**PRACTICAL FILE Of**

**Java Programming LAB**



**SUBMITTED BY: SUBMITTED TO:**

***GAURAV BHARDWAJ Dr. KRISHNA PAL SHARMA***

***18103034 ASSISTANT PROFESSOR***

***CSE/5th SEMESTER CSE DEPARTMENT***

***GROUP: G2***

**COMPUTER SCIENCE AND ENGINEERING DEPARTMENT**

**DR. B.R. AMBEDKAR NATIONAL INSTITUTE OF TECHNOLOGY JALANDHAR**

Table of Contents

|  |  |  |  |
| --- | --- | --- | --- |
| **S. No.** | **Topics** | **Page No.** | **Sign.** |
| 1. | Lab 1 | 3 – 11 |  |
| 2. | Lab 2 | 12 – 22 |  |
| 3. | Lab 3 | 23 – 32 |  |
| 4. | Lab 4 | 33 – 38 |  |
| 5. | Lab 5 | 39 – 48 |  |
| 6. | Lab 6 | 49 – 53 |  |
| 7. | Lab 7 | 54 – 59 |  |
| 8. | Lab 8 | 60 – 65 |  |
| 9. | Lab 9 | 66 – 72 |  |
| 10. | Lab 10 | 73 – 85 |  |
| 11. | Scientific Calculator | 86 – 96 |  |

# **Lab 1**

1. Find the prime numbers between a range from x to y where integers x and y are entered from the command line.

->| class PrimeN{

public static void main(String args[]) {

System.out.println("Name: Gaurav Bhardwaj\tRoll no.: 18103034\n");

int a = Integer.parseInt(args[0]);

int b = Integer.parseInt(args[1]);

int j;

System.out.println("Limit =" +a + " : "+ b);

for(int i = a;i<b;i++){

for(j=2;j<i;j++){

if(i%j == 0){j=i;}

}

if(j == i){

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

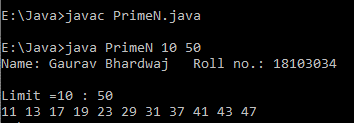
}

}

}

}

**Output:**



1. Write a program to differentiate the static and instance variable in java:

->| class Diff{

int a= 1;

static int b =2;

public static void main(String args[]){

System.out.println("Name: Gaurav Bhardwaj\tRollno.: 18103034\n");

Diff c1 = new Diff(); //class obj c1

Diff c2 = new Diff(); //class obj c2

System.out.println("Instance(c1): " + c1.a +"\tInstance(c2): " + c2.a);

System.out.println("Static(c1): " + c1.b + "\tStatic(c2): " + c2.b + "\n\nChanging c1=>\n");

c1.a = 10;c1.b = 6; // changing the values of c1 obj variables

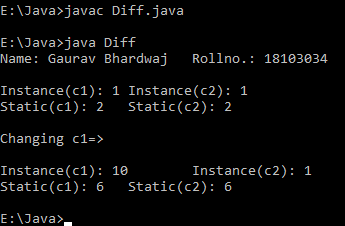
System.out.println("Instance(c1): " + c1.a +"\tInstance(c2): " + c2.a);

System.out.println("Static(c1): " + c1.b + "\tStatic(c2): " + c2.b);

}

}

**Output:**



1. WAP to demonstrate the operations of the bitwise operators in java:

->| class Operators{

public static void main(String args[]){

System.out.println("Name: Gaurav Bhardwaj\tRollno.: 18103034\n");

int a = Integer.parseInt(args[0]);

int b = Integer.parseInt(args[1]);

System.out.println("A: " + (a)+"\tB:" + (b));

System.out.println("AND: " + (a&b));

System.out.println("OR: " + (a|b));

System.out.println("NOT A: " + (~a)+"\tNOT B: " + (~b));

System.out.println("XOR: " + (a^b));

System.out.println("Left Shift A: " + (a<<1)+"\tLeft Shift B: " +(b<<1));

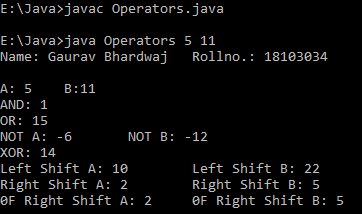
System.out.println("Right Shift A: " + (a>>1)+"\tRight Shift B: " +(b>>1));

System.out.println("0F Right Shift A: " + (a>>>1)+"\t0F Right Shift B: " +(b>>>1));

}

}

**Output:**



1. WAP to count the number of command line arguments passed and also print them.

->| class NoArgs{

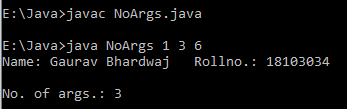
public static void main(String args[]){

System.out.println("Name: Gaurav Bhardwaj\tRollno.: 18103034\n");

System.out.println("No. of args.: " + args.length);

} }

**Output:**



1. Write a program to show the use of labelled break and continue statements:

->| class Labels{

public static void main(String args[]) {

System.out.println("Name: Gaurav Bhardwaj\tRoll no.: 18103034\n");

int k=4;

for(int i=k;i<10;i++){

if(i%3 == 0)

break;

if(i ==5)

continue;

else

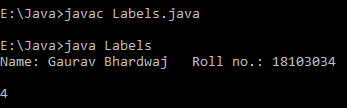
System.out.println(i);

}

}

}

**Output:**



1. Write a program to find the multiplication of two matrices using java.

->| class Matrix{

public static void main(String args[]){

System.out.println("Name: Gaurav Bhardwaj\tRollno.: 18103034\n");

int[][] A = {{1,0,0},{0,1,0},{0,0,1}};

int[][] B = {{1,0,0},{0,1,0},{0,0,1}};

int[][] C= new int[3][3];

int sum =0,k=0;

