)**
Objective1:write simple programs	Students will write programs to get used to java environment	1.by observations 2.by lab records 3.by viva-voice	1. 100% pass Rate 2. 80% passing rate 3. 82% positive response 4. 76% minimum	**B,cj**
Objective2: Write programs on inheritance	Students will execute different types of inheritances	1.by execution 2.by observation 3.by lab records 4.by viva-verse	1. 100% pass Rate 2. 80% passing rate 3. 82% positive response	**C,d,e,j**
Objective3:create packages & interfaces	Students will write programs of creating packages & interfaces	1.by execution 2.by observation 3.by lab record	1. 100% pass Rate 2. 80% passing rate 3. 82% positive response	**C,d,e,f,j**
Objective4:Exceptional handing	Students will write their own exception class	1.by observation 2.by lab record	1. 80% passing rate 2. 82% positive response 3. 76% minimum	**C,d,e**
Objective5:create multithreads	Students will perform execution of threads simultaneously	1.by observation 2.by lab record	1.80% passing rate 2.82% positive response 3.76% minimum	**A,c,e,f**

Objective6:create Applets	Students will create applets for web applications	1.by observation 2.by lab record	1. 100% pass Rate 2. 80% passing rate 3. 82% positive response	**g,I,j,k**
Objective7:Introducing AWT	Students will work with windows, graphics ,text	1.by observation 2.by lab record	4. 100% pass Rate 5. 80% passing rate 82% positive response	**f,I,j**

5.1 Add-On Contents

S.no	Name of the topics	No. of Classes	Sources
1	Usage of java.sql	3	Internet
2	Usage of javax.servlet	3	Internet

6. Program Outcomes

a) Graduates will demonstrate knowledge of mathematics, science and engineering.
b) Graduates will demonstrate an ability to identify, formulate and solve engineering problems.
c) Graduate will demonstrate an ability to design and conduct experiments, analyze and interpret data.
d) Graduates will demonstrate an ability to design a system, component or process as per needs and specifications.
e) Graduates will demonstrate an ability to visualize and work on laboratory and multidisciplinary tasks.
f) Graduate will demonstrate skills to use modern engineering tools, software's and equipment to analyze problems.
g) Graduates will demonstrate knowledge of professional and ethical responsibilities.
h) Graduate will be able to communicate effectively in both verbal and written form.
i) Graduate will show the understanding of impact of engineering solutions on the society and also will be aware of contemporary issues.
j) Graduate will develop confidence for self education and ability for life-long learning.
k) Graduate who can participate and succeed in competitive examinations.

7. Program Objective/Outcome Matrix

.

Program Objective	Program Outcomes										
	a	b	c	d	e	f	g	h	i	j	k
1	√		√				√		√	√	√
2	√		√		√				√	√	√
3	√	√	√	√	√				√	√	√
4	√		√	√	√				√	√	√
5	√		√						√	√	√
6	√	√	√	√	√				√		√

COURSE/LABORATORY SCHEDULE: 40 hours laboratory facility with video lectures

Week-1(a):

```
/*TO PRINTS REAL SOLUTIONS TO THE QUADRATIC EQUATION*/

 import java.io.*;
 class QudEq
 {
  public static void main(String args[])throws IOException
  {
   BufferedReader br;
   int a,b,c;
   double x,y,z;
   System.out.println("ENTER a VALUE:");
   br= new BufferedReader(new InputStreamReader(System.in));
   a=Integer.parseInt(br.readLine());
   System.out.println("ENTER b VALUE:");
   b=Integer.parseInt(br.readLine());
   System.out.println("ENTER c VALUE:");
   c=Integer.parseInt(br.readLine());
   z=Math.sqrt(b*b-4*a*c);
   x=(-b+Math.sqrt(b*b-4*a*c))/(2*a);
   y=(-b-Math.sqrt(b*b-4*a*c))/(2*a);
   if(z>0)
    {
      System.out.println("THE REAL SOLUTIONS ARE X="+x+"\t"+"Y="+y);
    }
    else
     {
       System.out.println("THERE ARE NO REAL SOLUTIONS");
     }
   }
  }
/*////////////////////////////////////////////////////////
INPUT:ENTER a VALUE:1
      ENTER b VALUE:-4
      ENTER c VALUE:3
OUTPUT:THE REAL SOLUTIONS ARE X=3.0     Y=1.0
  */
```

