**5) Inheritance**

**A. Write a java program to implement single level inheritance.**

**Code: -**

class A

{

int i, j;

void showij()

{

System.out.println("i and j" + i + " " + j);

}

}

class B extends A

{

int k;

void showk()

{

System.out.print("k" +k);

}

void sum()

{

System.out.println(" i + j + k"+(i + j + k));

// return i+j+k;

}

}

class SimpleInheritance

{

public static void main(String[] args)

{

A superOb = new A();

B subOb = new B();

superOb.i = 11;

superOb.j = 23;

System.out.println("content of superclass :::");

superOb.showij();

System.out.println();

subOb.i = 7;

subOb.j = 8;

subOb.k = 9;

System.out.print("content of subob :::");

subOb.showij();

subOb.showk();

subOb.sum();

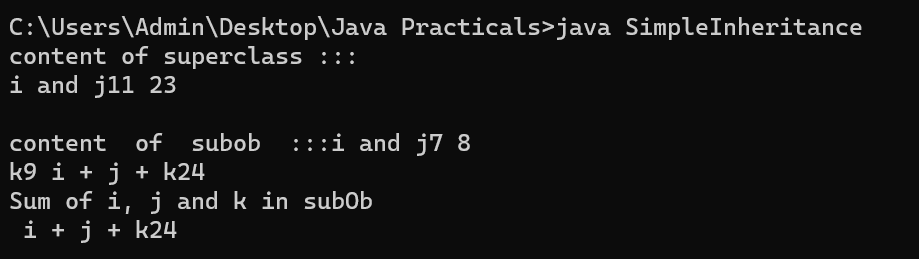
System.out.println("Sum of i, j and k in subOb");

subOb.sum();

}

}

**Output: -**

****

**B) Write a java program to implement multilevel inheritance**

**Code: -**

class A

{

int i, j;

void showi()

{

System.out.println("i " + i );

}

}

class B extends A

{

int j;

void showj()

{

System.out.print("j" +j);

}

// void sum()

// {

// System.out.println(" i + j + k"+(i + j + k));

// // return i+j+k;

// }

}

class C extends B{

int k;

void showk()

{

System.out.println("k : " + k);

}

void sum()

{

System.out.println("i + j + k" + (i+j+k));

}

}

class MultiLevelInheritance

{

public static void main(String[] args)

{

C subOb = new C();

subOb.i = 10;

subOb.j = 20;

subOb.k = 30;

System.out.print("content of subob :");

subOb.showi();

subOb.showj();

subOb.showk();

System.out.println();

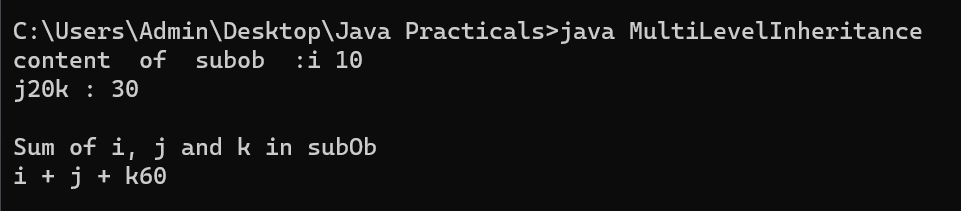
System.out.println("Sum of i, j and k in subOb");

subOb.sum();

}

}

**Output: -**

****

**C. Write a java program to implement multiple inheritance**

**Code: -**

interface A

{

void showA();

}

interface B

{

void showB();

}

class C implements A,B

{

public void showA()

{

System.out.println("Implements Interface A");

}

public void showB()

{

System.out.println("Implements Interface A");

}

void showC()

{

System.out.println("Class C method ");

}

}

class MultipleInheritance

{

public static void main(String[] args)

{

C subOb = new C();

System.out.println("Contents of subOb : ");

subOb.showA();

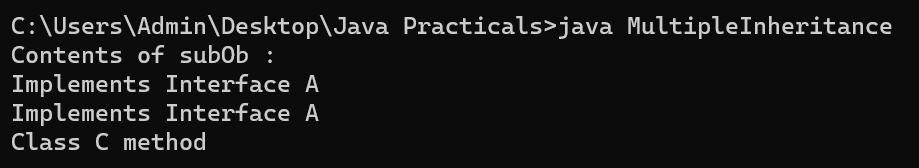
subOb.showB();

subOb.showC();