System.out.println("Matrix A:");

for(int i =0 ;i<3;i++){

for(int j=0;j<3;j++){System.out.print(A[i][j]+" ");}

System.out.println();}

System.out.println("Matrix B:");

for(int i =0 ;i<3;i++){

for(int j=0;j<3;j++){System.out.print(B[i][j]+" ");}

System.out.println();}

System.out.println("Matrix Multiplication =>");

for(int i =0 ;i<3;i++){

for(int j=0;j<3;j++){

for(k=0;k<3;k++){

sum += A[i][j]\*B[j][k];

C[i][j] = sum;}sum = 0;

}

}

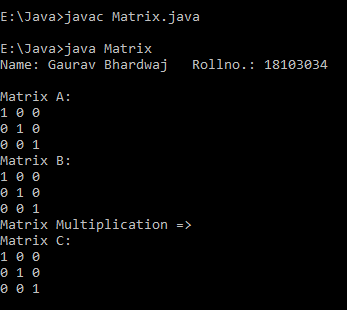
System.out.println("Matrix C:");

for(int i =0 ;i<3;i++){

for(int j=0;j<3;j++){System.out.print(C[i][j]+" ");}

System.out.println();}

}}



1. Write a program to find the sum of all elements in an array using for-each version of for loop.

->|class ArrayD{

public static void main(String args[]){

System.out.println("Name: Gaurav Bhardwaj\tRollno.: 18103034\n");

int A[] ={1,2,1,0,1,0,-1,3,2};

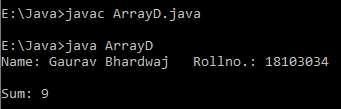
int sum=0;

for(int x: A){sum += x;}

System.out.println("Sum: " + sum);

}}

**Output:**

****

1. Write a program to define and implement a class Room which has parameters height, width and depth. Add two methods in the class to find volume of the room and area of the floor. The volume method will print the volume, while the area method will return the area to the caller of the method. Use the default, parameterised and copy constructor to initialize the objects.

->|class Room{

int length,width,height;

public void Volume(){System.out.println("Volume: " + (length\*width\*height));}

public int Area(){

int area = length\*width;

return area;

}

Room(){length=1;width=1;height=1;}

Room(int l,int w,int h){length=l;width=w;height=h;}

Room(Room r){length=r.length;width=r.width;height=r.height;}

public static void main(String args[]){

System.out.println("Name: Gaurav Bhardwaj\tRollno.: 18103034\n");

Room R1 = new Room();

Room R2 = new Room(3,4,5);

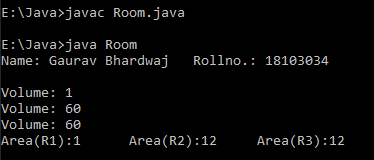
Room R3 = new Room(R2);

R1.Volume();R2.Volume();R3.Volume();

int arr1 = R1.Area(),arr2 = R2.Area(),arr3= R3.Area();

System.out.println("Area(R1):" + arr1+"\tArea(R2):" + arr2+"\tArea(R3):" + arr3);

}}



# **Lab 2**

1. **Write a program to demonstrate the use of static block, static variables and static methods. Also demonstrate that static variables are not the part of the object and they are accessed using class names.**

-> | class StaticDemo{

int a;

static int c;

static void method(){

c = 20;

System.out.println("static c = " + c);

System.out.println("This is statc method.");

}

static {

System.out.println("This is a static Block.");

c = 10;

System.out.println("static c = " + c);

}

public static void main(String args[]){

System.out.println("Gaurav Bhardwaj\t\t18103034");

StaticDemo sd = new StaticDemo();

sd.a = 2; //True : else error will come if we write (a =2)

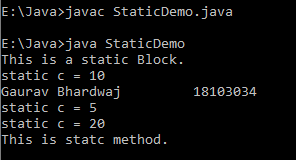
sd.c = 5;

System.out.println("static c = " + c);

StaticDemo.method();

} }

**Output:**

****

1. **Write a program to show the use of this keyword.**

-> | class ThisDemo{

int a;

int b;

ThisDemo(){

a=1;

b=2;

}

void method(int a,int b){

this.a = a;

this.b = b;

}

public static void main(String args[]){

System.out.println("Gaurav Bhardwaj\t\t18103034");

ThisDemo td = new ThisDemo();

System.out.println("Initially a: " + td.a + " b: " + td.b);

td.a = 2;

td.b = 5;

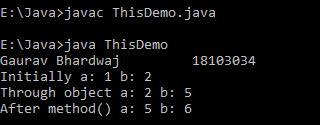
System.out.println("Through object a: " + td.a + " b: " + td.b);

td.method(5,6);

System.out.println("After method() a: " + td.a + " b: " + td.b);

} }

**Output:**

****

1. **Write a program to demonstrate the two ways of method calling (Call by value and call by reference).**

-> | class ValVsRef\_Demo{

int a;

int b;

ValVsRef\_Demo(){

a=1;

b=2;

}

void method(ValVsRef\_Demo obj){

a = obj.a;

b = obj.b;

}

void method(int i,int j){

a = i;

b = j;

}

public static void main(String args[]){

System.out.println("Gaurav Bhardwaj\t\t18103034");

ValVsRef\_Demo vr1 = new ValVsRef\_Demo();

System.out.println("Initially a: " + vr1.a + " b: " + vr1.b);

vr1.method(10,20);

System.out.println("Passing Values a: " + vr1.a + " b: " + vr1.b);

ValVsRef\_Demo vr2 = new ValVsRef\_Demo();

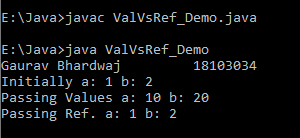
vr1.method(vr2);

System.out.println("Passing Ref. a: " + vr1.a + " b: " + vr1.b);

}

}

**Output:**

****

1. **Demonstrate using a java program that an object can be returned from the method.**

-> | class Demo{

int a;

int b;

Demo(){

a=1;

b=2;

