Camellia Institute of Technology

DEPT OF COMPUTER SCIENCE AND ENGG

JAVA PROGRAMMING LAB MANUAL

for

B.Tech in Computer Science & Engineering
B.Tech in Information Technology
B.Tech in Electronics and Communication Engineering

Prepare By.
Mr. Sukhendu Shekhar Mondal
Dept of CSE

Camellia Institute of Technology

JAVA PROGRAMMING

Ex.no	Name of the Exercises
1	Classes and Objects
2	Command Line Argument
3	Bitwise Operators
4	Method Overloading
5	Packages
6	Interface
7	Inheritance
8	Exception Handling
9	User Defined Exception Handling
10	Multithreading:join() and isAlive()
11	Multithreading:Inter thread communications
12	Addition of Two Numbers using Applet

Ex.No: 1

CLASSES AND OBJECTS

Aim: To write a java program to work with the creation of objects for the class with overloaded constructor and user defined methods returning a value.

Algorithm:

1. Start
2. Create an object called room.
3. Create an object with overloaded constructor and get length and breadth.
4. Create a function room where only length is passed and breadth remains constant.
5. Create a function getdata() to return the area.
6. Create another class ex1 that includes the main function.
7. Declare variable int l=0,b=0,float area.
8. Get the dynamic input using exceptional try and catch.
9. Pass the input values to the constructor.
10. Calculate the input values to the getdata().
11. Display the result.

Source code:

```
import java.io.*;
import java.io.DataInputStream;
class Room
{
    float length,breadth;
    Room(int a,int b)
   {
      length=a;
      breadth =b;
    }
    Room(int a)
   {
       length =a;
       breadth=100;
    }
```

```
    float getdata()
   {
      Return(length*breadth);
     }
}
class Ex1
{
    public static void main(String args[])
   {
     float area;
     int l=0,b=0;
     DataInputStream in=new DataInputStream(System.in);
    try
   {
      System.out.println("\n enter the value for Room length:");
      l=Integer.parseInt(in.readLine());
      System.out.println("\n enter the value for Room breadth:");
      b=Integer.parseInt(in.readLine());
   }
   Catch(Exception e)
  {}
   Room room1=new Room(l,b);
   area=room1.getdata();
   System.out.println("\n length="+room1.length);
   System.out.println("\n breadth="+room1.breadth);
   System.out.println("\n Area="+area);
   System.out.println("\n passing only length of the room with breadth=100");
    Room room2=new Room(l);
   area =room2,getdata();
   System.out.println("\n length="+room2.length);
   System.out.println("\n breadth="+room2.breadth);
   System.out.println("\n Area="+area);
  }
}
```

Sample input and output:

```
Enter the value for Room length:100
Enter the value for Room breadth:200
Length =100.0
Breadth=200.0
Area=20000.0
```

Passing only length of the room with breadth=100
Length=100.0
Breadth=100.0
Area=10000.0

Result:
Thus the java program was executed successfully and the output was verified.

ExNo: 2

COMMAND LINE ARGUMENT
Aim: To write a java program to get and sort names by command line argument.

Algorithm:
1. Create a class name Ex2 and declare the variables int n and string s[]
2. Allocate the memory of names for variable s.
3. Get total number of names to be entered in the variable n.
4. Using compareTo function sort the names in alphabetical order.
5. Use forloop to sort the name.
6. Print the sorted name list.

