OOJ REPORT (LAB 2)

Lab Program 6

Solve this program and write the procedure you have used to Execute this in your observationCreate a package CIE which has two classes- Student and Internals. The Class Personal has members like usn, name, sem. The class Internals has an Array that stores the internal marks scored in five courses of the current Semester of the student. Create another package SEE which has the class External which is a derived class of Student. This class has an array that Stores the SEE marks scored in five courses of the current semester of the Student. Import the two packages in a file that declares the final marks of n

Date
Page 195

LAB Program 6 -

```
package CIE;
public class Student {
    public int usn;
    public String name;
    public int sem;
public ~~class~~ Student(int usn, String name,
    int sem){
        this.usn = usn;
        this.name = name;
        this.sem = sem;
}}
```

```
public class internals extends Student{
    public int[] cieMarks = new int[5];
  public internals(int usn, String name,
  int sem, int[] cieMarks){
    super(usn, name, sem)
      this.cieMarks = cieMarks;
  }}
```

```
package SEE;
 import CIE.*;
  public class externals extends Student {
   int[] seeMarks = new int[5];
public externals(int usn, String name, int sem,
   int[] seeMarks){
   super(usn, name, sem);
   this.seeMarks = seeMarks;
}}
```

Date
Page 200

```
// In default package
import SEE.*;
import CIE.*;
class Main{
  public static void main(String[]args){
int usn = 131;
String name = "Rituka";
int sem = 3;
int[] cie = {44, 48, 46, 40, 41};
int[] see = {90, 88, 89, 92, 91};
Internals in = new internals(usn, name,
  sem, cie);
externals ex = new externals(usn, name,
sem, see);
System.out.println("name:"+
  in.name+" usn:"in.usn+" sem"
  +in.sem);
int final = 0;
  for(int i=0; i<5; i++)
{ final = in.cieMarks[i] + ex.seeMarks
[i];
System.out.println("final marks are:"
  +final+"  ");
}}}
```

```
Package CIE;
Import java.util.Scanner;
Public class Internals extends CIE.Student
{
   Public int ciem[]=new int[5];
   Scanner xx =new Scanner (System.in);
   Public void accept()
   {
      For(int i=0;i<5;i++)
      {
         System.out.println("Enter the cie marks of subject"+(i+1)+" out of 50");
         {
            Ciem[i]=xx.nextInt();
         }
      }
   }


}
```

```
Package SEE;
Import CIE.*;
Import java.util.Scanner;

Public class Externals extends CIE.Student
{
```

```
    Public int seem[]=new int[5];
    Scanner xx =new Scanner (System.in);
    Public void accept()
    {
        For(int i=0;i<5;i++)
        {
            System.out.println(“Enter the see marks of subject”+(i+1)+” out of 100”);
            {
                Seem[i]=xx.nextInt();
            }
        }
    }

}
```

```
Package CIE;
Import java.util.Scanner;

Public class Student
{
    String name,usn;
    Int sem;
    Scanner xx=new Scanner(System.in);
    Public void accept()
    {
        System.out.println(“Enter name:”);
```

```
        Name=xx.nextLine();
        System.out.println(“Enter usn:”);
        Usn=xx.next();
        System.out.println(“Enter sem:”);
        Sem=xx.nextInt();
    }
    Public void display()
    {
        System.out.println(“Name :”+name);
        System.out.println(“Usn :”+usn);
        System.out.println(“Sem :”+sem);
    }
}
```

```
Enter number of students:-
2
Enter name:
Ishan
Enter usn:
123
Enter sem:
5
Enter CIE marks of 5 subjects:-
Marks in subject 1:
50
Marks in subject 2:
50
Marks in subject 3:
40
Marks in subject 4:
30
Marks in subject 5:
25
Enter SEE marks of 5 subjects:-
Marks in subject 1:
100
Marks in subject 2:
90
Marks in subject 3:
90
Marks in subject 4:
100
Marks in subject 5:
100
Total marks:
100.0
```

```
Marks in subject 4:
30
Marks in subject 5:
25
Enter SEE marks of 5 subjects:-
Marks in subject 1:
100
Marks in subject 2:
90
Marks in subject 3:
90
Marks in subject 4:
100
Marks in subject 5:
100
Total marks:
100.0
95.0
85.0
80.0
75.0
Enter name:
Hello
Enter usn:
21
Enter sem:
3
Enter CIE marks of 5 subjects:-
Marks in subject 1:
100
```

```
Enter usn:
21
Enter sem:
3
Enter CIE marks of 5 subjects:-
Marks in subject 1:
100
ERROR! MARKS CANNOT BE OVER 50! PL
Marks in subject 1:
50
Marks in subject 2:
50
Marks in subject 3:
505
ERROR! MARKS CANNOT BE OVER 50! PL
Marks in subject 3:
30
Marks in subject 4:
30
Marks in subject 5:
100
ERROR! MARKS CANNOT BE OVER 50! PL
Marks in subject 5:
40
Enter SEE marks of 5 subjects:-
Marks in subject 1:
100
Marks in subject 2:
200
ERROR! MARKS CANNOT BE OVER 100! P
Marks in subject 2:
90
```

```
Marks in subject 2:
200
ERROR! MARKS CANNOT BE OVER 100! P
Marks in subject 2:
90
Marks in subject 3:
84
Marks in subject 4:
80
Marks in subject 5:
98
Total marks:
100.0
95.0
72.0
70.0
89.0
```