}

Demo methods(){

Demo obj= new Demo();

obj.a = a;

obj.b = b;

return obj;

}

void method(int i,int j){

a = i;

b = j;

}

public static void main(String args[]){

System.out.println("Gaurav Bhardwaj\t\t18103034");

Demo d1 = new Demo();

System.out.println("Initially a: " + d1.a + " b: " + d1.b);

d1.method(10,20);

System.out.println("Values (object 1) a: " + d1.a + " b: " + d1.b);

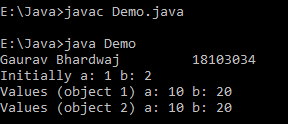
Demo d2 = d1.methods();

System.out.println("Values (object 2) a: " + d2.a + " b: " + d2.b);

}

}

**Output:**

****

1. **Demonstrate the method overloading and constructor overloading in java.**

-> | class Overloader{

int a;

int b;

Overloader(){

a=1;

b=2;

}

Overloader(int a,int b){

this.a=a;

this.b=b;

}

Overloader(int n){

a=n;

b=n;

}

Overloader(Overloader obj){

a=obj.a;

b=obj.b;

}

void method(int i,int j){

a = i;

b = j;

}

void method(Overloader obj){

a = obj.a;

b = obj.b;

}

void method(){a = 100;}

public static void main(String args[]){

System.out.println("Gaurav Bhardwaj\t\t18103034");

Overloader o1 = new Overloader(); // }

Overloader o2 = new Overloader(10,20); // }-- Constructor Overloading

Overloader o3 = new Overloader(25); // }

Overloader o4 = new Overloader(o2); // }

System.out.println("Initially a: " + o1.a + " b: " + o1.b);

System.out.println("Values (object 1) a: " + o1.a + " b: " + o1.b);

System.out.println("Values (object 2) a: " + o2.a + " b: " + o2.b);

System.out.println("Values (object 3) a: " + o3.a + " b: " + o3.b);

System.out.println("Values (object 4) a: " + o4.a + " b: " + o4.b);

o1.method(); // }

o2.method(54,26); // }-- Method Overloading

o3.method(o4); // }

System.out.println("\nValues (object 1) a: " + o1.a + " b: " + o1.b);

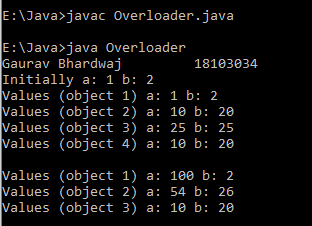
System.out.println("Values (object 2) a: " + o2.a + " b: " + o2.b);

System.out.println("Values (object 3) a: " + o3.a + " b: " + o3.b);

}

}

**Output:**

****

# **Lab 3**

**1. Write a program to demonstrate the overloading of methods with variable arguments. Also demonstrate the situation where the overloading of variable arity methods is not possible.**

**->|** class VariableArgs{

VariableArgs(int ... v){

for(int x : v){System.out.println(x);}

System.out.println();

}

VariableArgs(boolean a, int ... b){

System.out.println(a);

for(int x : b){System.out.println(x);}

System.out.println();

}

VariableArgs(float ... b){

for(float x : b){System.out.println(x);}

System.out.println();

}

public static void main(String args[]){

System.out.println("Name: Gaurav Bhardwaj\t\tRoll no.: 18103034");

VariableArgs v1 = new VariableArgs(1,2,7,8,4,0);

VariableArgs v2 = new VariableArgs(5,2,4);

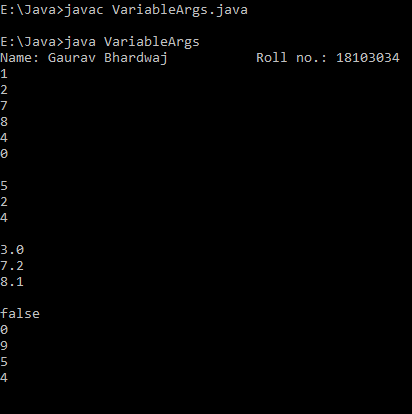
VariableArgs v3 = new VariableArgs(3,7.2f,8.1f);

VariableArgs v = new VariableArgs(false,0,9,5,4);

}

}

**Output:**



**2. Write a single program to demonstrate the concept of single, multilevel and hierarchical inheritance.**

**->|** class A{

A(){System.out.println("A");}

}

class B extends A{

B(){System.out.println("B");}

}

class C extends B{

C(){System.out.println("C");}

}

class D extends B{

D(){System.out.println("D");}

}

class Inheritance{

public static void main(String args[]){

System.out.println("Name: Gaurav Bhardwaj\t\tRoll no.: 18103034");

System.out.println("Single level Inheritance:");

B b = new B();

System.out.println("\nMulti-level Inheritance:");

C c = new C();

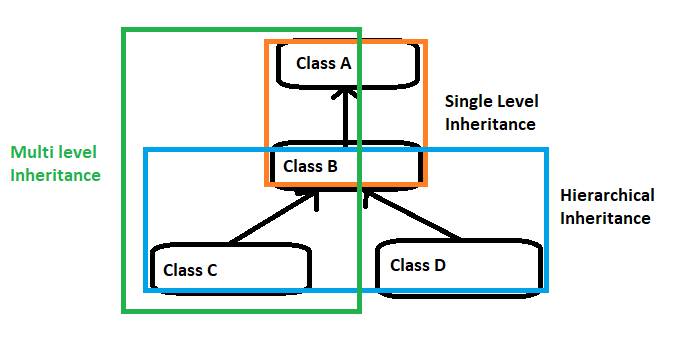
System.out.println("\nHierarchical Inheritance:");

c = new C();

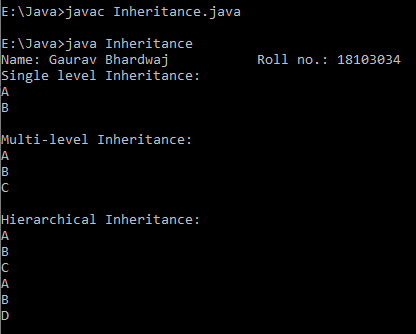
D d = new D();