Source code:

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

Public class Ex2{
public static void main(String args[])throws IOException
{
  String t;
  int n=args.length;
  int i,j;
  System.out.println("\n you have entered"+n+"names to sort it in ascendingorder");
  for(i=0;i<n-1;i++)
 {
    for(j=i+1;j<n;j++)
   {
      if(args[i].compareTo(args[j])>0)
     {
```

```
          t=args[i];
          args[i]=args[j];
         args[j]=t;
      }
   }
}
System.out.println("\n sorted name list is…\n);
for (i=0;i<n;i++)
{
    System.out.println(args[i]);
}
}
}
```

Sample input and output:

E:\>java ex2 one two three four five six seven eight nine ten eleven twelve
You have entered 12 names to sort it in ascending order
Sorted name list is…
Eight
Eleven
Five
Four
Nine
One
Seven
Six
Ten
Three
Twelve
two

Result:

Thus the java program was executed successfully and the output was verified

ExNo: 3

BITWISE OPERATOR

Aim: To write a java program to understand the concept of functionalities of different Bitwise operators.

Algorithm:

1. Create a class called Ex3.
2. Use DataInputStream to get stream of data.
3. Display the menu and get choice from the user.
4. Now,call the appropriate care.
5. Display the output.

Source code:

```
import java.io.*;
public class Ex3
{
public static void main(String args[])throws IOException
{
    DataInputStream d=new DataInputStream(System.in);
    System.out.println("\n enter value for x:");
    int x=Integer.parseInt(d.readLine());
    System.out.println("\n enter value for y:");
    int y= Integer.parseInt(d.readLine());
    int c;
   do
  {
    System.out.println("\n BITWISE OPERATOR\n");
    System.out.println("1.AND");
    System.out.println("2.OR");
    System.out.println("3.XOR");
    System.out.println(" 4.LEFT SHIFT");
    System.out.println(" 5.RIGHT SHIFT");
    System.out.println(" 6.NOT");
    System.out.println(" 7.EXIT\n");
    System.out.println(" \n enter your choice:");
    c=Integer.parseInt(d.readLine());
    switch(c)
   {
     case 1:
       System.out.println("AND OPERATION:"+(x+y));
       break;
   case 2:
      System.out.println("OR OPERATION:"+(x|y));
```

```
        break;
    case 3:
        System.out.println("XOR OPERATION:"+(x^y));
        break;
    case 4:
        System.out.println("LEFT SHIFT:"+(x<<y));
        break;
    case 5:
         System.out.println("RIGHT SHIFT:"+(x>>y));
         break;
    case 6:
         System.out.println("NOT of x :"+(~x));
         System.out.println("NOT of y :"+(~y));
         break;
     case 7:
          System.exit(1);
          break;
     default:
         System.out.println("\n enter only 1 to 7");
         break;
    }
  }while(c!=7);
}
}
```

Sample input and output:

```
Enter value for x:10
Enter value for y:6
BITWISE OPERATOR
1.AND
2.OR
3.XOR
4.LEFT SHIFT
5.RIGHT SHIFT
6.NOT
7.EXIT
Enter your choice:1
AND OPERATION:2
BITWISE OPERATOR
1.AND
2.OR
3.XOR
4.LEFT SHIFT
5.RIGHT SHIFT
6.NOT
```

7.EXIT
13
Enter your choice:2
OR OPERATION:14
BITWISE OPERATOR
1.AND
2.OR
3.XOR
4.LEFT SHIFT
5.RIGHT SHIFT
6.NOT
7.EXIT
Enter your choice:3
XOR OPERATION:12
BITWISE OPERATOR
1.AND
2.OR
3.XOR
4.LEFT SHIFT
5.RIGHT SHIFT
6.NOT
7.EXIT
Enter your choice:4
LEFT SHIFT OPERATION:640
BITWISE OPERATOR
1.AND
2.OR
3.XOR
4.LEFT SHIFT
5.RIGHT SHIFT
6.NOT
14
7.EXIT
Enter your choice:5
RIGHT SHIFT OPERATION:0
BITWISE OPERATOR
1.AND
2.OR
3.XOR
4.LEFT SHIFT
5.RIGHT SHIFT
6.NOT
7.EXIT
Enter your choice:6
NOT of x :-11
NOT of y :-7

BITWISE OPERATOR
1.AND
2.OR
3.XOR
4.LEFT SHIFT
5.RIGHT SHIFT
6.NOT
7.EXIT
Enter your choice:7

Result:
Thus,the java program has been written to understand the concept of functionalities of different bitwise operators and output were verified.

ExNo: 4

METHOD OVERRIDING
Aim: To write a java program to understand the concept of Method Overriding.

Algorithm:
1. Create a base class "Vehicle" and declare the variable reg_no as string and model as integer.
2. Create a function read()inside the class to read data for the variables.
3. Create a derived class"TwoWheeler" and declare the variable no_gear,power as integer.
4. Create another deriver class"Scooter" and declare variable manufacturer,owner as string.
5. Create a function read()to get the data for no_gear.
6. Create another function print that prints the values of the reg_no,model,power,no_gear

Source code:
```
import java.io.*;
class Vehicle
{
    String reg_no;
    int model;
    void read()throws IOException
   {
      DataInputStream d=new DataInputStream(System.in);
      System.out.println("\n enter reg_no and model :");
      reg_no=d.readLine();
      model=Integer.parseInt(d.readLine());
    }
}
```