}

}

**Output: -**

****

**6) Packages and Arrays**

**A. Create a package MyMath, Add the necessary classes and import the package in java class.**

**Code:-**

PACKAGE 1 ADDITION CODE:

package maths;

public class addition

{

public int add(int x,int y)

{

return(x+y);

}

}

PACKAGE 2 SUBTRACTION CODE:

package maths;

public class subtraction

{

public int sub(int x,int y)

{

return(x-y);

}

}

PACKAGE 3 MULTIPLICATION CODE:

package maths;

public class multiplication

{

public int mul(int x,int y)

{

return(x\*y);

}

}

PACKAGE 4 DIVISION CODE:

package maths;

public class division

{

public int div(int x,int y)

{

return(x/y);

}

}

PACKAGE MAIN CODE:

import maths.\*;

class demo1

{

public static void main(String[] args)

{

addition a=new addition();

subtraction s=new subtraction();

multiplication m=new multiplication();

division d=new division();

System.out.println("addition is"+a.add(10,20));

System.out.println("subtraction is"+s.sub(10,20));

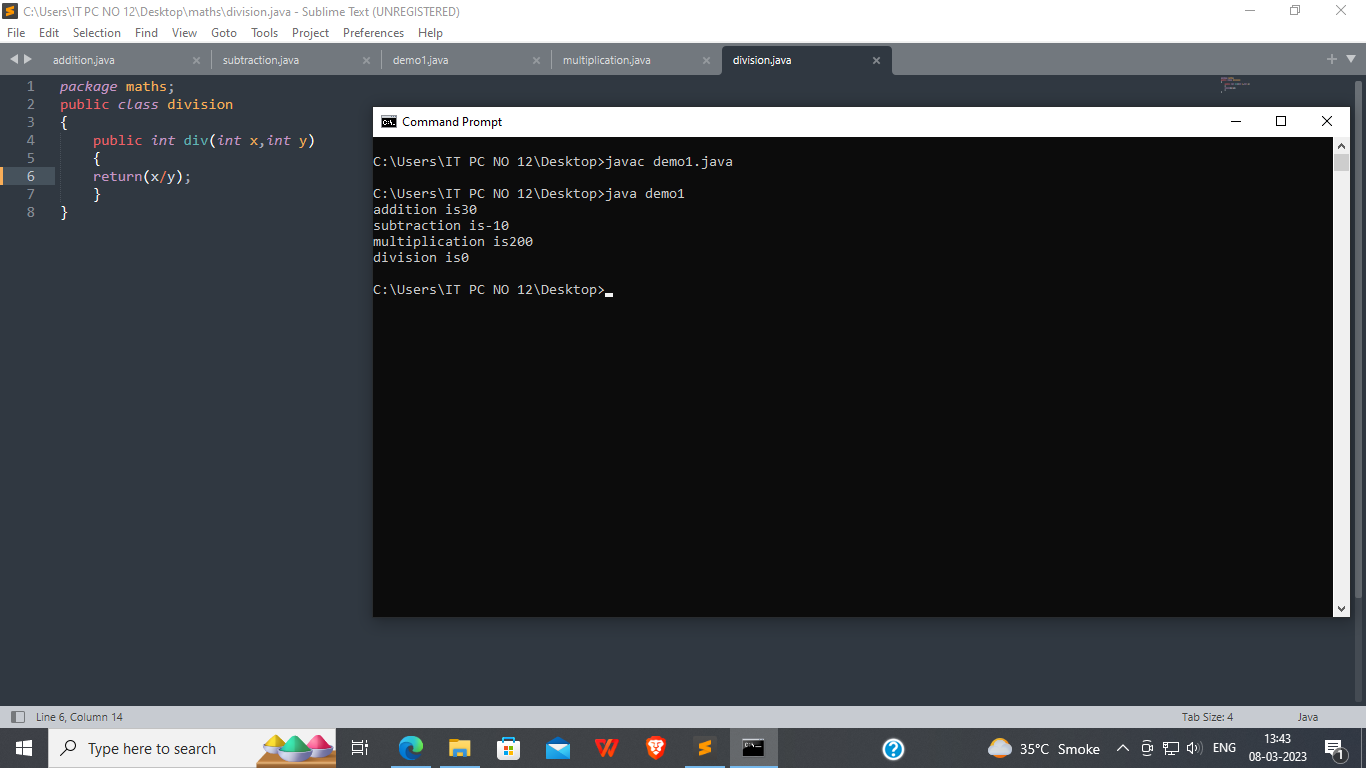
System.out.println("multiplication is"+m.mul(10,20));