Week-1(b):

```
/*FIBONACCI SEQUENCE USING RECURSIVE*/

import java.io.*;
public class FibRec
{
  static int fib(int i)
  {
   if(i==1||i==2)
   {
    return i-1;
   }
   else
    {
     return fib(i-1)+fib(i-2);
    }
   }

 public static void main(String args[])throws IOException
 {
 BufferedReader br;
 int n;
 System.out.println("ENTER nth VALUE");
 br = new BufferedReader(new InputStreamReader(System.in));
 n=Integer.parseInt(br.readLine());
 System.out.print("FIBONACCI SEQUENCE:");
 for(int i=2;i<=n+1;i++)
  {
  System.out.print(fib(i)+"\t");
  }
 }
}
/*//////////////////////////////////////////////////////
INPUT:ENTER nth VALUE:5
OUTPUT:FIBONACCI SEQUENCE:1   1    2    3    5
*/

/*FIBONACCI SEQUENCE USING NON RECURSIVE*/
  import java.io.*;
  class FibNonRec
  {
   public static void main(String args[])throws IOException
   {
     BufferedReader br;
     int low=0;
     int heigh=1,t,n;
```

```
    br=new BufferedReader(new InputStreamReader(System.in));
    System.out.println("ENTER nth VALUE");
    n=Integer.parseInt(br.readLine());
    System.out.print("FIBONACCI SEQENCE IS:");
    while(heigh<=n)
     {
      System.out.print(heigh+"\t");
       t=heigh;
       heigh=heigh+low;
       low=t;
     }
   }
  }
/*//////////////////////////////////////////////////////
INPUT:ENTER nth VALUE:5
OUTPUT:FIBONACCI SEQENCE IS:1  1   2    3    5
*/
```

Week-2(a):

```
/*TO PRINT PRIME NUMBERS UP TO nth VALUE*/
import java.io.*;
class Primenos
{
 public static void main(String args[])throws IOException
 {
  BufferedReader br;
  int i,t,flag,n;
  br=new BufferedReader(new InputStreamReader(System.in));
  System.out.print("ENTER nth VALUE:");
  n=Integer.parseInt(br.readLine());
  System.out.println("PRIME NUMBERS UP TO"+" "+n+":");
  for(i=2;i<=n;i++)
  {
   flag=1;
   for(t=2;t<i;t++)
   {
    if(i%t==0)
    {
     flag=0;
     break;
    }
   }
  if(flag==1)
  System.out.println(i);
  }
```

```
 }
}
/*//////////////////////////////////////////////////////
INPUT:ENTER nth VALUE:20
OUTPUT:PRIME NUMBERS UP TO 20:
2
3
5
7
11
17
19
*/
```

Week-2(b):

```
/*TO MULTIPLY TWO GIVEN MATRICES*/
import java.io.*;
class Matmul
{
 public static void main(String args[])throws IOException
 {
  int r,c,r1,c1,r2,c2,r3,c3,t;
  int m1[][]=new int[10][10];
  int m2[][]=new int[10][10];
  int m3[][]=new int[10][10];
  System.out.println("ENTER  MATRICES1 DETAIL ");
  System.out.println("ENTER NO OF ROWS:");
  BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
  r1=Integer.parseInt(br.readLine());
  System.out.println("ENTER NO OF COLUMNS");
  c1=Integer.parseInt(br.readLine());
  System.out.println("ENTER  MATRICES2 DETAIL");
  System.out.println("ENTER NO ROWS");
  r2=Integer.parseInt(br.readLine());
  System.out.println("ENTER NO OF COLUMNS:");
  c2=Integer.parseInt(br.readLine());
   if(c1==r2)
 {
 System.out.println("ENTER MAT1 ELEMENTS:");
 for(r=0;r<r1;r++)
  {
    for(c=0;c<c1;c++)
    {
```

```
      m1[r][c]=Integer.parseInt(br.readLine());
    }
  }
    System.out.println("ENTER MAT2  ELEMENT:");
  for(r=0;r<r2;r++)
   {
    for(c=0;c<c2;c++)
    {
     m2[r][c]=Integer.parseInt(br.readLine());
    }
   }
System.out.print("THE RESULT ");
 for(r=0;r<r1;r++)
  {
   for(c=0;c<c2;c++)
    { m3[r][c]=0;
     for(t=0;t<c1;t++)
      m3[r][c]+=m1[r][t]*m2[t][c];
    }
  }
 for(r=0;r<r1;r++)
  {
   System.out.println(" ");
    for(c=0;c<c2;c++)
     {
         System.out.print(m3[r][c]);
         System.out.print(" ");
      }
  }
 }

 else
 {
  System.out.println("MATRICES MULTIPLY NOT POSSIBLE");
 }
 }
}
/*//////////////////////////////////////////////////////*/
```