```
class TwoWheeler extends Vehicle
{
    int no_gear,power;
    void read()throws IOException
   {
      super.read();
      DataInputStream d=new DataInputStream(System.in);
      System.out.println("\n enter no_gear and power :");
      no_gear=Integer.parseInt(d.readLine());
      power= Integer.parseInt(d.readLine());
   }
}
class Scooter extends TwoWheeler
{
    String manufacturer,owner;
    void read()throws IOException
   {
      super.read();
     DataInputStream d=new DataInputStream(System.in);
     System.out.println("\n enter manufacturer and owner :");
     manufacturer=d.readLine();
     owner= d.readLine();
   }
   void print()
  {
     System.out.println("\n");
     System.out.println("\n Reg_no :"+reg_no);
     System.out.println("\n Model :"+model);
     System.out.println("\n No_gear :"+no_gear);
     System.out.println("\n Power :"+power);
     System.out.println("\n Manufacturer :"+manufacturer);
     System.out.println("\n Owner :"+owner);
   }
}
class Overriding
{
  public static void main(String args[])throws IOException
 {
    Scooter s=new Scooter();
    s.read();
    s.print();
 }
}
```

Sample input and output:
Enter reg_no and model:TN31AD1224 2010
Enter no_gear and power:5 798
Enter manufacturer and owner : Maruthi Sauresh.A
Reg_no : TN31AD1224
Model : 2010
No_gear : 5
Power : 798
Manufacturer : Maruthi
Owner : Sauresh.A

Result:
Thus, the java program has been written to understand the concept of Method Overriding and output was verified.

ExNo: 5

PACKAGES
Aim: To write a java program to understand the steps in the creation of packages.

Algorithm:
1. Include the package complex.
2. Create a class Arith and declare rp,ip as integer.
3. Define the function arith(),set rp and ip to 0
4. Another function arith(int rp,int ip) set this.rp=rp and this.ip=ip.
5. Create a function add() pass arguments with arith a1 and arith a2.
6. Add a1.rp and a2.rp store in rp.
7. Similarly add a1.ip and a2.ip and store in ip.
8. Create a function sub(arith a1,arith a2)
9. Subtract a1.rp and a2.rp store it in rp
10. Subtract a1.ip and a2.ip store it in ip

Source code:

```
package pkg;
public class arith
{
    public int rp,ip;
    public arith()
   {
      rp=0;
      ip=0;
    }
    public arith(int rp,in rip)
   {
       this.rp=rp;
```

```
    this.ip=ip;
  }
  public void add(arith a1,arith a2)
 {
        rp=a1.rp + a2.rp;
        ip=a1.ip +a2.ip;
  }
  public void sub (arith a1,arith a2)
 {
        rp=a1.rp - a2.rp;
        ip=a1.ip -a2.ip;
  }
}
```

Main code:

