# Lab Manual

**Program 1 :** Setup a Java programming development environment by using :

(a) Command prompt. (Classpath and path setup)

(b) Any IDE (Eclipse, Jcreator etc.)

**Solution :**

### (a) Path setup

1. After installing jdk. Open command prompt.
2. Set Path using.

```
Set CLASSPATH = C:\jdk\bin
```

### (b) Launch

**Step 0 : Launch Eclipse**

1. Eclipse by running "eclipse.exe" from the Eclipse installed directory.
2. Choose an appropriate directory for your *workspace*, i.e., where you would like to save your files (e.g., c:\myproject\eclipse for Windows).
3. If the "Welcome" screen shows up, close it by clicking the "cross" button next to the "Welcome" title.

**Step 1 : Create a new Java Project**

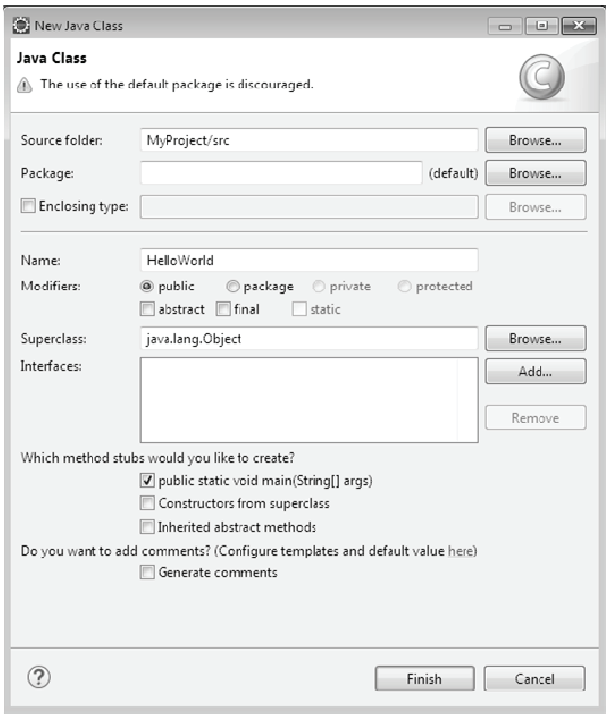
**To create a new Java project**

1. Select "File" menu ⇒ "New" ⇒ "Java project" (or "File" ⇒ "New" ⇒ "Project" ⇒ "Java project").
2. The "New Java Project" dialog pops up.
3. In "Project name", enter "MyProject".
4. Check "Use default location".
5. In "JRE", select "Use default JRE (currently 'JDK10.0.x')". But make sure that your JDK is 1.8 and above.
6. In "Project Layout", check "Use project folder as root for sources and class files".
   Click "Next" button.
7. Uncheck "Create module-info.java file" box (if it is checked) ⇒ Finish.

**Step 2 : Write a Hello-world Java Program**

1. In the "Package Explorer" (left side pane) ⇒ Right-click on "MyProject" ⇒ New ⇒ Class.
2. The "New Java Class" dialog pops up.
3. In "Source folder", keep the "MyProject".

4. In "Package", delete the content if it is not empty.
5. In "Name", enter "HelloWorld".
6. Check "public static void main(String[] args)".
7. Don't change the rest.



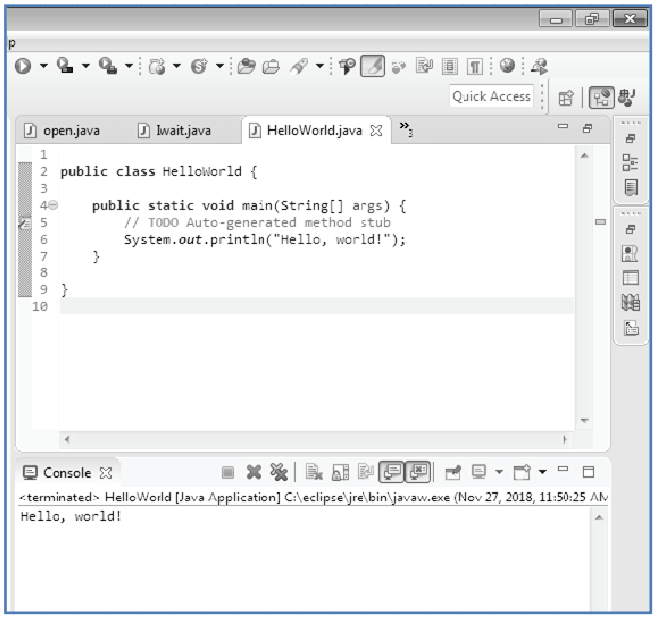

Push "Finish" button.

8. The source file "HelloWorld.java" opens on the editor panel (the center pane). Enter the following codes :

```
public class HelloWorld {
  public static void main(String[] args) {
    System.out.println("Hello, world!");
  }
}
```

**Step 3 : Compile and Execute the Java Program**

1. There is no need to compile the Java source file in Eclipse explicitly. It is because Eclipse performs the so-called incremental compilation, i.e., the Java statement is compiled as and when it is entered.
2. To run the program, right-click anywhere on the source file "HelloWorld.java" ⇒ Run As ⇒ Java Application.
3. The output "Hello, world!" appears on the Console panel (the bottom pane).

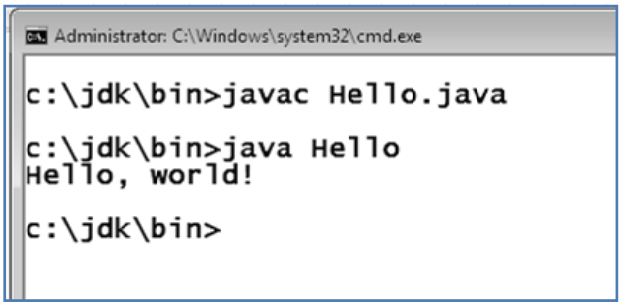



**Program 2 :** Test the JDE setup by implementing a small program.
**Solution :**

**Hello.java**

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

            System.out.println("Hello, world!");

     }
}
```

**Output**

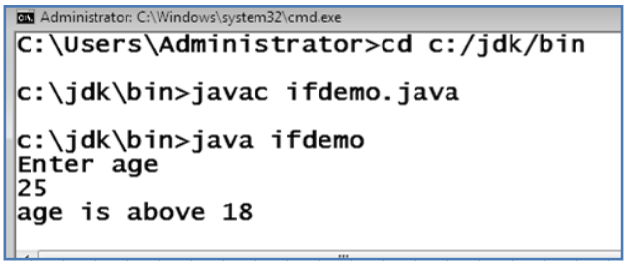



**Program 3 :** Develop programs to demonstrate use of if statements and its different forms.

**Solution : The if statement**

```
import java.util.Scanner;
class ifdemo
{
public static void main(String args[])
{
int age;
Scanner s1=new Scanner(System.in);
System.out.println("Enter age");
age=s1.nextInt();
if(age>18)
System.out.println("age is above 18");
}
}
```

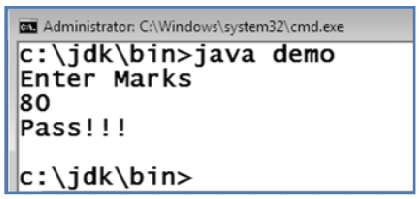
**Output**



**The if else Statement**

```
import java.util.Scanner;
class demo
{
public static void main(String args[])
{
int marks;
Scanner s1=new Scanner(System.in);
System.out.println("Enter Marks");
marks=s1.nextInt();
if(marks>40)
System.out.println("Pass!!!");
else
System.out.println("Fail!!!");
}
}
```

**Output**




**else if Ladder**

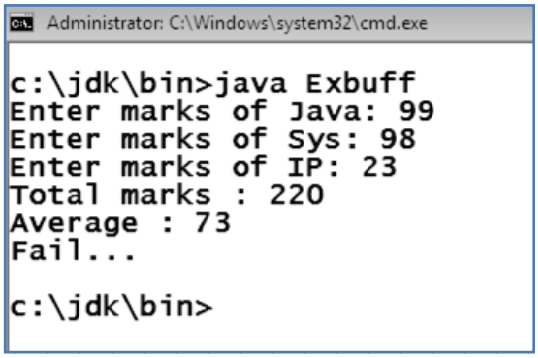
```
import java.io.*;
class Exbuff
{
public static void main(String args[]) throws IOException
{
int Java,Sys,IP,total,avg;
BufferedReader br = new BufferedReader(new
InputStreamReader(System.in));
```

```
System.out.print("Enter marks of Java: ");
Java = Integer.parseInt(br.readLine());
System.out.print("Enter marks of Sys: ");
Sys = Integer.parseInt(br.readLine());
System.out.print("Enter marks of IP: ");
IP = Integer.parseInt(br.readLine());
total = Java + Sys + IP;
avg = total / 3;
System.out.println("Total marks : "+total);
System.out.println("Average : "+avg);
if(Java>=40 && Sys>=40 && IP>=40)
{
if(avg>=80)
System.out.println("Grade : A");
else if(avg>=60)
System.out.println("Grade : B");
else if(avg>=40)
System.out.println("Grade : C");
}
else
System.out.println("Fail...");
}
}
```

Marks are accepted and converted in int

Student should pass in all subjects

**Output**



**Program 4 :** Develop programs to demonstrate use of

(a) Switch - Case statement (b) Conditional if (?)

**Solution :**

**(a) Switch Case statement**