Week-2(c):

```
/*To read a line of integers and display each integer and sum of all integers*/
import java.util.*;
import java.io.*;
class StrTok{
 public static void main(String args[])throws IOException{
 BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
 System.out.println("Enter String in numbers");
 String str=br.readLine();
 StringTokenizer s=new StringTokenizer(str);
 int x,sum=0;
 while(s.hasMoreTokens()){
  String str1=s.nextToken();
  x=Integer.parseInt(str1);
  System.out.println(x);
   sum+=x;
  }
 System.out.println("total sum:"+sum);
 }
}
/*////////////////////////////////////////////////
INPUT:Enter string in numbers:
     23 2 5 6
OUTPUT:23
          2
          5
          6
          total sum:36
*/
```

Week-3(a):

```
//** Palindrome check **////

import java.io.*;
public class Palindrome
{
 public static void main(String args[]) throws IOException
 {
  BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
  System.out.println("Enter String:");
  String str=br.readLine();
  char ch[]=new char[str.length()];
```

```
 for(int i=str.length()-1,j=0;i>=0;i--,j++)
 ch[j]=str.charAt(i);
 String restr=new String(ch);
 System.out.println("Reverse of String "+str+" is "+restr);
 if(str.compareTo(restr)==0)
 System.out.println(str+" "+"is a Palindrome");
 else
 System.out.println(str+" "+"is not a Palindrome");
 }
 }
```

Week-3(b):

```
//** Sorting Strings **//
class SortStrings
 {
  public static void main(String args[])
   {
    String temp;
    for(int i=0;i<args.length;i++)
    {
     for(int j=i+1;j<args.length;j++)
     {
       if(args[j].compareTo(args[i])<0)
       {
        temp=args[i];
        args[i]=args[j];
        args[j]=temp;
        }
      }
     System.out.println(args[i]);
     }
    }
   }
```

Week-3(c):

```
// *** To count number of words in a given text ****/////

import java.io.*;
import java.util.*;

class CountWords
{
```

```
public static void main(String[] args) throws IOException
{
BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

System.out.println("Enter a string : ");
String str=br.readLine();
int count=0;
if(str.length()==0)
{
System.out.println("No.of words in given text:" + count);
}
else
{
for(int i=0;i<str.length();i++)
{
       if(str.charAt(i)==' ')
       {
              count++;
       }
}

System.out.println("No.of words in given text:" + (count+1));
}
}
}
```

Week-4(a):

```
//**** Information about a file ****////

import java.io.*;
class FileInfo
{
 static void p(String s)
 {
  System.out.println(s);
 }
 public static void main(String args[]) throws IOException
 {
  InputStreamReader isr=new InputStreamReader(System.in);
  BufferedReader br=new BufferedReader(isr);
  p("Enter File Name:");
  File f1=new File(br.readLine());
  p("Filename:"+f1.getName());
  p("AbsPath:"+f1.getAbsolutePath());
```

```
  p(f1.exists()?"exists":"doesnotexist");
  p(f1.canWrite()?"iswritable":"not writable");
  p(f1.canRead()?"isreadable":"notreadable");
  p(f1.isDirectory()?"isDir":"not Dir");
  p("file size"+f1.length()+"bytes");
 }
}
```

Week-4(b):

```
//**** Displaying contents of a file with line number ****///

import java.io.*;
class FileWithLineNo
{
 public static void main(String args[]) throws IOException
 {
  BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
  System.out.println("Enter File Name to read:");
   FileInputStream fin=new FileInputStream(br.readLine());
  int n=fin.read();
  int no=1;
  if(n!=-1)
  {
   System.out.println(no+"\t");
  }
  while(n!=-1)
  {
   System.out.print((char)n);
   if((char)n=='\n')
   {
       no++;
       System.out.print(no+"\t");
   }
   n=fin.read();
 }
}
}
```