```
import pkg.Arith;
import java.io.*;
class Package1
{
    public static void main(String args[])throws IOException
   {
        int rp,ip;
        DataInputStream d=new DataInputStream(System.in);
        System.out.println("\n enter real part and imaginary part of 1st complex no :");
        rp=Integer.parseInt(d.readLine());
        ip= Integer.parseInt(d.readLine());
        arith a1=new arith(rp,ip);
        System.out.println("\n enter real part and imaginary part of 2nd complex no :");
        rp=Integer.parseInt(d.readLine());
        ip= Integer.parseInt(d.readLine());
        arith a2=new arith(rp,ip);
        arith a3=new arith();
        System.out.println("\n a1="+a1.rp+ "+" +a1.ip+ "i");
        System.out.println("\n a2="+a2.rp+ "+" +a2.ip+ "i");
        a3.add(a1,a2);
        System.out.println("\n Added value:" +a3.rp+ "+" +a3.ip+ "i");
        a3.sub(a1,a2);
        System.out.println("\n Subtacted value :" +a3.rp+ "-" a3.ip + "i");
   }
}
```

Sample input and output:

```
Enter real part and imaginary part of 1st complex no : 10 5
Enter real part and imaginary part of 2nd complex no : 3 6
a1= 10 + 5i
a2= 3 +6i
```

Added value : 13 +11i
Subtracted value : 7 – 1i

Result:
Thus, the java program has been written to create a package and output were verified by importing it in another program

Ex.No : 6

INTERFACE
Aim: To write java program to implement the concept of interface.

Algorithm:
1. Create a class income,declare iamount as integer.
2 Create a function get() to get the values for the variable.
3 Create a class expenditure and declare the function get1().
4. Create a derived class net_income that extends income and implements expenditure.
5. Declare variables eamount,namount as float.
6. Create a function print() to print the values.
7. Create a main class and create an object for net_income.
8. Now,get the values using n.print().
9. End.

Source code:

```
import java.io.*;
class Income
{
        float iamount;
        void get()throws IOException
        {
            DataInputStream d=new DataInputStream(System.in);
            System.out.println("\n enter the income :");
            iamount=Float.pareseFloat(d.readLine());
        }
}
interface Expenditure
{
        public void get1();
}
class NetIncome extends Income implements Expenditure
{
        float eamount,namount;
        public void get1()
        {
          try
```

```
        {
           DataInputStream d=new DataInputStream(System.in);
           System.out.println("\n enter expenditure:");
           eamount=Float.parseFloat(d.readLine());
         }
         Catch(IOException e)
         {}
       }
       void print()
       {
          System.out.println("\n income :" +iamount);
          System.out.println("\n expenditure :" +eamount);
          System.out.println("\n net income :" +(iamount-eamount));
       }
  }

class Interface6
{
    public static void main(String args[])throws IOException
   {
         net_income n=new net_income();
         n.get();
         n.get1();
         n.print();
   }
}
```

Sample input and output:

Enter income : 1000
Enter expenditure : 200
Income : 1000.0
Expenditure : 200.0
Net income : 800.0

Result:

Thus, the java program was created and executed successfully and the output is verified.

ExNo : 7

INHERITANCE

Aim: To write a java program to handle the situation of exception multi inheritance.

Algorithm:

1. Create a class student and declare variable rollno.
2. Create function getnumber() and display().
3. Create another class test which inherit student.

4. Create function display().
5. Create another interface sports with variable sport and function putsp().
6. Create variable total and function putsp(),display().
7. Create another class result which extends the test and sport.
8. Create main class inhetitance7 and which has the student as object of class

Source code:

```
import java.io.*;
class Student
{
        int roll_no;
        void getnumber(int n)
        {
           roll_no=n;
        }
        void display()
        {
           System.out.println("\n Rollno:' +roll_no);
        }
}
class Test extends Student
{
        int m1,m2;
        void getmarks(int p,int q)
        {
           m1=p;
           m2=q;
        }
        void display()
        {
            System.out.println("\n Rollno:' +roll_no);
            System.out.println("\n Marks Achieved");
            System.out.println("\n Mark1 :" +m1);
            System.out.println("\n Mark2 :" +m2);
        }
}
interface Sports
{
        float sport=6.0f;
        void putsp();
}
class Results extends Test implements Sports
{
        float total;
        public void putsp()
        {
```