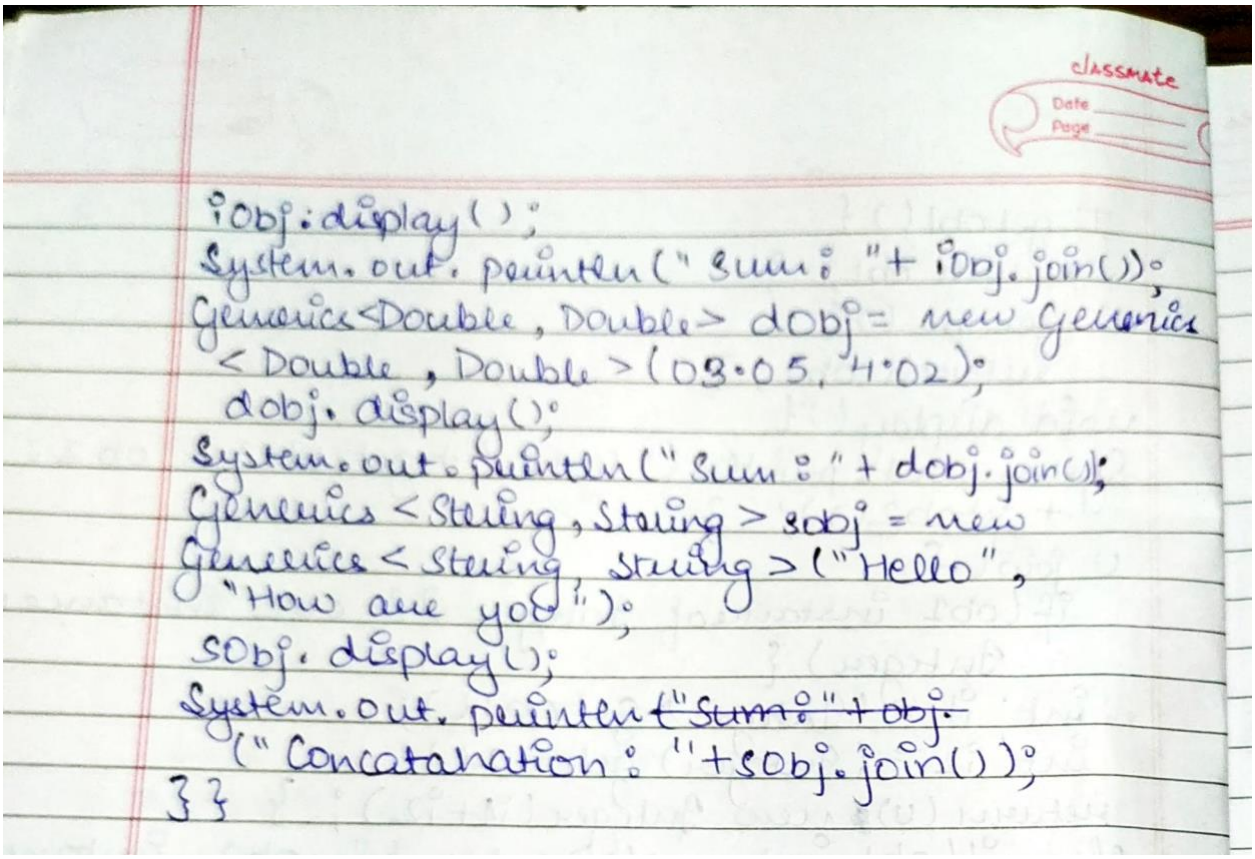
Program 7

1. Write a program to demonstrate generics with multiple object parameters.

LAB-7

```
class Generics<T, U>{
T ob1;
U ob2;
Generics(T x, U y){
    ob1 = x;
    ob2 = y; }
```

```
T getob1() {
 return ob1; }
U getob2() {
   return ob2; }
void display() {
System.out.println("Ob1: " + getob1() + "ob2: "
  + getob2()); }
U join() {
  if(ob1 instanceof Integer && ob2 instanceof
     Integer) {
  int i1 = (Integer) getob1();
  int i2 = (Integer) getob2();
  return (U) new Integer(i1+i2); }
 else if(ob1 instanceof Double && ob2 instanceof
 Double) {
 double d1 = (Double) getob1();
 double d2 = (Double) getob2();
 return (U) new Double(d1+d2); }
 else if(ob1 instanceof String && ob2 instanceof
   String)
   String s1 = (String) getob1();
   String s2 = (String) getob2();
 return (U) new String(s1+s2); }
else {
 return (U) new String("ERROR! ob1 and ob2
 Type Mismatch");
 }}}

class MyMain{
 public static void main(){
   Generics < Integer, Integer > Obj = new Generics
 <Integer, Integer> (5,4);
```

