Object Oriented Programming with Java

Lab Exercise

Lab Unit – 1 (2 Hrs actual Time)

1.1 Write a program to display any message:

1.2 Write a Java program to display default value of all primitive data types of Java.

1.3 Write a program check two strings are equal or not.

Lab Unit – 2 (2 Hrs actual Time)

2.1 Write a program to give the examples of operators.

2.1.1 Increment and decrement operators.
2.1.2 Bitwise Complement Operator.
2.1.3 Arithmetic operator.
2.1.4 Relational Operator
2.1.5 Bitwise operator.
2.1.6 Conditional Operator.

2.2 Write a program to give the example of control statements.

2.2.1 If statements.
2.2.2 Switch Statements.
2.2.3 For loop.
2.2.4 While Statements.
2.2.5 Do statements

2.3 Write a program to calculate the following

2.3.1 Find the length of array.
2.3.2 Demonstrate a one-dimensional array.
2.3.3 Demonstrate a two-dimensional array.
2.3.4 Demonstrate a multi-dimensional array.

2.4 Write a program give example for command line arguments.

2.4.1 To find the sum of command line arguments and count the invalid integers entered.

2.4.2 To get the name using command line.

2.5 Write a program to print the following triangle of binary digits.

2.5.1

```
1
1 0 1
1 0 0 0 1
1 0 0 0 0 0 1
1 0 0 0 0 0 0 0 1
```

2.5.2

```
5
4 5
3 4 5
2 3 4 5
1 2 3 4 5
0 1 2 3 4 5
```

2.5.3

```
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
```

2.6. Write a program to the find the following

2.6.1 Prime number checking

2.6.2 Sum of digit

2.7 Write a program to arrange the numbers in ascending order.

2.8 Write a program to calculate the roots of Quadratic equations.

2.9 Write a program for calculating Matrix Operations.

2.9.1 Addition.

2.9.2 Multiplication.

Lab Unit – 3 (2 Hrs Real Time)

3.1 Write a program to create a room class, the attributes of this class is roomno, roomtype, roomarea and ACmachine. In this class the member functions are setdata and displaydata.

3.2 Write a program create a class 'simpleobject'. Using constructor display the message.

3.3 Write a program for the following

1. Example for call by value.
2. Example for call by reference.

3.4 Write a program to give the example for 'this' operator. And also use the 'this' keyword as return statement.

Lab – 4 (2 Hrs Real Time)

4.1 Write a program to demonstrate static variables, methods, and blocks.

4.2 Write a program for reuse class. For this program use the above 'room class' program.

4.3 Create class named as 'a' and create a sub class 'b'. Which is extends from class 'a'. And use these classes in 'inherit' class.

4.4 Write a program to give the example for method overriding concepts.

4.5 Write a program to give the example for 'super' keyword.

Lab – 5 (2 Hrs Real Time)

5.1 Write a program to create a class named shape. In this class we have three sub classes circle, triangle and square each class has two member function named draw () and erase (). Create these using polymorphism concepts.

5.2 Write a program to give a simple example for abstract class.

5.3 Write a program suppose, it is required to build a project consisting of a number of classes, possibly using a large number of programmers. It is necessary to make sure that every class from which all other classes in the project will be inherited. Since any new classes in the project must inherit from the base class, programmers are not free to create a different interface. Therefore, it can be guaranteed that all the classes in the project will respond to the same debugging commands.

Lab – 6 (2 Hrs Real Time)

6.1 Write a program to create interface A in this interface we have two method meth1 and meth2. Implements this interface in another class named MyClass.

6.2 Write a program to give example for multiple inheritance in Java.

6.3 Write a program to create interface named test. In this interface the member function is square. Implement this interface in arithmetic class. Create one new class called ToTestInt in this class use the object of arithmetic class.

6.4 Create an outer class with a function display, again create another class inside the outer class named inner with a function called display and call the two functions in the main class.

Lab – 7 (2 Hrs Real Time)

7.1 Write a class called ColourChecking. Define a color with red = 193, green =255 and blue = 183. No separate the rgb values. Find the Hue, saturation and brightness of this color.

7.2 Write a program to check the font class method as follows: Create a font TimesRoman bold and Italic size 12. In this object use the font methods. Then display the attributes of the font.

7.3 Write a program to find the solution for the following problems using Recursion.

7.3.1 Find the maximum of an array. Let a[] be an array of integers. if n= 1, a[0] is the only number in the array and so, maximum = a[0]. if n > 1 , then do the following: find the maximum of n-1 entries of the array. Compare this maximum with the last entry a[n-1] and finalize.

7.3.2 Find the Fibonacci numbers are defined as $F_0=1$,$F_1=1$ and $F_i=F_{i-1}+F_{i-2}$ for i >2.

Lab – 8 (2 Hrs Real Time)

8.1 Create class point with following instance variable and methods.

Instance variable: private int x,y
Constructors : public Point(), Point(int x, int y)
Methods : public void setX(int x), setY(int y), setXY(int x, int y)

8.2 Create class Number with only one private instance variable as a double primitive type. To include the following methods (include respective constructors) isZero( ), isPositive(), isNegative( ), isOdd( ), isEven( ), isPrime(), isAmstrong() the above methods return boolean primitive type. getFactorial(), getSqrt(), getSqr(), sumDigits(), getReverse() the above methods return double primitive type. void listFactor(), void dispBinary().

8.3 Write a program to create a package named mypack and import it in circle class.

8.4 Write a program to create a package named pl, and implement this package in ex1 class.

Lab – 9 (2 Hrs Real Time)

9.1 Write a program to create automatic type conversions apply to overriding.
9.2 Create class box and box3d. box3d is extended class of box. The above two classes going to pull fill following requirement

- Include constructor.
- set value of length, breadth, height
- Find out area and volume.

Note: Base class and sub classes have respective methods and instance variables.

9.3 Write a program using vector class.

Lab – 10 (2 Hrs Real Time)

10.1 Write a program for example of try and catch block. In this check whether the given array size is negative or not.

10.2 Write a program for example of multiple catch statements occurring in a program.

10.3 Write a program to illustrate sub class exception precedence over base class.

10.4 Write a program to illustrate usage of try/catch with finally clause.

10.5 Write a program to describe usage of throws clause.

10.6 Write a program for creation of user defined exception.

Lab – 11 (2 Hrs Real Time)

11.1 Write a program to create a text file in the path c:\java\abc.txt and check whether that file is exists. Using the command exists(), isDirectory(), isFile(), getName() and getAbsolutePath().

11.2 Write a program to rename the given file, after renaming the file delete the renamed file. (Accept the file name using command line arguments.)

11.3 Write a program to create a directory and check whether the directory is created.

11.4 Write a program to open one application using process class.

11.5 Write a program using modifiers.

Lab – 12 (2 Hrs Real Time)

12.1 Write a program to create a file and write data into it using the methods OutputStream class.

12.2 Write a program to accept specified number of characters as input and converts them into uppercase characters.

12.3 Write a program to get the input from the user and store it into file. Using Reader and Writer file.

Lab – 13 (2 Hrs Real Time)

13.1 Write a program to illustrate creation of threads using runnable class.(start method start each of the newly created thread. Inside the run method there is sleep() for suspend the thread for 500 milliseconds).

13.2 Write a program to create a class MyThread in this class a constructor, call the base class constructor, using super and starts the thread. The run method of the class starts after this. It can be observed that both main thread and created child thread are executed concurrently.

Lab – 14 (2 Hrs Real Time)

14.1 Write a program to get the reference to the current thread by calling currentThread() method.

14.2 Write a program to create two threads. In this class we have one constructor used to start the thread and run it. Check whether these two threads are run are not.

Lab – 15 (2 Hrs Real Time)

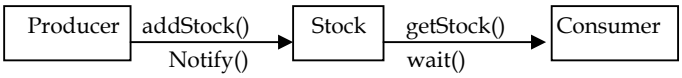
15.1 Create a multithreaded program by creating a subclass of Thread and then creating, initializing, and staring two Thread objects from your class. The threads will execute concurrently and display Java is hot, aromatic, and invigorating to the console window.

15.2 Create a multithreaded program as in the previous exercise by creating the MyThread subclass of Thread. But create threads as objects of the class MyClass, which is not a subclass of Thread. MyClass will implement the runnable interface and objects of MyClass will be executed as threads by passing them as arguments to the Thread constructor.

Lab – 16 (2 Hrs Real Time)

16.1 Write a program for inventory problem in this to illustrates the usage of synchronized keyword.

16.2 Write a program for interthread communication process. In this they have three classes consumer, producer and stock.



16.3 Write a program to show how synchronized methods and objects monitors are used to coordinate access to a common object by multiple threads. Clue use first program of this section for use will synchronized methods.

16.4 Write a complex program to illustrate how the thread priorities? Imagine that the first thread has just begun to run, even before it has a chance to do anything. Now comes the higher priority thread that wants to run as well. Now the higher priority thread has to do its work before the first thread starts.

Lab – 17 (2 Hrs Real Time)

17.1 Write a Applet program to display the "Hello World " in the browser.

17.2 Write a Applet program that automatically display the text with Font Style, Font type .

Lab – 18 (2 Hrs Real Time)

18.1 Write a Applet program that automatically display the text with Font Style, Font type Using getParameter Method.

18.2 Write a program that displays the menu bar and when You click the options it has to display a dialog box stating which option has been clicked.

18.3 Write a program that has menubar and also a quit option and if the user clicks the quite option the applet should quit.

Lab – 19 (2 Hrs Real Time)

19.1 Write a program to create a dialogbox and menu.

19.2 Write a program to create a grid layout control.

19.3 Write a program to create a border layout control.

19.4 Write a program to create a padding layout control.

Lab – 20 (2 Hrs Real Time)

20.1 Write a program to give the example for button control.

20.2 Write a program to give the example for panel control.

20.3 Write a program that will display check boxes and option buttons they are numbered from1 to. Use a textbox to display the number those corresponding boxes or button checked.

20.4 Write a program to create a simple calculator.

20.5 Write a program as above with combo box and list boxes instead.

20.6 Write a program that displays the x and y position of the cursor movement using Mouse.

20.7 Write a program to create a canvas.

20.8 Write a program that displays the x and y position of the cursor movement using Keyboard.

Lab – 21 (2 Hrs Real Time)

21.1 Write a program to create a text box control.

21.2 Write a program to create a analog clock.

Lab – 22 (2 Hrs Real Time)

22.1 Write a program to create a Applet life cycle.

22.2 Write a program for card Layout control.

Object Oriented Programming with Java

Lab Solutions

Lab - 1 (2 hrs real time)

Ex - 1.1

```
class Simout
{
      public static void main (String args[ ] )
       {
         System.out.println (“Welcome to Java programming”);
            }
        }
```

Ex - 1.2

```
class Default
{
    private short s;
    private int i;
    private long l;
    private float f;
    private double d;
    private char c;
    private String str;
    private boolean b;

public static void main (String args[ ])
{
     Default df = new Default( );
      System.out.println (“\n short s =” + df.s);
      System.out.println (“\n int  i =” + df.i);
      System.out.println (“\n long l =” + df.l );
      System.out.println (“\n float f =” + df.f);
      System.out.println (“\n double d =”  + df.d);
      System.out.println (“\n char c =” + df.c);
      System.out.println (“\n String s =” + df.str);
      System.out.println(“\n boolean b =” + df.b);
      }
}
```

Ex - 1.3

```
class Streq
{
  public static void main (String args [ ])
```

```
  {
 String str1 = "Good";
 String str2 = "Good";
 System.out.println ("\n str1 :"+str1);
 System.out.println ("\n str2 :"+str2);
 System.out.println ("\n str1 == str2  : " + str1 == str2);
System.out.println ("\n str1.equals(str2): " + str1.equals(str2));
    }
}
```

Lab – 2 (2 Hrs Real Time)

Ex - 2.1.1

Increment and Decrement Operators

```
class IncDec
 {
   public static void main (String args [ ] )
    {
        int x = 8, y = 13;
        System.out.println ("x =" + x);
        System.out.println ("y =" +y);
        System.out.println ("++x =" + ++x);
         System.out.println ("y++ =" + y++);
        System.out.println ("x =" + x);
        System.out.println ("y =" + y);
          }
    }
```

Ex - 2.1.2

Bitwise Complement Operator :

```
  class BitWiseComplement
      {
     public static void main (String args [ ] )
         {
              int  x = 8;
              System.out.println ("x  =" + x);
              int y  = ~x;
              System.out.println ("y =" + y);
            }
      }
```

Ex - 2.1.3

Arithmetic operators:

```
 class FloatMath
   {
   public static void main  ( String args  [ ] )
```

```
            {
                float  x = 23.5f,  y = 7.3f;
                System.out.println ("x  ="  + x);
                System.out.println ("y ="    + y);
                System.out.println ("x + y ="  + ( x + y) ) ;
                System.out.println ("x  - y ="   + (x - y)  ) ;
                 System.out.println (" x * y ="   +( x* y) );
                System.out.println (" x / y ="   + ( x / y )  );
                System.out.println (" x % y ="  + ( x  % y ) );
                  }
      }
```

Ex - 2.1.4

```
      Relational Operators:
          class Relational
            {
             public static void main (String args [  ] )
                  {
                  int x = 7, y = 11, z = 11;
                  System.out.println (" x=" + x);
                  System.out.println ("y =" + y);
                  System.out.println ("x < y =" + ( x < y ) );
                  System.out.println (" x > z ="   + (x > z)  );
                  System.out.println (" x <= z ="   + (y <= z)  );
                  System.out.println (" x >= y ="  + (x >= y ) );
                  System.out.println ( " y == z =" + (y ==z) );
                  System.out.println (" x != z ="  +   (x != z)  );
                 }
          }
```

Ex - 2.1.5

```
      Bitwise Operator :
        class Bitwise
           {
          public static void main  ( String  args [ ] )
              {
                  int  x = 5, y = 6;
                  System.out.println (" x =" +x);
                  System.out.println (" y =" + y );
                  System.out.println (" x & y ="   + ( x & y) ) ;
                  System.out.println (" x  |  y ="   + ( x  | y ) );
                  System.out.println (" x  ^ y ="    +( x ^ y)  );
              }
      }
```

Ex - 2.1.6

```
      Conditional Operators
```

```
   class Conditional
      {
       public static void main  (String args [ ] )
            {
              int  x  = 0;
              boolean isEven  = false;
              System.out.println ("x =" + x);
             x = isEven ? 4: 7;
             System.out.println ("x =" + x);
         }
}
```