Week-4(c):

```
//***** Counting number of words, lines and characters in a given file***///

import java.io.*;
class FCWL
{
	public static void main(String[] args) throws IOException
	 {
	   BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
	   System.out.println("Enter the file name : ");
		FileInputStream fin=new FileInputStream(br.readLine());
	   int n=fin.read();
	   int c=0,w=0,l=0;
	   boolean flag=true;
	   if(n!=-1)
	   {
	   w=1;l=1;
	   while(n!=-1)
	   {
			if(((char)n=='\n') || ((char)n=='\t') || ((char)n==' ') || ((char)n=='\r'))
	  {
			if(flag==true)
			{
			    w++;
			    flag=false;
			}
			   }
			   else
			   {
			flag=true;
				c++;
		}
		if((char)n=='\n')
		{
			l++;

		}

		n=fin.read();
	   }

	   System.out.println("No.of char's : " + c);
	   System.out.println("No.of word's : " + w);
	   System.out.println("No.of line's : " + l);
```

```
            }
            else
            {
            System.out.println("file is empty");
            }
      }
}
```

Week-5:

//*** To implement STACK ADT***///

```
import java.io.*;
interface IntStack {
  void push(int item);
  void pop();
}

class DynStack implements IntStack {
  private int stck[];
  private int tos;

  // allocate and initialize stack
  DynStack(int size) {
    stck = new int[size];
    tos = -1;
  }

  // Push an item onto the stack
  public void push(int item) {
    // if stack is full, allocate a larger stack
    if(tos==stck.length-1) {
      int temp[] = new int[stck.length * 2]; // double size
      for(int i=0; i<stck.length; i++) temp[i] = stck[i];
      stck = temp;
      stck[++tos] = item;
    }
    else
      stck[++tos] = item;
  }

  // Pop an item from the stack
  public void pop() {
    if(tos < 0) {
```

```
      System.out.println("Stack underflow.");

    }
    else
      System.out.println("Popped element: "+ stck[tos--]);
  }


 public void display()
{
 if(tos==-1)
 System.out.println("Stack is EMPTY");
 else
 System.out.println("Elements in Stack are:");
 for(int i=0;i<=tos;i++)
 System.out.println(stck[i]);
}
}
class Stack1 {
  public static void main(String args[]) throws IOException {
    int ch,ele;
    DynStack mystack = new DynStack(5);
    BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
while(true)
{
System.out.println("1.PUSH 2.POP 3.DISPLAY otherwise EXIT");
ch=Integer.parseInt(br.readLine());
switch(ch)
{
 case 1: System.out.println("Enter element you want to push:");
         ele=Integer.parseInt(br.readLine());
         mystack.push(ele);break;
 case 2: mystack.pop();

         break;
 case 3: mystack.display();
         break;
 default: System.exit(0);

  }
}
}
}
```

Week-6(a):

//***** Applet that displays a simple message ***///

```
import java.applet.*;
import java.awt.*;
/*
<applet code="AppletMsg" height=300 width=300>
</applet>
*/
public class AppletMsg extends Applet
{
public void init()
{
}
public void start()
{
}
public void paint(Graphics g)
{
g.drawString("This is an APPLET",50,30);
}
}
```

Week-6(b):

//*** Computing factorial using Applets ***///

/* Week 6 : b) Develop an applet that receives an integer in one text field, and computes its factorial Value and returns it in another text field, when the button named “Compute” is clicked.
*/

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

/*
<applet code=FactGUI width=300 height=300>
</applet>
*/

public  class FactGUI extends Applet
```

```
        implements ActionListener
{

   Button btn,clearbtn;
   Label lbl1,lbl2;
   TextField tf1,tf2;


   public void init()
   {


      btn=new Button("COMPUTE");
      btn.addActionListener(this);
      clearbtn=new Button("CLEAR");
      clearbtn.addActionListener(this);

       tf1=new TextField(30);
       tf2=new TextField(30);

       lbl1=new Label("NUMBER");

       lbl2=new Label("RESULT");

//       setLayout(new GridLayout(3,2));
         add(lbl1); add(tf1);
      add(lbl2); add(tf2);

       add(btn); add(clearbtn);
   }

   public void actionPerformed(ActionEvent e)
   {
     if(e.getSource()==btn)
     {
      int a=Integer.parseInt(tf1.getText());
      int fact=1;
      for(int i=1;i<=a;i++)
        fact*=i;
      tf2.setText(""+fact);
     }
     else
     {
       tf1.setText("");
       tf2.setText("");
     }
```