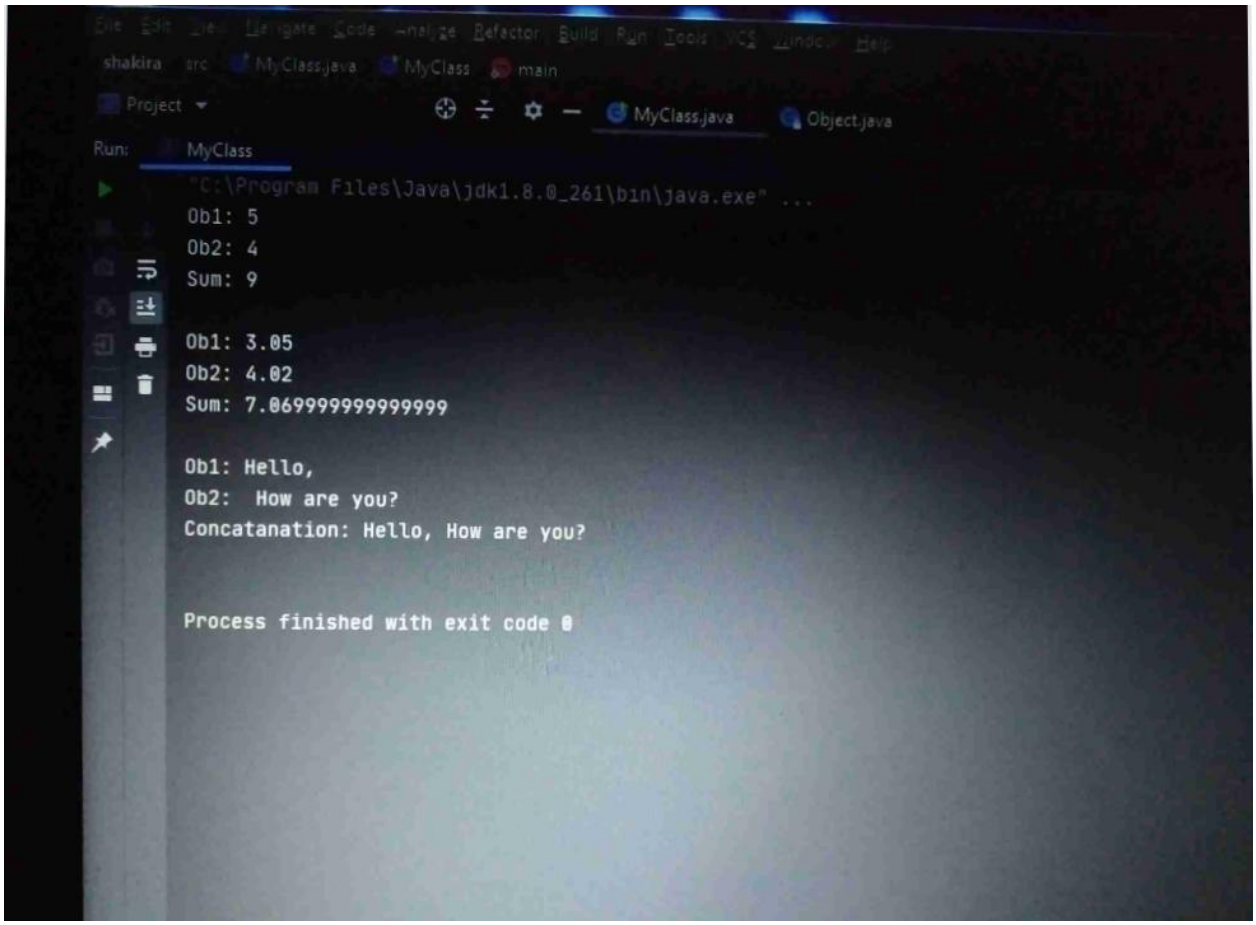


```
iObj.display();
System.out.println("Sum: "+iObj.join());
Generics<Double, Double> dObj = new Generics
<Double, Double>(03.05, 4.02);
dobj.display();
System.out.println("Sum: "+dobj.join());
Generics<String, String> sobj = new
Generics<String, String>("Hello",
"How are you");
sObj.display();
System.out.println("Sum: "+obj.
("Concatanation: "+sObj.join());
}}
```

```
Class Generics<T, U> {
        T ob1;
        U ob2;


        Generics(T x, U y) {
                Ob1 = x;
                Ob2 = y;
        }
```

```
T getob1() {
        Return ob1;
}

U getob2() {
        Return ob2;
}

Void display() {
        System.out.println(“Ob1: “ + getob1());
        System.out.println(“Ob2: “ + getob2());
}

U join() {
        If (ob1 instanceof Integer && ob2 instanceof Integer) {
                Int i1 = (Integer)getob1();
                Int i2 = (Integer)getob2();
                Return (U) new Integer(i1 + i2);
        }
        Else if (ob1 instanceof Double && ob2 instanceof Double) {
                Double d1 = (Double)getob1();
                Double d2 = (Double)getob2();
                Return (U) new Double(d1 + d2);
        }
        Else if (ob1 instanceof String && ob2 instanceof String) {
                String s1 = (String)getob1();
                String s2 = (String)getob2();
                Return (U) new String(s1 + s2);
        }
```

```
                Else {
                        Return (U) new String("ERROR! Ob1 and ob2 Type Mismatch…");
                }
        }
}

Class Lab7 {
        Public static void main(String[] args) {

                Generics<Integer, Integer> iObj = new Generics<Integer, Integer>(5,4);
                iObj.display();
                System.out.println("Sum: " + iObj.join());
                System.out.println();

                Generics<Double, Double> dObj = new Generics<Double, Double>(3.05,4.02);
                dObj.display();
                System.out.println("Sum: " + dObj.join());
                System.out.println();

                Generics<String, String> sObj = new Generics<String, String>("Hello,", " How are you?");
                sObj.display();
                System.out.println("Concatanation: " + sObj.join());
                System.out.println();

        }
}
```



Program 8

2. Write a program that demonstrates handling of exceptions in inheritance tree. Create a

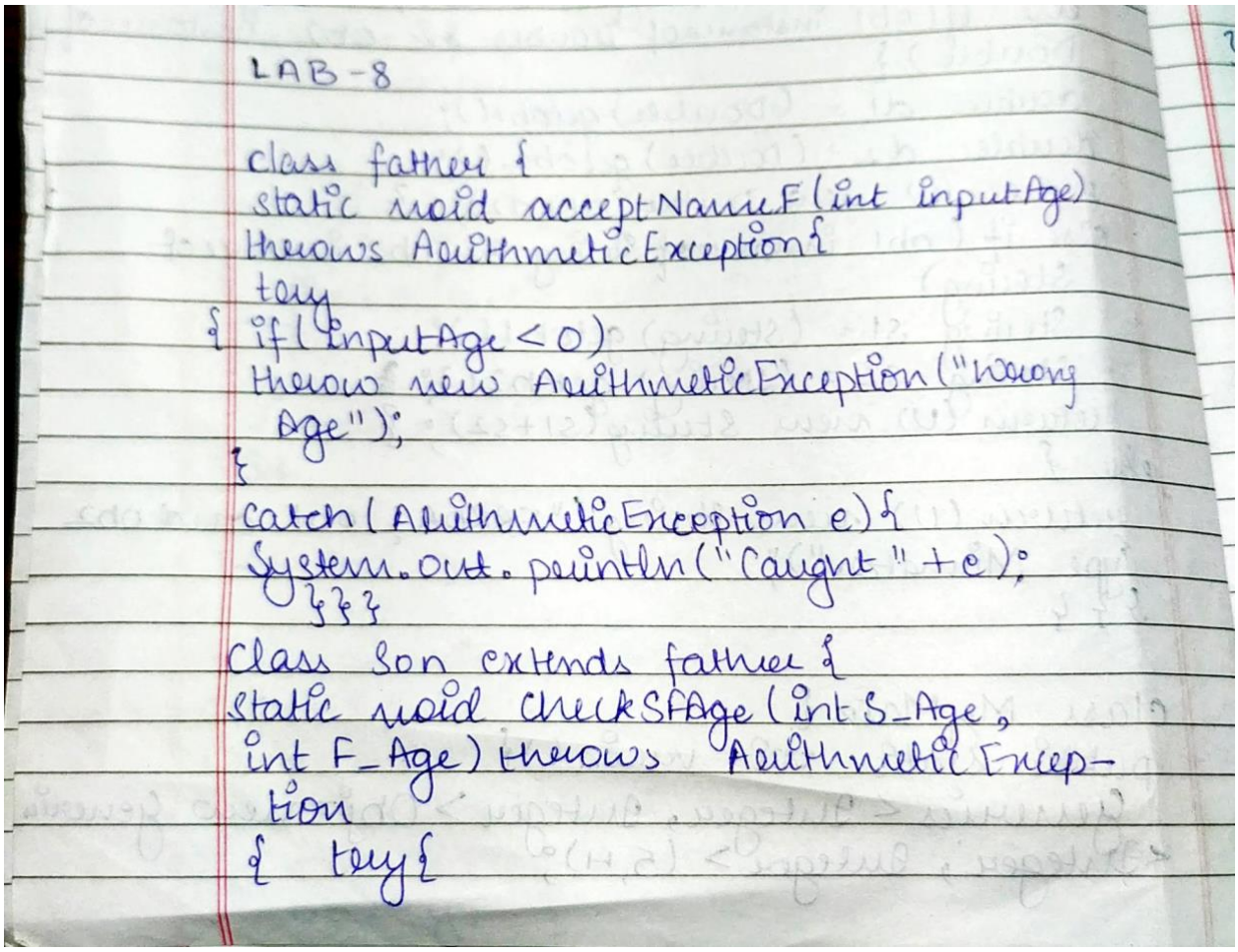
Base class called “Father” and derived class called “Son” which extends the base

Class. In Father class, implement a constructor which takes the age and throws the

Exception WrongAge( ) when the input age<0. In Son class, implement a constructor

That cases both father and son's age and throws an exception if son's age is >=father's

Age.