}

}



**Output:**

****

**3. Write a program to show the concept of method overriding with dynamic method dispatch (runtime polymorphism) and method overloading in a single program which have multiple related classes.**

**-> |** class A{

final int a=10;

int b=20;

A(){System.out.println("A:[a,b]=("+(a)+","+(b)+")");}

void sum(){System.out.println("Sum (A.a + A.b):"+(a+b));}

}

class B extends A{

int a=2;

void print(){

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

sum();

}

final void sum(){

System.out.println("Sum (A.a + B.a):" + (a + super.a));

}

}

class C extends B{

int c =20;

C(){a =15;System.out.println("C.a:" + a);}

final void print(){

System.out.println("a:"+(super.a)+" c:"+(c));

sum();

}

}

class FinalClass{

public static void main(String args[]){

System.out.println("Name: Gaurav Bhardwaj\t\tRoll no.: 18103034");

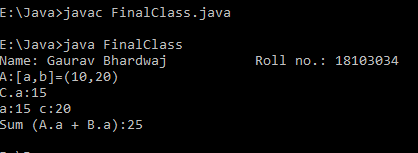
C c = new C();

c.print();

}

}

**Output:**

****

**4. Write a program to show the all three uses of super keyword.**

**-> |** class A{

int a=10,b=20;

A(){System.out.println("A:[a,b]=("+(a)+","+(b)+")");}

void sum(){System.out.println("Sum (A.a + A.b):"+(a+b));}

}

class B extends A{

int a=2,k=5;

B(){

super();

System.out.println("B:[a,k]=("+(a)+","+(k)+")");

}

void sum(){

super.sum();

System.out.println("Sum (A.a + B.k):" + (super.a+k));

}

}

class SuperClass{

public static void main(String args[]){

System.out.println("Name: Gaurav Bhardwaj\t\tRoll no.: 18103034");

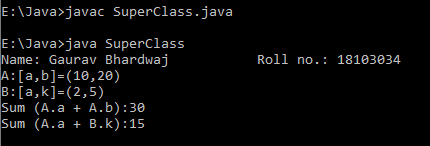
B b = new B();

b.sum();

}

}

**Output:**

****

**5. Write a program to show the all three uses of the final keyword.**

**-> |** final class A{

final int a=10;

int b=20;

A(){System.out.println("A:[a,b]=("+(a)+","+(b)+")");}

void sum(){System.out.println("Sum (A.a + A.b):"+(a+b));}

}

class B extends A{

a=2;

final void print(){

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

sum();

}

void sum(){

System.out.println("Sum (A.a + B.a):" + (a + super.a));

}

}

class C extends B{

int c =20;

C(){a =15;System.out.println("C.a:" + a);}

void print(){

System.out.println("a:"+(a)+" c:"+(c));

sum();

}

}

class FinalClass{

public static void main(String args[]){

System.out.println("Name: Gaurav Bhardwaj\t\tRoll no.: 18103034");

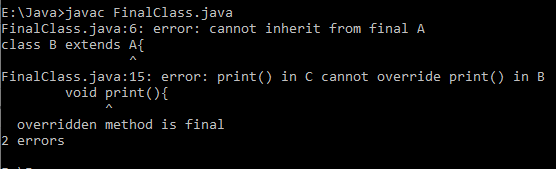
C c = new C();

c.print();

}

}

**Output:**

****

# **Lab 4**

**1. Write a java program to demonstrate the difference amongst constructor,  static and non static initializer blocks. Your program should have at least one constructor in super and subclass.**

**-> |** class Parent{

Parent(){

System.out.println("This is Super class\n");

}

static {System.out.println("This is Super Class's Static block");}

{System.out.println("This is Super Class's Instance block");}

void print(){

System.out.println("This is Super Class's method");

}

}

class Child extends Parent{

Child(){

System.out.println("This is Sub class\n");

}

  static {System.out.println("This is Sub Class's Static block\n");}

{System.out.println("This is Sub Class's Instance block");}

void print(){

super.print();

System.out.println("This is Sub Class's method");

}

}

class Super\_Sub{

public static void main(String args[]){

System.out.println("Gaurav Bhardwaj\tRoll no.: 18103034\n");

Child ch1 = new Child();

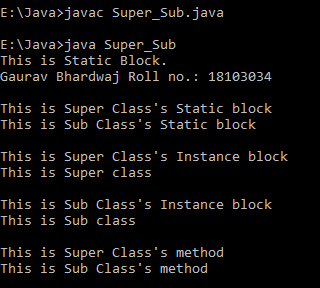
ch1.print();

}

static {System.out.println("This is Static Block.");}

}

**Output:**

****

**2. Write a program to show the usefulness of abstract classes. Inherit an abstract class  in another abstract class and implement the methods of the child abstract class in another class and show the calling of the methods defined in abstract classes using dynamic method dispatching. Your abstract classes must have constructors.**

**-> |** abstract class A1{

A1(){System.out.println("Abstract class A1");}

abstract void print();

}

abstract class A2 extends A1{

A2(){System.out.println("Abstract class A2");}

abstract void show();

public void print(){System.out.println("Hi There!!");}

}

class A3 extends A2{

A3(){System.out.println("Abstract class A3");}

void show(){System.out.println("Show() is called!");}

}

class AbsClasses{

public static void main(String args[]){

System.out.println("Gaurav Bhardwaj\tRoll no.: 18103034\n");

A2 obj = new A3();

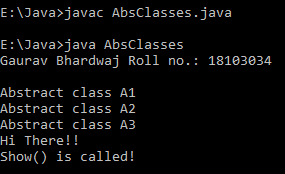
obj.print();

obj.show();