```
import java.io.*;

class Exbuff
{
public static void main(String args[]) throws IOException
{
int n1=0,n2=0,r,ch;
BufferedReader br = new BufferedReader(new
InputStreamReader(System.in));
System.out.println("1 : Addition");
System.out.println("2 : Subtraction");
System.out.println("3 : Multiplcation");
System.out.println("4 : Division");
System.out.print("Select your choice : ");
ch = Integer.parseInt(br.readLine());
if(ch>=1 && ch<=4)
{
System.out.print("Enter first number : ");
n1 = Integer.parseInt(br.readLine());
System.out.print("Enter second number : ");
n2 = Integer.parseInt(br.readLine());
}
switch(ch)
{
case 1:
r = n1 + n2;
System.out.println("Summation is "+r);
break;
case 2:
r = n1 - n2;
System.out.println("Subtraction is "+r);
break;
```

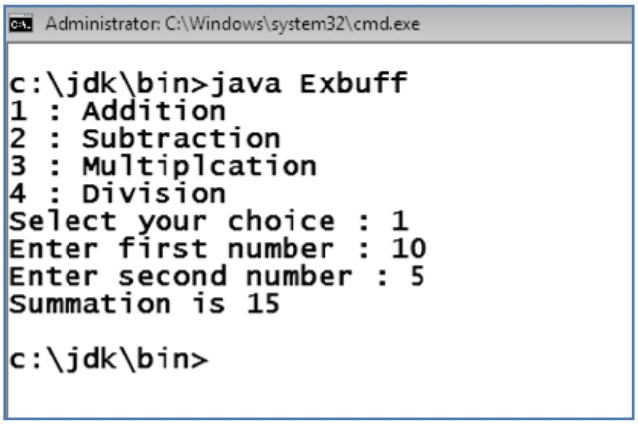
Displaying menu

Accept numbers only for valid choice

```
case 3:
r = n1 * n2;
System.out.println("Multiplication is "+r);
break;
case 4:
if(n2!=0)
{
r = n1 / n2;
System.out.println("Division is "+r);
}
else
System.out.println("Cannot divide by zero");
break;
default :
System.out.println("Invalid choice");
}
}
}
```

Division by zero is not allowed

**Output**

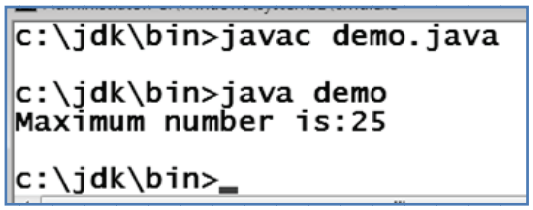



**(b) Conditional if (? :)**

```
import java.util.Scanner;
class demo
{
public static void main(String args[])
```

```
{
int x=10,y=25,max;
max = (x>y) ? x: y;
System.out.println("Maximum number is:" +max);
}
}
```

**Output**

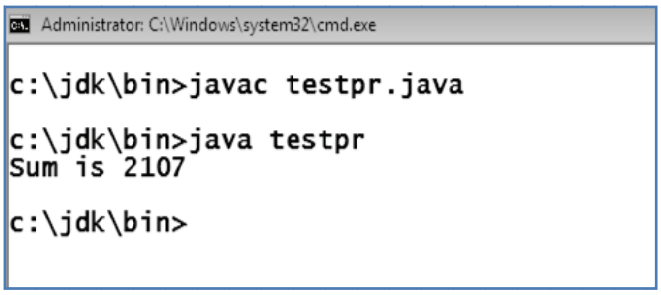



**Program 5 :** Develop programs to demonstrate use of Looping Statement 'for '

**Solution :**

```
class testpr
{
public static void main(String args[])
{
int i,sum;
sum = 0;
for(i=101;i<200;i++)
{
if(i%7==0)
{
 sum = sum + i;
}
}
System.out.println("Sum is "+sum);
}
}
```

**Output**

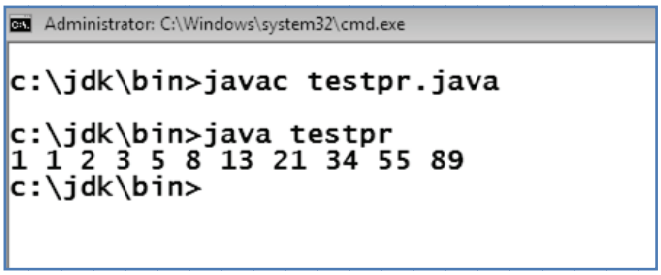



**Program 6 :** Develop programs to demonstrate use of 'while'. do-while.

**Solution : Do -While**

```
class testpr
{
public static void main(String args[])
{
int a,b,c,i;
        a = 1;
        b = 2;
        i = 1;
        System.out.print("1 1 2");
        do
        {
              c = a + b;
              System.out.print(" "+c);
              a = b;
              b = c;
              i = i + 1;
        }while(i<9);
}
}
```

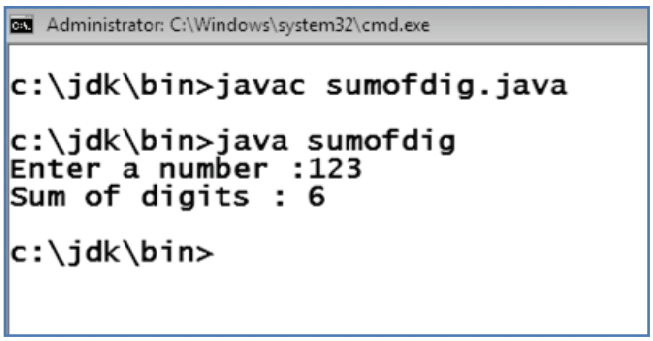
**Output**




**While loop**

```
import java.util.*;
class sumofdig
{
public static void main(String args[])
{
Scanner sc = new Scanner(System.in);
int n,rem,sum;
sum = 0;
System.out.print("Enter a number :");
n = sc.nextInt();
while(n>0)
{
rem = n % 10;
sum = sum + rem ;
n = n / 10;
}
System.out.println("Sum of digits : "+sum);
}
}
```

**Output**

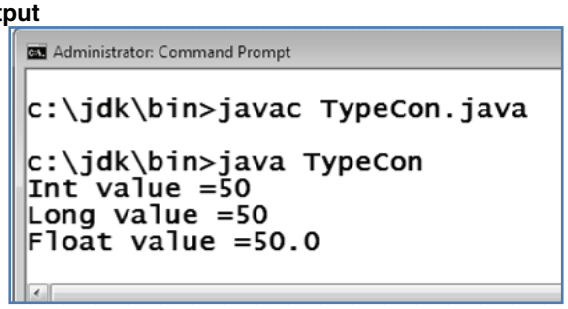


**Program 7 :** Develop a program for implementation of implicit type casting in Java part – I.

**Solution :**

```
public class TypeCon
{
   public static void main(String[] args)
   {
    int a = 50;
    long l = a;
    float f = a;
    System.out.println("Int value ="+a);
    System.out.println("Long value ="+l);
    System.out.println("Float value ="+f);
   }
}
```

**Output**



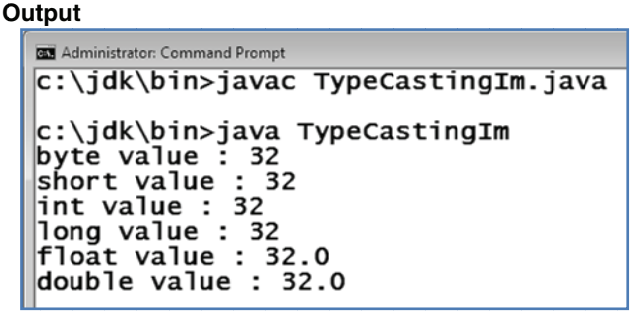

**Program 8 :** Develop a program for implementation of implicit type casting in Java. Part - II.

**Solution :**

```
class TypeCastingIm
{
      public static void main(String args[])
      {
         byte b = 32;

         short j = b;
```

```
        int k = j;
        long l = k;
        float m = l;
        double n = m;

        System.out.println("byte value : "+b);
        System.out.println("short value : "+j);
        System.out.println("int value : "+k);
        System.out.println("long value : "+l);
        System.out.println("float value : "+m);
        System.out.println("double value : "+n);
    }
}
```

**Output**



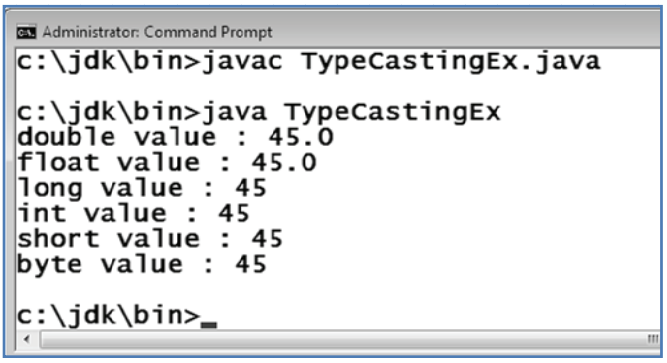

**Program 9 :** Develop a program for implementation of explicit type conversion in Java.

**Solution :**

```
public class TypeCastingEx
{
    public static void main(String args[])
    {
        double d = 45.0;

        float f = (float) d;
        long l = (long) f;
        int i  = (int) l;
        short s = (short) i;
```

```
            byte b = (byte) s;

            System.out.println("double value : "+d);
            System.out.println("float value : "+f);
            System.out.println("long value : "+l);
            System.out.println("int value : "+i);
            System.out.println("short value : "+s);
            System.out.println("byte value : "+b);
        }
}
```

**Output**




**Program 10 :** (a) Develop a program for implementation of constructor.

(b) Develop a program for implementation of multiple constructors in a class.

**Solution :**

**(a) Develop a program for implementation of constructor**

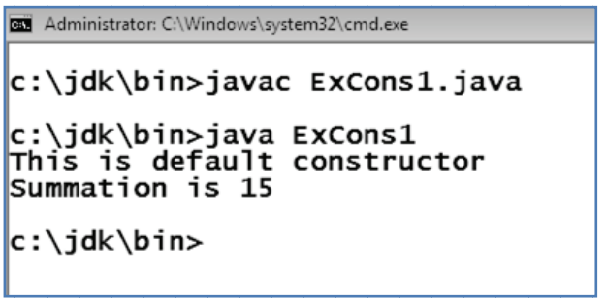
```
class ExCons
{
int n1,n2,sum;
ExCons()
{
System.out.println("This is default constructor");
n1 = 10;
n2 = 5;
}
```

Definition of default constructor

```
void cal()
{
sum = n1 + n2;
System.out.println("Summation is "+sum);
}
}
class ExCons1
{
public static void main(String args[])
{
ExCons obj = new ExCons();
obj.cal();
}
}
```

Default constructor get called

**Output**




**(b) Develop a program for implementation of multiple constructors in a class**