System.out.println("division is"+d.div(10,20));

}

}

**OUTPUT:**

****

**7) Vectors and Multithreading**

1. **Write a java program to implement the vectors.**

**Code: -**

import java.util.\*;

class vectorDemo {

public static void main(String[] args)

{

Vector<Integer> v = new Vector<Integer>(3, 2);

System.out.println("Initial size: " + v.size());

System.out.println("Initial capacity: " + v.capacity());

v.addElement(1);

v.addElement(2);

v.addElement(3);

v.addElement(4);

System.out.println("Capacity after four additions: " + v.capacity());

v.addElement(5);

System.out.println("Current capacity: " + v.capacity());

v.addElement(6);

v.addElement(7);

System.out.println("Current capacity: " + v.capacity());

v.addElement(9);

v.addElement(10);

System.out.println("Current capacity: " + v.capacity());

v.addElement(11);

v.addElement(12);

System.out.println("First Element: " + v.firstElement());

System.out.println("Last Element: " + v.lastElement());

if (v.contains(3))

{

System.out.println("Vectors contains 3.");

}

Enumeration<Integer> vEnum = v.elements();

System.out.println("\nElements in vector: ");

while (vEnum.hasMoreElements())

{

System.out.print(vEnum.nextElement() + " ");

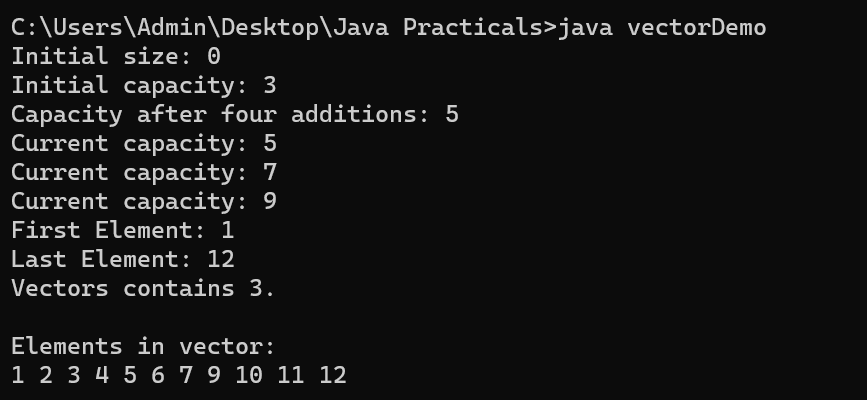
}

System.out.println();

}

}

**Output: -**

****

**B)Write a java program to implement thread life cycle**

**Code:-**

public class Main2{

public static void main(String args[]) {

Thread t=Thread.currentThread();

System.out.println("Current thread:"+t);

t.setName("my Thread");

System.out.println("After changing name"+t);

try{

for(int n=5;n>0;n--){

System.out.println(n);

Thread.sleep(5000);

}

}

catch(InterruptedException e){

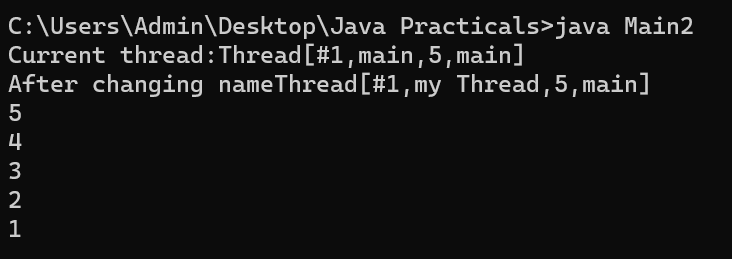
System.out.println("main thread interrupted");