}

}

**Output:**

****

**3. Write a program to demonstrate the use of interfaces. Add the methods and fields in an interface and implement that interface in a class.**

**-> |** interface Int1{ void print(); }

interface Int2 extends Int1{ void show(); }

class A1 implements Int1{

public void print(){ System.out.println("Class A1 <- Interface Int1"); }

}

class A2 implements Int2{

public void print(){ System.out.println("Class A2 <- Interface Int1"); }

public void show(){ System.out.println("Class A2 <- Interface Int2"); }

}

class Interfaces\_In\_Java{

public static void main(String[] args){

System.out.println("Gaurav Bhardwaj\tRoll no.: 18103034\n");

A1 obj = new A1();

obj.print();

A2 obj2 = new A2();

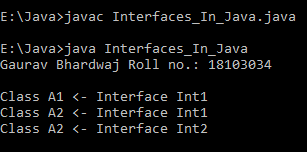
obj2.print();

obj2.show();

}

}

**Output:**



**4. How do you achieve multiple inheritance in java using interfaces? Write a program to show the multiple inheritance implementation using interfaces.**

**-> |** interface Int1{ void print(); }

interface Int2{ void show(); }

class A1 **implements** Int1{

public void print(){

System.out.println("Class A1 <- Interface Int1");

}

}

class A2 **implements** Int1**,**Int2{

public void print(){

System.out.println("Class A2 <- Interface Int1");

}

public void show(){

System.out.println("Class A2 <- Interface Int2");

}

}

class B{

public static void main(String[] args){

System.out.println("Gaurav Bhardwaj\tRoll no.: 18103034\n");

A1 obj = new A1();

obj.print();

A2 obj2 = new A2();

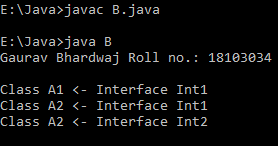
obj2.print();

obj2.show();

}

}

**Output:**

****

**5. Demonstrate how an abstract class can inherit an interface and another abstract class.**

**-> |** interface Int{ public void print(); }

abstract class Ab1 implements Int{

public abstract void show();

public void print(){

System.out.println("Abstract Class Ab1 <- Interface Int");

}

}

abstract class Ab2 extends Ab1{

void display(){super.print();

System.out.println("Abstract Class Ab2 <- Abstract Class Ab1");}

}

class C extends Ab2{

{ display();

System.out.println("Class C <- Abstract Class Ab2");

}

public void show(){

System.out.println("Class C <- Abstract Class Ab1");

}

}

class Abstract\_Interface{

public static void main(String args[]){

System.out.println("Gaurav Bhardwaj\tRoll no.: 18103034\n");

C obj = new C();

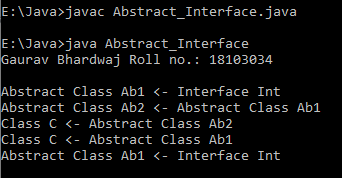
obj.show();

obj.print();

}

}

**Output:**

****

# **Lab 5**

1. **Write a java program that implements educational hierarchy using inheritance.**

**>|** class Office{

int employno,salary;

String employname;

public void getvalue(){

employno = 1;

salary=1000;

employname="Gaurav";

}

}

class Teaching extends Office{

String Designation;

public void setvalue(){

Designation="Professor";

System.out.println("Emplyee Name:" + employname);

System.out.println("Emplyee No.:" + employno);

System.out.println("Salary:" + salary);

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

}

}

class Non\_Teaching extends Office{

String Designation;

public void setvalue(){

Designation="Manager";

System.out.println("Emplyee Name:" + employname);

System.out.println("Emplyee No.:" + employno);

System.out.println("Salary:" + salary);

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

}

}

class Staff{

public static void main(String args[]){

System.out.println("Name: Gaurav Bhardwaj\t Roll no.: 18103034");

Teaching t1 = new Teaching();

t1.getvalue();

t1.setvalue();

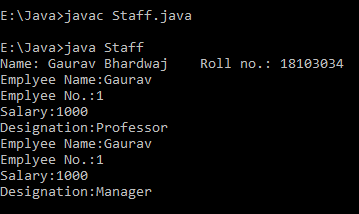
Non\_Teaching t2 = new Non\_Teaching();

t2.getvalue();

t2.setvalue();

}

}

** Output:**

1. **Write a program to demonstrate the different access specifies. Create two packages mypack1 and mypack2 and create at least two classes in each package with members of different visibility (using different access specifiers). Access the members to demonstrate the visibility associated with each of them.**

**>| Java => (MainPack.java):**

import mypack1.\*;

import mypack2.\*;

public class MainPack {

public static void main(String[] args) {

System.out.println("\n\nFrom mypack1, Call Class:\n");

call2 obj = new call2();

System.out.println("\nFrom mypack2, Call2 Class:\n");

call obj2 = new call();

System.out.println("\nFrom mypack1, Derived Class of Protection Class:\n");

derived obj3 = new derived();

}

}

**Package: Java => mypack1(Protection.java):**

package mypack1;

public class Protection{

int num = 1;

private int num\_priv = 2;

protected int num\_prot = 3;

public int num\_pubc = 4;

public Protection(){

System.out.println("Package: mypack1\n Class: protection");

System.out.println("Num: "+num);

System.out.println("Num\_Private"+num\_priv);

System.out.println("Num\_Protected"+num\_prot);

System.out.println("N\_Public"+num\_pubc);

}

}

**(Derived.java):**

package mypack1;

public class Derived extends Protection{

public derived(){

System.out.println("Package: mypack1\n Class: Derived\n SuperClass:Protection");

System.out.println("Num: "+num);

**// Private value are accessible within class only**

**// System.out.println("Num\_Private"+num\_priv);**

System.out.println("Num\_Protected"+num\_prot);

System.out.println("N\_Public"+num\_pubc);