Ex - 2.2.1

```
class IfTest
   {
    public static void main ( String args [ ] )
      {
         int x = 4;
         int y = 10;
         if (x > y )
            {
            System.out.println ("x is greater than y" );
              }
         else  {
            System.out.println ("X is lesser than y");
              }
          }
      }
```

Ex - 2.2.2

```
class SwitchTest
  {
   public static void main (String args [ ] )
      {
         char  ch = 'A';
         switch (ch)
             {
              case 'A':
                 System.out.println  ("Value is A");
                 break;
              case  'B':
                 System.out.println ("Value is B");
                 break;
               default:
                 System.out.println ("Unknown Value");
                }
          }
```

```
}
```

Ex – 2.2.3

```
class ForTest
 {
  public static void main (String args [ ] )
     {
       int i= 0;
       int sum = 0;
       for( i = 0; i <= 10; i++)
       sum += i;
       System.out.println ("The sum of first 10 Nos ="  + sum );
     }
  }
```

Ex – 2.2.4

```
class WhileTest
  {
  public static  void main (String args [ ] )
      {
          int i=1;
          while (i<=5)
            {
            System.out.println ("i ="  + i);
           i++;
            }
        }
  }
```

Ex – 2.2.5

```
class BreakLoop
  {
  public static void main (String args [ ])
      {
           int  i= 0;
           do  {
           System.out.println ("I'm  stuck !" ) ;
           i++;
           if (i > 5)
           break;
              } while (true);
        }
}
```

Ex – 2.3.1

```
class Length
 {
  public static void main (String args [ ] )
      {
         int a1[ ]  = new int [10];
         int a2 [ ] = { 3,5,7, 1, 8, 99 , 44, -10 };
          int a3 [ ] = { 4, 3, 2, 1 };

   System.out.println ("Length of a1 is"  + a1.length);
   System.out.println (" Length of a2 is" + a2.length);
   System.out.println ("Length of a3 is"  + a3. length);
          }
}
```

Ex – 2.3.2

```
// Demonstrate a one-dimensional array.

 class Array
  {
    public static void main (String args [ ] )
     {
        int num[];
        int size =5;
        num = new int [size];
       num [0]  = 10;
       num [1]  = 20;
       num[2]  =  30;
       num [3]   = 40;
       num [4]=  50;
       for  (int  i =0; i<size; i+ +)
       System.out.println ("num [" + i  +" ] ="  + num [ i ]);
      }
}
```

Ex – 2.3.3

```
// ACDemo - Using arraycopy ( )
class ArrayCopy
 {
    static char a [ ] = { 'H', 'E', 'L', 'L', 'O'} ;
     static char b [ ] = { 'W', 'O', 'R', 'L', 'D'};
  public static void main ( String args [ ])
   {
      System.out.print ("Before ArrayCopy  a - -  >" );
      System.out.println (a);
      System.out.print  ("Before ArrayCopy  b - - >" );
      System.out.println (b);
```

```
            System.arraycopy (a, 0, b, 0 ,a.length);
           System.out.print ("After ArrayCopy a - - >" );
           System.out.print (a);
           System.out.print ("After ArrayCopy  b - ->" );
           System.out.println (b);
           }
      }
```

Ex – 2.3.4

```
      // Demonstrate a two-dimensional array.

      class TwoDArray
        {
          public static void main (String args[])
           {
               int twoD[][] = new int[3][3];
               int i, j , k = 0;
               for (i=0; i<3; i++)
               for (j=0; j<3; j++)
                 {
                      twoD[i][j] = k;
                      k++;
                       }
               for (i=0; i< 3; i++)
                    {
                  for ( j= 0;  j < 3; j++)
               System.out.print (twoD[i][j] + " ");
               System.out.println();
              }
          }
      }
```

Ex – 2.4.1

```
      public class Summation
        {
       public static void main(String a[])
         {
            int sum = 0;
            int invalid = 0;
            for(int I=0; I<a.length;I++)
            {
            try
             {
               sum += Integer.parseInt(a[I]);
              }
             catch(NumberFormatException e)
                {
                  invalid++;
```

```
                }
            }
            System.out.println("Total no. of arguments :" + a.length);
            System.out.println("Invalid Integers:" + invalid);
            System.out.println("Sum :"+sum);
              }
      }
```

Ex – 2.4.2

```
      class cmdline
        {
          public static void main (String args [ ])
             {
               for ( int i=0 ;  i < args.length ; i++)
                 System.out.println (args [ i ]);
            }
      }
```

Ex – 2.5.1

```
      public class BinaryTriangle
      {
         public static void main (String arg [ ] )
          {
            String  k = "1", l = " ", s = "1";
            int m =  0;
            int  n =  5; //* if necessary change the value of n** //
            for (int  i  =  0; i  < n; i++)
              {
                for (int j = 1; j < m; j++)
                   {
                        l+= "0";
                      }
                System.out.println (k + l + s + "\n");
                l = "";
                m += 2;
                }
           }
         }
```

Ex – 2.5.2

```
      public class NumberReverseTriangle
      {
         public static void main (String arg[ ])
           {
             String k= "" ;
```

```
            int n = 5; /* if necessary change the value of n* */
            for (int i = 5; i  >= 0; i--)
             {
               k = i + " "  + k;
               System.out.println (k + "\n");
               }
            }
        }
```

Ex – 2.5.3

```
      public class NumberTriangle
       {
         public static void main (String  arg [ ])
          {
             String k = " ";
             int n = 6; /* if necessary change the value of n * */
             for ( int i = 1; i <= n; i++)
             {
               k += i + " ";
               System.out.println (k + "\n");
               }
           }
       }
```

Ex - 2.6.1

```
      class sumdig
      {
          public static void main (String args [ ] )
           {
             int i, s;
             i = 927489;
             s=0;
             System.out.println (i);
             while (i>10)
              {
               s += i%10;
               i/=10;
               }
              s += i;
            System.out.println ("the sum of the digits is :" + s);
          }
      }
```

Ex – 2.6.2

```
      import java.lang.*;
```

```
class prime
{
   public static void main (String args [ ])
    {
      int i, j, n, lastn;
     double a;
     boolean flag;
     for (i=0;i<1000;i++)
      {
        a = i;
        a = Math.sqrt (a);
        lastn = (int)a;
        flag =true;
        for (j=2;j<=lastn; j+ +)
        {
         if(i != j)
          {
           if(i % j = =0)
            {
           flag = false;
           break;
            }
          }
        }
     if (flag) System.out.println (“\n” + i );
    }
  }
 }
```

Ex – 2.7

```
class exarray
{
  public static void main (String args [ ] )
    {
   int [ ] arr = {234,6,846,85,96,198,545,12,60,34,4,87,7,1};
   int i, j, l, temp;
   l= arr.length;
   for (i=0;i<l-1;i++)
    {
    for (j=i+1;j<l;j++)
     {
      temp = arr [i];
      arr[i] = arr[j];
      arr[j] = temp;
      }
    }
for (i=0;i<l;i++)
 {
   System.out.println (arr[i]);
```

```
    }
   }
  }
```

Ex – 2.8

```
public class Quad
{
double a, b, c;
Quad(double a, double b, double c)
 {
  a = a;
  b = b;
  c = c;
}
void root()
{
  double r1, r2, d, rp, ri;
  d= a* a – 4 * a*c;
  if (d<0)
   {
    rp = -b/(2.0 * a);
   ri = Math.sqrt(Math.abs(d))/(2.0*a);
  System.out.println(“ The roots are complex conjugates”);
  System.out.println(“Roots1 = “+ rp + “i” + ri);
  System.out.println(“Roots2 = “+ rp + “i”+ ri);
  }
  if (d==0)
  {
   r1= -b/(2.0*a);
   System.out.println(“The roots are real and equal”);
   System.out.println(“Root = “ + r1);
  }
  if (d>0)
  {
   r1 = (-b +Math.sqrt(d))/(2.0*a);
   r2 = (-b-Math.sqrt(d))/(2.0*a);
   System.out.println(“Root1 = “ + r1 + “\n Root2 = “ + r2);
   }
  }
  public static void main(String a[])
  {
   Quad q1,q2,q3;
   q1 = new Quad(1.0, -5.0, 6.0);
   q2 = new Quad(4.0, -20.0, 25.0);
   q3 = new Quad(2.0, 1.0, 2.0);
   q1.root();
   q2.root();
   q3.root();
  }
```

```
}
```

Ex – 2.9.1

```
public class MatAdd
{
public static void main(String a1[])
{
int I, j, k;
int a[][]={ {4,7,9,8,3},{2,4,7,8,1},{1,1,8,1,2},{0,0,1,0,4}};
int b[][]={ {1,2,8,4,3},{4,1,8,3,1},{2,1,0,0,5},{1,2,1,1,7}};
int c[][];
c = new int[4][5];
for(I=0;I<4;I++)
 for(j=0;j<5;j++)
  c[I][j]=a[I][j]+b[I][j];
for(I=0;I<4;I++)
{
  System.out.println("\n");
  for(j=0;j<5;j++)
    System.out.print(c[I][j]);
  }
 }
}
```

Ex – 2.9.2

```
public class MatMult
{
public static void main(String a1[])
{
 int a[][]={ {0,1,4},{2,1,7} };
 int b[][]={ { 1,7,2,3},{4,1,-2,3},{4,1,7,2} };
 int c[][];
 c = new int[2][4];
 int I, j, k;
 for(I=0;I<2;I++)
  for(j=0;j<4;j++)
  {
   c[I][j]=0;
   for(k=0;k<3;k++)
   c[I][j]+=a[I][k]*b[k][j];
  }
for(I=0;I<2;I++)
{
 System.out.println("\n");
 for(j=0;j<4;j++)
 System.out.print("\b" + c[I][j]+" ");
}
```

```
}
}
```

Lab - 3 (2 hrs Real time)

Ex - 3.1

```
class Room
{
      int roomNo;
      String roomType;
      float roomArea;
      boolean acMachine;

void setData(int rno, String rt, float area, boolean ac)
  {
     roomNo = rno;
      roomType = rt;
      roomArea = area;
      acMachine = ac;
       }

 void displayData()
 {
     System.out.println("The room #. Is "   + roomNo);
     System.out.println ("The room Type is "  + roomType);
     System.out.println ("The room area is "   + roomArea);
     String s = (acMachine)  ? "yes " : "no ";
     System.out.println ("The A/c Machine needed "  + s);

      }
 public static void main(String arg[])
     {
           Room room1 = new Room ( );
           room1. setData (101, "Deluxe", 240.0f, true);
           room1.displayData ( );
     }
}
```

Ex – 3.2

```
class SimpleObject
  {
   SimpleObject()
     {
      System.out.println ("No argument Constructor");
       }
   SimpleObject(int a)
     {
```

```
            System.out.println ("One argument Constructor");
             }
      }

        public class Constructor
             {
            public static void main(String arg[])
               {
                 new SimpleObject();
                 new SimpleObject(100);
                  }
           }
```

Ex – 3.3

```
      // simple types are passed by value
       class Test
          {
           void meth(int i, int j)
            {
               i *=2;
               j /=2;
            }
      }

      class CallByValue
        {
       public static void main (String args[])
             {
           Test ob = new Test();
           int a =15, b = 20;
           System.out.println ("Before call  :a = "+ a +"b = "  + b);
           ob.meth (a, b );
           System.out.println ("After call :a = " + a+" b = " +  b );
           }
      }

      //  simple types are passed by value

      class Test
        {
         int a, b;
         Test(int i, int j)
           {
               a  = i;
               b  = j;
            }

      // pass an object
      void meth(Test o)
```

```
  {
     o.a *=2;
     o.b /= 2;
    }
}

class CallByRef
  {
   public static void (String args[])
     {
     Test ob = new Test(15,20);
 System.out.println("Before call :ob.a  = "+ob.a+" ob.b "+ob.b);
 ob.meth (ob);
 System.out.println("After call :ob.a = "+ob.a+" ob.b  "+ ob.b);
     }
}
```

Ex – 3.4

```
class Point  {
    int x;
    int y;

 point ( int   x, int  y)
{
        x = x;
        y = y;
}

void displayPoint  ( )  {
     System.out.println ("The x value is" + x);
     System.out.println ("The y value is"  + y);
 }
public static void main ( String arg [ ] )   {
       Point point   = new point (10,20);
        point. displayPoint  ( );

    }
}

//  using this keyword
class ThisReturn
 {
   private int i = 0;
ThisReturn increment()
 {
   i++;
   return this ;
  }

void print()
```

```
       {
       System.out.println ("The  i value is"  + i);
        }

      public static void main (String arg[])
       {
       ThisReturn  tr = new ThisReturn();
       tr.increment().increment().increment().increment().print();
           }
      }
```

Lab – 4 (2 Hrs Real Time)

Ex – 4.1

```
      // Demonstrate static variable , methods, and blocks.
      class UseStatic
        {
            static int a=3;
            static int b;
            static void meth(int x)
                  {
                  System.out.println ("x  =" +x);
                  System.out.println ("a ="  +a);
                  System.out.println ("b ="  + b);
                    }
            static
              {
                  System.out.println ("Static block Initialized");
                  b = a * 4;
                   }
            public static void main (String args [ ])
                  {
                   meth (42);
                     }
            }
      //  Using Method

      class StaticDemo
        {
           static int a = 42;
           static int b = 99;
           static void callme()
              {
                System.out.println ("a =" + a);
              }
        }
       class StaticByName
          {
           public static void main( String args[])
```

```
        {
          StaticDemo.callme();
          System.out.println  ("b =" + StaticDemo.b);
        }
}
```

Ex – 4.2

```
class House
 {
       Room r;
     void createHouse()
       {
         r = new Room();
       }

void displayHouse()
    {
     r.displayData();
     }

public static void main (String args[])
 {
     House h  = new House();
     h.createHouse();
     h.displayHouse();
     }
}
```