```
class ExCons
{
int n1,n2,sum;
ExCons()
{
n1 = 0;
n2 = 0;
}
ExCons(int x, int y)
{
```

Definition of default constructor

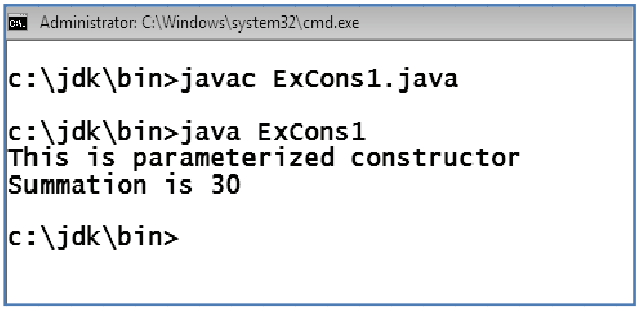
```
System.out.println("This is parameterized constructor");
n1 = x;
n2 = y;
}
void cal()
{
sum = n1 + n2;
System.out.println("Summation is "+sum);
}
}
class ExCons1
{
public static void main(String args[])
{
new ExCons();
ExCons obj = new ExCons(10,20);
obj.cal();
}
}
```

Definition of parameterized constructor

Default constructor get called

Parameterized constructor get called

**Output**



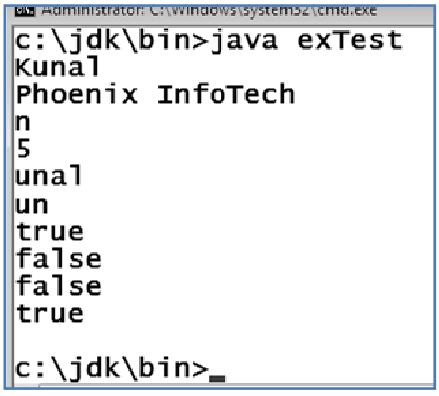

**Program 11 :** Develop a program for implementation of different functions of string class. Part - I.

**Solution :**

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

```
String str = "Kunal";
String str1 = "Phoenix InfoTech";
System.out.println(str);
System.out.println(str1);
System.out.println(str.charAt(2));
System.out.println(str.length());
System.out.println(str.substring(1));
System.out.println(str.substring(1,3));
System.out.println(str.startsWith("Ku"));
System.out.println(str.endsWith("abc"));
System.out.println(str.equals("KUNAL"));
System.out.println(str.equalsIgnoreCase("KUNAL"));
}
}
```

**Output**



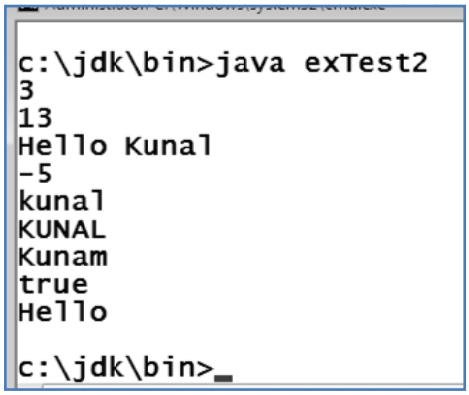

**Program 12 :** Develop a program for implementation of different functions of string class. Part - II.

**Solution :**

```
class exTest2
{
public static void main(String args[])
{
String str = "Kunal";
String str1 = "Phoenix InfoTech";
System.out.println(str1.indexOf('e'));
System.out.println(str1.lastIndexOf('e'));
```

```
System.out.println("Hello ".concat(str));
System.out.println(str.compareTo(str1));
System.out.println(str.toLowerCase());
System.out.println(str.toUpperCase());
System.out.println(str.replace('l','m'));
System.out.println(str1.contains("Info"));
System.out.println(" Hello ".trim());
}
}
```

**Output**




**Program 13 :** Develop a program for implementation of Arrays in Java.

**Solution : One dimensional array**

```
class Exarray
{
   public static void main (String[] args)
   {
    int[] myArray;
```

Declares an Array of integers

```
    myArray = new int[5];
```

Allocating memory for 5 integers.

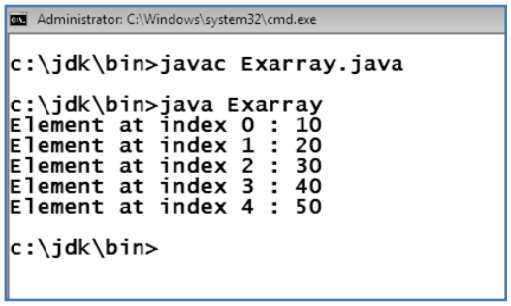
```
    myArray[0] = 10;
    myArray[1] = 20;
```

Initialize the first elements of the array

```
    myArray[2] = 30;
    myArray[3] = 40;
    myArray[4] = 50;
    for (int i = 0; i < myArray.length; i++)
      System.out.println("Element at index " + i +
                         " : "+ myArray[i]);
  }
}
```

Accessing the elements of the specified array

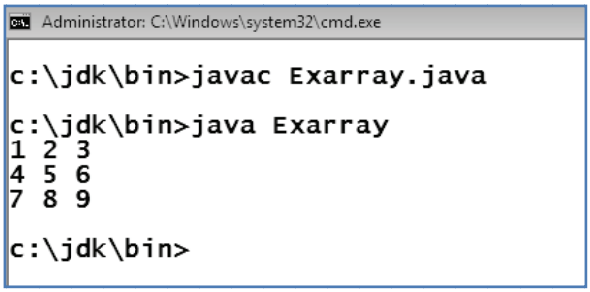
**Output**




**Two dimensional array**

```
class Exarray
{
   public static void main (String[] args)
   {
   int myarr[][]={{1,2,3},{4,5,6},{7,8,9}};
for(int i=0;i<3;i++)
{
 for(int j=0;j<3;j++)
 {
  System.out.print(myarr[i][j]+" ");
 }
 System.out.println();
}
   }
}
```

**Output**

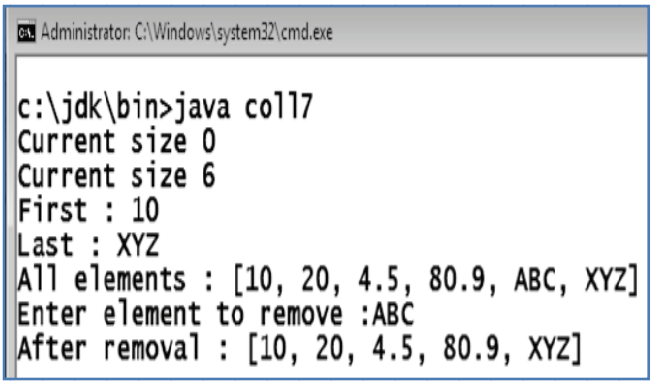



**Program 14 :** Develop a program for implementation of Vectors in Java.

**Solution :**

```
import java.util.*;
class coll7
{
public static void main(String args[])
{
Scanner sc = new Scanner(System.in);
Vector v = new Vector(5);
System.out.println("Current size "+v.size());

v.addElement(new Integer(10));
v.addElement(new Integer(20));
v.addElement(new Float(4.5));
v.addElement(new Float(80.90));
v.addElement("ABC");
v.addElement("XYZ");

System.out.println("Current size "+v.size());
System.out.println("First : "+v.firstElement());
System.out.println("Last : "+v.lastElement());
System.out.println("All elements : "+v);

System.out.print("Enter element to remove :");
String ele = sc.next();
v.removeElement(ele);
System.out.println("After removal : "+v);
}
}
```

**Output**




**Program 15 :** Develop a program for implementation of Wrapper Class to convert primitive into object.

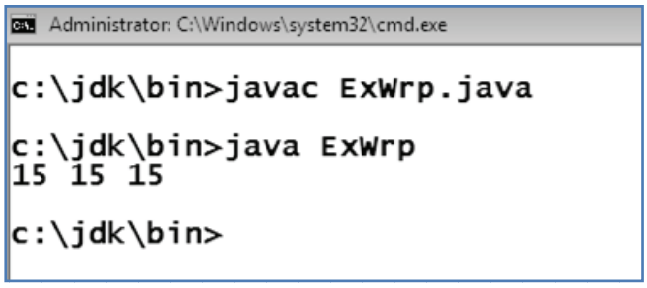
**Solution :**

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

int x=15;
Integer y = Integer.valueOf(x);
Integer z = x;

System.out.println(x+" "+y+" "+z);
}
}
```

Converting int to integer

autoboxing, now compiler will write Integer.valueOf(x) internally

**Output**

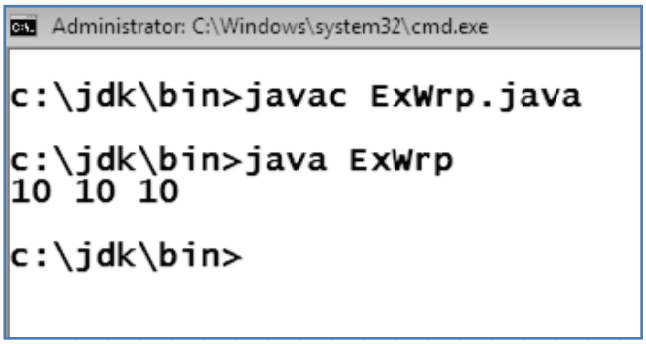


**Program 16 :** Develop a program for implementation of Wrapper Class to convert object into primitive.

**Solution :**

```
public class ExWrp
{
public static void main(String args[]){
Integer x=new Integer(10);

int y=x.intValue();   <-- Converting integer to int

int z=x;   <-- Unboxing, now compiler will write x.intValue internally.

System.out.println(x+" "+y+" "+z);
}
}
```

**Output**




**Program 17 :** Develop program which implements the concept of overriding.

**Solution :**

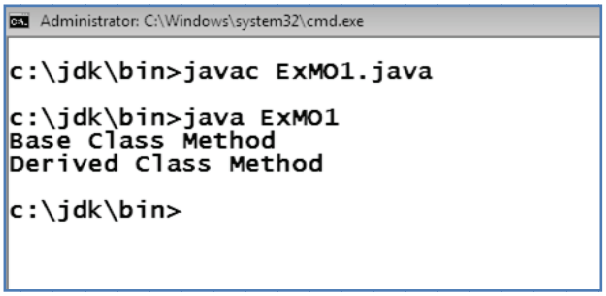
```
class base
{
      void display()
      {
      System.out.println("Base Class Method");
      }
}
class derived extends base
{
```

```
    void display()
    {
    super.display();
    System.out.println("Derived Class Method");
    }
}

class ExMO1
{
    public static void main(String args[])
    {
     derived obj = new derived();
     obj.display();
    }
}
```