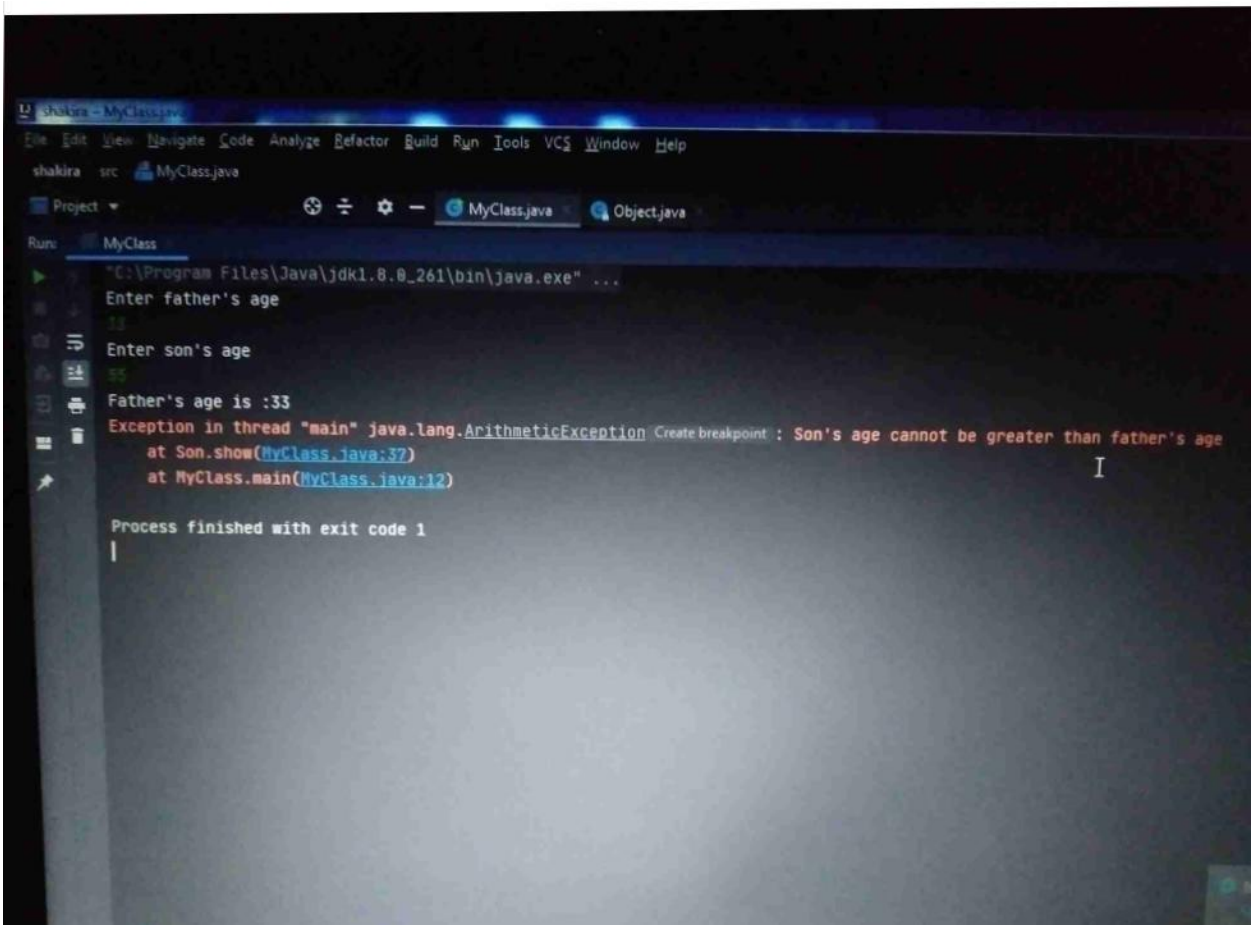


```
LAB-8

class father {
static void acceptNameF(int inputAge)
throws ArithmeticException{
try
{ if(inputAge<0)
throw new ArithmeticException("wrong
 Age");
}
catch(ArithmeticException e){
System.out.println("Caught "+e);
}}}
class Son extends father {
static void checkSfAge(int S_Age,
int F_Age) throws ArithmeticExcep-
tion
{ try{
```

```
if (S_Age >= F_Age)
throw new ArithmeticException ("Son's age
should be smaller than father's age");
System.out.println("Son's age is " + S_Age +
"Father's Age is" + F_Age);
}
catch (ArithmeticException e)
System.out.println("F Caught" + e);
}}}
public class MyClass {
public static void main (String[] args) {
  Father.acceptNameF(-10);
  Son.CheckSfage(30,20);
}}
```