Ex – 4.3

```
// a simple example of inheritance
 class A
   {
        int i, j;
       void showij()
        {
      System.out.println ("  i and j: "  + i +" "+ j );
         }
       }

class B extends A
  {
     int k;
 void showk()
   {
       System.out.println  ( "K:  "  + k);
      }

void sum()
  {
```

```
   System.out.println  (“ i +  j  + k: “ + ( i  + j  + k ) );
   }
}

class Inherit
 {
   public static void main (String args[])
     {
        A superOb  = new A();
        B subOb  = new B();

 // the superclass may be used by itself
        superOb.i    = 10;
        superOb.j   = 20;
        System.out.println(“Contents of superOb:”);
        super Ob.showij()  ;
        System.out.println  ( );

//the subclass has access to all public members of its superclass
        subOb. i = 1;
        subOb. j = 2;
        subOb. k = 3;
        System.out.println  (“Contents of supOb:”  );
        subOb.showij ( ) ;
        subOb.showk ( );
        System.out.println ( );
  System.out.println (“Sum of i, j and k in subob:” );
        subOb.sum ( );
  }
}
```

Ex – 4.4

```
// method overriding

  class A
    {
      int i, j;
      A()
       {
        i = 0;
        j = 0;
        }
A(int a, int b)
  {
       i = a;
       j = b;
  }
void show()
  {
```

```
        System.out.println ("i and j : "  +  i +"   "+ j);
              }
      }
      class B extends A
        {
            int  k;
      B()
       {
          i = 0;
          j = 0;
          k = 0;
         }

      B(int a, int b, int c)
       {
          i = a;
          j = b;
          k = c;
        }
      void show()
       {
         System.out.println  ( "i  and j : " + i +"   "  +   j  );
         System.out.println  (" k:"  + k);
         }
      }
      class Override
        {
         public static void main (String args[])
           {
                B subob  = new B( 1,2,3);
                  subob.show() ;

                 }
      }
```

Ex – 4.5

```
      //  method  overriding

      class A
       {
          int i,  j;
          A()
           {
               i = 0;
               j =  0;
             }
      A(int a, int b)
          {
               i = a;
               j = b ;
```

```
        }

    void show()
       {
         System.out.println  ("i and  j : "+ i +"  " + j);
         }
    }

    class B extends A
       {
          int  k;
          B()
           {
              super();
              k = 0;
            }
    B(int a, int b, int c)
     {
         super(a, b);
         k = c;

       }

     void show()
      {
         super.show();
         System.out.println("k:" + k);
          }
    }

    class Override
      {
        public static void main ( String args[])
           {
                B subob  = new B( 1, 2, 3 );
                subob.show();
                }
    }
```

Lab – 5 (2 hrs Real Time)

Ex – 5.1

```
    class Shape
     {
      void draw()
        {
         System.out.println("Shape draw()");
          }
      void erase()
```

```
   {
     System.out.println (“ Shape erase()”);
      }
  }
class Circle extends Shape
 {
  void draw()
   {
     System.out.println (“Circle draw()”);
     }

void erase()
 {
    System.out.println (“Circle erase()”);
      }
}
class Triangle extends Shape
   {
     void draw()
      {
   System.out.println(“Triangle erase()”);
        }
}
class Square extends Shape
 {
   void draw()
    {
      System.out.println (“Square draw()”);
     }
void erase()
 {
    System.out.println (“Square erase()”);
     }
}
public class Shapes
   {
    public static Shape randshape()
      {
       switch((int)(Math.random()*3))
         {
             case  0:  return new Circle();
             case 1:  return new Square();
             case 2:  return new Triangle();
             default : System.out.println(“default”);
             return new Shape();
              }
}
 public static void main (String arg[])
  {
     Shape s[] = new Shape[9];
     for(int i = 0;i< s.length; i++)
```

```
            s[i] = randshape();
           for(int i= 0;i < s.length; i++)
            s[i].draw();
           }
      }
```

Ex – 5.2

```
      // A simple demonstration of abstract
       abstract class  A
         {
            abstract void callme();
            void callmetoo()
              {
               System.out.println (“This is a concrete method .”);
                 }
           }
      class B extends  A
       {
        void callme()
         {
           System.out.println (“B’ s implementation of callme.”);
            }
         }
      class AbstractDemo
       {
       public static void main (String args[])
          {
             B b  = new B();
             b.callme();
             b.callmetoo();
            }
      }
```

Ex – 5.3

```
      abstract class debuggable{
      abstract void dump()
      {
      System.out.println("debuggable error: no dump() defined for the
      class");
      }
      }
      class X extends debuggable{
      private int a,b,c;
      public:
       X( int aa =0,int bb=0,int cc=0)
      {
      a = aa;
      b = bb;
      c = cc;
```

```
}
void dump()
{
Systen.out.println( "a = " + a +"b=" +b+ "c=" +c);
}
}
class Y extents debuggable{
private int i,j,k;
public:
Y( int ii =0,int jj=0,int kk=0)
{
i = ii;
j = jj;
k = kk;
}
void dump()
{
Systen.out.println( "i = " + i +"j=" +j+ "k=" +k);
}
}
class Z extents debuggable{
private int p,q,r;
public:
Y( int pp =0,int qq=0,int rr=0)
{
p = pp;
q = qq;
r = rr;
}
void dump()
{
Systen.out.println( "p = " + p +"q=" +q+ "r=" +r);
}
}

class abstdemo
{
public static void main(String arg[])
{
X x(1,2,3);
Y y(2,4,5);
Z z;
x = new X;
y = new Y;
z = new Z;
x.dump();
y.dump();
z.dump();
}
```

Lab – 6 (2 hrs Real Time)

Ex - 6.1

```
// One interface an extend another.

interface  A
   {
      void meth1();
      void meth2();
       }

//  B now includes meth1() and meth2()--it adds meth3().
interface B extends A
  {
     void meth3();
       }

 // This class must implement all of A and B

 class MyClass implements B
   {
      public void meth1 ( )
         {
            System.out.println("Implement meth1().");
                }

public void meth2()
   {
       System.out.println ("Implement meth2().");
         }

public void meth3()
   {
       System.out.println ("Implement  meth().” );
         }
      }

class IFExtend
  {
    public static void main(String arg[])
      {
        MyClass ob = new MyClass();
        ob.meth1();
        ob.meth2();
        ob.meth3();
         }
    }
```

Ex – 6.2

```
class Number
   {
     protected int x;
     protected int y;
     }
interface Arithmetic
  {
        int add(int a, int b);
        int sub(int a, int b);
    }
class UseInterface extends Number implements Arithmetic
{
      public int add(int a, int b)
        {
          return(a + b);
           }
public int sub(int a, int b)
   {
     return (a - b);
     }
public static void main(String args[])
    {
    UseInterface ui  =  new UseInterface();
    System.out.println("Addition --- >" + ui.add(2,3));
    System.out.println("Subtraction ----- >" + ui.sub(2,1));
      }
}
```

Ex – 6.3

```
   //   Interface

public interface Test
{
     public int square(int a);
   }

// Implements
class arithmetic implements Test
   {
      int s = 0;
      public int square(int b)
       {
System.out.println("Inside arithmetic class – implemented method  square");
         System.out.println("Square of " + " is "+s);
         return s;
         }
      void armeth()
         {
```

```
            System.out.println("Inside method of class Arithmetic");
            }
      }

      //  use the object
      class ToTestInt
       {
        public static void main(String a[])
          {
         System.out.println("calling from ToTestInt class main
                                                method");
         Test t = new arithmetic();
         System.out.println("==============================");
       System.out.println("created object of test interface – reference
      Arithmetic class ");
         System.out.println("Hence Arithmetic class method square
      called");
         System.out.println("This object cannot call armeth method of
      Arithmetic class");
         System.out.println("==================================");
         t.square(10);
         System.out.println("==================================");
         }
      }
```

Ex – 6.4

```
      class Outer{

      String so = ("This is Outer Class");
      void display()
      {
      System.out.println(so);
      }
      void test(){
      Inner inner = new Inner();
      inner.display();
      }
      //this is an inner class
      class Inner{
      String si =("This is inner Class");
      void display(){
      System.out.println(si);
      }
      }
      }
      class InnerClassDemo{
      public static void main(String args[]){
      Outer outer = new Outer();
      outer.display();
      outer.test();
      }
```

```
}
```

Lab – 7 (2 hrs Real time)

Ex – 7.1

```
COLOUR CHECKING
import java.awt. *;
public class ColourChecking
{
  public static void main(String arg [ ])
    {
       Color rgb, hsb;
       rgb  = new Color (193,255,183);
       int red, green, blue;
      red  = rgb.getRed ( );
      green  =  rgb.getGreen ( );
      blue    =  rgb.getBlue ( );
      float x [ ] = {0.0f,0.0f,0.0f};
      rgb.RGBtoHSB(red, green, blue, x);
System.out.println ("RGB Combination");
System.out.println ("Red : "+red+"Green: "+green+"Blue : "+blue);
System.out.println ("Hue : "+x[0]+ "Saturation: "+x[1] +
                        "Brightness: "+x[2]);
int r = rgb.HSBtoRGB (0.75f,0.2375f,0.95f);
  System.out.println ("\n HSB Combination");
  System.out.println(" Red, Blue and Green Combination  : "+ r);
        }
}
```

Ex - 7.2

```
FONT CHECKING
import java.awt.*;
public class FontChecking
{
  public static void main (String arg [ ])
   {
     Font  f;
     f  = new Font ("TimesNewRoman", Font.BOLD+Font.ITALIC,12);
      String font  = f. getName( );
      int style    = f.getStyle ( );
      int size     = f.getSize ( );
      boolean bold = f.isBold ( );
      boolean normal = f.isPlain ( );
      boolean italic = f.isItalic ( );
System.out.println ("Font : "+ font + "\n Style  :  "+ style + "\n
Size: "+ size + "\n Bold  : "bold + "\n Italic : "+italic);
                }
}
```

Ex – 7.3.1

```
ORIAL RECURSION
public class Factorial
{
   int fact (int n)
     {
        if (n <= 1) return 1;
          else return ( n * fact(n – 1)) ;
         }
        public static void main  (String arg [ ])
          {
            int fa, fb, fc;
            int a = 4, b = 5, c = 6;
            Factorial f;
            f = new Factorial ( );
            fa = f.fact (a);
            fb =f.fact (b);
            fc = f.fact (c);
       System.out.println("Factorial of "+ a + "is" + fa);
       System.out.println ("Factorial of "+  b + " is" + fb);
       System.out.println("Factorial of "+ c + "is" + fc);
              }
}
```

Ex – 7.3.2

```
FIBONACCI NUMBERS : RECURSION
public class Fibonacci
{
   long fibo (int  n)
       {
         if (n  <= 1) return 1;
          else
           return (fibo(n – 1)  + fibo (n –2) );
 }
public static void main (String arg [ ])
{
     Fibonacci f;
     long l;
          f = new Fibonacci ( );
          l = f.fibo (5);
          System.out.println ("5th Fibonacci number is : "+1);

        }
}
```

Lab – 8 (2 hrs Real Time)

Ex – 8.1

```
class ThisRetrun
 {
   private int i = 0;

ThisReturn increment()
 {
   i+ +;
   return this ;
  }

void print()
 {
 System.out.println ("The  i value is"  + i);
  }

public static void main (String arg[])
 {
 ThisReturn  tr = new ThisRetrun();
 tr.increment().increment().increment().increment().print();
     }
}
import java.lang.*;
class point
  {
          private int x;
          private int y;

           public point ( ) { }

public point (int x, int y)
{
         this.x = x;
         this.y = y;
}
public void setX(int x)
{
            this.x = x;
}
public void setY (int y)
{
       this.y = y;
}
public void setXY (int x, int y)
{
     this.x = x;
     this.y = y;
}
public int getX( )
{
```

```
        return x;
    }
    public int gety ( )
    {
        return y;
    }
    public int addXY ( )
    {
        return (x + y);
    }
    public void display( )
    {
        System.out.println (x);
        System.out.println (y);
    }
    public double distance (point  p2)
    {
     return(Math.sqrt( (x-p2.x)* (x-p2.x) + (y-p2.y)  * (y – p2.y)) );
    }

    public static void main (String args[ ])
     {
        double a, b, c;
        point op,op1,op2;
        op = new point(5, 6);
        op1 =  new point();
        op2 = new point();
        op1.setX(3);
        op1.setY(6);
        op1.getX();
        op1.gety();
        op2.setXY(8,9);
        op2.addXY();
        a = op1.distance(op2);
        b = op2.distance(op);
        c = op.distance(op1);
        System.out.println("Distance between point op1 to op2 is"+a);
        System.out.println("Distance between point op2 to op is" +b);
        System.out.println("Distance between point op to op1 is" +c);
        System.out.println(" Points Of op, op1, op2");
        op.display();
        op1.display();
        op2.display();
        }
    }
```

Ex – 8.2

```
    import java.lang.Math. *;
    import java.lang.Number.*;
    class Num4
```

```
{
      private double db1;
      private long lg;
      public Num4 ( )
       {
           db1 = 108.0d;
           lg = 249;
         }
public Num4(double d, long l)
  {
     db1 = d;
     lg = 1;
    }
 public boolean isZero ( )
   {
    if (db1 == 0.0)
     return true;
    else
     return false;
    }
public boolean isPositive  ( )
  {
    if(db1 > 0.0)
      return true;
    else
      return false;
   }

public boolean isNegative ( )
  {
    if (db1  < 0.0)
     return true ;
    else
     return false;
    }

public boolean isodd( )
  {
    if (db1 % 2 != 0.0)
      return true;
    else return false;
    }
public boolean isEven ( )
  {
    if (db1 % 2 == 0.0)
      return true ;
    else return false;
   }
public boolean isPrime ( )
 {
     int i, lastn;
```

```
        double a;
        boolean flag;
        a = Math.sqrt(lg);
        lastn = (int)a;
        flag  = true;
        for (i=2; i<lastn; i++)
         {
           if (lg != i)
            {
              if( lg % i ==0)
                {
                  flag = false;
                  break;
                  }
              }
            }
        if (flag)
         return true;
        else return false;
    }
   public boolean isAmstrong ( )
     {
       if (db1 == 0.0)
        return true;
       else return false;
      }

   public double getFactorial ( )
    {
      double d=1;
      for(int i = 1;i <lg; i++)
      d *=i;
      return d;
     }
   public double getSqrt ( )
   {
      double d;
      d = (double) lg;
      d= Math.sqrt(d);
      return d;
    }

   public double getSqr ( )
   {
     double d;
     d = (double) lg;
     d = d * d;
     return d;
    }
   public double sumDigits( )
   {
```

```
    double d=0;
    while( lg>9)
    {
      d  += lg % 10;
      lg  = lg/10;
     }
    d +=lg;
    return d;
 }

public double getReverse ( )
{
    double d =0;
    double temp;
    while (lg>9)
    {
      temp = lg%10;
      d = d * 10 + temp;
      lg = lg/10;
      System.out.println ("\n"+ temp + "\t"  + d +" \t "+lg);
    }
  d  = d * 10 +lg;
  System.out.println ("Inside class" + d);
  return d;
}
 public void dispBinary ( )
  {
System.out.println("ByteValue of lg :" + Long.toBinaryString(lg));
    }
public static void main (String args [ ])
  {
    Num4 mynum  = new Num4( );
    double d = 199;
    System.out.println(" The given numbers are 108.0d and 249");
    System.out.println ("isZero "  + mynum.isZero()  );
    System.out.println  ("isPositive " + mynum. isPositive());
    System.out.println ("isNegative " + mynum.isNegative() );
    System.out.println ("isOdd "  + mynum.isodd());
    System.out.println ("isEven "  + mynum.isEven());
    System.out.println (" isPrime " +mynum.isPrime());
    System.out.println ("getFactorial " + mynum.getFactorial());
    System.out.println  ("getSqrt " + mynum. getSqrt( ));
    System.out.println ("getSqr " + mynum.getSqr( )  );
    System.out.println ("sumDigits "  + mynum.sumDigits( ));
    System.out.println ("getReverse "  + mynum.getReverse( ));
    mynum.dispBinary();
    System.out.println (" isPrime " +mynum.isPrime());
  }
}
```