```
        System.out.println("\n sports result ="+sport);
    }
    void display()
    {
        total=m1+m2+sport;
        display();
        putsp();
        System.out.println("\n total score =" +total);
    }
}
class Inheritance7
{
    public static void main(String args[])
    {
        result student1 = new results();
        student1.getnumber(1256);
        student1.getmarks(30,40);
        student1.display();
    }
}
```

Sample input and output:

Rollno :1256
Marks Achieved
Marks1 = 30
Marks2 =40
Sports result =6.0
Total score = 76.0

Result:

Thus, the java program was created and executed successfully and output is verified.

ExNo : 8

EXCEPTION HANDLING

Aim: To write a java program to implement the concept of exception handling.

Algorithm:

1. Create a class exception8
2. Declare the integer a,b,c inside the class.
3. Store length of the argument in a and get the variable for b.
4. Calculate c=b/a.
5. Print the value of a,b,c.
6. Declare variable d[] = new int [a-2].
7. End.

Source code:

```
import java.io.*;
class exception8
{
        public static void main(String args[])throws IOException
        {
           DataInputStream din= new DataInputStream(System.in);
           int a,b,c;
        try
        {
           a=args.length;
           System.out.println("\n enter b value :");
           b= Integer.pareseInt(din.readLine());
           c=b/a;
           System.out.println("\n a="+a);
           System.out.println("\n b=" +b);
           System.out.println("\n b/a =" +c);
           int d[] = new int[a-2];
           d[42]=99;
        }
        catch(ArithmeticException e3)
        {
           System.out.println(e3);
        }
        catch(NegativeArraySizeException e2)
        {
           System.out.println(e2);
        }
        catch(ArrayIndexOutOfBoundsException e1)
        {
           System.out.println(e1);
        }
        catch(NumberFormatException e)
        {
          System.out.println(e);
        }
}
```

Sample input and output:

```
C:\>java exception8
Enter b value:
1
java.lang. ArithmeticException:/by zero
C:\>java exception8
Enter b value: 4
a = 1
```

b = 4
b/a = 4
java.lang. NegativeArraySizeException
C:\>java exception8 3 5
Enter b value: 5
a = 2
b = 5
b/a =2
java.lang. ArrayIndexOutOfBoundsException: 42
C:\>java exception8 3 5
Enter b value: t
java.lang. NumberFormatException: for input string : “t”.

Result:
Thus, the java program was created and executed successfully and output is verified.

ExNo : 9

USER DEFINED EXCEPTION HANDLING
Aim: To write a java program to implement the concept of user defined exception.

Algorithm:
1. Create a class myexception that extends exception ,create variable and function to read and return a message.
2. Create another class exception , create variable eno ,ename ,job ,count ,as static int and string.
3. Create a function display() to display the employee details.
4. The main function call the get() and display() function to red and print the values.

Source code:

```
import java.io.*;
import java.lang.*;
class MyException extends Exception
{
     String msg;
     public MyException(String message)
    {
       msg = message;
     }
     public StringtoString()
    {
       return msg;
   }
 }
class Exception9
{
```