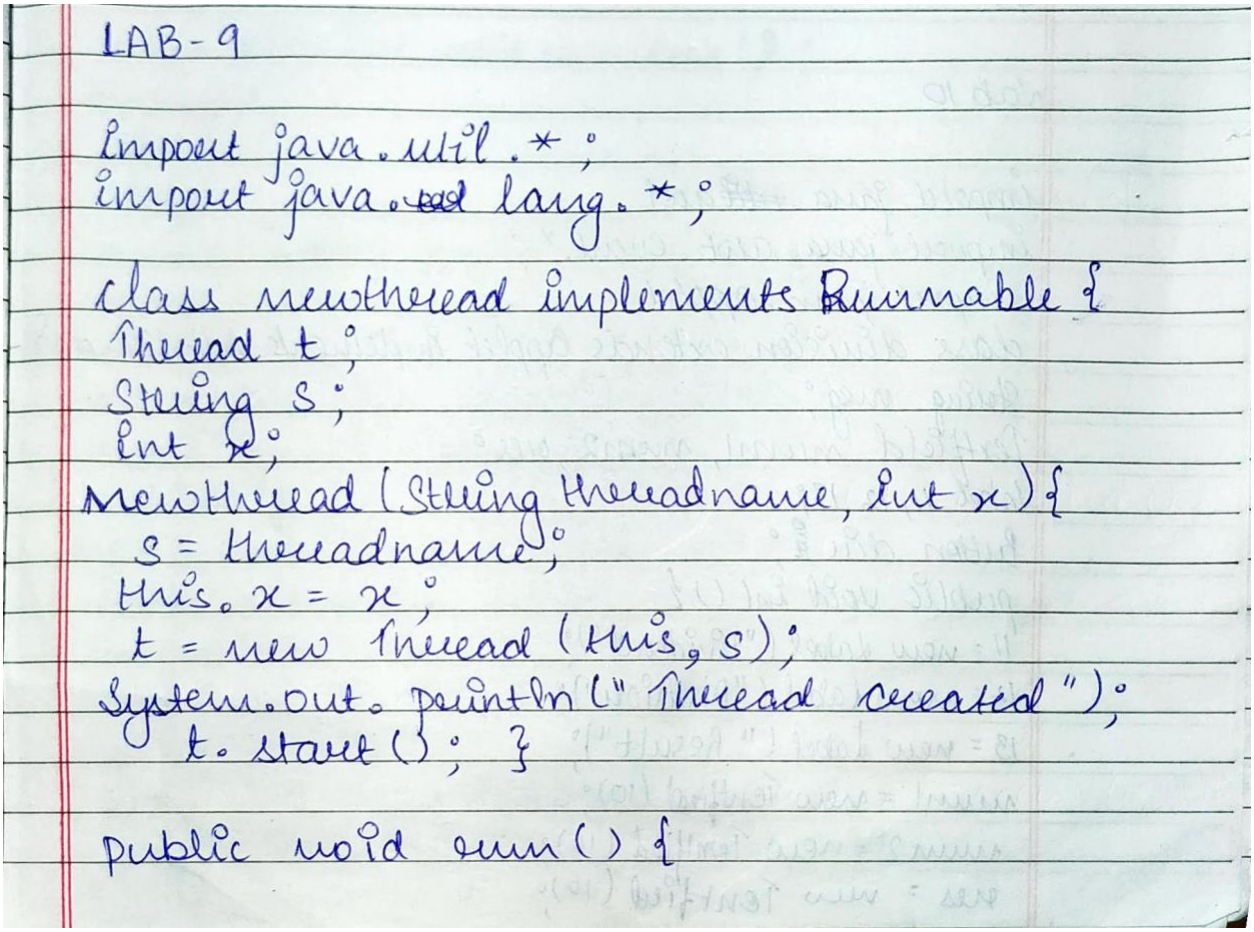
```
Import java.util.Scanner;
Public class MyClass{
   Public static void main(String[] args){
      Scanner sc=new Scanner(System.in);
      System.out.println(“Enter father’s age”);
      Int Fage=sc.nextInt();
      System.out.println(“Enter son’s age”);
      Int Sage=sc.nextInt();
Father o1=new Father(Fage);
Son o2=new Son(Sage,Fage);
O1.display();
O2.show();
   }
}

Class Father{
   Int age;
   Father(int age){
      This.age=age;
   }
   Void display(){
   If(age<0)
      Throw new ArithmeticException(“wrong age”);
   Else
      System.out.println(“Father’s age is :”+age);
}
}
Class Son extends Father{
   Int SonAge;
```

```
    Son(int SonAge,int age){
        Super(age);
        This.SonAge=SonAge;

    }
    Void show(){
    If(SonAge>age )
        Throw new ArithmeticException("Son's age cannot be greater than father's age");
    Else
    System.out.println("Father's age is :"+age+" and fathers age is "+age);
}}
```



Lab 9

Write a program which creates 2 threads, one thread displaying “Bms College of Engineering” once after every 10 seconds and another displaying “CSE” once in every 2 seconds.