Ex – 8.3

```
//create a package
package mypack;

class Circle
{
   double r;
   void area()
   {
     System.out.println("Area of the circle = " + (3.14 * r * r));
   }
}
class Square
{
   double s;
   void area()
   {
       System.out.println("Area of the Square = " + (s * s));
   }
}
class Rectangle
{
   double l,b;
   void area()
   {
       System.out.println("Area of the circle = " + (l * b));
   }
}

//implements the package

import mypack.Circle;
class Eg
{
   public static void main(String a[])
   {
      Circle c = new Circle();
      c.area();
   }
}
```

Ex – 8.4

```
package p1;
public class ex
{
      public ex()
      {
            System.out.println("Created");
      }
```

```
            public void display()
            {
                  System.out.println("Hello");
            }
      }

      import p1.*;
      class ex1
      {
            public static void main(String[] a)
            {
                  ex e=new ex();
                  e.display();
            }
      }
```

Lab - 9 (2 Hrs Real Time)

Ex – 9.1

```
         // Automatic type conversions apply to  overloading
        class OverloadDemo
            {
             void test()
               {
               System.out.println ("No parameters" );
                 }
             void test (int a, int b)
                {
          System.out.println ("a and b:   "  + a  +"  "  +  b);
                  }
             void test (double a)
                {
          System.out.println ("Inside test (double) a  :  " + a);
                }
           }
       class Overload
        {
         public static void main (String args[])
          {
            OverloadDemo  ob = new OverloadDemo();
            ob.test();
            ob.test(5);
            ob.test(10,20);
            ob.test(23,56);
           }
      }
```

Ex – 9.2

```
class Box
{
   private int length;
   private int breadth;
   public Box ( )
    {
      length =0;
      breadth =0;
     }
public Box (int x, int y)
{
 length = x;
 breadth =y;
  }
public void setval (int x, int y)
{
  length = x;
  breadth = y;

public int area ( )
{
 return (length * breadth);
      }
}
class Box3d extends Box
{
   private int height;
   public Box3d ( )
   {
    super ( );
    height  = 0;
     }
public Box3d(int x, int y, int z)
{
     super (x, y);
     height = z;
}
public void setval (int x, int y, int z)
{
      super.setval(x, y);
      height = z;
}
public int volume ( )
{
     return (super.area ( ) * height );
}
public static void main(String arga [ ])
{
   Box  b1 = new Box ( );
   Box3d b2 = new Box3d(12,34,18);
   b1.setval (25,30);
```

```
         System.out.println (“The area of b1 is : “ + b1.area( ) );
         System.out.println (“The volume of b2 is : “+ b2.volume ( ));
        }
      }
```

Ex – 9.3

```
      import java.util.*;
      class Vectdemo
      {
        public static void main(String ar[])
         {
          String s = "Delhi";
          String ss = "Chennai";
          Vector v = new Vector();
          v.addElement(s);
          v.addElement(ss);
          System.out.println("Size is " + v.size());
         }
      }
```

Lab – 10 (2 hrs Real time)

Ex – 10.1

```
      class NegTest
      {
      public static void main(String a[])
          {
            try
              {
                 int a1[] = new int[-2];
                System.out.println(“first element : “+a1[0]);
               }
           catch(NegativeArraySizeException n)
             {
               System.out.println(“ generated exception : “ + n);
                  }
           System.out.println(“ After the try block”);
         }
      }
```

Ex – 10.2

```
      import java.io.*;
      class MultiCatchException
       {
         public static void main( String a[])
           {
              int a1[] = { 100,200,300,400,500 };
              System.out.println(“enter a number as array index and
```

```
                                        find out its value");
            System.out.println("enter and to come out of the programs");
              try
                  {
                  String line;
                  int x;
                  BufferedReader d = new BufferedReader( new
                    InputStreamReader(System.in));
            while (( line = d.readLine()) != null)
            {
              if (line.equals("end"))
                  break;
              else  {
              try {
                x = Integer.parseInt(line);
                System.out.println("valid element and
                                    it is : "+a1[x]);
                  }
                catch(ArrayIndexOutOfBoundsException e)
                  {
                        System.out.println("invalid elements ");
                  System.out.println("exception generated : "+e);
                        }
                  catch(NumberFormatException n)
                  {
                  System.out.println("sorry no characters");
                  System.out.println("generated exception : " + n);
                        }
                  }
             }
            }catch(IOException i){ }
          }
        }
```

Ex – 10.3

```
      // Accept  file name as  command line arguments.
      import java.io.*;
      class BaseSubException
        {
          public static void main(String a[])
           {
             if(a.length == 0)
               {
                  System.out.println("invalid usage");
             System.out.println("usage : Java <filename> file1
             file2 file3 ...");
                  return;
                  }
             for(int I=0;I<a.length; I++)
               {
                  File f = new File(a[I]);
```

```
                try {
                   String line;
                   DataInput d = new DataInputStream(new
                            FileInputStream(a[I]));
                   if (f.exists() && f.isFile())
                    {
                      System.out.println("file exists");
                      System.out.println(f + "is ordinary file");
                System.out.println("printing the contents of file
                                          named : " + a[I]);
                      System.out.println("========================");
                      while((line = d.readLine()) != null)
                       {
                            System.out.println(line);
                            }
                      }
           } catch(FileNotFoundException e)
            {
             if(f.exists() && f.isDirectory())
              {
               System.out.println("======================");
               System.out.println("Name : " + f + "is a directory");

      System.out.println("inside catch of FileNotFoundException");
      System.out.println("========================");
      }
      else {
         System.out.println("====================");
         System.out.println("Name : " + a[I] + "does not exists");
         System.out.println("generated exception :"+e);
         System.out.println("======================");
         }
         } catch(IOException p) {
      System.out.println("super class is higher up in the program");
            }
           }
          }
         }
```

Ex – 10.4

```
      import java.io.*;
      class FinallyException
      {
         public static void main(String a[]) throws Exception
            {
              try
               {
                  BufferedInputStream f1 = null;
                  int size = f1. available();
                  for(int I = 0; I< size; I++)
                   {
```

```
                    System.out.print((char)f1.read());
                    }
                  }
                   catch(NullPointerException n)
                    {
              System.out.println("exception generated : " +n);
                    }
              finally
                {
                 System.out.println("==================");
                 System.out.println("Inside finally");
                 System.out.println("hay I always have the final");
                 System.out.println("====================");
                 }
              }
        }
```

Ex – 10.5

```
class ThrowsException
{
 public static void main(String a[]) throws ArithmeticException
      {
            System.out.println("Inside main");
            int I = 0;
           int j = 40/I;
            System.out.println("this statement is not printed");
            }
      }
/*  Output of this program is
 *  inside main
 *  java.lang.ArithmeticException: / by Zero
 *  at  ThrowsException.main(ThrowsEception.java:5) */
```

Ex – 10.6

```
import java.io.*;
class MyException extends Exception
  {
      private int a;
      MyException(int b)
        {
            a = b;
            }
      public String toString()
         {
            return "MyException [ " + a + "]";
          }
       }
class UserdefException
  {
```

```
            public int x;
            final int k = 5;
            void getInt()
             {
             try {
               BufferedReader d = new BufferedReader(new
                   InputStreamReader(System.in);

             System.out.println("do some guess work to generate your
                  own exception ");

            System.out.println("enter a number between 1 and 10");

            System.out.println("between these numbers lies the number
            to generate your own exception ");

             String line;
             while((line = d.readLine()) != null)
                  {
                    x = Integer.parseInt(line);
                     if ( x == 5)
                        {
                         System.out.println(" congrats ! you have
                                          generated an exception !");
                        throw new MyException(x);
                        }
                        else
                    System.out.println("Wrong guess ! try again");
                          continue;
                        }
                  } catch(MyException m) {
                  System.out.println("Generated exception: " +m);
                  } catch(NumberFormatException n)
                        {
                          System.out.println("sorry ! no characters");
                          System.out.println("Exiting ...");
                    System.out.println("Generated exception :"+n);
                        }
                  catch(IOException e) { }
                  }
            public static void main(String a[]) {
                  UserdefException u = new UserdefException();
                  u.getInt();
            }
      }
```

Lab - 11 (2 Hrs Real Time)

Ex – 11.1

```
      import java. io.*;
```

```
class FileMethods1
  {
  public static void main(String args[ ] )
    {
      File f1 = new File( "c:\\java", "abc.txt") ;
      System.out.println("File name :" + f1.getName());
      System.out.println("path :" + f1.getPath());
      System.out.println("Absolute path :"+f1.getAbsolutePath());
      System.out.println(f1.exists() ? "file exists"
                         : "file does not exists");
      System.out.println(f1.isDirectory() ? "file is a directory"
                         :    "file is not" + " a directory");
      System.out.println(f1.isFile()? "file is an ordinary file"
                        :  "file may be a named pipe") ;
      }
 }
```

Ex – 11.2

```
import java.io.*;
class FileMethods2
  {
   public static void main(String args[])
    {
      for (int i = 0; i<args.length; i++)
       {
       File f = new File("c:\\java", args[i]);
      File f1 = new File("c:\\java\\renfile");
      if (f.exists())
      {
       System.out.println(f + " does exists.");
       System.out.println("Its size is" + f.length()+ "bytes");
       f.renameTo(f1);
        System.out.println("Renamed file name :" + f1 + (i+1));
       System.out.println("deleting the renamed file"+f1+(i+1));
       System.out.println ("======================");
       f.delete();
        }
      else
        System.out.println(f +" does not exists");
            }
       }
 }
```

Ex – 11.3

```
import java.io.*;
class FileMethods4
  {
 public static void main (String args[]) throws IOException
     {
```

```
                File f = new File (“c:/java/temp”);
                if (f.mkdir())
                    System.out.println(“created a directory”);
                else
                   System.out.println (“unable to create a directory”);
             }
         }
```

Ex – 11.4

```
       import java.io.*;
       class Lang
       {
        public static void main(String a[])throws IOException
        {
         Runtime r;
         r = Runtime.getRuntime();
        System.out.println(r.freeMemory());
         int x[] = {1};
         r.gc();
          System.out.println(r.freeMemory());
       Process p = r.exec("pbrush.exe");
       System.out.println(r.freeMemory());
        }
       }
```

Ex – 11.5

```
      import java.lang.reflect.*;
      import java.util.*;
      class Modi
      {
         public static void main(String a[])throws ClassNotFoundException
         {
             Class c = Class.forName("Lang");
             Method m[] = c.getDeclaredMethods();
             for(int i =0; i < m.length; i++)
             {
               int mo = m[i].getModifiers();
               if (Modifier.isPublic(mo))
                 System.out.println(m[i].getName());
             }
         }
      }
```

Lab - 12(2 hrs Real time)

Ex – 12.1

```
      import java.io.*;
      class ReadWriteFile
      {
       public static byte getInput()[] throws Exception
```

```
        {
          byte inp[ ]  = new byte[50];
          System.out.println("enter text.");
          System.out.println("only 50 bytes i.e. about 2 lines …");
          System.out.println("press enter after each line to get
                                input into the program");
          for(int i=0; i<50; i++)
            {
               inp[i] = (byte)System.in.read();
                  }

          return inp;
        }
      public static void main(String args[])throws Exception
        {
           byte input[] = getInput();
            OutputStream f = new FileOutputStream("c:/java/write.txt");
            for(int i=0; i<50; i++)
             {
                f.write(input [i]);
                  }
            f.close();
            int size;
            InputStream f1 = new FileInputStream ("c:/java/write.txt");
            size = f1.available();
            System.out.println("reading contents of file write.txt…");
            for(int i=0; i<size; i++)
            {
              System.out.print((char)f1.read());
                  }
            f.close();
            }
       }
```

Ex – 12.2

```
      import java.io.*;
      class ByteArray
       {
       public static void main(String args[]) throws Exception
        {
            ByteArrayOutputStream f = new ByteArrayOutputStream(12);
           System.out.println ("enter 10 characters and a return");
           System.out.println ("These will be converted to uppercase and
           displayed");
               while (f.size() != 10)
                {
                 f.write(System.in.read( ) );
                  }
               System.out.println("Accepted characters into an array");
               byte b[] = f.toByteArray();
```

```
            System.out.println("displaying characters in the array");
            for(int l=0;l<b.length;l++)
            {
             System.out.println((char)b[l]);
            }
            ByteArrayInputStream inp = new ByteArrayInputStream(b);
            int c;
            System.out.println("converted to upper case characters");
            for (int i=0;i<1;i++)
          {
            while((c=inp.read()) != -1)
               {
           System.out.print(Character.toUpperCase((char)c));
               }
         System.out.println();
         inp.reset();
         }
       }
      }
```