}

}

}

**Output: -**

****

**C. Write a java program to implement multithreading**

**Code:-**

class NewThread implements Runnable{

String name;

Thread t;

NewThread(String threadName){

name=threadName;

t=new Thread(this,name);

System.out.println("New thread:"+t);

t.start();

}

public void run(){

try{

for(int i=5;i>0;i--)

{

System.out.println(name+": "+i);

Thread.sleep(10000);

}

}catch(InterruptedException e){

System.out.println(name+"Interrupted");

}

System.out.println(name+" Exiting");

}

}

public class multithread {

public static void main(String args[]) {

new NewThread("one");

new NewThread("Two");

new NewThread("three");

try{

Thread.sleep(10000);

}catch(InterruptedException e){

System.out.println("Main thread Interrupted");

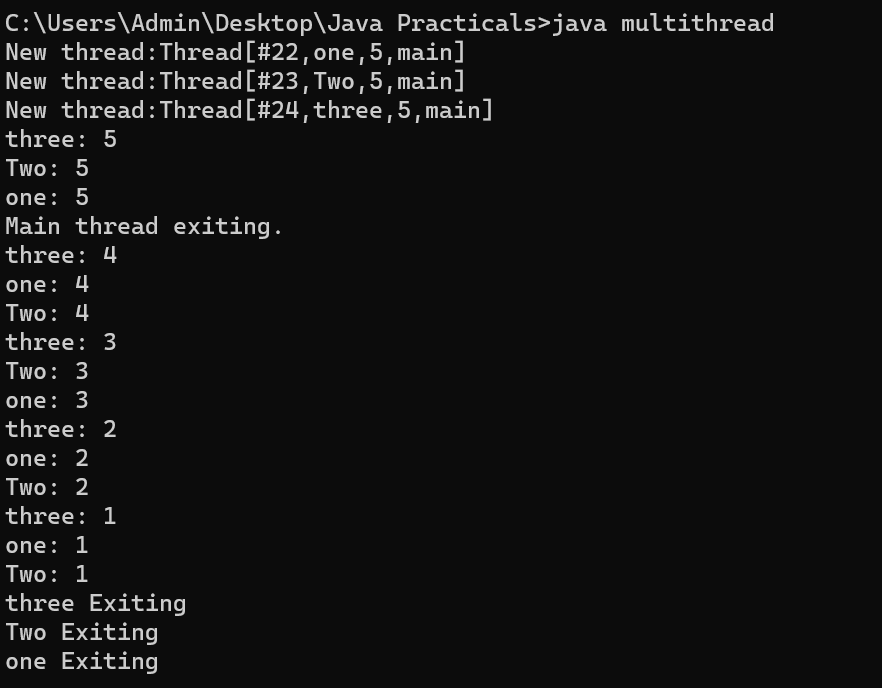
}

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

}

}

**Output: -**

****

**8) File Handling**

**A)Write a java program to open a file and display the contents in the console window**

**Code: -**

import java.io.\*;

class showfile{

public static void main(String[] args) {

int i;

FileInputStream fin;

if(args.length !=1){

System.out.println("usage showfile name ");

return;

}

try{

fin=new FileInputStream(args[0]);

}

catch(FileNotFoundException e){

System.out.println("Cannot not open file");

return;

}

try{

do{

i=fin.read();

if(i!=-1){

System.out.println((char) i);

}

}while(i!= -1);

}

catch(IOException e){

System.out.println("error reading file ");

}

try{

fin.close();

}

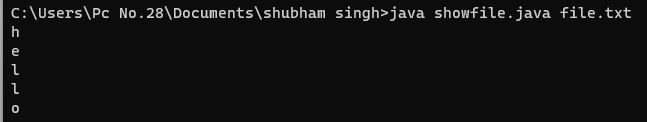
catch(IOException e){

System.out.println("error closing file ");

}

} }

**Output: -**



**B) write a java program to copy the contents from one file to another file.**

**Code:**

import java.io.\*;

class copyfile {

public static void main(String[] args) throws IOException

{

int i;

FileInputStream fin = null;

FileOutputStream fout = null;

if (args.length != 2)

{

System.out.println("Usage: Copy file from to");

return;