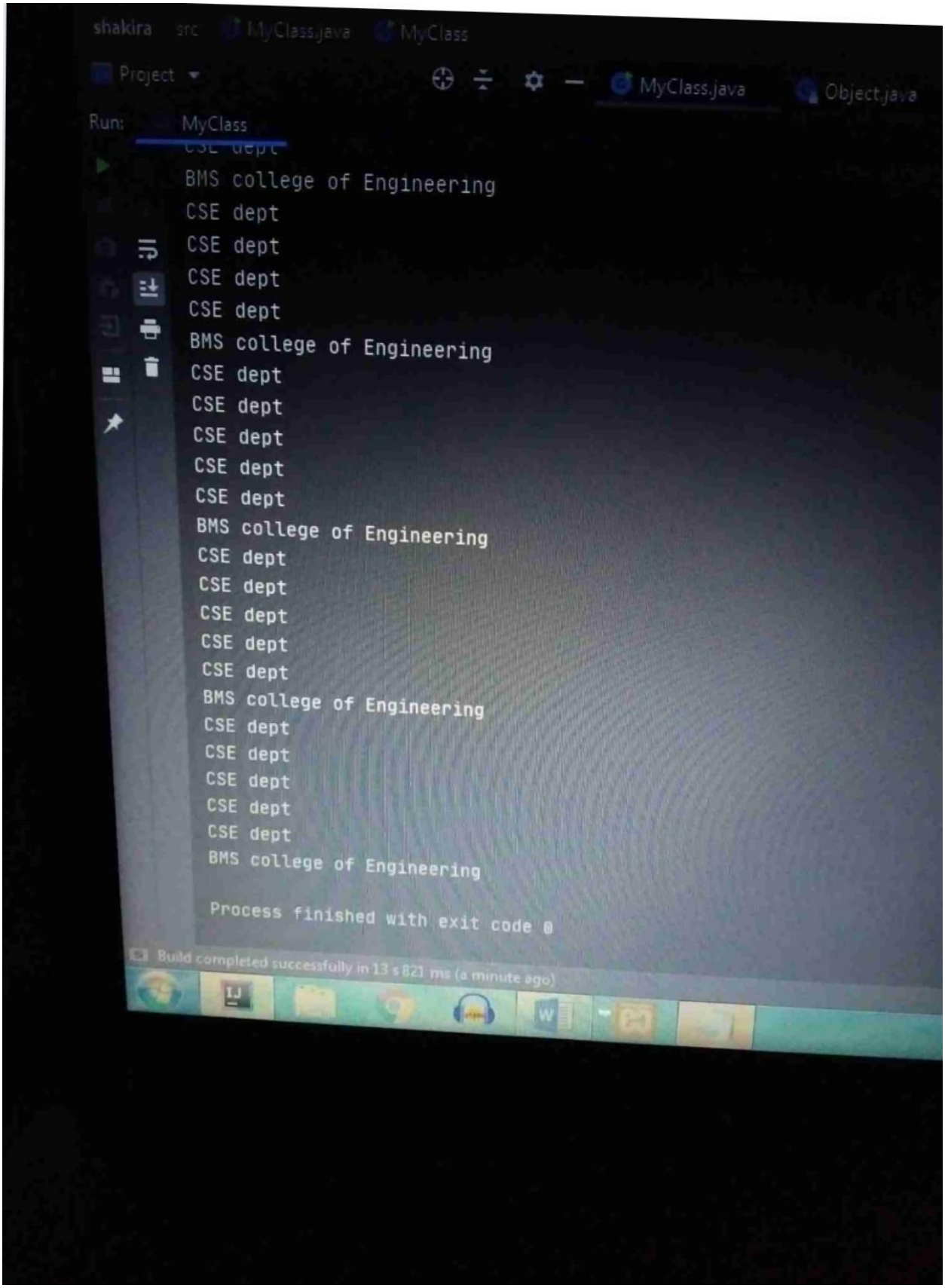


```
LAB-9

Import java.util.*;
import java.lang.*;

class newthread implements Runnable {
Thread t;
String s;
Int x;
Newthread (String threadname, Int x){
  s = threadname;
this.x = x;
  t = new Thread (this, s);
System.out.println(" Thread created ");
    t.start(); }

public void run() {
```

```
try {
for (int i=0; i<=10 ; i++) {
 System.out.println (S);
 Thread.sleep (x);
}}
catch (InterruptedException e) {
System.out.println("Thread interrupted");
}}}

public class MyClass {
public static void main (String[] args){
 new NewThread ("BMS College of
 Engineering", 10000);
 new NewThread ("CSE", 2000);
}}
```

```
Class Thread1 implements Runnable{
   Thread t;
   String name;
   Thread1(String name){
      This.name = name;
      T = new Thread(this,this.name);
      t.start();
   }
   Public void run(){
      Try{
         For(int i=0;i<20;i++){
            System.out.println(“CSE dept”);
            Thread.sleep(2000);
         }
      }catch(InterruptedException e){
         System.out.println€;
      }
   }
}

Class Thread2 implements Runnable{
   Thread t;
   String name;
   Thread2(String name){
      This.name = name;
      T = new Thread(this,this.name);
      t.start();
   }
   Public void run(){
```

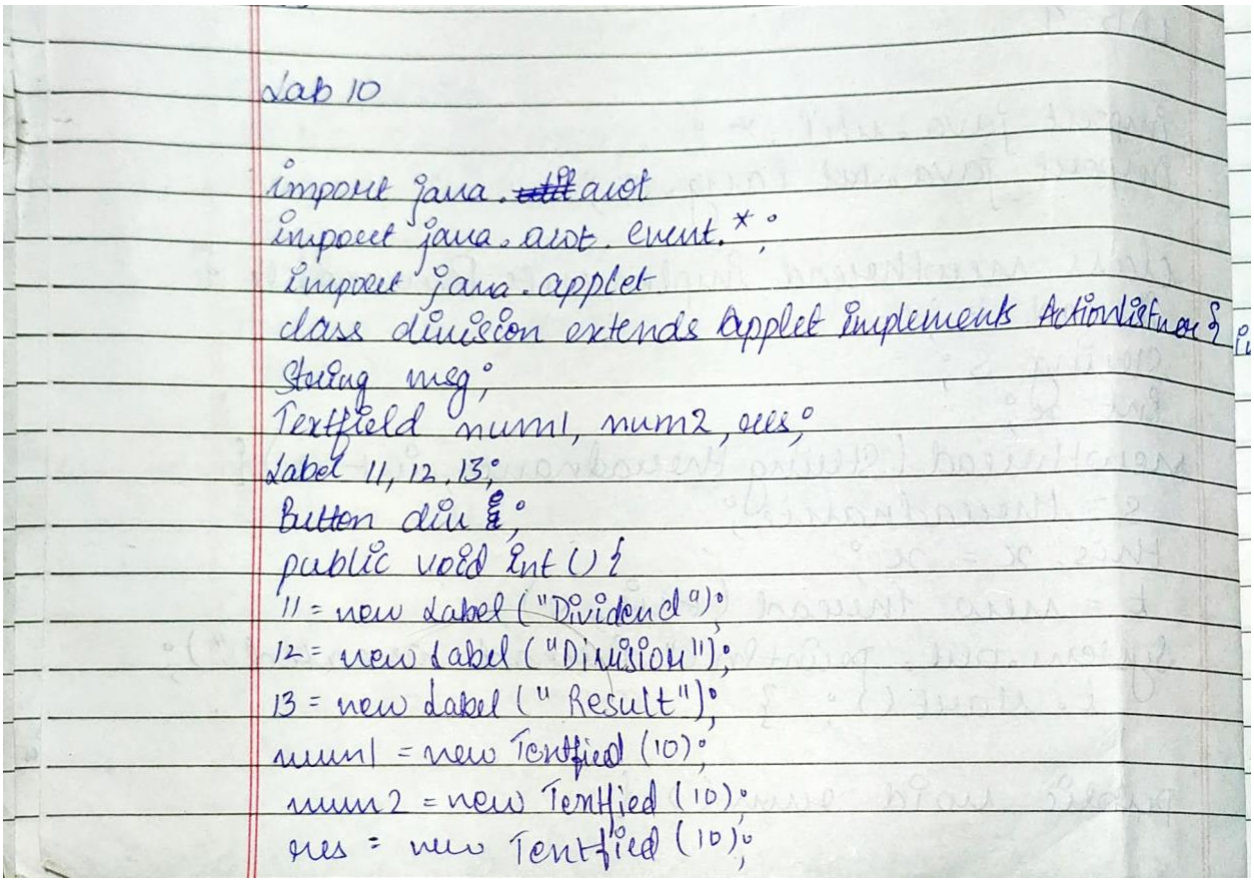
```
        Try{
            For(int i=0;i<5;i++){
                System.out.println(“BMS college of Engineering”);
                Thread.sleep(10000);
            }
        }catch(InterruptedException e){
            System.out.println€;
        }
    }
}

Class labProgram9{
    Public static void main(String[] args){
        Thread1 obj1 = new Thread1(“Dept. name”);
        Thread2 obj2 = new Thread2(“College name”);
        //System.out.println(obj1.name+” “+obj1.t.isAlive());
        //System.out.println(obj2.name+” “+obj2.t.isAlive());
        Try{
            Obj1.t.join();
            Obj2.t.join();
        }catch(Exception e){
            System.out.println(“Interrupted”);
        }
    }
}
```