Ex – 12.3

```
      import java.io.*;
      public class ReaderWriter
        {
      public static void main(String args[])
        {
       try{
     BufferedReader in = new BufferedReader(new FileReader(args[0]));
      String s, s1 = new String();
      while ((s = in.readLine())!= null)
         s1 += s + "\n";
         in.close();
      BufferedReader stdin =new BufferedReader(new InputStreamReader
                                              (System.in));
      System.out.println("enter line");
      System.out.println ("usage of BufferedReader and
                                        InputStreamReader…");
      System.out.println (stdin.roadLine());
      StringReader in2= new StringReader(s1);
      int c;
      System.out.println ("printing individual characters of the file"
                                      + args[0]);
      while (( c = in2.read()) != -1)
           System.out.println((char)c);
           BufferedReader in4 = new BufferedReader(new
                                           StringReader (s1));
           PrintWriter p = new PrintWriter (new BufferedWriter(new
                                   FileWriter("demo.out")));
      while((s = in4.readLine()) != null)
      p.println("output   " + s );
      p.close();
```

```
      }catch (IOException e ) { }

           }
      }
```

Lab - 13 (2 hrs Real Time)

Ex – 13.1

```
          class IntThread implements Runnable
        {
           Thread  t;
           IntThread()
             {
                t = new Thread ( this, “Test thread”);
                System.out.println (“ Child thread :” + t);
                t.start();
                   }
            public void run()
             {
              try {
                 for (int i= 5; i<0; i--)
                    {
                    System.out.println (“Child thread :” + i);
                    Thread.sleep (500);
                      }
                     }catch (InterruptedException e) { }
                     System.out.println (“Exiting child thread …” );
                      }
           public static void main (String args[])
           {
              IntThread  i = new IntThread();
                 try {
                   for ( int j=5; j >0; j--)
                     {
                     System.out.println  (“Main thread  :” + j);
                     Thread.sleep (1000);
                         }
                       } catch (InterruptedException  e)  { }
                   System.out.println ( “Main thread exiting …”);
                          }
             }
```

Ex – 13.2

```
         class MyThread extends Thread
          {
              MyThread()
                {
                super (“Using Thread class”);
                System.out.println (“Child thread:” + this);
```

```
            start();
              }
           public void run()
              {
               try
                {
                for ( int  i =5; i > 0; i--)
                   {
                   System.out.println (“Child thread” + i);
                   Thread.sleep (500);
                    }
                } catch (InterruptedException e)  { }
              System.out.println (“exiting child thread …”);
             }
       }

     class TestMyThread
       {
         public static void main(String args[])
          {
              new MyThread();
                try {
                  for ( int k = 5; k < 0; k--)
                    {
              System.out.println  (“Running main thread :”  + k);
              Thread.sleep(1000);
                }

              }catch (InterruptedException  e)  { }
             System.out.println (“Exiting main thread  . . .”);
              }
         }
```

Lab - 14(2 hrs Real Time)

Ex – 14.1

```
     class ThreadMethod
       {
          public static void main(String args[])
              {
                  Thread t = Thread.currentThread();
                  System.out.println (“current thread:” + t);
    System.out.println(“Name of the current thread:” + t.getName());
        System.out.println (“Priority :”  + t.getPriority());
                  t.setName(“mythread”);
                  System.out.println (“after name change :” + t);
                  t.setPriority (2);
                  System.out.println (“after priority change :” + t);
```

```
    System.out.println (“number of active thread :” + t.activeCount());
                  }
   }
```

Ex – 14.2

```
        class CreateThread extends Thread
          {
            String tname;
            Thread t;

            CreateThread(String s)
              {
                 tname  = s;
                 t = new  Thread(this, s);
                 System.out.println (“New thread :”  +  t);
                 t.start();
                  }
             public void run()
                {
                  try
                   {
                   for(int i = 5; i > 0; i--)
                     {
                    System.out.println (tname + “:” + i );
                    Thread.sleep (500) ;
                    }
                  } catch (InterruptedException e) { }
                    System.out.println(tname + “exiting….”);
        }
    }
   class ThreadMethod2
     {
        public static void main(String args[])
         {
           CreateThread  m1 =new CreateThread(“one”);
           CreateThread  m2 = new CreateThread (“two”);
        System.out.println(“Thread m1 is alive :” + m1.t.isAlive());
       System.out.println (“Thread m2 is alive:” + m1.t.isAlive());
           try  {
            System.out.println (“Waiting for thread to finish ….”);
            m1.t.join();
            m2.t.join();
               } catch (InterruptedException e) { }
      System.out.println(“ Thread m1 is alive :” + m1.t.isAlive());
      System.out.println(“ Thread m2 is alive :” + m2.t.isAlive());
             System.out.println (“ Main thread exiting …”);
                  }
      }
```

Lab - 15 (2 hrs Real Time)

Ex – 15.1

```
import java.lang.Thread;
import java.lang.System;
import java.lang.Math;
import java.lang.*;


class ThreadTest1
{
   public static void main (String args [ ])
    {
        MyThread thread1 = new MyThread (“thread1:”);
        MyThread thread2 = new MyThread (“thread2:”);
        thread1.start ( );
        thread2.start ( );
        boolean thread1IsAlive  = true;
        boolean thread2IsAlive = true ;
        do
            {
            if (thread1IsAlive && !thread1.isAlive ( ) )
              {
                thread1IsAlive = false;
                System.out.println (“Thread1 is dead.”);
                 }
          if (thread2IsAlive && ! thread2.isAlive ( ))
            {
               thread2IsAlive = false;
              System.out.println (“Thread  2 is dead.”);
               }
                }while (thread1IsAlive || thread2IsAlive);
         }
}

class MyThread extends Thread
{

static String message [ ] = {“Java”, “is”, “hot”, “aromatic”,
                                     “and”, “invigorating.”};
       public MyThread (String id)
       {
                super (id);
        }
        public void run ( )
        {
            String name = getName ( );
            for (int i=0;i<message.length; ++i )
              {
```

```
                    randomWait ( );
                    System.out.println (name + message[i] );
                   }
              }
       void randomWait ( )
        {
          try
           {
              sleep ((long)(3000*Math.random ( ) ) );
                  }
          catch (InterruptedException x )
           {
              System.out.println (“Interrupted!”);
             }
          }
      }
```

Ex – 15.2

```
      import java.lang.Thread;
      import java.lang.System;
      import java.lang.Math;
      import java.lang.*;
      import java.lang.Runnable;

      class ThreadTest2
      {
        public static void main (String args[ ])
          {
            Thread thread1 = new Thread (new MyClass (“thread1:”) );
            Thread thread2 = new Thread (new MyClass (“thread2:”) );
            thread1.start ( );
            thread2.start ( );
            boolean thread1IsAlive = true;
            boolean thread2IsAlive = true;
            do {
             if (thread1IsAlive && !thread1.isAlive ( ) )
              {
                   thread1IsAlive = false;
                   System.out.println (“Thread 1 is dead.”);
                   }
             if (thread2IsAlive && !thread2.isAlive ( ))
              {
                   thread2IsAlive = false;
                   System.out.println (“Thread 2 is dead.”);
                }
               } while (thread1IsAlive || thread2IsAlive );
          }
      }
      class MyClass implements Runnable
      {
       static String message [ ] = {“Java”, “is”, “hot”, “aromatic”,
```

```
                                        "and", "invigorating."};
      String name;
      public MyClass (String id)
       {
           name = id;
             }
      public void run ( )
        {
           for (int i =0;i<message.length;++i)
             {
                randomWait ( );
                System.out.println (name + message[i]) ;
             }
          }

    void randomWait ( )
    {
     try
      {
       Thread.currentThread ().sleep((long)(3000*Math.random()));
           }
      catch (InterruptedException  x)
       {
           System.out.println ("Interrupted !");
             }
       }
    }
```

Lab - 16 (2 hrs Real Time)

Ex – 16.1

```
   class Inventory
     {
        static int qoh = 500;
        static int req = 0;
        static public synchronized void request(int order)
          {
             if ( order <= qoh)
               {
               System.out.println ("Quantity ordered :"  + order);
               qoh -= order;
               req += order;
               System.out.println ("Quantity on hand :"  + qoh);
               System.out.println("Total quantify taken away by way
                                   of order :"+ req);
                   }
               else  {
            System.out.println ("Ordered quantity more than quantity
```

```
                                        on hand");
                        }
              }

             public static void main(String args[])
                {
                  new OurThread();
                  new OurThread();
                   try {
                     for(int p = 3; p > 0; p--)
                       {
                       System.out.println ("======================");
                       System.out.println (" main thread  :"  + p);
                       System.out.println ("======================");
                       Thread.sleep (1000);
                      }
              } catch (InterruptedException  e)  { }

              System.out.println (" exiting main thread . ..");
               }
          }

        class OurThread extends Thread
         {
            OurThread()
              {
                 super ("test thread");
                 System.out.println("child thread :" + this);
                 start();
                   }
            public void run()
              {
              for(int i=5; i > 0; i--)
               {
                try
                 {
                 sleep(100);
                 } catch(InterruptedException  e ) { }

                 Inventory.request((int)(Math.random()*100));
                   }
             }
       }
```

Ex – 16.2

```
1:          class Consumer implements Runnable
           {
             Stock c;
             Thread t;
             Consumer (Stock c)
              {
```

```
                this.c = c;
                t = new Thread(this, " producer thread");
                t.start();
                 }
             public void run()
               {
                 while (true)
                  {
                 try  {
                   t.sleep(750);
                   } catch (InterruptedException  e)  { }
                c. getstock((int)(Math.random()*100));
                   }
           }

           void stop()
            {
              t.stop();
             }
       }

2:         class Producer implements Runnable
     {
            Stock s;
            Thread t;
            Producer(Stock s )
              {
                 this.s = s;
                 t = new Thread(this, "producer thread");
                 t.start();
               }
             public void run()
              {
                while(true)
                  {
                    try
                     {
                    t.sleep(750);
                    } catch (InterruptedException e) { }
                   s.addstock((int)(Math.random()*100));
                    }
              }

      void stop()
        {
        t.stop();
           }
      }


3:
```

```
class Stock
  {
    int goods = 0;
    public synchronized void addstock(int i)
      {
        goods = goods + i;
       System.out.println("Stock added :" + i);
       System.out.println("present stock :" + goods );
       notify();
        }
  public synchronized int getstock(int j)
      {
     while(true)
          {
       if(goods >= j)
            {
             goods  = goods - j ;
             System.out.println("Stock taken away :" + j);
             System.out.println("Present stock  :" + goods);
             break;
             }
       else {
          System.out.println("Stock not enough …" );
          System.out.println  (" Waiting for stocks to come …");
          try  {
             wait();
               }catch(InterruptedException e)  { }
           }
  } / / end of while

    return goods;
  }

public static void main(String args[])
  {
   Stock j = new Stock();
   Producer p = new Producer(j);
   Consumer c = new Consumer(j);
     try {
        Thread.sleep(10000);
        p.stop();
        c.stop();
        p.t.join();
        c.t.join();
        System.out.println("Thread stopped");
       } catch(InterruptedException e) { }
        System.exit(0);
          }
   }
```

Ex – 16.3

```
import java.lang.Thread;
import java.lang.System;
import java.lang.Math;
import java.lang.InterruptedException;
class ThreadSynchronization
{
   public static void main (String args [ ])
    {
      MyThread thread1 = new MyThread (“thread1:”);
      MyThread thread2 = new MyThread (“thread2:”);
      thread1.start ( );
      thread2.start ( );
      boolean thread1IsAlive = true;
      boolean thread2IsAlive = true;
      do
      {
         if (thread1IsAlive && !thread1.isAlive ( ) )
           {
              thread1IsAlive = false;
               System.out.println (“Thread 1 is dead.”);
               }
         if (thread2IsAlive && !thread2.isAlive ( ))
           {
              thread2IsAlive = false;
            System.out.println (“Thread 2 is dead.”);
                }
          }while (thread1IsAlive  | | thread2IsAlive);
     }
}

class MyThread extends Thread
{
 static String message [ ] = {“Java”, “is”, “hot”, “aromatic,”,
                                      “and”, “invigorating.” };
   public MyThread (String id)
     {
         super (id);
          }
   public void run ( )
     {
          SynchronizedOutput.displayList (getName(),message);
          }

   void randomWait ()
     {
     try
      {
        sleep ( (long ) (3000*Math.random ( )) );
       }
```

```
                catch (InterruptedException  x )
                 {
                 System.out.println (“Interrupted!”);
                    }
            }
      }
      class SynchronizedOutput
      {
         public static synchronized void displayList(String name,
                                                String list [ ])
          {
             for (int i=0;i<list.length; + +i)
                 {
         MyThread t = (MyThread) Thread.currentThread ( );
          t.randomWait ( );
          System.out.println (name + list [i]);
               }
          }
      }
```

Ex – 16.4

```
      public class ComplexThread extends Thread
       {
          private int delay;

          ComplexThread (String name, float seconds)
          {
             super (name);
             delay = (int) seconds * 1000;// delays are in milliseconds
             start();                    // start up ourself!

            }
      public void run ( )
      {
        while (true)
        {
         System.out.println (Thread.currentThread ( ).getName ( ) );
         try
         {
          Thread.sleep (delay);
          }
         catch (InterruptedException   e)
         {
           return ;
            }
          }
      }

      public static void main (String argc[ ])
       {
```

```
            new ComplexThread (“one potato”,     1.1F);
            new ComplexThread (“two potato”,     1.3F);
            new ComplexThread (“three potato”,    0.5F);
            new ComplexThread (“four”,       0.7F);
      }
      }
```

Lab - 17(2 hrs Real Time):

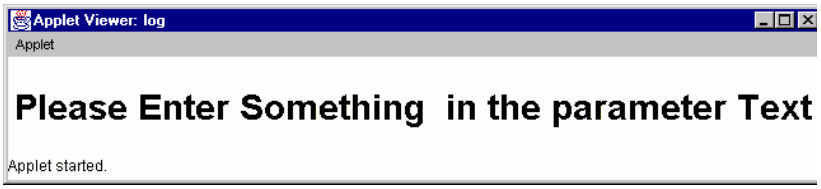
Ex – 17.1

```
      import java.awt.Graphics;
      public class HelloWorld  extends java.applet.Applet
       {
          public void paint (Graphics g )
          {
          g.drawString (“Hello World” , 5, 50 );
          }
      }

      // After compiling this run the program using appletviewer.
      // create a following HTML coding.
      // appletviewer filename (html file name).

      < HTML >
      < APPLET CODE = HelloWorld WIDTH=200 HEIGHT = 100>
      < / APPLET >
      < / HTML >
```