```
        static int eno[] = new int[];
        static String ename[] = new String[3];
        static String job[] = new String[3];
        static int sal[] =new int[3];
        static int count = 0;
        static void get()throws IOException
        {
           int i;
           DataInputStream d = new DataInputStream(System.in);
           try
          {
            for(i=count;i<3;i++)
           {
              System.out.println("\n enter " +(i+1)+" employee detail :");
              System.out.println("\n enter employee no and name :");
              eno[i] = Integer.parseInt(d.readLine());
              if(eno[i]<=0) throw new MyEception(" enter valid number");
              ename[i] = d.readLine();
              System.out.println("\n enter job and salary :");
              job[i] = d.readLine();
              sal[i] = Integer.parseInt(d.readLine());
              if(sal[i]<0) throw new MyEception(" enter valid salary");
             count++;
           }
         }
        catch(MyException e)
        {
           System.out.println(" Exception caught :" +e);
        }
    }
}
static void disp()
{
        for(int i=0;i<3;i++)
        {
           System.out.println(eno[i]+ "\t\t" +ename[i]+ "\t\t" +job[i]+ "\t\t"+ sal[i]);
        }
}
public static void main(String args[])throws IOException
{
        get();
        System.out.println("\n ENO\t\t ENAME\t\t JOB\t\t SALARY\n");
        disp();
}
}
```

Sample input and output:

```
Enter 1 employee details:
Enter employee no and name : -5554
Exception caught : enter valid number
Enter 1 employee detail :
Enter employee no and name : 3001
R.Haricharan
Enter job and salary : lecturer 25000
Enter 2 employee detail :
Enter employee no and name : 5001
S.Kalaiselvan
Enter job and salary : reader
20000
Enter 3 employee detail :
Enter employee no and name : 7801
S.Rajadurai
Enter job and salary : director
23000
35
ENO ENAME JOB SALARY
3001 R.Haricharan lecturer 25000
5001 S.Kaliselvan reader 20000
7801 S.Rajadurai director 23000
```

Result:

Thus, the java program was created and executed successfully and output is verified.

ExNo : 10

MULTI THREADING – join() and isAlive()

Aim: To create a java program in a multithread environment and implement join() and isAlive() functions.

Algorithm:

1. Create a class table that is derived from Thread.
2. Declare name as string.
3. Create a constructor table , pass thread name.
4. Call super(threadname) , start() , start the thread.
5. Create a function run() , print the multiples of 5.
6. Create another class fseries that extends thread.
7. Declare name as string and t1,t2,t3 as int and set f1=-1,f2=1.
8. Create a constructor fseries pass thread name call super thread name()
9. In the run() function display the Fibonacci series.
10. Create a class multi in the main function , create object f for table.
11. f.join() waits for thread to find
12. create object f for fseries and do f.join().

Source code:

```
import java.io.*;
class Table extends Thread
{
        String name;
        Table(String threadname)
        {
          super(threadname);
          name = threadname;
          System.out.println("\n New thread :' +this);
          start();
        }
        public void run()
        {
          System.out.println("\n Five Table");
          for(int i=0;i<=10;i++)
          {
            System.out.println("\n i+ "*5=" +(i*5));
          }
 }
class Fseries extends Thread
{
        String name;
        int f1 = -1 , f2 = 1 , f3;
        Fseries(String threadname)
        {
          super(threadname);
          name = threadname;
          System.out.println("\n New thread :" + this);
          start();
        }
        public void run()
        {
           System.out.println("\n Fibonacci Series");
           for(int i= 1;i<=5 ;i++)
          {
             f3 = f1 +f2;
             f1 = f2;
             f2 = f3;
             System.out.println( f3);
```