shakira src MyClass.java MyClass
Project
MyClass.java
Object.java
Run: MyClass
BMS college of Engineering
CSE dept
CSE dept
CSE dept
CSE dept
BMS college of Engineering
CSE dept
CSE dept
CSE dept
CSE dept
CSE dept
BMS college of Engineering
CSE dept
CSE dept
CSE dept
CSE dept
CSE dept
BMS college of Engineering
CSE dept
CSE dept
CSE dept
CSE dept
CSE dept
BMS college of Engineering
Process finished with exit code 0
Build completed successfully in 13 s 821 ms (a minute ago)

Lab 10

Write a program that creates a user interface to perform integer division. The user enters two numbers in the textfield, num1 and num2. The division of num1 and num2 is displayed in the result field when the divide button is clicked. If num1 and num2 word not an integer, the program would throw a a number format exception. If num2 were 0, the program would throw an arithmetic exception.

Display the exception in a message dialogue box.



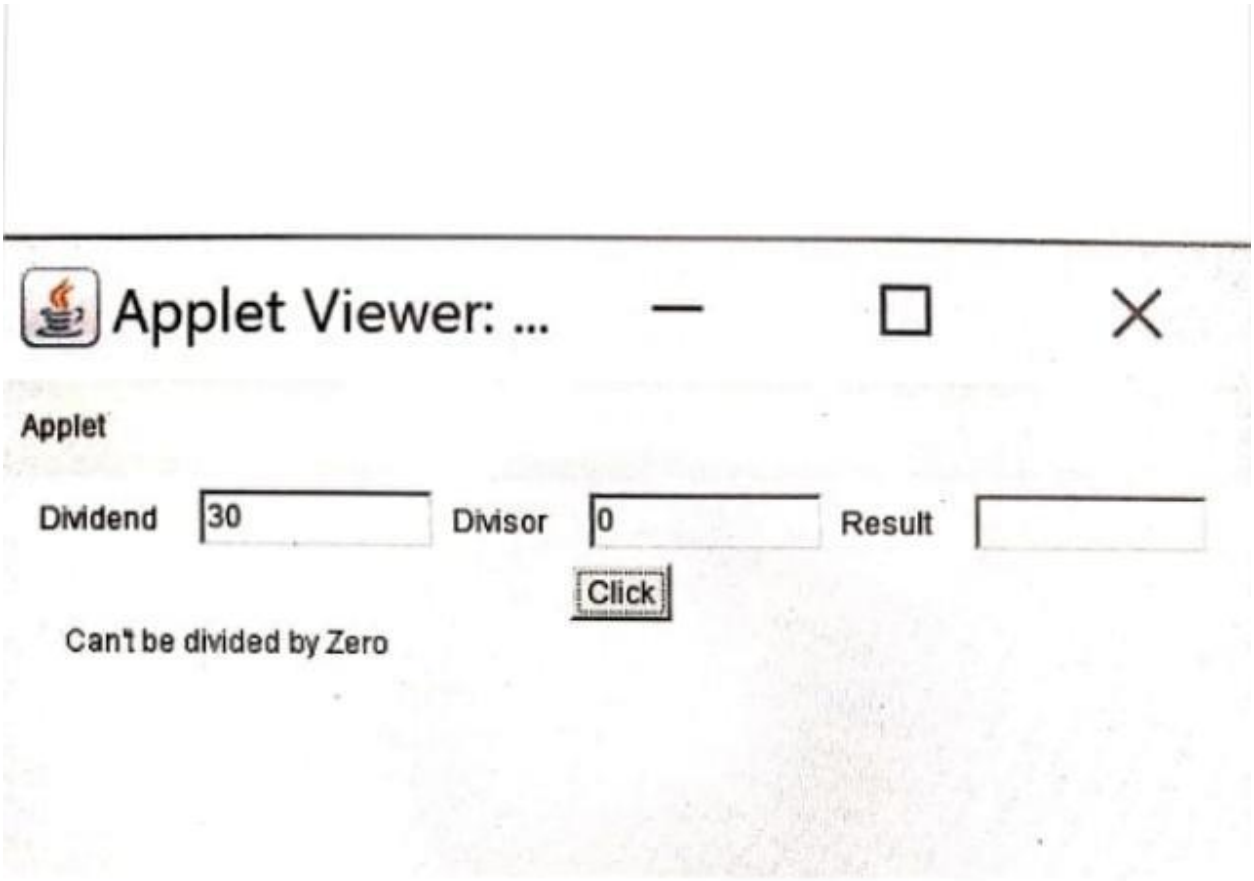
```
Lab 10

import java.awt.*
import java.awt.event.*;
Import java.applet
class division extends Applet implements ActionListener {
String msg;
Textfield num1, num2, res;
Label l1, l2, l3;
Button div;
public void Int () {
l1 = new Label ("Dividend");
l2 = new Label ("Division");
l3 = new Label ("Result");
num1 = new TextField (10);
num2 = new TextFied (10);
res = new TextFied (10);
```