Call to base method

Calls derived method

**Output**




**Program 18 :** Develop a program for implementation of single and Multilevel inheritance.

**Solution : Single Inheritance**

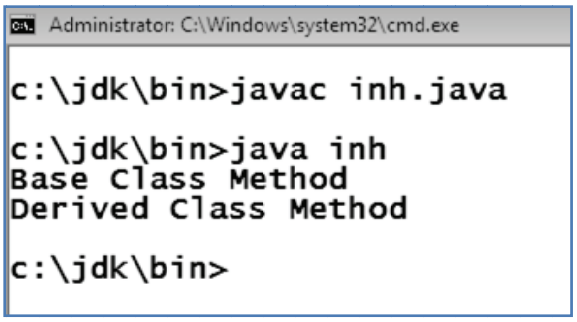
```
class A
{
    void displayBase()
    {
    System.out.println("Base Class Method");
    }
}
```

```
class B extends A
{
        void displayDerived()
        {
        System.out.println("Derived Class Method");
        }
}
class inh
{
        public static void main(String args[])
        {
         B obj = new B();
         obj.displayBase();
         obj.displayDerived();
        }
}
```

Extends keyword is used to inherit a class

Base class method can be accessed using derived class object

**Output**



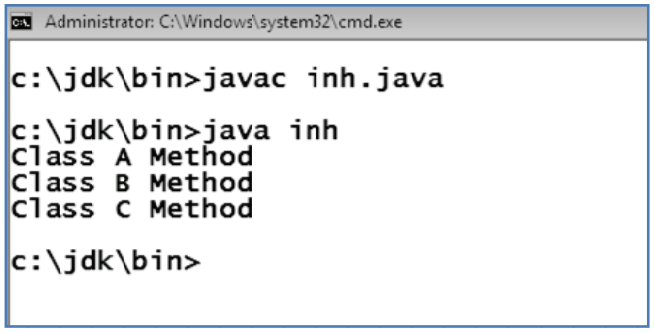

**Multilevel Inheritance**

```
class A
{
        void display1()
        {
        System.out.println("Class A Method");
        }
}
```

```
class B extends A
{
      void display2()
      {
      System.out.println("Class B Method");
      }
}
class C extends B
{
      void display3()
      {
      System.out.println("Class C Method");
      }
}
class inh
{
      public static void main(String args[])
      {
       C obj = new C();
       obj.display1();
       obj.display2();
       obj.display3();
      }
}
```

Object of class C can access members of both class A and B

**Output**



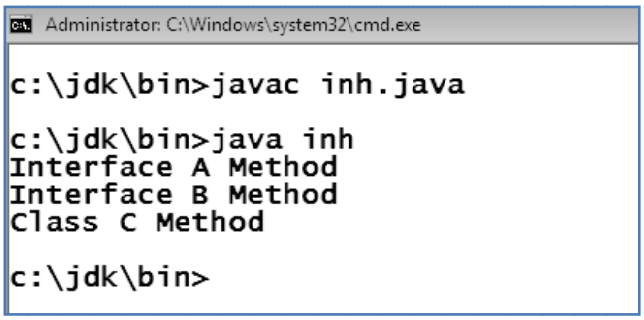
**Program 19 :** Develop a program for implementation of multiple inheritances.

**Solution : Multiple Inheritances**

```
interface A
{
      void display1();
}
interface B
{
      void display2();
}
class C implements A,B

{
      public void display1()
      {
      System.out.println("Interface A Method");
      }
      public void display2()
      {
      System.out.println("Interface B Method");
      }
      void display3()
      {
      System.out.println("Class C Method");
      }
}
class inh
{
      public static void main(String args[])
      {
       C obj = new C();
       obj.display1();
       obj.display2();
       obj.display3();
      }
}
```

Implements keyword is used to inherit class from interface

**Output**




**Program 20 :** Develop a program to import different classes in package.

**Solution :**

```
import mypack.*;
class thirdcl
{
public static void main(String args[])
{
firstcl obj1 = new firstcl();
secondcl obj2 = new secondcl();
obj1.add(10,5);
obj2.sub(10,5);
}
}
```

**Program 21 :** Develop a program for implementation of multhreading operation Part - I.

**Solution :**

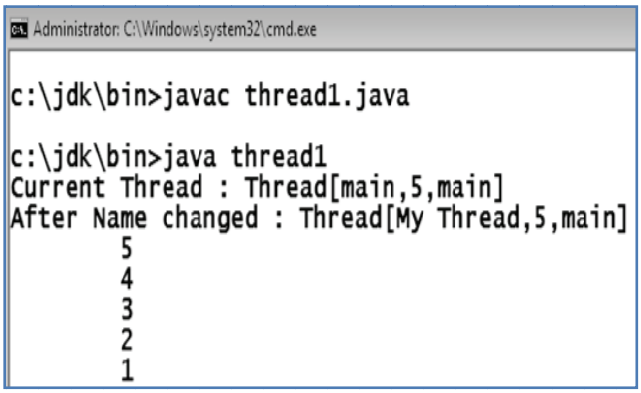
```
class thread1
{
public static void main(String args[])
{
Thread t = Thread.currentThread();
System.out.println("Current Thread : "+t);
t.setName("My Thread");
```

t prints : name of thread(default-main), priority (default-5), group of thread(default-main)

```
System.out.println("After Name changed : "+t);
try
{
for(int n=5;n>0;n--)
{
System.out.println("\t"+n);
Thread.sleep(1000);
}
}
catch(InterruptedException e)
{
System.out.println("Thread interrupted");
}
}
}
```

Sleep() halts the execution for 1 second

**Output**



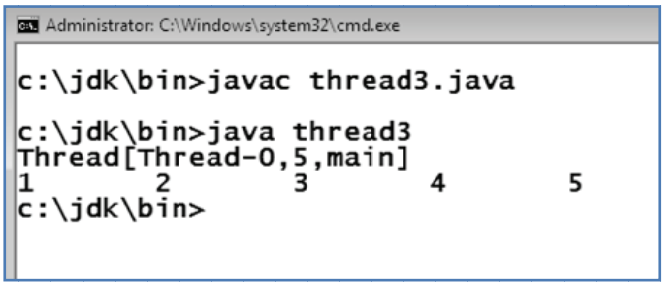

**Program 22 :** Develop a program for implementation of multithreading operation Part - II.

**Solution :**

```
class MyThread extends Thread
{
public void run()
{
for(int i=1;i<6;i++)
System.out.print(i+"\t");
```

```
}
}
class thread3
{
public static void main(String args[])
{
MyThread t = new MyThread();
System.out.println(t);
t.start();
}
}
```

**Output**




**Program 23 :** Develop a program for implementation of try, catch block Part - I.

**Solution :**

```
class Exception1
{
 public static void main (String args[])
  {
    int a=0,b=0,res;
   try
   {
      a=Integer.parseInt(args[0]);
      b=Integer.parseInt(args[1]);
     res = a/b;
     System.out.println("Division is "+res);
```

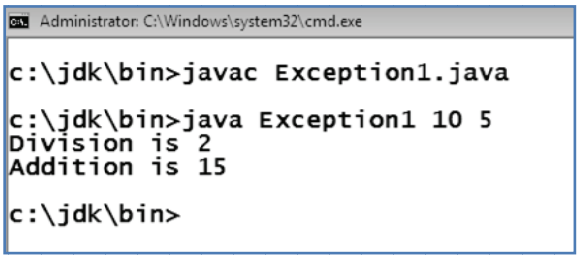
The code in which exception may occur

```
    }
   catch(ArithmeticException e)
    {
System.out.println("Cannot divide by zero");
    }
res = a + b;
System.out.println("Addition is "+res);
  }
}
```

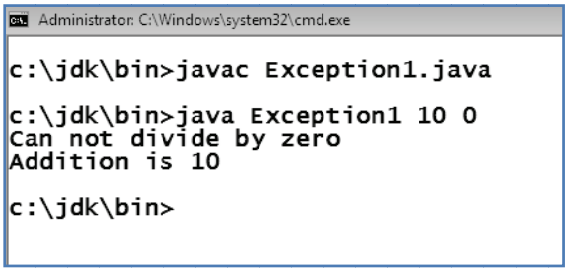
Execute if exception occurs and caught

Executes even if exception occurs

If user input is correct, then it will produce proper output :




And if user enters second number as 0 :



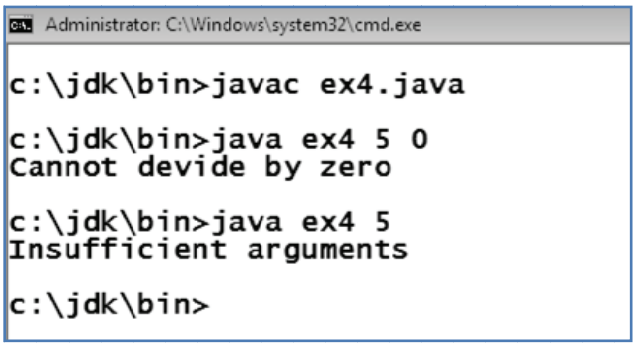

**Program 24 :** Develop a program for implementation of try, catch block. Part - II.

**Solution : nested try catch**

```
class ex4
{
 public static void main (String args[])
  {
```

```
    int a,b,div;
try                                          Outer try
{
    try                                      Inner try
    {
       a=Integer.parseInt(args[0]);
       b=Integer.parseInt(args[1]);
      div = a/b;
      System.out.println("Division is "+div);
    }
   catch(ArithmeticException e)              Inner catch
   {
    System.out.println("Cannot devide by zero");
   }
}
 catch(ArrayIndexOutOfBoundsException e)     Outer catch
   {
    System.out.println("Insufficient arguments");
   }
  }
}
```