```
    }


}
```

Week-7:

```
// **** Designing Calculator ***///

import java.awt.*;
import java.awt.event.*;
import java.applet.*;
/*
<applet code="Calc" height=300 width=300>
</applet>
*/
public class Calc extends Applet implements ActionListener
{
TextField dis;
double arg=0;
String op="=";
boolean start=true;
public void init()
{
setLayout(new BorderLayout());
dis=new TextField("0");
dis.setFont(new Font("Arial",Font.BOLD,24));
dis.setEditable(false);
add(dis,BorderLayout.NORTH);
Panel p=new Panel();
p.setLayout(new GridLayout(4,4));
String buttons="789/456*123-0.=+";
for(int i=0;i<buttons.length();i++)
{
Button b=new Button(buttons.substring(i,i+1));
b.setFont(new Font("Arial",Font.BOLD,24));
p.add(b);
b.addActionListener(this);
}
add(p,BorderLayout.CENTER);
}
public void actionPerformed(ActionEvent ae)
{
String s=ae.getActionCommand();
if('0'<=s.charAt(0)&&s.charAt(0)<='9'||s.equals("."))
```

```
{
if(start)
dis.setText(s);
else
dis.setText(dis.getText()+s);
start=false;
}
else
{
if(start)
{
if(s.equals("."))
{
dis.setText(s);
start=false;
}
else
op=s;
}
else
{
calcu(Double.parseDouble(dis.getText()));
op=s;
start=true;
}
}
}
public void calcu(double n)
{
if(op.equals("+"))
arg+=n;
else
if(op.equals("-"))
arg-=n;
else
if(op.equals("*"))
arg*=n;
else
if(op.equals("/"))
arg/=n;
else
if(op.equals("="))
arg=n;


dis.setText(""+arg);
```

```
}
public void paint(Graphics g)
{
}
}
```

Week-8:

//*** Handling Mouse Events***///

```
// Demonstrate the mouse event handlers.
import java.awt.*;
import java.awt.event.*;
import java.applet.*;
/*
  <applet code="MouseEvents" width=300 height=100>
  </applet>
*/

public class MouseEvents extends Applet
  implements MouseListener, MouseMotionListener {

  String msg = "";
  int mouseX = 0, mouseY = 0; // coordinates of mouse

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

  // Handle mouse clicked.
  public void mouseClicked(MouseEvent me) {
    // save coordinates
    mouseX = 0;
    mouseY = 10;
    msg = "Mouse clicked.";
    repaint();
  }

  // Handle mouse entered.
  public void mouseEntered(MouseEvent me) {
    // save coordinates
    mouseX = 0;
    mouseY = 10;
    msg = "Mouse entered.";
    repaint();
```

```
  }

  // Handle mouse exited.
  public void mouseExited(MouseEvent me) {
    // save coordinates
    mouseX = 0;
    mouseY = 10;
    msg = "Mouse exited.";
    repaint();
  }

  // Handle button pressed.
  public void mousePressed(MouseEvent me) {
    // save coordinates
    mouseX = me.getX();
    mouseY = me.getY();
    msg = "Down";
    repaint();
  }

  // Handle button released.
  public void mouseReleased(MouseEvent me) {
    // save coordinates
    mouseX = me.getX();
    mouseY = me.getY();
    msg = "Up";
    repaint();
  }

  // Handle mouse dragged.
  public void mouseDragged(MouseEvent me) {
    // save coordinates
    mouseX = me.getX();
    mouseY = me.getY();
    msg = "*";
    showStatus("Dragging mouse at " + mouseX + ", " + mouseY);
    repaint();
  }

  // Handle mouse moved.
  public void mouseMoved(MouseEvent me) {
    // show status
    showStatus("Moving mouse at " + me.getX() + ", " + me.getY());
  }

  // Display msg in applet window at current X,Y location.
```

```
  public void paint(Graphics g) {
    g.drawString(msg, mouseX, mouseY);
  }
}
```

Week-9(a):

```
// *** implementing Multithreading concept****////

class NewThread implements Runnable {
  String name; // name of thread
  Thread t;

  NewThread(String threadname) {
    name = threadname;
    t = new Thread(this, name);
    System.out.println("New thread: " + t);
    t.start(); // Start the thread
  }