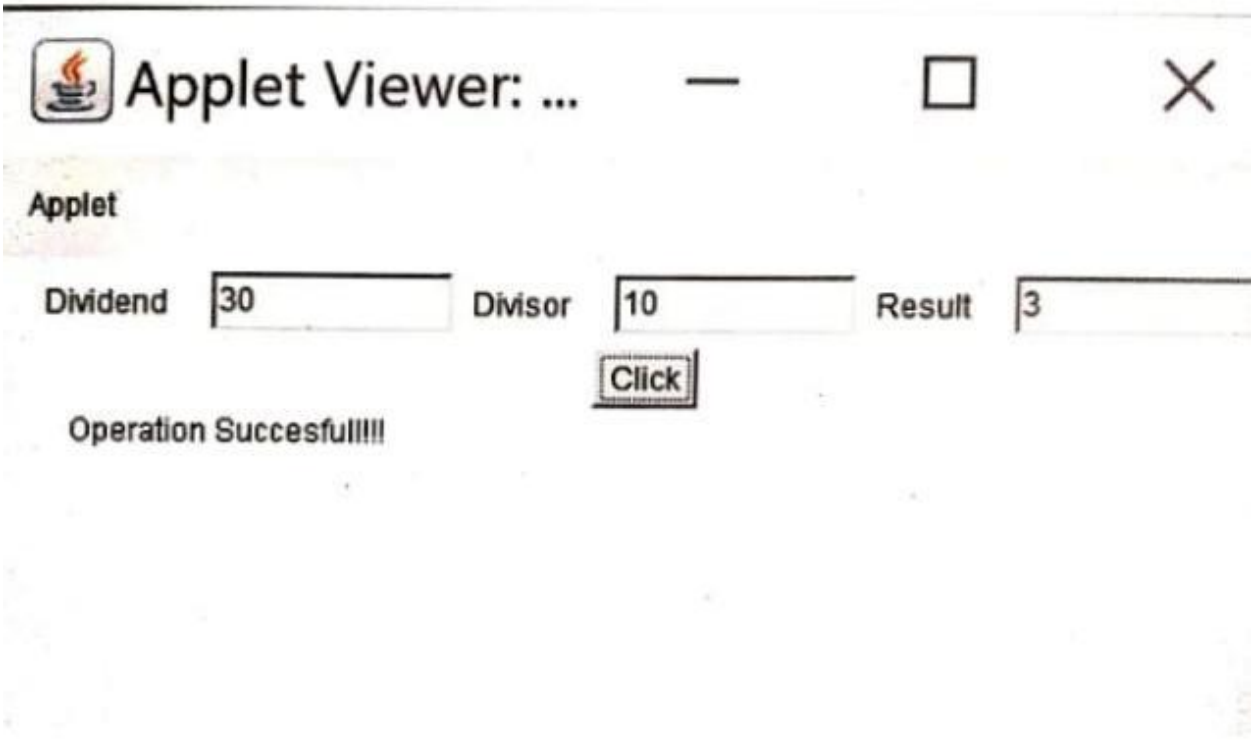
```
div = new Button ("click");
div = addActionListner("this");
add (num1); add (num1);
add (12);
add (num2);
add (13);
add (res);
add (div);
}
Public void actionperformed (ActionEvent as) {
string arg = ae.getActionCommand();
int num1 = 0, num2 = 0;
if (arg.equals ("click")) {
if (this.num1.getText().isEmpty() && this.num2.getText().
    isEmpty()) {
msg = "Enter the valid numbers";
repaint();
else {
try {
num1 = Integer.parseInt (this.num1.getText());
num2 = Integer.parseInt (this.num2.getText());
int num3 = num1 / num2;
res.setText (String.valueOf (num3));
msg = "Operation successfull";
repaint() } }
catch (NumberFormatException ex) {
System.out.println (ex);
res.setText ("");
msg = "cant be divided by 0"}
repaint(); }}}
public void paint (Graphics g) {
    g.drawString (msg, 30, 70); }}
```

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

/*<applet code="DivisionExample"width=230 height=250></applet>*/

 Class DivisionExample extends Applet implements ActionListener {
        String msg;
        TextField num1, num2, res;
        Label l1, l2, l3;
        Button div;

        Public void init() {
                L1 = new Label("Dividend");
                L2 = new Label("Divisor");
                L3 = new Label("Result");
                Num1 = new TextField(10);
                Num2 = new TextField(10);
                Res = new TextField(10);
                Div = new Button("Click");
                div.addActionListener(this);
                add(l1);
                add(num1);
                add(l2);
                add(num2);
                add(l3);
                add(res);
                add(div);
        }
```

```
Public void actionPerformed(ActionEvent ae) {
        String arg = ae.getActionCommand();
        Int num1 = 0, num2 = 0;
        If (arg.equals("Click")) {
                If (this.num1.getText().isEmpty() && this.num2.getText().isEmpty()) {
                        Msg = "Enter the valid numbers!";
                        Repaint();
                } else {
                        Try {
                                Num1 = Integer.parseInt(this.num1.getText());
                                Num2 = Integer.parseInt(this.num2.getText());

                                Int num3 = num1 / num2;

                                Res.setText(String.valueOf(num3));
                                Msg = "Operation Succesfull!!!";
                                Repaint();
                        } catch (NumberFormatException ex) {
                                System.out.println(ex);
                                Res.setText("");
                                Msg = "NumberFormatException – Non-numeric";
                                Repaint();
                        } catch (ArithmeticException e) {
                                System.out.println("Can't be divided by Zero" + e);
                                Res.setText("");
                                Msg = "Can't be divided by Zero";
                                Repaint();
                        }
```

```
                    }
                }
            }

Public    void paint(Graphics g) {
                g.drawString(msg, 30, 70);
            }
}
```





LAB Program 6 -

```
package CIE;
public class Student {
    public int usn;
    public String name;
    public int sem;
public Student(int usn, String name,
    int sem) {
        this.usn = usn;
        this.name = name;
        this.sem = sem;
}}

public class internals extends Student {
    public int[] cieMarks = new int[5];
public internals(int usn, String name,
int sem, int[] cieMarks) {
    super(usn, name, sem)
        this.cieMarks = cieMarks;
}}

package SEE;
import CIE.*;
public class externals extends Student {
    int[] seeMarks = new int[5];
public externals(int usn, String name, int sem,
    int[] seeMarks) {
    super(usn, name, sem);
this.seeMarks = seeMarks;
```

Date
Page 200

```
// In default package
import SEE.*;
import CIE.*;
class Main{
  public static void main(String[]args){
int usn = 131;
String name = "Rituka";
int sem = 3;
int[] cie = {44, 48, 46, 40, 41};
int[] see = {90, 88, 89, 92, 91};
Internals in = new internals(usn, name,
  sem, cie);
Externals ex = new externals(usn, name,
sem, see);
System.out.println("name:"+
  in.name+"usn:"in.usn+"sem"
  +in.sem);
int final = 0;
  for(int i=0; i<5; i++)
{ final = in.cieMarks[i] + ex.seeMarks
[i];
System.out.println("final marks are:"
  +final+"  ");
}}}
```