Ex – 17.2

```
      // Logo version 1.0 rev

      import java.awt.Graphics;
      import java.awt.Font;

      public class log extends java.applet.Applet
         {

       // Declare the object variable array StrLine with 3 values.

           String StrLine[] = new String[1];
       // Declare the Font object called appFont.
           Font appFont;
          public void init()
             {
             // Get the value for the
```

```
   String att = getParameter("Text");
  StrLine[0] = (att == null) ? "Please Enter Something  in
                      the parameter Text !" : att;

// Construct the font with the following attributes :

// Font      Attrib    Size    appFont = new Font(" Helvetica",
Font.BOLD, 28);
      }

public void paint(Graphics  g)
  {
      // Set the Font for object g
       g.setFont(appFont);
      // Display the variable on the screen
       g.drawString(StrLine[0], 5, 50);
      }
}
```



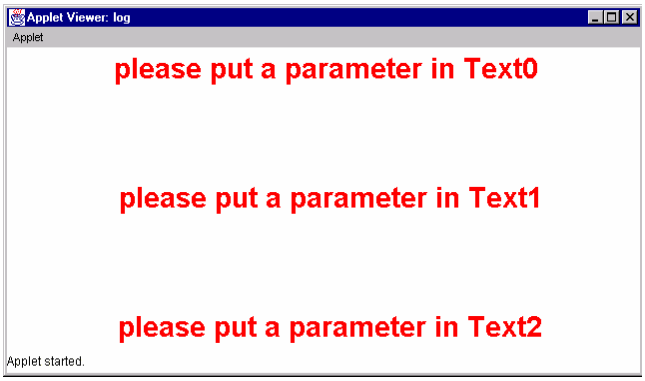
Lab - 18(2 hrs Real Time)

Ex – 18.1

```
/ / aniLogo rev

import java.awt.*;
public class anilogo extends java.applet.Applet implements
Runnable
 {
     Rectangle  r;

 // Declare the object variable array StrLine with 3 values .
    String StrLine[] = new String[3];

    Font appFont;

// Create a Thread object called myThread
    Thread myThread  = null;

// declare width array
     int width[] = new int[3];

    public void init()
```

```
      {

// A for loop to find the three lines of text.
       for(int i=0;i < 3;i++)
        {
          String att = getParameter("Text" + i);
          StrLine[i] = (att == null) ? ("please put a parameter
                         in Text" + i ) : att;
       }
      appFont  = new Font("Helvetica", Font.BOLD, 28);

 // Set r = to the bounds of the applet
      r = bounds();
}

// Override this methods from the interface Runnable
public void run()
 {

   // Set the current Threads priority.
   Thread.currentThread().setPriority(Thread.NORM_PRIORITY-1);

  // Initialize the locations of the two lines of text to be

  // of the screen.

    width[1] = 2000;
    width[2] = 2000;
    repaint();

 // Algorithm to send the first line of Text across the screen.

    for(int i = -50;i < r.width/2; i+=7)
      {
         width[0] = i;
         repaint();

 // Rest
        Rest(1);
      }

// Algorithm to send the second line of text across the screen.
    for(int i=r.width + 20; i > r.width/2; i -= 7)
        {
          width[1] = i;
          repaint();
          Rest(1);
         }

// Algorithm to send the third line of text across the screen
    for(int i = 90; i < r.width/2; i += 7)
```

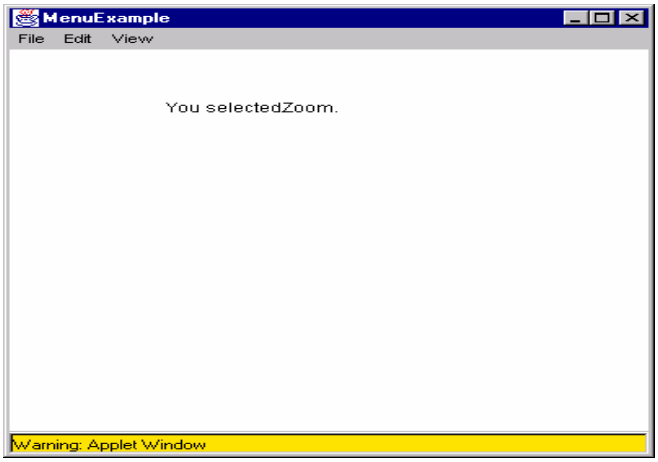
```
        {
             width[2] = i;
             repaint();
            // Rest
             Rest(1);
            }
}
      void Rest(int r)
        {

        // Rest for a period of time

            try
              {
                myThread.sleep(100*r);
                }catch(InterruptedException e)
                   {
                     return;
                     }
               }
public void start()
  {
   if (myThread == null)
    {
      myThread = new Thread(this);
      myThread.start();
    }
}

public void stop()
  {
    if (myThread != null)
      {
         myThread.stop();
         myThread = null;
         }
   }


public void paint(Graphics g)
  {
    g.setColor(Color.red);
    g.setFont(appFont);
    for(int i =0;i < 3;i++)
     {
      g.drawString(StrLine[i],width[i] +(10+i),30 +(125*i));
       }
    }
}
```



Ex – 18.2

```
import java.awt. *;
public class MenuExample extends Frame
 {
   String menuSelection = “Select a menu item.”;
   public static void main (String args [ ] ) {
   MenuExample  win = new MenuExample ( );
  }
public MenuExample ( )
 {
  super (“MenuExample”);
  pack ( );
  resize (400,400);
  addMenu( );
  show ( );
  }
void addMenu()
 {
  MenuBar menubar = new MenuBar ();
  Menu file = new Menu (“File”);
  Menu edit = new Menu (“Edit”);
  Menu view = new Menu (“View”);
  file.add (“Open”);
  file.add(“Save”);
  file.add(“Close”);
  file.add(“Quit”);
  edit.add( “Copy”);
  edit.add(“Cut”);
  edit.add (“Zoom”);
  menubar.add (file);
  menubar.add(edit);
  menubar.add(view);
  setMenuBar(menubar);
}
public void paint(Graphics g)
 {
```

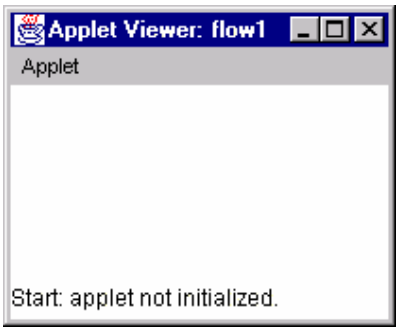
```
    g.drawString (menuSelection,100,100);
    }
  public boolean handleEvent(Event event)
    {
     if (event.id = =Event.WINDOW_DESTROY)
     {
      System.exit (0);
      return true;
      }
    else if (event.id = = Event.ACTION_EVENT &&
              event. target instanceof MenuItem )
    {
      if ("Quit".equals (event.arg) )
      {
       System.exit (0);
       return true;
        }
       else
       {
       menuSelection = "You selected" +event.arg.toString ( )
       repaint ( );
       return true;
       }
     }
     else
      return false;
     }
  }
```

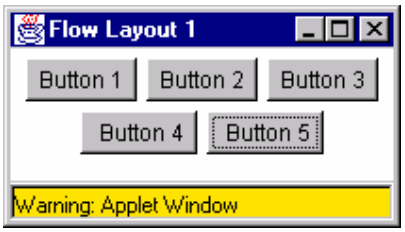


Ex – 18.3

```
import java.awt.*;

public class flow1 extends Frame
{
public flow1()
{
super ("Flow Layout 1");
```

```
setLayout  (new FlowLayout ( ) );
add (new Button ( “Button 1”)  );
add (new Button (“Button 2”)   );
add ( new Button (“Button 3”)   );
add ( new Button (“ Button 4”)  );
add (new Button  (“ Button 5”)  );
setBounds (100,100,200,100);
setVisible (true);
}
public static void main (String arg [ ])
{
new flow1();
  }
}
```





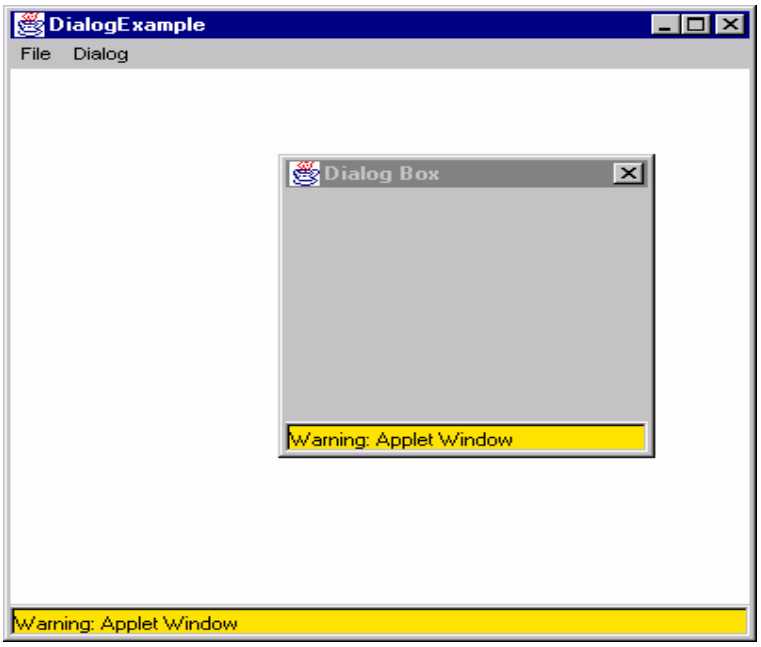
Lab - 19 (2 Hrs Real Time)

Ex 19.1

```
import java.awt.*;
public class DialogExample extends Frame
{
       Dialog dialog;
        public static void main (String args [ ])
        {
           DialogExample win = new DialongExample();
          }
      public DialogExample ( )
          {
            super (“DialogExample”);
            pack ( );
            resize (400,400);
```

```
            addMenus();
            createDialog();
            show();
           }
         void addMenus()
          {
           MenuBar menubar = new MenuBar();
           Menu file = new Menu ("File");
           Menu dialog  = new Menu ("Dialog");
           file.add ("Quit");
           dialog.add("Show");
           dialog.add("Hide");
           menubar.add (file);
           menubar.add(file);
           menubar.add(dialog);
           setMenuBar (menubar);
          }
         void createDialog ( )
         {
          dialog = new Dialog (this, "Dialog Box", false);
          dialog.resize (200,200);
          }
public boolean handleEvent(Event event)
{
   if (event.id = =Event.WINDOW_DESTROY)
      {
        System.exit (0);
        return true;
       }
    else if (event.id == Event.ACTION_EVENT &&
      event.target instanceof MenuItem)
       {
          if ("Quit." equals (event.arg) )
           {
             System.exit (0);
             return true;
            } else if ("Show".equals (event.arg ) ) {
             dialog.show ( );
             return true;
             } else
                 {
                  dialog.hide ();
                  return true;
                   }
                    }
                  else
                   return false;
                   }
```

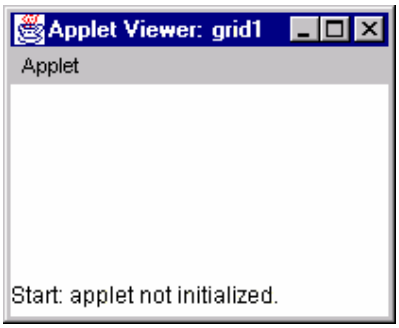
```
}
```

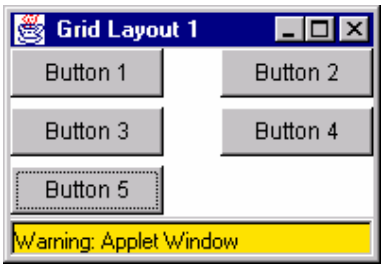


Ex – 19.2

Grid Layout

```
import java.awt.*;
public class grid1 extends Frame
{
public grid1()
{
super (“ Grid Layout 1”);
setLayout (new GridLayout (3, 3, 30, 5) );
add (new Button (“Button 1” ) );
add (new Button (“Button 2”)  );
add (new Button ( “Button 3”) );
add (new Button ( “Button 4” ) );
add (new Button (“Button 5” )  );
setBounds(100,100,200,100);
setVisible (true);
}
public static void main(String arg [ ])
 {
new grid1();
   }
}
```

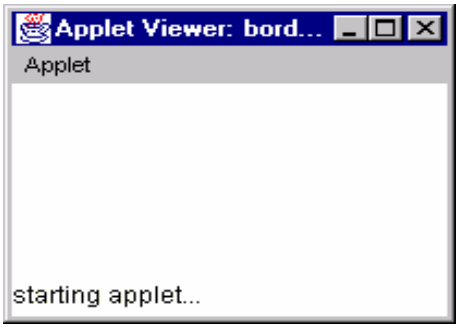


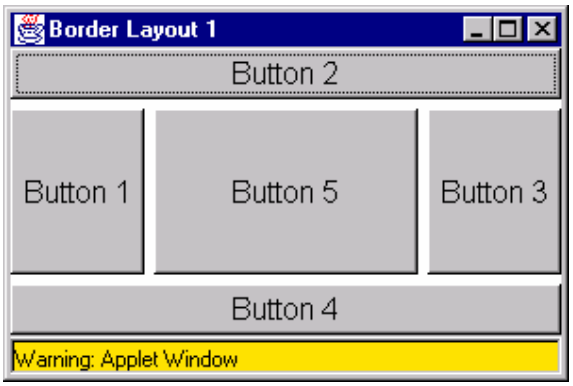


Ex – 19.3

```
Border Layout
import java.awt.*;
public class border1 extends Frame
{
public border1()
{
super ("Border Layout 1");
setFont (new Font ("Helvetica", Font. PLAIN, 16 ) );
setLayout (new BorderLayout (5, 5));
add ( new Button ("Button 1"),BorderLayout.WEST);
add ( new Button ("Button 2"),BorderLayout.NORTH);
add ( new Button ("Button 3"),BorderLayout.EAST);
add ( new Button ("Button 4"),BorderLayout.SOUTH);
add ( new Button ("Button 5" ),BorderLayout.CENTER);
setBounds(100, 100, 300, 200);

setVisible(true);
}
public static void main (String arg [ ])
    {
new border1( );
  }
```

```
}
```