```
        }
    }
}
class Multi
{
        public static void main(String args[])
        {
        try
        {
            Table t= new table(“Five Table”);
            System.out.println(“\n First thread is alive :’ +t.isAlive());
            t.join();
            System.out.println(“\n First thread is alive :’ +t.isAlive());
            Fseries f = new fseries(“Fibonacci Series”);
            System.out.println(“\n Second thread is alive :” +f.isAlive());
        }
        catch(InterruptedException e)
        {
            System.out.println(“\n Main thread is interrupted”);
        }
        System.out.println(“\n Main thread is exiting”);
      }
}
```

Sample input and output:

New thread : Thread[Five Table , 5, main]
First thread is alive : true
Five Table
1*5=5
2*5=10
3*5=15
4*5=20
5*5=25
6*5=30
7*5=35
8*5=40
9*5=45
40
10*5=50
First thread is alive : false

New thread [Fibonacci Series , 5, main]
Fibonacci series
Second thread is alive : true
0 1 1 2 3
Second thread is alive : false
Main thread is exiting.
Result:
Thus, the java program was created and executed successfully and output is verified.

ExNo : 11

MULTI THREADING : INTER-THREAD COMMUNICATON

Aim:To write a java program to implement the concept of inter-thread communication.
Algorithm:
∗ creating a class q and declare variable n as int and declare value set(Boolean) as false.
∗ Create synchronized method in get().
∗ Create another synchronized method void put(int n).
∗ If (value set) then go to try and catch . inside catch call notify() this tells produces that it away put more about in the queue.
∗ Create a class producer , that implements runnable.
∗ Create object q. create a constructor producer where object is passed.
∗ Create a function run() , initialize and declare i= 0.
∗ Create a class consumer that implements runnable.
∗ Create constructor consumer , pass the object and create a function run().
∗ Create a class multi2 inside the main function and create object q.
Source code:

```
import java.io.*;
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(InterrupedException e)
{}
c.getstock((int) (Math.random() * 100);
} }
void stop()
{
t.stop();
} }
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(InterrupedException e)
{}
c.getstock((int) (Math.random() * 100);
```

```
}
}
void stop()
{
44
t.stop();
} }
class stock
{
int goods = 0;
public synchronized void addstock(int i)
{
goods = goods +i ;
System.out.println("\n stock added :" +i);
System.out.println("\n present stock :' +goods);
notify();
}
public synchronized int getstock(int j)
{
while(true)
{
if(goods>=j)
{
goods = goods-j;
System.out.println("\n stock taken away :" +j);
System.out.println("\n present stock :" +goods);
break;
45
}
else
{
System.out.println("\n stock not enough…");
System.out.println("\n waiting for stock to come..");
try
{
wait();
}
catch(InterrupedException e)
{}
}
```

```
}
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(20000);
```

```
p.stop(); c.stop();p.t.join();c.t.join();
System.out.println("\n Thread stopped");
}
catch(InterrupedException e)
{}
System.exit(0);
} }
```
Sample input and output:

```
Stock added : 58
Present stock : 58
Stock taken away : 32
Present stock : 26
Stock not enough..
Waiting for stock to come..
Stock added : 15
Present stock : 41
Stock not enough …
Waiting for stock to come ..
Thread stopped.
```
Result:

Thus, the java program was created and executed successfully and output is verified.

Ex.No:12

ADDITION OF TWO NUMBERS USING APPLET

Aim:

To write a java program to implement applet concept.

Algorithm:

1. Start

2. Create a object for input stream class
3. Compile the program in java
4. Run the program using applet viewer
5. Enter the values
6. End.

Source Code:

```
import java.awt.*;
import java.applet.*;
public class userin extends applet
{
TextField text1,text2;
public void init()
{
text1=new TextField(8);
text2=new TextField(8);
add(text1);
add(text2);
```



```
text1.setText("0");
text2.setText("0");
}
public void paint(Graphics g)
{
int x=0,y=0,z=0;
String s1,s2,s;
g.drawString("input a number in each box",10,50);
try
{
s1=text1.getText();
x=Integer.parsseInt(s1);
s2=text2.getText();
y=Integer.parseInt(s2);
}
catch(Exception ex)
{}
z=x+y;
s=String.valueOf(z);
g.drawString("the sum is:",10,75);
g.drawString(s,100,75);
}
```

```
public boolean action(Event event,Object object)
49
{
repaint();
return true;
}
/*<applet
code="userin"
width=300
height=200>
</applet>*/
}
```

Execution Method:

C:\j2sdk1.4.1_06\bin>javac userin.java

Note:userin.java uses or overrides a deprecated API

Note:recpmpile with-deprecation for details

C:\j2sdk1.4.1_06\bin>appletviewer userin.java

Result:

Thus the java program to add two numbers using applet was created and executed successfully and the output was verified.