}

try

{

fin = new FileInputStream(args[0]);

fout = new FileOutputStream(args[1]);

do

{

i = fin.read();

if (i != -1)

{

fout.write(i);

}

}while (i != -1);

}

catch(IOException e)

{

System.out.println("I/O Error: " + e);

}

finally

{

try

{

if (fin != null)

{

fin.close();

}

}

catch(IOException e2)

{

System.out.println("Error! closing Input File");

}

try

{

if (fout != null)

{

fout.close();

}

}

catch(IOException e2)

{

System.out.println("Error! cloing ouput file");

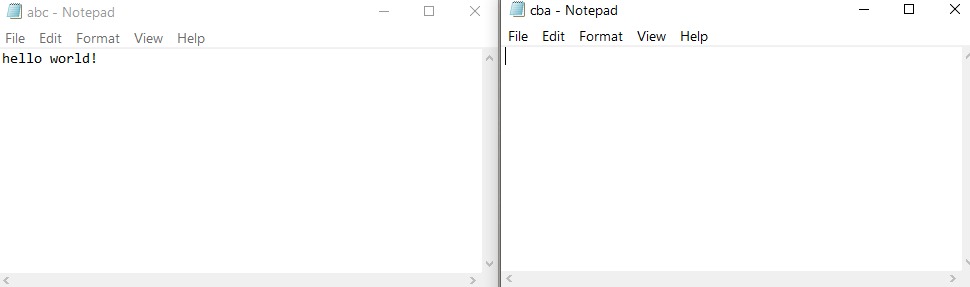
}

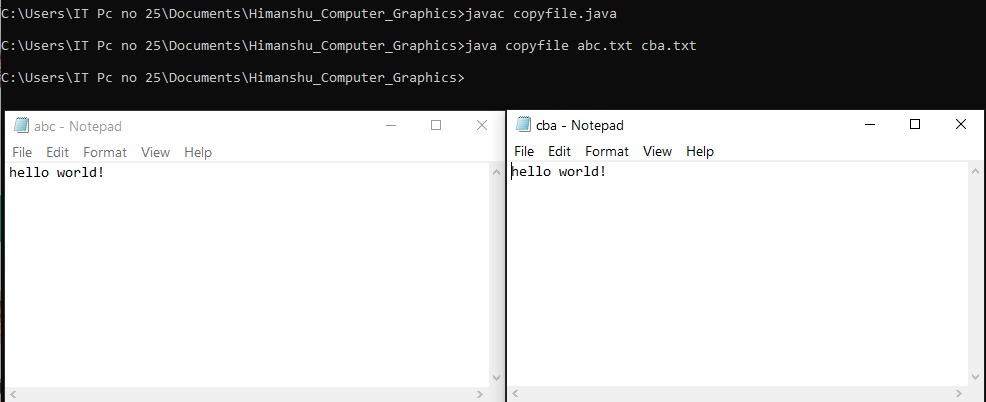
}

}

}

**Output: -**





**9) Exception Handling**

1. **Write a java program to implement exception handling.**

**CODE:**

class exc2

{

public static void main(String args[])

{

int d,a;

try{

d=0;

a=42/d;

System.out.println("This will not printed");

}

catch(ArithmeticException e)

{

System.out.println("Division by zero");

}

System.out.println("After catch Statement");

}

}

**OUTPUT: -**



**B)Write a java program to implement exception handling with multiple catches.**

**CODE:**

class MultipleCatches

{

public static void main(String args[])

{

try

{

int a= args.length;

System.out.println("a="+a);

int b=42/a;

int c[]={1};

c[42]=99;

}

catch(ArithmeticException e)

{

System.out.println("Divide by 0:"+e);

}

catch(ArrayIndexOutOfBoundsException e)

{

System.out.println("Array index oob:"+e);

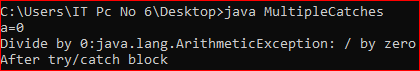
}

System.out.println("After try/catch block");

}

}

**OUTPUT: -**



**10) GUI Programming**

**A) Construct a simple calculator using the JAVA awt with 3 text fields (the 3rd text field should be read-only) and 4 buttons to add, subtract, multiply, divide with minimum functionality.**