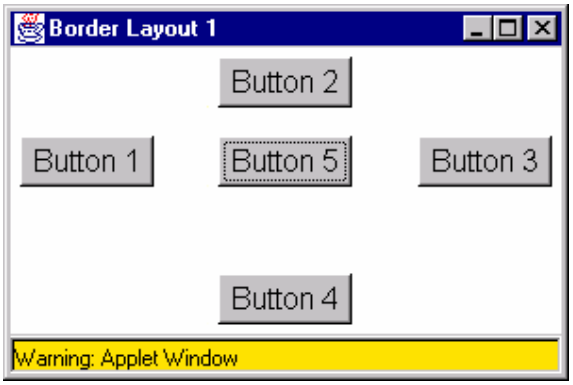




Ex – 19.4

```
Padding Layout
import java.awt.*;
public class border2 extends Frame
{
public border2 ( )
{
super (“Border Layout 1”);
setFont (new Font (“Helvetica”, Font.PLAIN, 16) );
setLayout (new BorderLayout (5,  5) );
Panel p1  = new Panel();
p1.add (new Button (“Button 1”) );
add (“West”, p1);
Panel p2 = new Panel();
p2.add ( new Button (“Button 2” ) );
add( “North”, p2);
Pane1 p3 = new Panel();
p3.add ( new Button (“Button 3”) );
add (“East”, p3);
Pane1 p4  = new Panel();
p4.add ( new Button (“Button 4”) );
add (“South”,  p4);
Pane1 p5 = new Panel();
p5.add ( new Button (“Button 5”) );
add (“Center”,  p5);
setBounds (100,  100,  300, 200);
setVisible (true);
```

```
}
static public void main (String [ ]argv )
{
new border2 ( );
   }
}
```



Lab - 20 (2 Hrs Real Time)

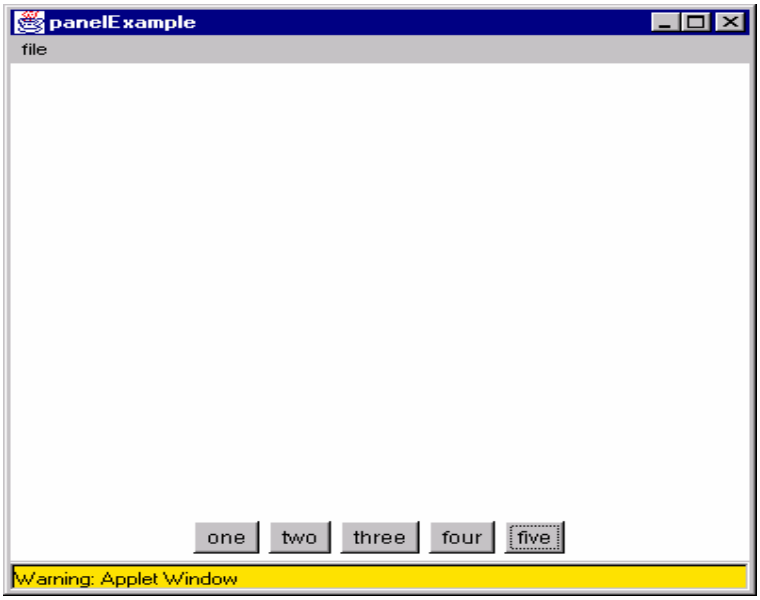
Ex – 20.1

```
//Win version 1.0 rev
import java.awt.*;
public class Win extends java.applet.Applet
   {
      public void init()
      {
   // Set the Layout to FlowLayout
      setLayout(new FlowLayout(FlowLayout.CENTER, 10, 10));

  // Add a Label and a Button
      add (new Label("Please press the button below:"));
      Button show = new Button("Show Window");
      add(show);
     }
}

// After compiling this run the program using appletviewer.
// create a following HTML coding.
// appletviewer filename (html file name).

<applet code = Win width=200 height=100>
</applet>
```

Ex – 20.2

```
import java.awt. *;
public class PanelExample extends Frame
 {
      public static void main (String args[] )
      {
            panelExample win  = new panelExample();
        }
public panelExamaple  ()
  {
      super ("panelExample");
      pack ( );
      resize (400,400);
      addMenus ( );
      addpanel ( );
      show ( );
    }
void addMenus ( ) {
MenuBar menubar = new MenuBar ( );
Menu file = new Menu("File");
file.add ("Quit" );
menubar.add (file);
setMenuBar (menubar);
}
void addpanel ( )  {
Panel panel = new Panel();
panel.add(new Button ("one") );
panel.add (new Button ("two")  );
panel.add (new Button ("three") );
panel.add (new Button ("four")  );
panel.add (new Button ("five")   );
add ("South", panel );
}
public boolean handleEvent (Event event )  {
if(event.id == Event.WINDOW_DESTROY) {
System.exit ( 0);
return true;
}else if (event.id = =Event.ACTION_EVENT &&
```

```
event.target instanceof MenuItem )  {
if("Quit", equals (event .arg) ) {
System.exit (0);
return true;
}else {
return false;
}
}else return false;
}
}
```

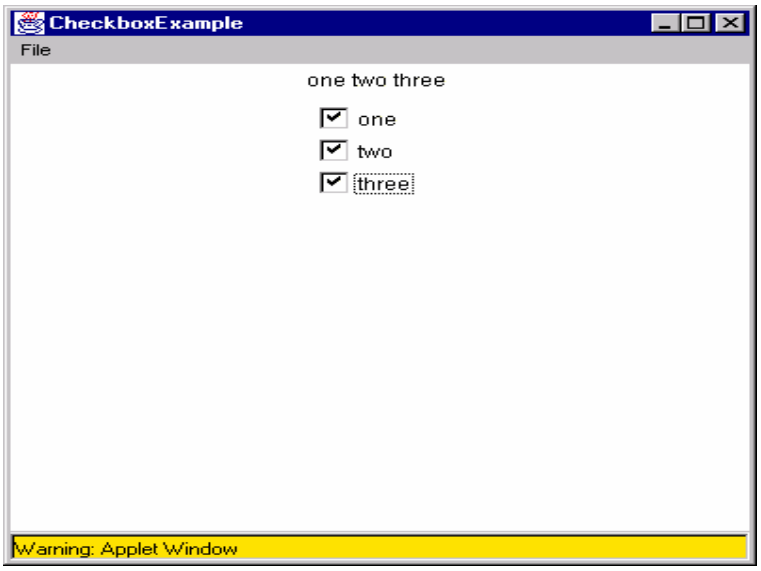


Ex – 20.3

```
import java.awt.*;

public class CheckboxExample extends Frame {
Label label = new Label ("Default Text");
Checkbox checkbox[] = new Checkbox[6];
public static void main (String args [] )  {
CheckboxExample win  = new CheckboxExample();
}
public CheckboxExample()
 {
super ("CheckboxExample");
addMenus();
addComponents();
pack();
resize(400,400);
show();
 }
void addMenus() {
```

```
MenuBar menubar = new MenuBar();
Menu file = new Menu("File");
file.add("Quit");
menubar.add(file);
setMenuBar (menubar);
}
void addComponents ( )  {
add ("North", label );
label.setAlignment(Label.CENTER);
Panel panel = new Panel();
Panel panel1 = new Panel();
panel1.setLayout (new GridLayout (3,1) );
Panel panel2 = new Panel();
panel2. setLayout (new GridLayout (3,1)  );
checkbox [0] = new Checkbox ("one");
checkbox [1] = new Checkbox ("two");
checkbox [2] = new Checkbox ("three");
CheckboxGroup group  = new CheckboxGroup  ( ) ;
checkbox [3] = new Checkbox ("four", group, false );
checkbox [4] = new  Checkbox ("five", group, false );
checkbox [5] = new Checkbox ("six", group, false );
for (int i=0;i<3;++i )
 panel1.add (checkbox [i]);
panel.add (panel1) ;
panel.add(panel2);
add("Center", panel);
}
public boolean handleEvent (Event event)
  {
if (event.id == Event.WINDOW_DESTROY) {
System.exit (0);
return true;
}else if (event.id == Event.ACTION_EVENT && event.target instanceof
MenuItem)  {
if ("Quit".equals(event.arg ) ) {
System.exit(0);
return true;
}
else
{
return false;
}
}
else if (event.id == Event.ACTION_EVENT && event.target instanceof
Checkbox )  {
String text ="";
for (int i=0;i<6; ++i)  {
if (checkbox [i].getState ( )  )
text += checkbox[i].getLabel ( ) + " ";
}
label.setText(text);
```

```
return true;
}
else return false;
}
}
```

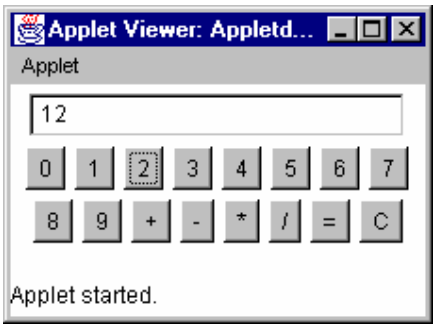


Ex - 20.4

```
import java.applet.*;
 import java.awt.*;
import java.awt.event.*;
public class Appletdemo extends Applet
{
 Button b[];
 TextField t1;
String txt="";
 int no1=0,no2=0,no3=0;
 String oprt="";
 public void init()
   {
     b = new Button[16];
     for(int i =0; i <= 9; i++)
     {
         b[i] = new Button(i + "");
     }
     b[10] = new Button("+");
     b[11] = new Button("-");
```

```
        b[12] = new Button("*");
        b[13] = new Button("/");
        b[14] = new Button("=");
        b[15] = new Button("C");

        t1 = new TextField(25);
        add(t1);

        for(int i =0; i <= 15; i++)
        {
          add(b[i]);
          b[i].addActionListener(new Bh());
        }
      }
     class Bh implements ActionListener
     {
       public void actionPerformed(ActionEvent e)
         {
          String s = e.getActionCommand();
          if (s.equals("+") || s.equals("-") ||
    s.equals("*") || s.equals("/") )
          {
              no1 = Integer.parseInt(t1.getText());
              oprt = s;
              t1.setText(no1+ "");
              txt = "";
          }
          else if (s.equals("C"))
          {
             no1 = no2 = 0;
             txt = "";
             t1.setText("");
          }
          else if (s.equals("="))
          {
              no2 = Integer.parseInt(t1.getText());
               if (oprt.equals("+"))
                    t1.setText((no1 + no2) + "");
               if (oprt.equals("-"))
                    t1.setText((no1 - no2) + "");
               if (oprt.equals("*"))
                    t1.setText((no1 * no2) + "");
               if (oprt.equals("/"))
                    t1.setText((no1 / no2) + "");
               txt = "";
          }
          else
          {
              txt = txt + s;
              t1.setText(txt);
          }
```

```
    }
   }
}
```

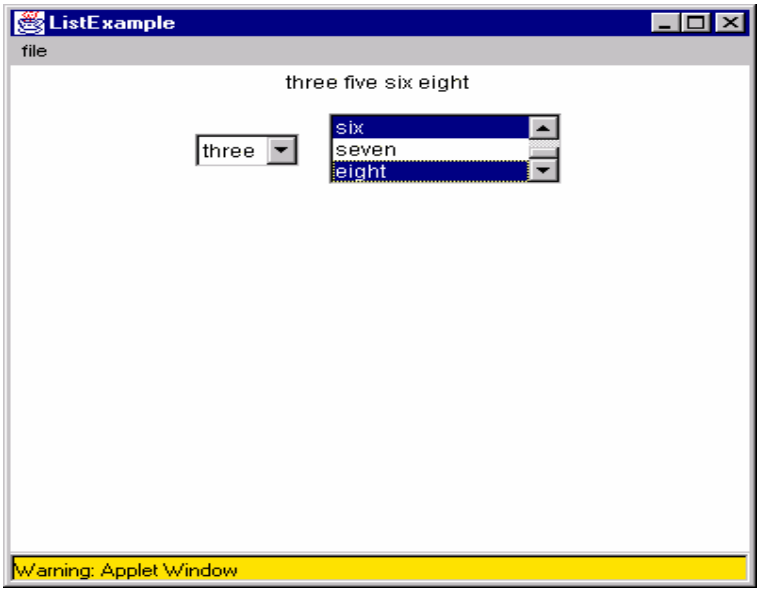


Ex- 20.5

```
import java.awt.*;
public class ListExample extends Frame
{
Label label  = new Label ("Default Text");
Choice choice = new Choice ( );
List list = new List (3, true);
public static void main (String args [ ])
  {
ListExample win = new ListExample ( );
}
public ListExample ( )
 {
super ("ListExample" );
addMenus();
addComponents();
pack();
resize(400,400);
show();
 }
void addMenus()
{
MenuBar menubar = new MenuBar();
Menu file = new Menu("file");
file.add("Quit");
menubar.add(file);
setMenuBar(menubar);
}
void addComponents()
 {
add("North", label );
label.setAlignment (Label.CENTER);
Panel panel = new Panel();
Panel panel1 = new Panel();
Panel panel2 = new Panel();
```

```
try  {
choice.addItem ("one");
choice.addItem("two");
choice.addItem ("three");
}catch (NullPointerException ex)
  {
  }

panel1.add (choice);
list.addItem ("four");
list.addItem ("five");
list.addItem ("six");
list.addItem ("seven");
list.addItem ("eight");
panel2. add (list );
panel.add(panel1);
panel.add(panel2);
add("Center", panel);
}
public boolean handleEvent (Event event)
 {
if (event.id == Event.WINDOW_DESTROY ) {
System.exit (0);
return true;
}
else if (event.id == Event.ACTION_EVENT && event .target instanceof
MenuItem)
  {
if ("Quit".equals(event.arg) )
{
System.exit (0);
return  true ;
}
     else
{
return false;
}
}
else
if (event.target instanceof Choice || event.target instanceof
      List )
 {
  String text = choice. getSelectedItem () + " ";
 for (int i=0;i<5;++i)
 {
  if (list.isSelected (i) )
    text += list.getItem (i) + " ";
   }
 label.setText(text);
 return true;
}
```