**Output**




**Multiple Catch Blocks**

```
class Exception2
{
 public static void main (String args[])
```

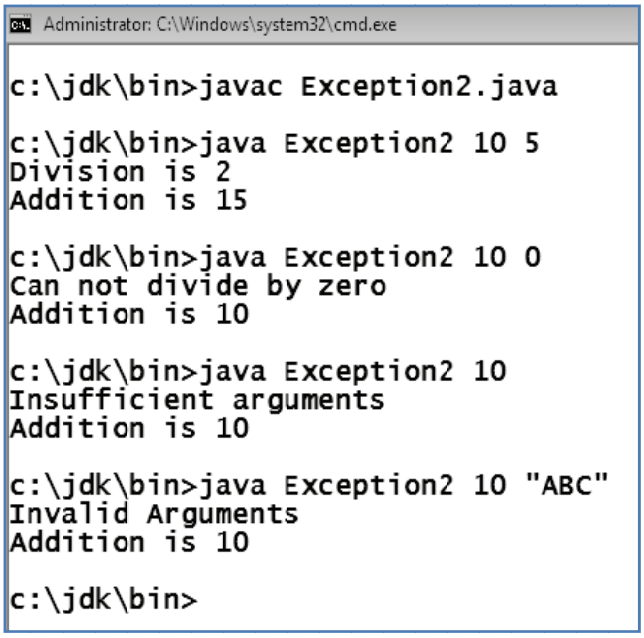
```
  {
    int a=0,b=0,res;
   try
   {
      a=Integer.parseInt(args[0]);
      b=Integer.parseInt(args[1]);
     res = a/b;
     System.out.println("Division is "+res);
   }
catch(ArithmeticException e)
   {
    System.out.println("Cannot divide by zero");
   }
catch(ArrayIndexOutOfBoundsException e)
   {
    System.out.println("Insufficient arguments");
   }
 catch(NumberFormatException e)
   {
    System.out.println("Invalid Arguments");
   }
     res = a + b;
     System.out.println("Addition is "+res);
  }
}
```

Exception if second argument is 0

Exception if all arguments are not provided

Exception if arguments are rather than integer type

**Output**




**Program 25 :** Develop a program for implementation of try, catch and finally block.

**Solution :**

```
class Exception2
{
 public static void main (String args[])
  {
    int a=0,b=0,res;
   try
   {
      a=Integer.parseInt(args[0]);
      b=Integer.parseInt(args[1]);
     res = a/b;
     System.out.println("Division is "+res);
   }
```

```
 catch(ArithmeticException e)
    {
     System.out.println("Can not divide by zero");
    }
 catch(ArrayIndexOutOfBoundsException e)
    {
     System.out.println("Insufficient arguments");
    }
finally

{

res = a + b;

System.out.println("Addition is "+res);

}

  }

}
```

Executes even if exception is not handled

**Output**

```
c:\jdk\bin>javac Exception2.java

c:\jdk\bin>java Exception2 10 5
Division is 2
Addition is 15

c:\jdk\bin>java Exception2 10 "ABC"
Addition is 10
Exception in thread "main" java.lang.NumberForma
BC"
        at java.lang.NumberFormatException.forI
java:48)
        at java.lang.Integer.parseInt(Integer.ja
        at java.lang.Integer.parseInt(Integer.ja
        at Exception2.main(Exception2.java:9)
```

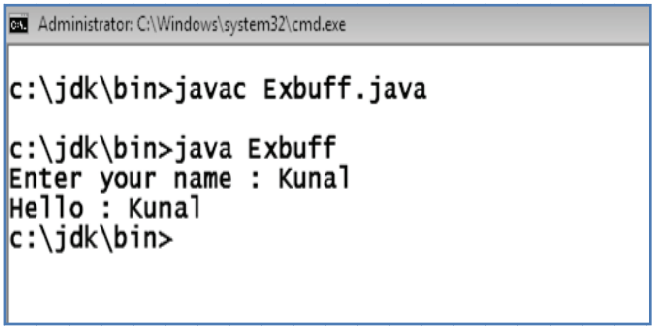
**Program 26 :** Develop programs for implementation of throw, throws clause. Part - I.

**Solution :**

```
import java.io.*;
class Exbuff
{
public static void main(String args[]) throws IOException
{
String nm;
BufferedReader br = new BufferedReader(new
InputStreamReader(System.in));
System.out.print("Enter your name : ");
nm = br.readLine();
System.out.print("Hello : "+nm);
}
}
```

Creating object for BufferedReader

**Output**




**Program 27 :** Develop programs for implementation of throw, throws clause. Part – II.

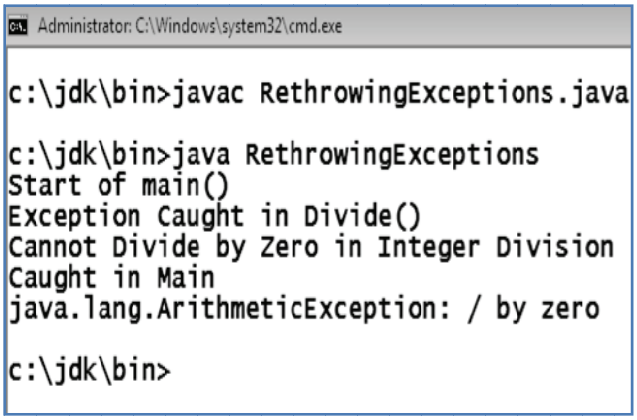
**Solution :**

```
class test
{
    static void divide()
  {
    int x,y,z;
    try
```

```
      {
        x = 5 ;
        y = 0 ;
        z = x/y ;
        System.out.println(x + "/"+ y +" = " + z);
       }
       catch(ArithmeticException e)
       {
        System.out.println("Exception Caught in Divide()");
        System.out.println("Cannot Divide by Zero in Integer Division");
        throw e
       }
    }
  }
public class RethrowingExceptions
{
   public static void main(String[] args)
    {
          System.out.println("Start of main()");
          try
          {
              test.divide();
          }
        catch(ArithmeticException e)
          {
             System.out.println("Caught in Main");
             System.out.println(e);
          }
     }
}
```

Re-throwing the exception

**Output**




**Program 28 :** Develop minimum two basic Applets. Display output with applet viewer and browser.

(a) Develop a program on basic applet.

(b) Develop a program using control loops in applets.

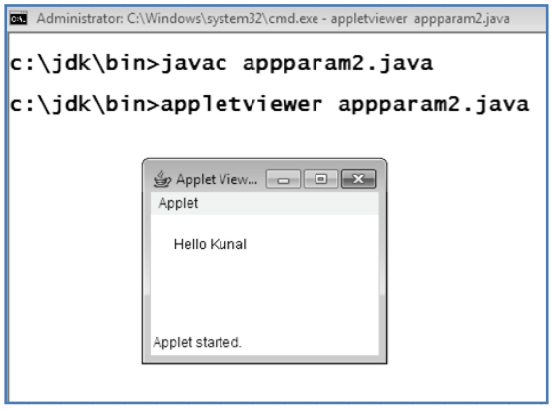
**Solution :**

**(a) Develop a program on basic applet**

```
import java.applet.*;
import java.awt.*;
/*
<applet code = "appparam2" width=200 height=100>
<param name = "pname" value="Kunal" />
</applet>
*/
public class appparam2 extends Applet
{
String name,msg;
public void init()
{
name = getParameter("pname");
}
public void paint(Graphics g)
{
```

```
msg = "Hello " + name;
g.drawString(msg,20,30);
}
}
```

**Output**

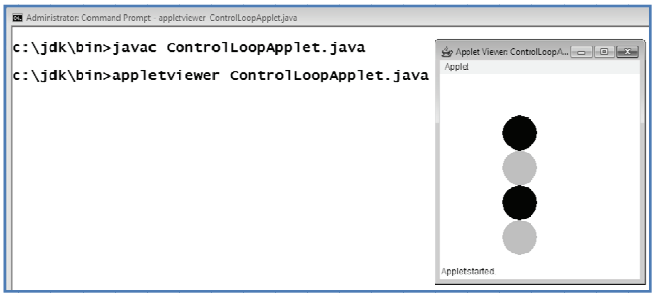



**(b) Develop a program using control loops in applets**

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

   public class ControlLoopApplet extends Applet
   {
    public void paint(Graphics g)
    {
     for(int i=1;i<=4;i++)
     {
      if(i%2==0)
      {
       g.fillOval(90,i*50+10,50,50);
       g.setColor(Color.black);
      }
      else
      {
       g.fillOval(90,i*50+10,50,50);
       g.setColor(Color.pink);
```

```
    }
   }
  }
 }
/* <applet code=ControlLoopApplet width=300 height=300>
  </applet> */
```

**Output**




**Program 29 :** Write a program to create animated shape using graphics and applets, You may use following shapes :

(a) Lines and Rectangles.

(b) Circles and Ellipses.

(c) Arcs

(d) Polygons with fillpolygon method.

**Solution :**

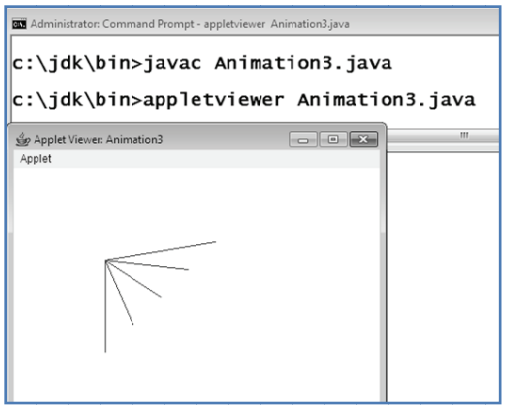
### (a) Lines and Rectangles

**1. Lines**

```
import java.applet.Applet;
import java.awt.Color;
import java.awt.Graphics;

public class Animation3 extends Applet {

   public void paint(Graphics g) {

      int a=100,b=100,c=100,d=200;
         g.setColor(Color.red);
      for(int i=0;i<5;i++)
```

```
      {
         try{
            Thread.sleep(1000);
         }catch(InterruptedException ex){}
g.drawLine(a,b,c,d);

         c+=30;
         d-=30;
      }
   }
}
/*<applet code=Animation3 width=400 height=400>
    </applet> */
```

**Output**

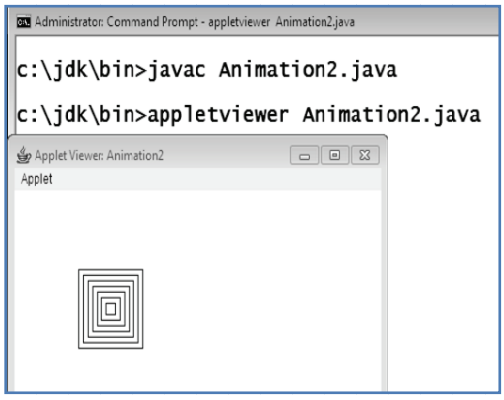



**2. Rectangles**