**Code:-**

import java.awt.\*;

import java.awt.event.\*;

class AwtMaths extends Frame implements ActionListener

{

TextField tf1, tf2;

Button btnAdd, btnSub,btnMul,btnDiv,btnClear;

Label n,l1,l2,r;

AwtMaths()

{

n = new Label("AWT Basic Maths Calculator Program");

l1 = new Label ("Enter First number");

l2 = new Label ("Enter Second number");

r = new Label();

tf1 = new TextField();

tf2 = new TextField();

btnAdd=new Button("Add");

btnSub = new Button("Sub");

btnMul = new Button("Mult");

btnDiv = new Button("Div");

btnClear = new Button("Clear");

n.setBounds(50, 40, 250, 20);

l1.setBounds(50, 70, 150, 20);

tf1.setBounds (50, 90, 190, 30);

l2.setBounds(50, 120, 150, 20);

tf2.setBounds(50, 140, 190, 30);

btnAdd.setBounds(50, 180, 50, 30);

btnSub.setBounds (100, 180, 50, 30);

btnMul.setBounds (150, 180, 50, 30);

btnDiv.setBounds (200, 180, 50, 30);

btnClear.setBounds (250, 180, 50, 30);

r.setBounds(50, 220, 200, 20);

add(n);

add(l1);

add(tf1);

add (l2);

add(tf2);

add(btnAdd);

add(btnSub);

add(btnMul);

add(btnDiv);

add (btnClear);

add(r);

setSize(350,280);

setLayout(null);//no layout manager

setVisible(true);//now frame will be visible, by default not visible

btnAdd.addActionListener(this); btnSub.addActionListener(this);

btnMul.addActionListener(this); btnDiv.addActionListener(this);

btnClear.addActionListener(this);

addWindowListener(new WindowAdapter()

{

public void windowClosing(WindowEvent e)

{

dispose();

}

});

}

public void actionPerformed (ActionEvent e)

{

int num1 = Integer.parseInt(tf1.getText());

int num2 = Integer.parseInt(tf2.getText());

int result;

if(e.getSource() ==btnAdd)

{

result=num1+num2;

r.setText("Addition is "+result);

}

if(e.getSource() ==btnSub)

{

result=num1-num2;

r.setText("Subtraction is "+result);

}

if(e.getSource() ==btnMul)

{

result=num1\* num2;

r.setText("Multiplication is "+result);

}

if(e.getSource() ==btnDiv)

{

result=num1/num2;

r.setText("Division is "+result);

}

if(e.getSource() ==btnClear)

{

r.setText("");

tf1.setText("");

tf2.setText("");

}

}

public static void main(String[] args)