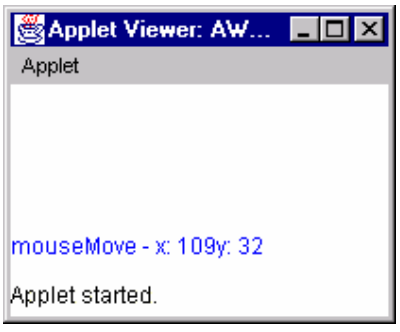
```
else return false;
}
}
```

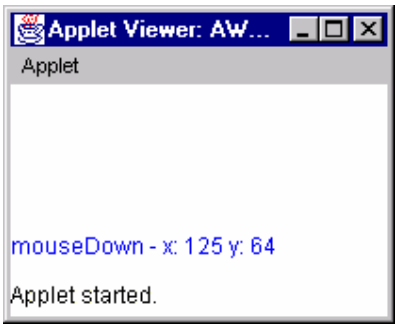


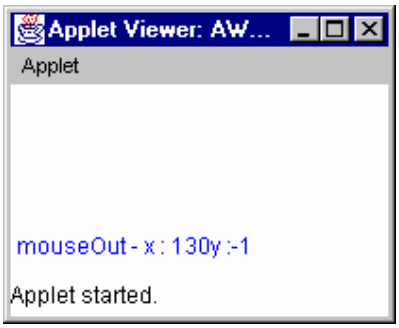
Ex – 20.6

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

public class AWTEventDemo extends Applet
{
     private String message = “Waiting for events . . .”;
      // Default constructor
       public AWTEventDemo ( )
       {
             // Call superclass
             super ( );
       }
    // Init method, called when applet first initializes public void
init ( )
  {
     setBackground(Color.white );
          }
              // Overridden paint method
        public void paint ( Graphics  g)
        {
           g.setColor( Color.blue );
           g.drawString (message, 0, size ( ).height – 10);
         }

          // Overridden methods for event handling
        public boolean mouseEnter (Event evt, int x, int y)
         {
```

```
                // Set  message ……
                message = "mouseEnter – x:" + x +" Y:" + y;
                / / … and repaint applet
                repaint();

                / / Signal we have handled the event

               return true;
            }
          public boolean mouseExit (Event evt, int x, int y)
            {
                  // Set message …….
             message = " mouseOut – x : " + x + "y :" + y;

                   // …. and repaint applet
             repaint();

                // Signal we have handled the event
            return true;
          }
        public boolean mouseMove (Event evt, int x, int y)
           {
          //  Set message  ……
               message = "mouseMove – x: " + x +"y:" + y;
                         // ….. and repaint applet
             repaint();
                  // Signal we have handled the event
             return true;
            }
             / /  Mouse click event
              public boolean mouseDown (Event evt, int x, int y )
               {
               //  Set message …..
                 message = "mouseDown – x: " + x + " y: "+ y;
                  / / … and repaint applet
                  repaint();

               / /  Signal we have handled the event
                return true ;
}
}
```



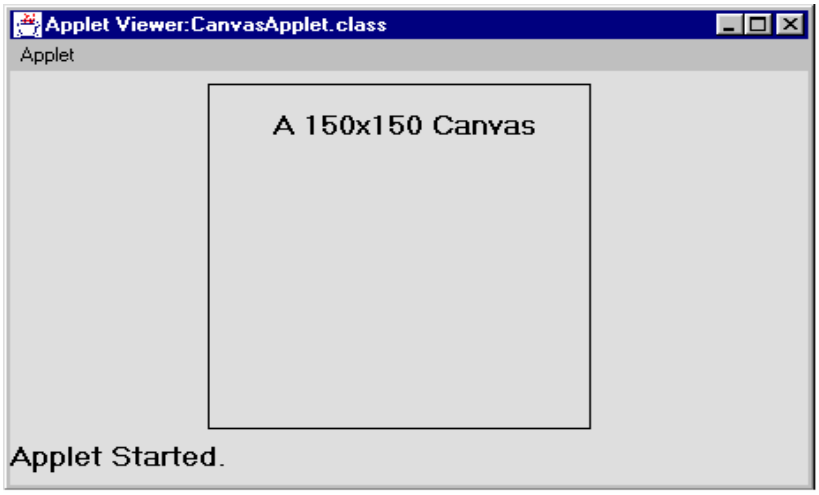




Ex – 20.7

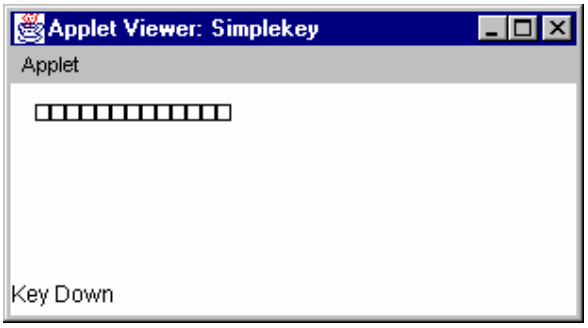
```
import java.awt.*;
import java.applet.*;
public class CanvasApplet extends Applet
{
      public void init( )
      {
            DrawingRegion region = new DrawingRegion ( );
            add(region);
      }
}
class DrawingRegion extends Canvas
{
      public DrawingRegion
      {
            setSize(150,150);
      }
      public void paint (Graphics g)
      {
      g.drawRect(0,0,149,149); //draw border around region
            g.drawstring("A 150 x 150 Canvas", 20,20);
      //draw string
      }
}
```

The above code draws a rectangular region on a canvas.



Ex – 20.8

```
// Demonstrate the key event handlers.

import java.awt.*;
import java.awt.event.*;
import java.applet.*;
/*
         < applet code = “SimpleKey” width=300 height=100>
         < /applet>
*/

public class Simplekey extends Applet implements KeyListener  {
 String msg = " ";
int  X = 10, Y = 20;
public void init()  {
   addKeyListener(this);
    requestFocus();
}

public void keyPressed(KeyEvent  ke) {
        showStatus("Key Down");
}

public void keyReleased(KeyEvent ke) {
     showStatus("Key Up");
}

public void keyTyped (KeyEvent  ke)  {
      msg  += ke.getKeyChar();
       repaint();
}

     public void paint(Graphics g) {
      g.drawString(msg, X, Y);
      }
}
```

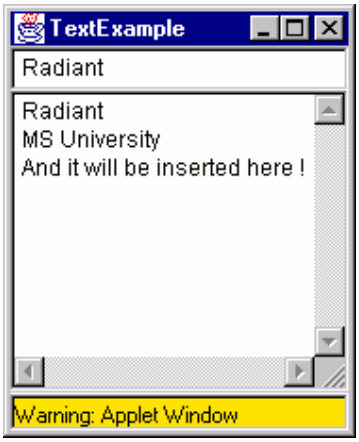


Lab - 21 (2 hrs Real Time)

Ex – 21.1

```
 import java.awt. *;
public class TextExample extends Frame   {
TextField textfield  = new TextField (“Enter text here.”);
TextArea textarea = new TextArea (“And it will be inserted here !”);
public static void main(String args[])
   {
            TextExample win = new TextExample();
   }

public TextExample()
 {
      super (“TextExample”);
      addComponents();
      pack();
      resize(400,400);
      show();
 }
void addMenus()
{
      MenuBar menubar = new MenuBar ( );
      Menu file = new Menu(“File”);
      file.add(“Quit”);
      menubar.add (file);
      setMenuBar(menubar);
      }
void addComponents()
{
add (“North”, textfield);
add(“Center”, textarea);
}
public boolean handleEvent(Event event)
 {
   if (event.id == Event.WINDOW_DESTROY)
   {
    System.exit(0);
    return true;
   }
 else
 if(event.id == Event.ACTION_EVENT && event target instanceof
             MenuItem )
 {
      if (“Quit”.equals(event.arg))
      {
            System.exit (0);
            return true;
            }
            else
```

```
            {
             return false ;
}
}
else
 if (event.id == Event.ACTION_EVENT && event.target instanceof
             TextField )
 {
      textarea.insertText(textfield.getText() + "\n", 0);
      return true;
      }
      else return false;
}
```



Ex – 21.2

```
import java.awt.*;
public class grid1 extends Frame
{
public grid1()
{
super (" Grid Layout 1");
setLayout (new GridLayout (3, 3, 30, 5) );
add (new Button ("Button 1" ) );
add (g.drawString(lastdate, 5, 125);
        }
        if (xm != lastxm || ym != lastym) {
            g.drawLine(xcenter, ycenter-1, lastxm, lastym);
           g.drawLine(xcenter-1, ycenter, lastxm, lastym); }
        if (xh != lastxh || yh != lastyh) {
            g.drawLine(xcenter, ycenter-1, lastxh, lastyh);
            g.drawLine(xcenter-1, ycenter, lastxh, lastyh); }
        g.setColor(numberColor);
        g.drawString("", 5, 125);
        g.drawString(today, 5, 125);
        g.drawLine(xcenter, ycenter, xs, ys);
```

```
            g.setColor(handColor);
            g.drawLine(xcenter, ycenter-1, xm, ym);
            g.drawLine(xcenter-1, ycenter, xm, ym);
            g.drawLine(xcenter, ycenter-1, xh, yh);
            g.drawLine(xcenter-1, ycenter, xh, yh);
            lastxs=xs; lastys=ys;
            lastxm=xm; lastym=ym;
            lastxh=xh; lastyh=yh;
            lastdate = today;
            currentDate=null;
        }

        public void start() {
            timer = new Thread(this);
            timer.start();
        }

        public void stop() {
            timer = null;
        }

        public void run() {
            Thread me = Thread.currentThread();
            while (timer == me) {
                try {
                    Thread.currentThread().sleep(100);
                } catch (InterruptedException e) {
                }
                repaint();
            }
        }

        public void update(Graphics g) {
            paint(g);
        }

        public String getAppletInfo() {
           return "Title: A Clock \nAuthor: Rachel Gollub, 1995 \nAn
    analog clock.";
        }

        public String[][] getParameterInfo() {
            String[][] info = {
                {"bgcolor", "hexadecimal RGB number", "The background
    color. Default is the color of your browser."},
                {"fgcolor1", "hexadecimal RGB number", "The color of the
    hands and dial. Default is blue."},
                {"fgcolor2", "hexadecimal RGB number", "The color of the
    seconds hand and numbers. Default is dark gray."}
            };
            return info;
```

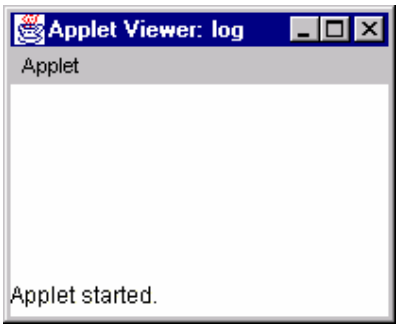
```
    }
}
```

Lab - 22(2 hrs Real Time)

Ex – 22.1

```
public class cycle extends  java.applet. Applet
 {
  public void init()
  {

// Display the  this statement at the bottom of the Window
   showStatus(" The applet is initializing …" );

  // Pause for a period of time
      for(int i = 1;i < 1000000; i++);
        }
     public void start()
      {
      showStatus(" The applet is starting …");
      for(int i =1;i < 1000000;i++);
        }
         public void destroy()
          {
         showStatus(" The applet is being destroyed ..." );
        for(int i = 1 ;i < 1000000;i++);
            }
}
// After compiling this run the program using appletviewer.

// create a following HTML coding.

// appletviewer filename (html file name).

< applet code = cycle width=200 height=200 >
```

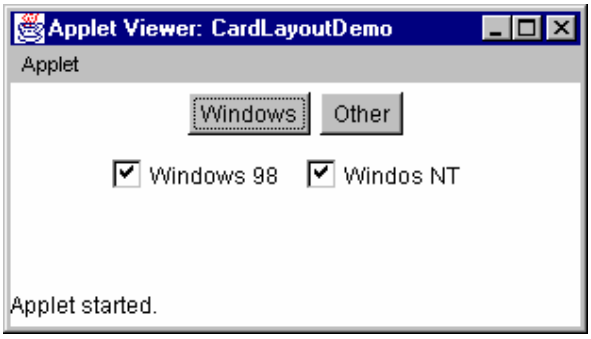
```
< / applet >
```

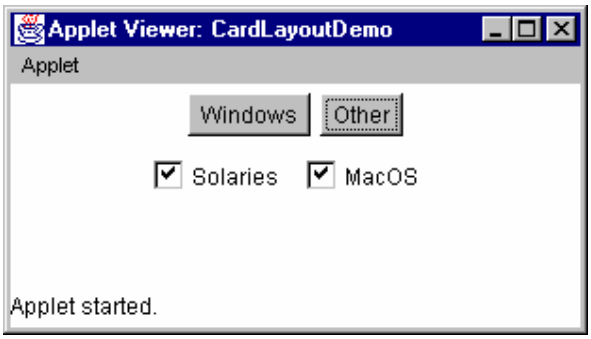


Ex – 22.2

```
/ / Demonstrate CardLayout.

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

<applet code = “CardLayoutDemo” width = 300 height = 100 >
         < /applet>
*/
public class CardLayoutDemo extends Applet implements
ActionListener,MouseListener  {
Checkbox Win98, winNT, solaris, mac;
Panel osCards;
CardLayout cardLo;
Button Win, Other;
public void init()
 {
     Win  = new Button ("Windows");
     Other = new Button ("Other") ;
     add (Win) ;
     add (Other);

cardLo  = new CardLayout();
osCards  = new Panel();
osCards.setLayout (cardLo);
Win98  = new Checkbox ("Windows 98", null, true);
winNT  =  new Checkbox ("Windos NT");
solaris = new Checkbox ("Solaries");
mac = new Checkbox ("MacOS");
Panel winPan = new Panel();
winPan.add(Win98);
winPan.add(winNT);
Panel otherPan = new Panel();
```

```
otherPan.add(solaris);
otherPan.add (mac);
osCards.add(winPan, "Windows");
osCards.add (otherPan, "Other");
add (osCards);
Win.addActionListener (this);
Other.addActionListener (this);
addMouseListener (this);
}
public void mousePressed (MouseEvent me)
{
   cardLo.next (osCards);
}
public void mouseClicked (MouseEvent  me)
{
}
public void mouseEntered (MouseEvent  me)
{
}
public void mouseExited (MouseEvent  me)
{
}
public void mouseReleased (MouseEvent  me)
{
}
public void actionPerformed (ActionEvent  ae)
{
  if (ae.getSource() == Win )
   {
   cardLo.show(osCards,"Windows");
    }
   else {
    cardLo.show(osCards, "Other");
        }
   }
}
```




Applet Viewer: CardLayoutDemo
Applet
Windows
Other
Solaries
MacOS
Applet started.