```
import java.applet.Applet;
import java.awt.Color;
import java.awt.Graphics;

public class Animation2 extends Applet {
   public void paint(Graphics g) {

      int a=100,b=100,c=10,d=10;
```

```
        g.setColor(Color.blue);
      for(int i=0;i<10;i++)
      {
        try{
          Thread.sleep(1000);
        }catch(InterruptedException ex){}
        g.drawRect(a, b, c, d);
        a-=5;
        b-=5;
        c+=10;
        d+=10;
      }
    }
}
/*<applet code=Animation2 width=400 height=400>
    </applet> */
```

**Output**



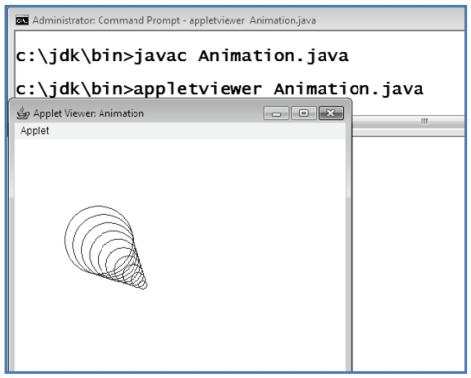

### (b) Circle and Ellipse

#### 1. Circle

```
import java.applet.Applet;
import java.awt.Color;
import java.awt.Graphics;
public class Animation extends Applet {
```

```
  public void paint(Graphics g) {
     int a=150,b=170,c=10,d=10;
     g.setColor(Color.red);
     for(int i=0;i<10;i++)
     {
        try{
           Thread.sleep(1000);
        }catch(InterruptedException ex){}
        g.drawOval(a, b, c, d);
        a-=10;
        b-=10;
        c+=8;
        d+=8;
     }
  }
}
/*<applet code=Animation width=400 height=400>
   </applet> */
```

**Output**

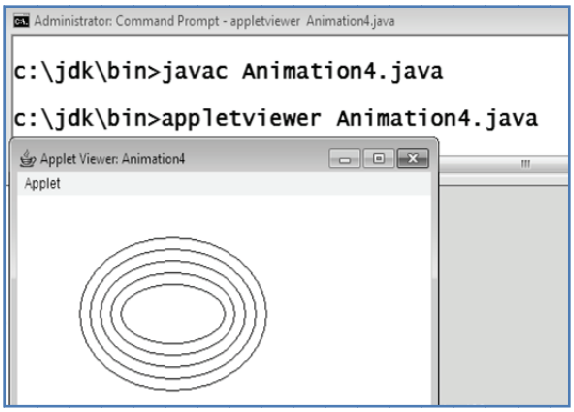



**2. Ellipse**

```
import java.applet.Applet;
import java.awt.Color;
import java.awt.Graphics;
public class Animation4 extends Applet {
```

```
  public void paint(Graphics g) {
     int a=100,b=75,c=100,d=50;
        g.setColor(Color.blue);
     for(int i=0;i<5;i++)
     {
        try{
           Thread.sleep(1000);
        }catch(InterruptedException ex){}
         g.drawOval(a,b,c,d);
        a=a-10;
        b=b-10;
        c=c+20;
        d=d+20;
     }
  }
}
/*<applet code=Animation4 width=400 height=400>
    </applet> */
```

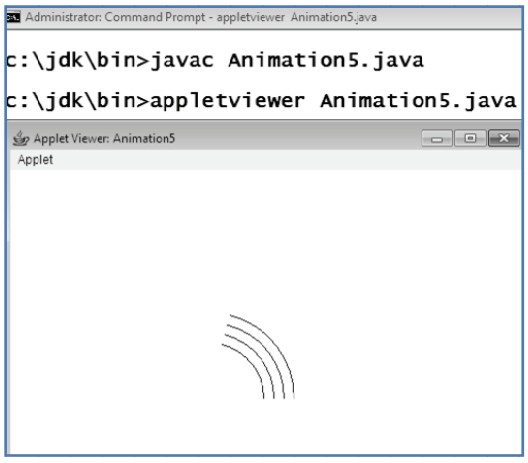
**Output**



**(c) Arcs**

```
import java.applet.Applet;
import java.awt.Color;
import java.awt.Graphics;
public class Animation5 extends Applet {
   public void paint(Graphics g) {
      int a=110,b=140,c=170,d=170,e=0,f=75;
         g.setColor(Color.blue);
      for(int i=0;i<5;i++)
      {
         try{
            Thread.sleep(1000);
         }catch(InterruptedException ex){}
           g.drawArc(a, b, c, d, e,f);
          a=a+10;
         b=b+10;
         c=c-20;
         d=d-20;
      }
   }
}
/*<applet code=Animation5 width=400 height=400>
    </applet> */
```

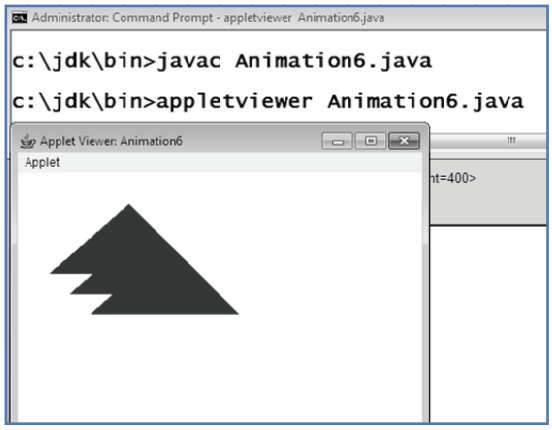
**Output**



**(d) Polygons with fillPolygon method**

```
import java.applet.Applet;
import java.awt.Color;
import java.awt.Graphics;
public class Animation6 extends Applet {
   public void paint(Graphics g) {
      int a1=110,b1=180,c1=30,d1=110,a2=30,b2=100,c2=100,d2=30;
      for(int i=0;i<5;i++)
      {
         try{
            Thread.sleep(1000);
         }catch(InterruptedException ex){}
         int xPoints[] = {a1, b1,c1, d1};
         int yPoints[] = {a2, b2, c2, d2};
 g.setColor(Color.RED);
 g.fillPolygon(xPoints, yPoints, 4);
          a1=a1+20;
         b1=b1+20;
        a2=a2+20;
         b2=b2+20;
        c1=c1+20;
         d1=d1+20;
        c2=c2+20;
        d2=d2+20;
      }
   }
}
/*<applet code=Animation6 width=400 height=400>
    </applet> */
```

**Output**




**Program 30 :** Develop a program to draw following shapes, graphics and applets.

(a) Cone (b) Cylinders (c) Cube

(d) Square inside a circle

(e) Circle inside a square.

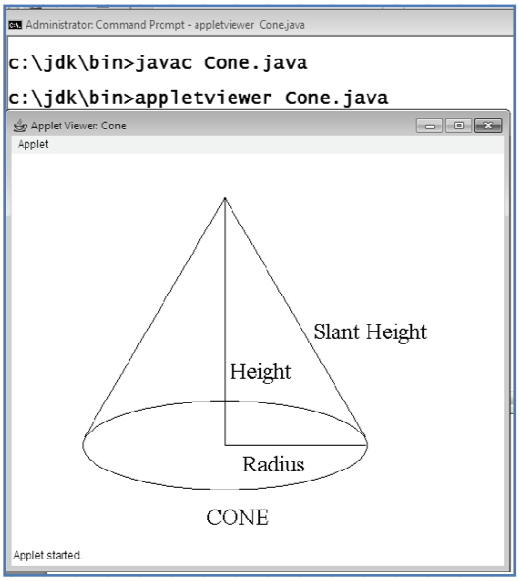
**Solution :**

**(a) Cone**

```
import java.applet.Applet;
import java.awt.*;
public class Cone extends Applet
{
   public void paint(Graphics g)
   {
      Font font = new Font("Serif", Font.PLAIN, 26);
      g.setFont(font);
     g.drawOval(80,280,320,100);
      g.drawLine(240,50,82,320);
      g.drawLine(240,50,398,320);
      g.drawLine(240,330,398,330);
```

```
        g.drawLine(240,50,240,330);
        g.drawString("Radius",260,360);
        g.drawString("Height",246,255);
        g.drawString("Slant Height",340,210);
        g.drawString("CONE",220,420);
    }
}
/*<applet code=Cone width=700 height=500>
    </applet> */
```

**Output**

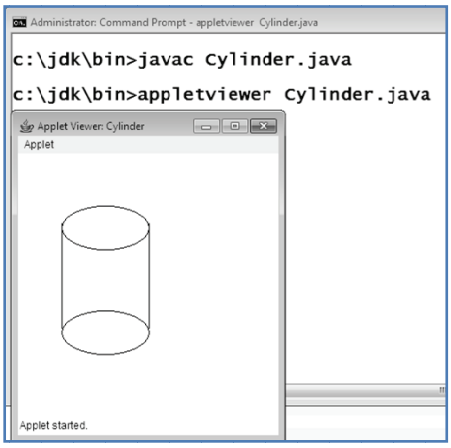



**(b) Cylinder**

```
import java.awt.*;
import java.applet.Applet;
public class Cylinder extends Applet
{
  public void paint(Graphics g)
  {
```

```
    g.drawOval(50,60,100,50);
    g.drawLine(50,80,50,200);
    g.drawLine(150,80,150,200);
    g.drawOval(50,180,100,50);
}
/*<applet code=Cylinder width=300 height=300>
     </applet> */
```

**Output**

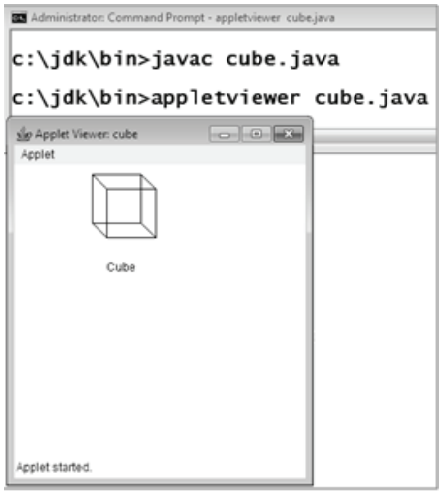



**(c) Cube**