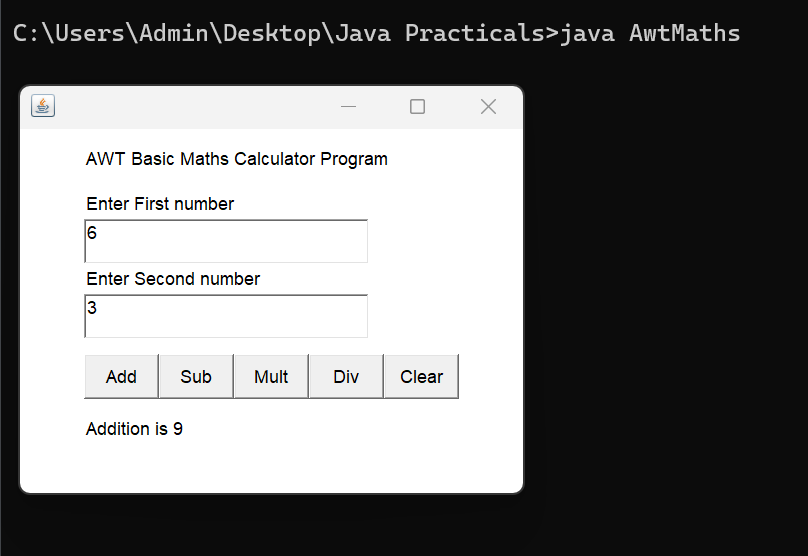
{

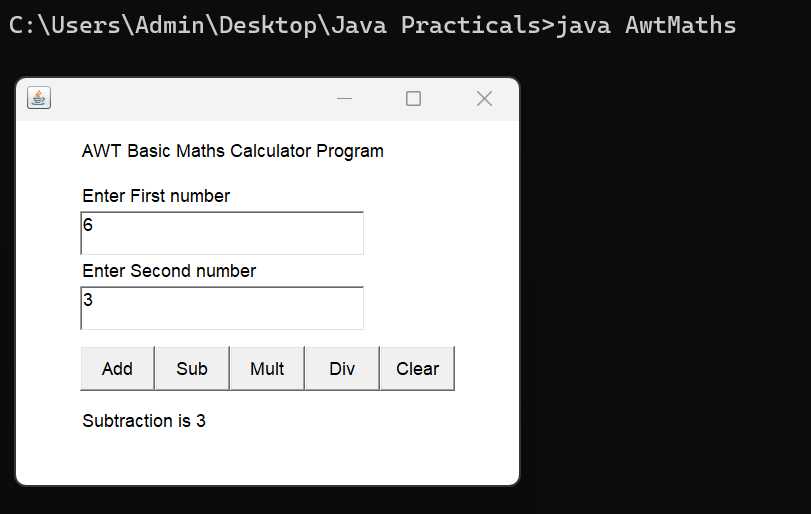
new AwtMaths();