}

}

**(call.java):**

package mypack1;

public class call {

public call(){

Protection obj = new Protection(); // Object of Protection class created

System.out.println("Package: mypack1\nClass:call(Creating Protection object within same package)");

System.out.println("Num: "+obj.num);

**// Within class only(private member)**

**// System.out.println("Num\_Private"+num\_priv)**;

System.out.println("Num\_Protected"+obj.num\_prot);

System.out.println("N\_Public"+obj.num\_pubc);

}

}

**Package: Java => mypack2 (Protection2.java):**

package mypack2;

import mypack1.\*;

public class Protection2 extends Protection{

public Protection2(){

System.out.println("Package: mypack2\n Class: protection2 (Extending Class from other package)");

**//Default value not accessible in different package**

**// System.out.println("Num: "+num);**

**// Private is no accessible in other classes**

**// System.out.println("Num\_Private"+num\_priv);**

**//Protected is not accessible in other package**

**// System.out.println("Num\_Protected"+num\_prot);**

System.out.println("Num\_Public"+num\_pubc);}

}

**(call2.java):**

package mypack2;

import mypack1.\*;

public class call2 {

public call2(){

Protection obj = new Protection();

System.out.println("Package: mypack1\nClass:call(Creating Protection object in different package)");

**//Default value not accessible in different package**

**// System.out.println("Num: "+obj.num);**

**// Private is no accessible in other classes**

**// System.out.println("Num\_Private"+obj.num\_priv);**

**//Protected is not accessible in other package, other class.**

**// System.out.println("Num\_Protected"+obj.num\_prot);**

System.out.println("Num\_Public"+obj.num\_pubc);

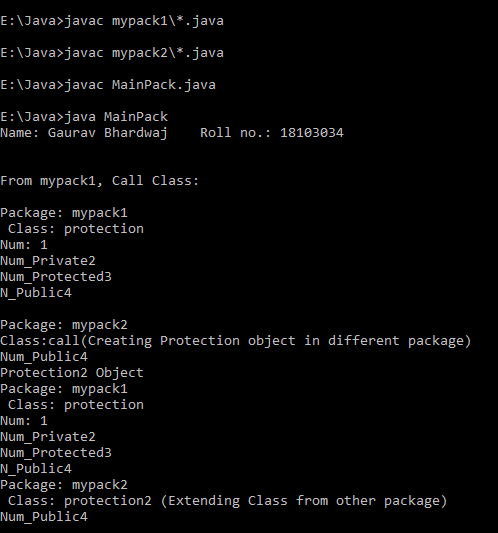
System.out.println("Protection2 Object");

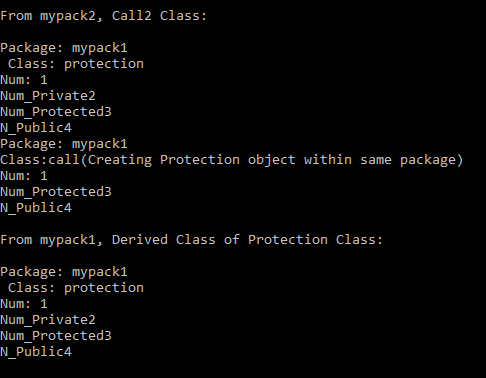
Protection2 obj2 = new Protection2();

}

}

**Output:**

****

****

1. **Write a program to generate the divide by zero exception and handle the exception.**

**>|** class Test{

public static void main(String args[]){

System.out.println("Name: Gaurav Bhardwaj\t Roll no.: 18103034");

try{

int a = 5/0;

}

catch(ArithmeticException A){

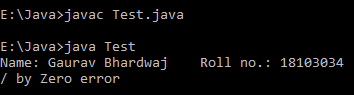
System.out.println("/ by Zero error");

}

}

}

**Output:**

****

1. **Write a java program that implements Array Index out of bound Exception using built-in-Exception.**

**>|** class Test{

public static void main(String args[]){

System.out.println("Name: Gaurav Bhardwaj\t Roll no.: 18103034");

int a[]= {1,2,3,4,5};

try{

a[5] = 6;

}

catch(ArrayIndexOutOfBoundsException A){

System.out.println("Cannot add value to array");

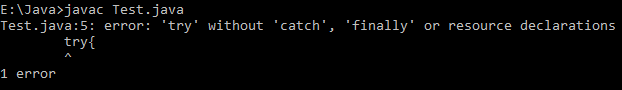
}

}

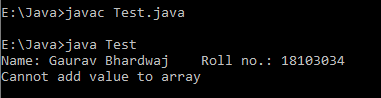
}

**Output:**

**Without catch block:**

****

**With catch block:**

****

# **Lab 6**

1. **Write a java program to show the termination of the program in the presence of ArrayIndexOutOfBoundsException and then write the code to catch the exception.**

**>|** class ExceptionTest{

public static void main(String args[]){

System.out.println("Name: Gaurav Bhardwaj\t Roll no.: 18103034");

int A[]={1,2,3,4,5};

try{

A[5]=6;

}

catch(Exception e){

System.out.println("Exception caught: "+e);

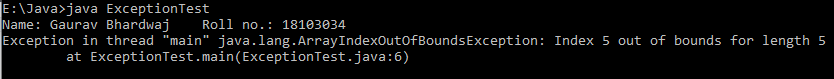
}

System.out.println("No Exceptions");

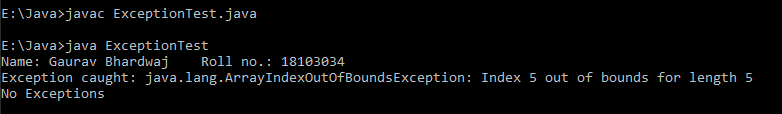
}

}

**Before Exception handling:**

****

**After Exception Handling:**

****

1. **Write a java program to generate multiple exceptions from the java code and handle them using multiple catch blocks and justify that the multiple catch blocks should contain the exception classes in a hierarchy from child to parent (the child class should be defined before the parent).**