```
import java.awt.*;
import java.applet.*;
public class cube extends Applet
{
public void paint(Graphics g)
{
g.drawString("Cube",95,110);
g.drawRect(80,10,50,50);
g.drawRect(95,25,50,50);
g.drawLine(80,10,95,25);
g.drawLine(130,10,145,25);
g.drawLine(80,60,95,75);
```

```
g.drawLine(130,60,145,75);
}
}
/*<applet code=cube width=300 height=300>
    </applet> */
```

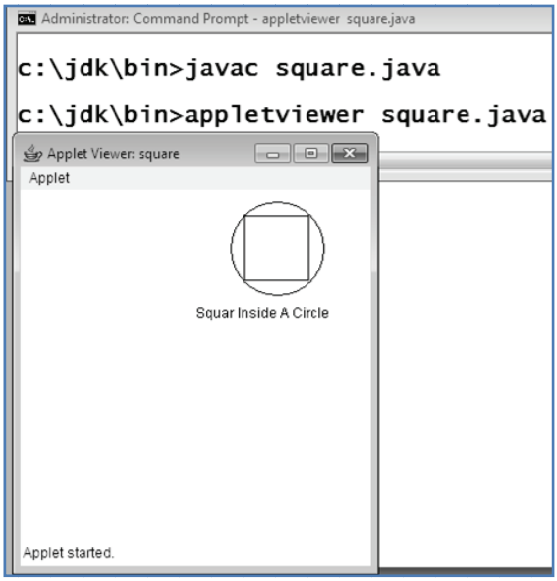
**Output**




**(d) Square inside a circle**

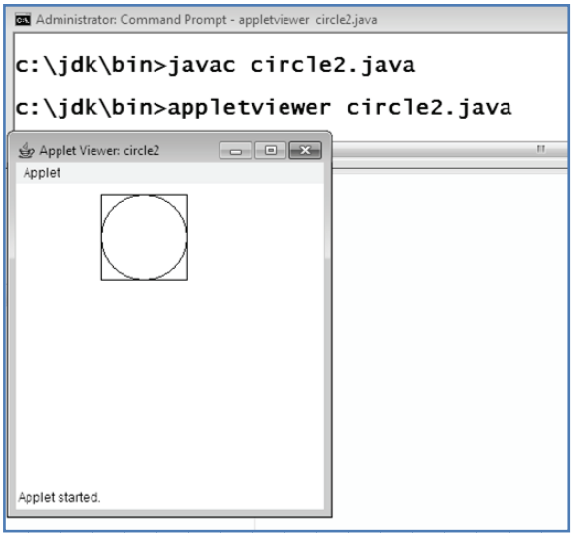
```
import java.awt.*;
import java.applet.*;
public class square extends Applet
{
public void paint(Graphics g)
{
g.drawString("Squar Inside A Circle",150,110);
g.drawOval(180,10,80,80);
g.drawRect(192,22,55,55);
}
}
/*<applet code=square width=300 height=300>
    </applet> */
```

**Output**




**(e) Circle inside a square**

```
import java.awt.*;
import java.applet.*;
public class circle2 extends Applet
{
public void paint(Graphics g)
{
g.drawString("Circle Inside a Squar",290,110);
g.drawRect(80,10,80,80);
g.drawOval(80,10,80,80);
}
}
/*<applet code=circle2 width=300 height=300>
    </applet> */
```




**Program 31 :** Develop a program for implementation of I/O stream classes.

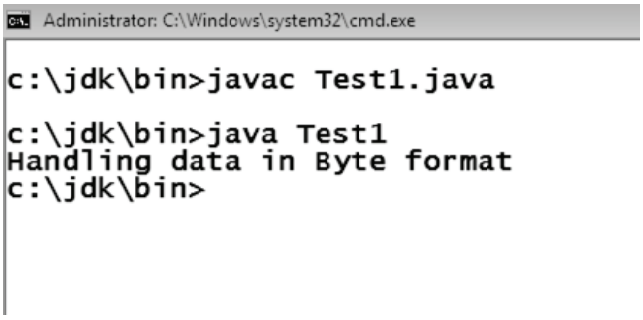
**Solution :**

**Input Stream**

```
import java.io.ByteArrayInputStream;

public class Test1
{
        public static void main(String[] args) {
                String msg = "Handling data in Byte format";

                ByteArrayInputStream bs=new
ByteArrayInputStream(msg.getBytes());
                int ch;
                while((ch=bs.read())!=-1)
                        {
                        System.out.print((char)ch);
                }
        }
}
```

Unless end of Data

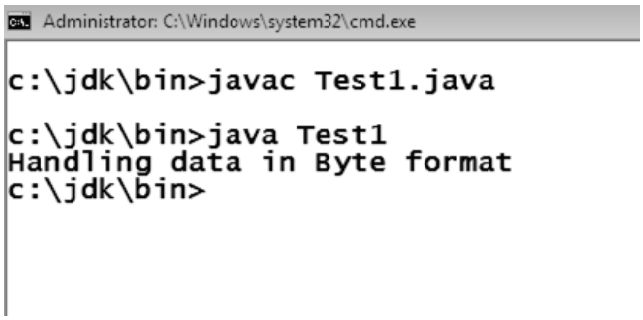
**Output**




**OutputStream**

```
import java.io.ByteArrayOutputStream;
import java.io.IOException;

public class Test1
{
	public static void main(String[] args) throws IOException {
		String s1 = "Handling Data ";
		String s2 = "in Byte Format";
		ByteArrayOutputStream bs = new ByteArrayOutputStream();
		bs.write(s1.getBytes());
		bs.write(s2.getBytes());
		System.out.println(bs.toString());
	}
}
```

**Output**



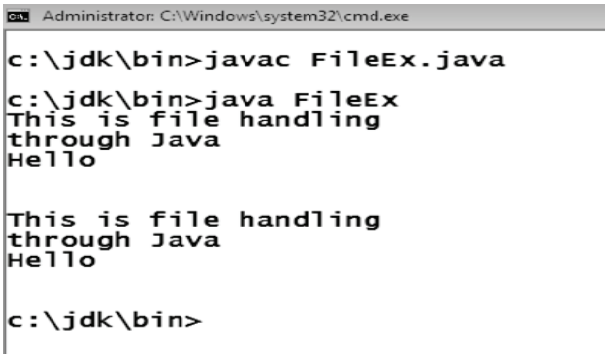
**Program 32 :** Develop a program for implementation of file stream classes.

**Solution :**

**Example on FileInputStream**

```
import java.io.*;
public class FileEx{
        public static void main(String[] args) throws IOException {
                File file = new File("p1.txt");
                FileInputStream fis = new FileInputStream(file);
                //Read file using read()
                int i=0;
                 while((i=fis.read())!=-1){
                  System.out.print((char)i);
                 }
                System.out.println("\n\n");
                fis.close();
//Read file using read(byte[] b)
                fis = new FileInputStream(file);
                byte[] b = new byte[256];
                while((i=fis.read(b))!=-1){
                  System.out.print(new String(b));
                 }
                fis.close();
        }
}
```

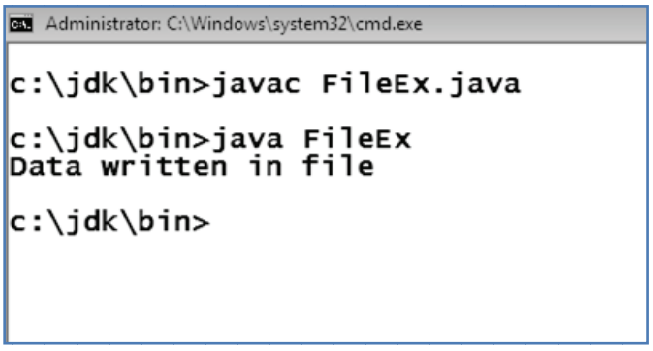
**Output**



**Example on FileOutputStream**

```
import java.io.*;

public class FileEx {
  public static void main(String[] args) throws IOException
      {
         File file = new File("p1.txt");
         FileOutputStream fos = new FileOutputStream(file);
         //write file using write(byte[] b)
         fos.write("Welcome To My Web ".getBytes());
         //write file using write(byte[] b, int off, int len)
         fos.write("Welcome To My Web".getBytes(),6,6);
         //write file using write(int b)
         byte[] b = " Welcome To My Web!".getBytes();
         for(int i=0;i<b.length;i++){
            fos.write(b[i]);
         }
         fos.close();
         System.out.println("Data written in file");
  }
}
```

**Output**

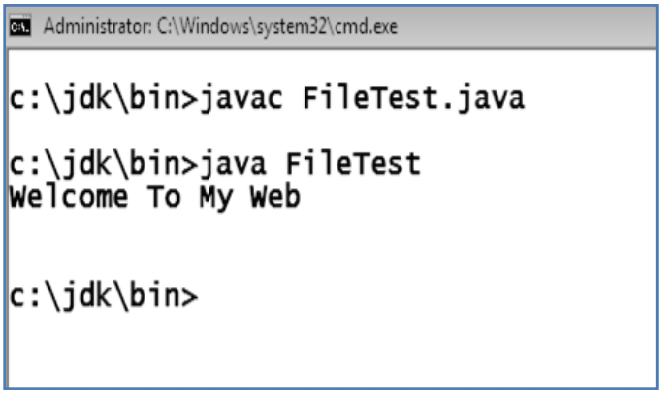


**Example on BufferedInputStream**

```
import java.io.*;
public class FileTest {
        public static void main(String[] args) {
                File file = new File("p1.txt");
                FileInputStream fn = null;
                BufferedInputStream bs = null;
                try {
                        fn = new FileInputStream(file);
                        bs = new BufferedInputStream(fn);
                        // Create buffer
                        byte[] buffer = new byte[1024];
                        int br = 0;
                        while ((br = bs.read(buffer)) != -1) {
                        System.out.println(new String(buffer, 0, br));
                        }
                } catch (IOException e) {
                        e.printStackTrace();
}
}
}
```

Message if error occurs

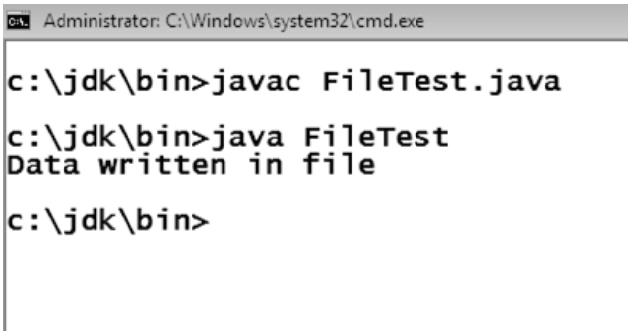
**Output**



**Example on BufferedOutputStream**