  // This is the entry point for thread.
  public void run() {
    try {
      for(int i = 5; i > 0; i--) {
        if(name.equals("Good Morning"))
        {
        System.out.println(name + ": " + i);
        Thread.sleep(1000);
        }
        else if(name.equals("Hello"))
        {
        System.out.println(name + ": " + i);
        Thread.sleep(2000);
        }
        else if(name.equals("Welcome"))
        {
        System.out.println(name + ": " + i);
        Thread.sleep(3000);
       }
      }
    } catch (InterruptedException e) {
      System.out.println(name + " interrupted.");
    }
    System.out.println(name + " exiting.");
  }
```

```
}

class ThreadsDemo {
  public static void main(String args[]) {
    NewThread ob1 = new NewThread("Good Morning");
    NewThread ob2 = new NewThread("Hello");
    NewThread ob3 = new NewThread("Welcome");

    // wait for threads to finish
    try {
      System.out.println("Waiting for threads to finish.");
      ob1.t.join();
      ob2.t.join();
      ob3.t.join();
    } catch (InterruptedException e) {
      System.out.println("Main thread Interrupted");
    }

    System.out.println("Main thread exiting.");
  }
}
```

Week-9(b):

```
// Program for producer-consumer problem using threads***////

class Q {
  int n;
  boolean valueSet = false;

  synchronized int get() {
    if(!valueSet)
      try {
        wait();

     } catch(InterruptedException e) {
        System.out.println("InterruptedException caught");
      }

      System.out.println("Got: " + n);
      valueSet = false;
      notify();
      return n;
  }
```

```
  synchronized void put(int n) {
    if(valueSet)
      try {
        wait();
      } catch(InterruptedException e) {
        System.out.println("InterruptedException caught");
      }

      this.n = n;
      valueSet = true;
      System.out.println("Put: " + n);
      notify();
  }
}

class Producer implements Runnable {
  Q q;

  Producer(Q q) {
    this.q = q;
    new Thread(this, "Producer").start();
  }

  public void run() {
    int i = 0;

    while(true) {
      q.put(i++);
    }
  }
}

class Consumer implements Runnable {
  Q q;

  Consumer(Q q) {
    this.q = q;
    new Thread(this, "Consumer").start();
  }

  public void run() {
    while(true) {
      q.get();
    }
  }
}
```

```
class PCFixed {
  public static void main(String args[]) {
    Q q = new Q();
    new Producer(q);
    new Consumer(q);

    System.out.println("Press Control-C to stop.");
  }
}
```

Week-10:

/* Week 10 : Write a program that creates a user interface to perform integer divisions. The user enters two numbers in the textfields, Num1 and Num2. The division of Num1 and Num2 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 NumberFormatException. If Num2 were Zero, the program would throw an ArithmeticException Display the exception in a message dialog box. */

```
import java.applet.*;
import java.awt.*;
import java.awt.event.*;
class SampleDialog extends Dialog implements ActionListener
{
SampleDialog(Frame parent,String title,String msg)
{
super(parent,title,false);
setLayout(new FlowLayout());
setSize(300,200);
add(new Label(msg));
Button b;
add(b=new Button("OK"));
b.addActionListener(this);
}
public void actionPerformed(ActionEvent ae)
{
dispose();
}
}
public  class Division extends Frame
            implements ActionListener
{
```

```
   Button btn;
   Label lbl1,lbl2,lbl3;
   TextField tf1,tf2,tf3;
  Division()
 {
    super("Exception Handler");

    btn=new Button("DIVIDE");
    btn.addActionListener(this);

       tf1=new TextField(30);
       tf2=new TextField(30);
       tf3=new TextField(30);
       lbl1=new Label("NUM 1");
       lbl2=new Label("NUM 2");
       lbl3=new Label("RESULT");
     setLayout(new FlowLayout());
         add(lbl1); add(tf1);
        add(lbl2); add(tf2);
         add(lbl3); add(tf3);
        add(btn);
  addWindowListener (new WindowAdapter()  {
  public void windowClosing (WindowEvent e)   {
  System.exit(0);
}
});
}

public void actionPerformed(ActionEvent e)
   {
    if(e.getSource()==btn)
    {
   try
   {
     int a=Integer.parseInt(tf1.getText());
     int b=Integer.parseInt(tf2.getText());
     int c=a/b;
     tf3.setText(""+c);
   }
   catch(NumberFormatException ex)
   {
         tf3.setText("--");
         SampleDialog d=new SampleDialog(this,"EXCEPTION","ONLY INTEGER
DIVISION");
         d.setVisible(true);
```