**>|** class MultipleExceptionTest{

public static void main(String args[]){

System.out.println("Name: Gaurav Bhardwaj\t Roll no.: 18103034");

int A[]={1,2,3,4,5};

int a = args.length;

try{

int n=3/a;

A[5]=n;

}

catch(ArrayIndexOutOfBoundsException Ae){

System.out.println("Cannot add to the array");

}

catch(ArithmeticException e){

System.out.println("divide by zero error");

}

catch(Exception e){

System.out.println("Error Occured:"+e);

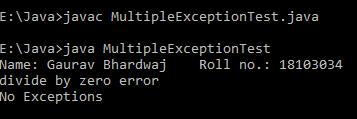
}

System.out.println("No Exceptions");

}

}

**Output:**

****

1. **Write a java program for showing the use of nested try blocks and finally block in the same program.**

**>|** class NestedExceptionTest{

public static void main(String args[]){

System.out.println("Name: Gaurav Bhardwaj\t Roll no.: 18103034");

int A[]={1,2,3,4,5};

try{

try{

int n=3/0;

A[5]=n;

}

catch(ArrayIndexOutOfBoundsException a){

System.out.println("Cannot add to the array");

}

finally{

System.out.println("Exception returned to outer Catch block");

}

}

catch(ArithmeticException e){

System.out.println("divide by zero error");

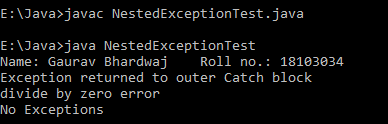
}

System.out.println("No Exceptions");

}

}

**Output:**

****

1. **Write a java program to find the factorial of an integer number and generate an arithmetic exception and handle it in case the user entered a negative number.**

**>|** class Fact{

int fact(int num){

if(num <= 0)

Throw new ArithmeticException();

int f=1,i=1;

while(i <= num){

f = f\*i;

i += 1;

}

return f;

}

public static void main(String args[]){

System.out.println("Name: Gaurav Bhardwaj\t Roll no.: 18103034");

int n= Integer.parseInt(args[0]);

Fact obj = new Fact();

try{

int b = obj.fact(n);

System.out.println("Factorial of "+ n +": "+ b);

}

catch(ArithmeticException e){

System.out.println("Entered no. < 1");

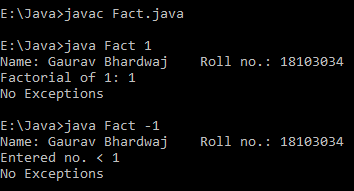
}

System.out.println("No Exceptions");

}

}

**Output:**

****

1. **Write a program to show the use of throws keyword by generating some checked exceptions in some methods and see what will happen if the caller of the methods will not handle the exception.**

**>|** class ThrowsExceptionTest{

void fun() throws InterruptedException{

throw new InterruptedException("Not-Valid");

}

public static void main(String args[]){

System.out.println("Name: Gaurav Bhardwaj\t Roll no.: 18103034");

int A[]={1,2,3,4,5};

ThrowsExceptionTest obj=new ThrowsExceptionTest();

try{

obj.fun();

int n=3/0;

A[5]=n;

}

catch(ArrayIndexOutOfBoundsException a){

System.out.println("Cannot add to the array");

}

catch(ArithmeticException e){

System.out.println("divide by zero error");

}

catch(InterruptedException e){

System.out.println(e);

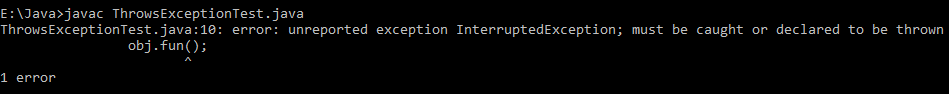
}

System.out.println("No Exceptions");

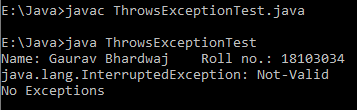
}

}

**Without Catch statement:**

****

**With Catch statement:**

****

# **Lab 7**

**1. Create a class to represent customers in a bank. The data to be stored is: integer Account number, string Name, integer Balance in account. Using the above class input data of 10 customers and perform the following operations.**

1. **Write a function to print the Account number and name of each customer with balance below Rs. 100. And generate a user defined exception “WrongAccountException” in case the account number entered by the user is not integer.**
2. **If a customer requests for withdrawal or deposit, it is given in the form: Account Number, Amount, Code (1 for deposit, 0 for withdrawal) and If on withdrawal the balance falls below Rs. 100 then the program should throw an user defined exception “InsufficientBalanceException”.**

**>|** class InsuffientBalanceException extends Exception{

public String toString(){

return ("You have insuffient amount in your account to withdraw!! (:()");

}

}

class WrongInputException extends Exception{

public String toString(){

return ("You have entered wrong account no.!! (:()");

}

}

class Bank{

int Acc\_no,Balance;

String Name;

public void register(int a,String n,int b){

Acc\_no=a;

Name=n;

Balance=b;

}

public void BelowMinimun(Bank obj[]){

for(Bank acc : obj){

if(acc.Balance < 100){

System.out.println(acc.Name);

}

}

}

// Transanction

public void Transanction(Bank obj[],int an,int amt,int c){

for(int i=0;i<obj.length;i++){

if(obj[i].Acc\_no == an){

if(c == 0){

try{

if(obj[i].Balance < 100 || (obj[i].Balance-amt < 100))

throw new InsuffientBalanceException();

else

obj[i].Balance -= amt;

}

catch(InsuffientBalanceException e){System.out.println("Exception: "+ e);}

}

else if(c == 1){

obj[i].Balance += amt;

}

}

}

}

}

class Banker{

public static void main(String args[]){

System.out.println("Name: Gaurav Bhardwaj \t Roll no.: 18103034");

int[] acc = new int[]{1,2,3,4,5,6,7,8,9,10};

int[] bal = {100,200,150,80,500,67,700,85,140,101};