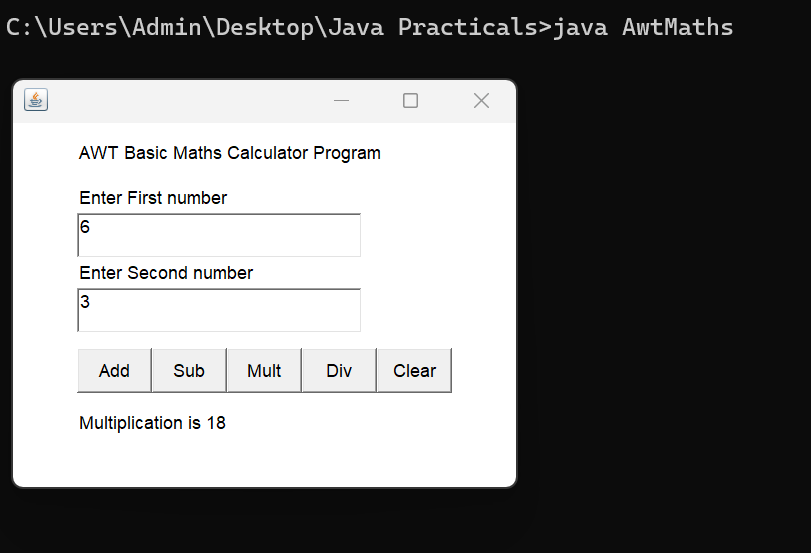
}

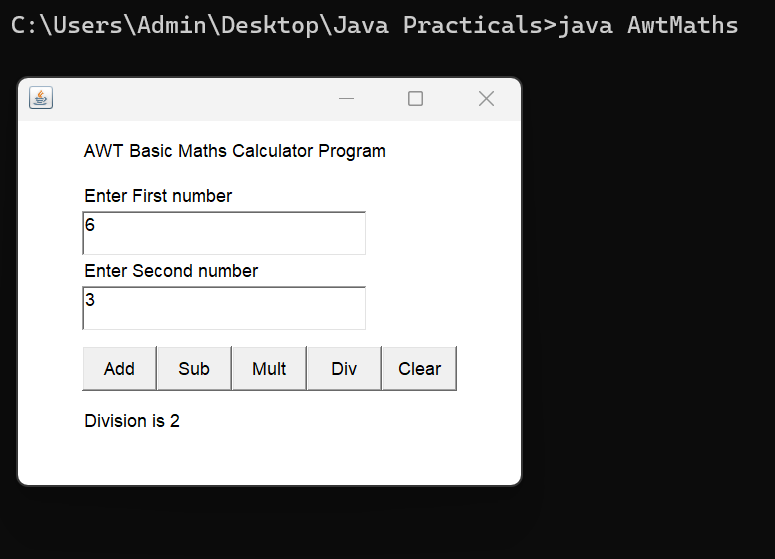
}

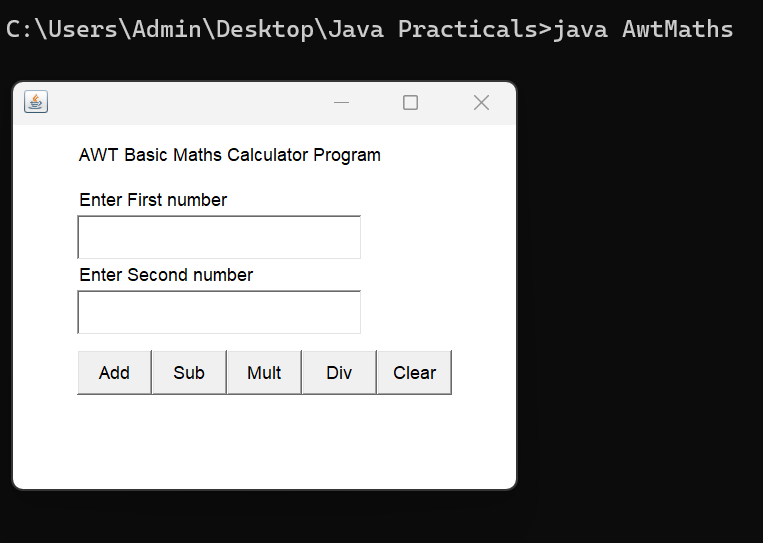
**Output: -**











**C) Design a AWT program to print the factorial for an input value**

**Code: -**

import java.awt.\*;

import java.awt.event.\*;

class AwtFactorials extends Frame implements ActionListener

{

Button btn;

TextField tf1;

Label n,lb1,r;

AwtFactorials()

{

n= new Label("Awt factorial program");

lb1= new Label("Enter the number");

r= new Label();

tf1= new TextField();

btn= new Button("Factorial");

n.setBounds(50,40,200,20);

lb1.setBounds(50,70,180,20);

tf1.setBounds(50,100,160,20);

btn.setBounds(50,130,140,20);

r.setBounds(50,160,120,20);

add(n);

add(lb1);

add(tf1);

add(btn);

add(r);

setSize(300,250);

setLayout(null);

setVisible(true);

btn.addActionListener(this);

addWindowListener(new WindowAdapter()

{

public void windowClosing(WindowEvent e)

{

dispose();

}

});

}

public void actionPerformed(ActionEvent e)

{

int num1 = Integer.valueOf(tf1.getText());

int fact=1;

while(num1>0)

{

fact=fact\*num1;

num1--;

}

r.setText("factorial is"+fact);

}

public static void main(String [] args)

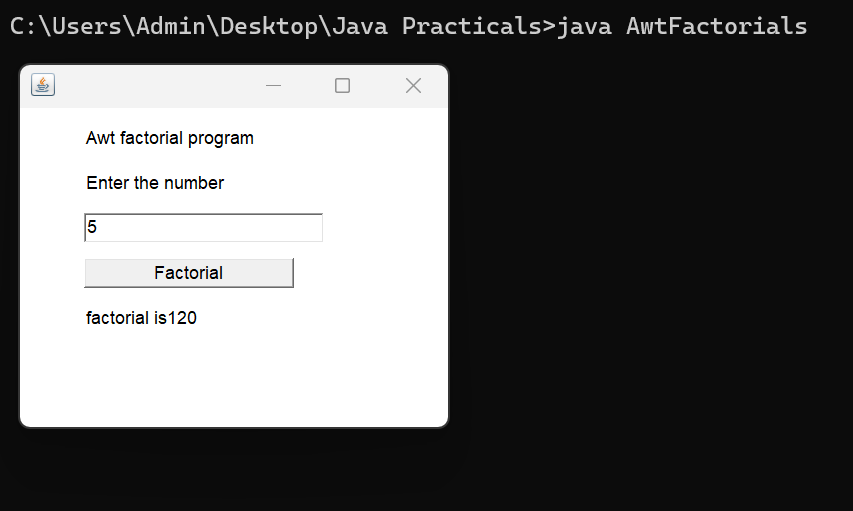
{

new AwtFactorials();

}

}

**Output: -**

****