```
          }
    catch(ArithmeticException ex)
    {
         tf3.setText("--");
         SampleDialog d=new SampleDialog(this,"EXCEPTION","DIVIDE BY ZERO");
          d.setVisible(true);
        }
    catch(Exception ex)
    {
         tf3.setText("--");
         SampleDialog d=new SampleDialog(this,"EXCEPTION",""+ex);
          d.setVisible(true);
   }
     }
    }

   public  static void main(String args[])
   {
     Division b=new Division();
     b.setSize(300,300);
     b.setVisible(true);
   }
}
```

Week-11:

```
// **** Program to simulate Traffic Light***///

import java.awt.*;
import java.awt.event.*;
import java.applet.*;
/*
  <applet code="Tr" width=350 height=350>
  </applet>
*/
public class Tr extends Applet implements ItemListener {

  Checkbox red, green, yellow;
  CheckboxGroup cbg;

  public void init() {
    cbg = new CheckboxGroup();
    red = new Checkbox("RED", cbg, false);
    yellow = new Checkbox("YELLOW", cbg, false);
    green = new Checkbox("GREEN", cbg, false);
```

```
    add(red);
    add(yellow);
    add(green);
    red.addItemListener(this);
    yellow.addItemListener(this);
    green.addItemListener(this);

  }
  public void itemStateChanged(ItemEvent ie) {
    repaint();
  }

  public void paint(Graphics g) {
    Checkbox sel;
    red.setBackground(Color.white);
    green.setBackground(Color.white);
    yellow.setBackground(Color.white);
    sel= cbg.getCurrent();
    g.drawOval(300-50,100-50,100,100);
    g.drawOval(300-50,200-50,100,100);
    g.drawOval(300-50,300-50,100,100);

    if(sel==red)
    {
    g.setColor(Color.red);
    g.fillOval(300-50,100-50,100,100);
    sel.setBackground(Color.red);
    }
    else
    if(sel==yellow)
    {
    g.setColor(Color.yellow);
    g.fillOval(300-50,200-50,100,100);
    sel.setBackground(Color.yellow);
    }
    else if(sel==green)
    {
    g.setColor(Color.green);
    g.fillOval(300-50,300-50,100,100);
    sel.setBackground(Color.green);
    }
}
}
```

Week-12:

```
//**** Program that allows the user to draw lines, rectangles and ovals***///

import java.awt.*;
import java.awt.event.*;
import java.applet.*;
/*
 <applet code="DrawShape" width=300 height=300>
</applet>
*/

public class DrawShape extends Applet
                implements MouseListener,ActionListener,MouseMotionListener
{
Button lines,ovals,rect;
int ch;
int x,y,x2,y2,k;

        public void mouseClicked(MouseEvent e)
        {

        }
        public void mousePressed(MouseEvent e)
        {

                x=e.getX();
                y=e.getY();

        }
        public void mouseReleased(MouseEvent e)
        {
                x2=e.getX();
                y2=e.getY();
           repaint();
        }
        public void mouseEntered(MouseEvent e)
        {

        }
        public void mouseExited(MouseEvent e)
        {

        }
```

```
    public void mouseDragged(MouseEvent e)
    {

}
    public void mouseMoved(MouseEvent e)
    {


    }
    public void init()
    {
    lines=new Button("line");
    ovals=new Button("oval");
    rect =new Button("rect");


    add(lines);add(ovals);add(rect);
    lines.addActionListener(this);
    ovals.addActionListener(this);
    rect.addActionListener(this);

    addMouseListener(this);
    addMouseMotionListener(this);
    }
    public void actionPerformed(ActionEvent ki)
    {
            if(ki.getSource()==lines)
             ch=1;
            else if(ki.getSource()==ovals)
             ch=2;
            else if(ki.getSource()==rect)
             ch=3;

    }
    public void paint(Graphics g)
    {
    switch (ch)
    {
    case 1:g.drawLine(x,y,x2,y2);
            break;
    case 2:g.drawOval(x,y,(x2-x),(y2-y));
            break;
    case 3:g.drawRect(x,y,(x2-x),(y2-y));
            break;
```

```
        }
}}
```