```
import java.io.*;
public class FileTest {
	public static void main(String[] args) {
		File file=new File("p2.txt");
		FileOutputStream fs=null;
		BufferedOutputStream bs=null;
		try {
			fs=new FileOutputStream(file);
			bs=new BufferedOutputStream(fs);
			bs.write("Writing data in a file".getBytes());
			bs.write(" using BufferedOutputStream".getBytes());
			bs.flush();
			System.out.println("Data written in file");

		} catch (IOException e) {
			e.printStackTrace();
		}
}
}
```

**Output**



# Appendix A

## Solved University Question Paper of Summer 2019

## Summer 2019

**Q. 1(a)** List any eight features of Java.

(Ans. : Refer section 1.2) **(Chap. 1, 2 Marks)**

**Q. 1(b)** State use of finalize( ) method with its syntax.OO.

(Ans. : Refer section 2.8) **(Chap. 2, 2 Marks)**

**Q. 1(c)** Name the wrapper class methods for the following :

(i) To convert string objects to primitive int.

(ii) To convert primitive int to string objects.

**(Chap. 2, 2 Marks)**

**Ans. :**

(i) To convert string objects to primitive int - Integer.parseInt()

(ii) To convert primitive int to string objects - Integer.toString()

**Q. 1(d)** List the types of inheritances in Java.

(Ans. : Refer section 3.2) **(Chap. 3, 2 Marks)**

**Q. 1(e)** Write the syntax of try-catch-finally blocks.

**(Chap. 4, 2 Marks)**

**Ans. :**

```
try
{
Statements;
}
catch(…)
{
Statements;
}
finally
{
Statements;
}
```

**Q. 1(f)** Give the syntax of < param > tag to pass parameters to an applet. (Ans. : Refer section 5.8) **(Chap. 5, 2 Marks)**

**Q. 1(g)** Define stream class. List its types.
(Ans. : Refer sections 6.1 and 6.2) **(Chap. 6, 2 Marks)**

**Q. 2(a)** Explain the concept of platform independence and portability with respect to Java language.
(Ans. : Refer section 1.2) **(Chap. 1, 4 Marks)**

**Q. 2(b)** Explain the types of constructors in Java with suitable example. (Ans. : Refer section 2.4) **(Chap. 2, 4 Marks)**

**Q. 2(c)** Explain the two ways of creating threads in Java. (Ans. : Refer section 4.15) **(Chap. 4, 4 Marks)**

**Q. 2(d)** Distinguish between Input stream class and output stream class. (Ans. : Refer section 6.2) **(Chap. 6, 4 Marks)**

**Q. 3(a)** Define a class student with int id and string name as data members and a method void SetData ( ). Accept and display the data for five students. **(Chap. 2, 4 Marks)**

**Ans. :**

```
import java.util.*;
class student
{
int id;
String name;
Scanner sc = new Scanner(System.in);
void setdata()
{
System.out.print("\nEnter id and name : ");
id = sc.nextInt();
name = sc.next();
}
void disp()
{
System.out.println(id  + "\t" + name);
}
}
```

```
class ExTest1
{
public static void main(String args[])
{
int i;
student[] obj;
obj = new student[5];
for(i=0;i<5;i++)
{
obj[i] = new student();
}
for(i=0;i<5;i++)
{
obj[i].setdata();
}
System.out.println("\nID \tName");
for(i=0;i<5;i++)
{
obj[i].disp();
}
}
}
```

**Output**

```
Command Prompt

c:\jdk\bin>javac ExTest1.java

c:\jdk\bin>java ExTest1

Enter id and name : 11 abc

Enter id and name : 12 pqr

Enter id and name : 13 xyz

Enter id and name : 14 rty

Enter id and name : 15 psk

ID      Name
11      abc
12      pqr
13      xyz
14      rty
15      psk

c:\jdk\bin>
```

**Q. 3(b)** *Explain dynamic method dispatch in Java with suitable example. (Ans. : Refer section 3.5.3)* **(Chap. 3, 4 Marks)**

**Q. 3(c)** *Describe the use of following methods :*

*(i) Drawoval ( ) (ii) getFont ( )*

*(iii) drawRect ( ) (iv) getFamily ( ).*

*(Ans. : Refer section 5.13)* **(Chap. 5, 4 Marks)**

**Q. 3(d)** *Write a program to count number of words from a text file using stream classes.* **(Chap. 6, 4 Marks)**

**Ans. :**

```
import java.io.FileReader;
import java.io.BufferedReader;
import java.util.StringTokenizer;
class ExTest {
public static void main(String[] args) throws Exception {
FileReader fr = new FileReader(args[0]);
BufferedReader br = new BufferedReader(fr);
String line = "", str = "";
int now = 0;
while ((line = br.readLine()) != null) {
str += line + " ";
}
StringTokenizer st = new StringTokenizer(str);
while (st.hasMoreTokens()) {
String s = st.nextToken();
now++;
}
System.out.println("Number of words  " + now);
}
}
```

**Output**

```
Command Prompt

c:\jdk\bin>javac ExTest.java

c:\jdk\bin>java ExTest "p1.txt"
Number of words  7

c:\jdk\bin>
```

**Q. 4(a)** Describe instance Of and dot ( . ) operators in Java with suitable example.
(Ans. : Refer sections 1.14.1(G),(H)) **(Chap. 1, 4 Marks)**

**Q. 4(b)** Explain the four access specifiers in Java.
(Ans. : Refer section 2.2) **(Chap. 2, 4 Marks)**

**Q. 4(c)** Differentiate between method overloading and method overriding. (Ans. : Refer section 3.5.1) **(Chap. 3, 4 Marks)**

**Q. 4(d)** Differentiate between Java Applet and Java Application (any four points). (Ans. : Refer section 5.1.1)
**(Chap. 5, 4 Marks)**

**Q. 4(e)** Write a program to copy content of one file to another file. (Ans. : Refer Program 6.4.7) **(Chap. 6, 4 Marks)**

**Q. 5(a)** Describe the use of any methods of vector class with their syntax. (Ans. : Refer Program 2.12) **(Chap. 2, 6 Marks)**

**Q. 5(b)** Explain the concept of Dynamic method dispatch with suitable example. (Ans. : Refer section 3.5.3)
**(Chap. 3, 6 Marks)**

**Q. 5(c)** Write a program to create two threads. One thread will display the numbers from 1 to 50 (ascending order) and other thread will display numbers from 50 to 1 (descending order). **(Chap. 4, 6 Marks)**

**Ans. :**

```
class NewThread implements Runnable
{
Thread t;
NewThread()
{
t = new Thread(this,"Demo Thread");
System.out.println("Child thread : "+t);
t.start(); //call to run method
}
public void run()
```

```
{
try
{
for(int n=50;n>=1;n--)
{
System.out.println("\tChild thread : "+n);
Thread.sleep(500);
}
}
catch(InterruptedException e)
{
System.out.println("Child Thread interrupted");
}
System.out.println("Exiting child thread");
}
}
class threadTest
{
public static void main(String args[])
{
new NewThread(); //call to constructor
try
{
for(int n=1;n<=50;n++)
{
System.out.println("Main thread : "+n);
Thread.sleep(1000);
}
}
catch(InterruptedException e)
{
System.out.println("Thread interrupted");
}
System.out.println("Main thread Exiting");
}
}
```

**Output**

```
Command Prompt

c:\jdk\bin>javac threadTest.java

c:\jdk\bin>java threadTest
Child thread : Thread[Demo Thread,5,main]
Main thread : 1
        Child thread : 50
        Child thread : 49
        Child thread : 48
Main thread : 2
        Child thread : 47
        Child thread : 46
Main thread : 3
        Child thread : 45
        Child thread : 44
Main thread : 4
        Child thread : 43
        Child thread : 42
Main thread : 5
        Child thread : 41
        Child thread : 40
Main thread : 6
        Child thread : 39
Main thread : 7
        Child thread : 38
        Child thread : 37
Main thread : 8
        Child thread : 36
        Child thread : 35
```

**Q. 6(a)** *Explain the command line arguments with suitable example. (Ans. : Refer section 1.17)* **(Chap. 1, 6 Marks)**

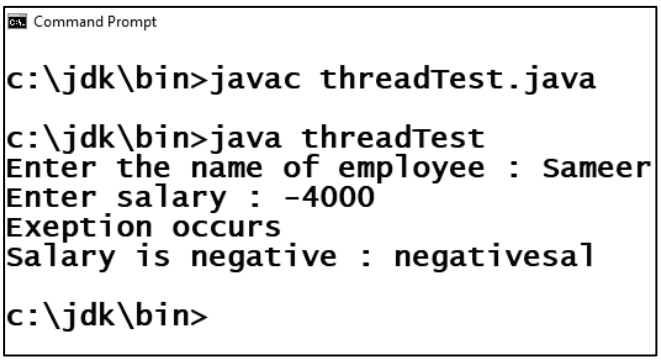
**Q. 6(b)** *Write a program to input name and salary of employee and throw user defined exception if entered salary is negative.* **(Chap. 4, 6 Marks)**

**Ans. :**

```
import java.io.*;
class negativesal extends Exception
{
public negativesal()
{
System.out.println("Exeption occurs");
}
}
class threadTest
{
public static void main(String args[]) throws IOException
```

```
{
BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
try
{
System.out.print("Enter the name of employee : ");
String name= br.readLine();
System.out.print("Enter salary : ");
int sal = Integer.parseInt(br.readLine());
if(sal<0)
throw new negativesal();
else
System.out.println("Thank you..."+name);
}
catch(negativesal e)
{
System.out.println("Salary is negative : "+e);
}
}
}
```

**Output**




**Q. 6(c)** *Describe the applet life cycle in detail.*

*(Ans. : Refer section 5.4)* **(Chap. 5, 6 Marks)**

❑❑❑