String[] name ={"John","Gaurav","Sukhi","Mannat","Kartik","Gauri","Gaurang","Gagan","Arsh","Amrit"};

Bank B[] = new Bank[10];

Bank obj = new Bank();

for(int i=0;i<10;i++){

B[i] = new Bank();

B[i].register(acc[i],name[i],bal[i]);

}

obj.Transanction(B,2,50,0);

try{

int ac = Integer.parseInt(args[0]);

int amt= Integer.parseInt(args[1]);}

catch(Exception e){System.out.println("Exception");}

}}

**2. Write a Java program to create three threads by extending Thread class in such a way that the first thread is finding the multiplication of two matrices, second is finding the prime numbers between 1 to 100000 and print the counting, and third is to find GCD of some integers values passed through a large integer array.**

**>|** class Thread1 extends Thread{

Thread1(String name){

new Thread(name);

}

public static int gcd(int a,int b){

if(b==0)

return a;

else

return gcd(b,a%b);

}

public void run(){

Thread1 T1 = new Thread1("Thread 1");

System.out.println(T1);

int[][] a,b,c = new int[3][3];

a = new int[][]{{1,2,3},{4,5,6},{7,8,9}};

b = new int[][]{{1,4,7},{2,5,8},{3,6,9}};

for(int i=0;i<3;i++){

for(int j=0;j<3;j++){

c[i][j] = 0;

for(int k=0;k<3;k++){

c[i][j] += a[i][j]\*b[j][k];

}

System.out.print(c[i][j] + " ");

//T1.sleep(100);

}

System.out.println();

}

Thread1 T2 = new Thread1("Thread 2");

System.out.println(T2);

int l = 100,cnt=0;

for(int i=2;i<l;i++){

int flag=0,K=0,L=l/2;

for(int j=2;j<L;j++){

if(i%j == 0){ flag=1;break; }

else{cnt += 1;K=i;break;}

}

if(flag == 0){

System.out.println("Number: "+ K + " Current count: "+ cnt);

//T2.sleep(300);

}

}

Thread1 T3 = new Thread1("Thread 3");

System.out.println(T3);

int arr[]= {2,5,10,23,13,45,78,100};

for(int i=0;i<7;i++){

int result = gcd(arr[i],arr[i+1]);

System.out.println("GCD("+arr[i] +","+arr[i+1]+"): "+ result);

//T3.sleep(200);

}

}

}

class Main\_Threads\_Thread{

public static void main(String args[]){

System.out.println("Name: Gaurav Bhardwaj \t Roll no.: 18103034");

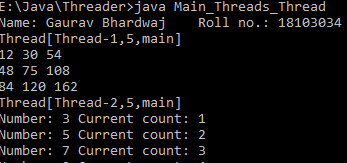
Thread1 t = new Thread1("Main Class");

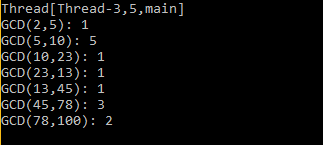
t.start();

}

}

**Output:**

****

****

**3. Write a java program to create three threads by implementing runnable interface, one is main thread and two other are (child) threads called by main threads in such a way that main thread always terminates after the termination of child threads.**

**>|** class Threads\_runable implements Runnable{

public void run(){

System.out.println("Hello");

Thread ct1 = new Thread("Child Thread 1");

Thread ct2 = new Thread("Child Thread 2");

int tm1 = 200,tm2 = 500,i=5;

System.out.println(ct1.getName() + "\t Time=0");

System.out.println(ct2.getName() + "\t Time=0");

while(i != 0){

System.out.println(ct1.getName() + "\t Time=" + tm1);

System.out.println(ct2.getName() + "\t Time=" + tm2);

i -= 1;

}

}

}

class Main\_Thread\_Runnable{

public static void main(String args[]){

System.out.println("Name: Gaurav Bhardwaj \t Roll no.: 18103034");

Threads\_runable tr = new Threads\_runable();

Thread t = new Thread(tr,"Main Thread");

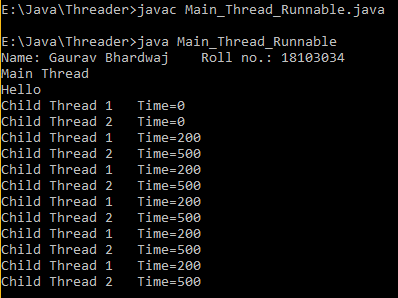
t.start();

System.out.println(t.getName());

}

}

**Output:**



# **Lab 8**

1. **Write a java program to create three threads: one main and two child threads. Control the termination of these threads such that the first thread should terminate first, then seconds and then main.**

**>|** class ChildThread implements Runnable{

Thread t;

ChildThread(){

t= new Thread(this,"child");

t.setPriority(10);

t.start();

}

public void run(){

System.out.println(t+" | Priority:"+t.currentThread().getPriority());

try{t.sleep(2000);}

catch(InterruptedException e){System.out.println("<->");}

}

}

class MainThread{

public static void main(String args[]){

System.out.println("Name: Gaurav Bhardwaj \t Roll no.: 18103034");

Thread T = new Thread("Main-Thread ");

T.start();

ChildThread ct = new ChildThread();

ChildThread ct2 = new ChildThread();

  try{T.sleep(10000);}

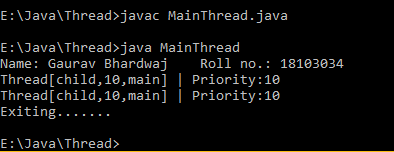
catch(InterruptedException e){System.out.println("<->");}

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

}

}

**Output:**

****

1. **Suppose there is a method to print the table of a number which is quite lazy such that it will wait for some time after printing one number. Two threads are using that common method. Write a program to synchronize the access of this method so that if one thread is using it another can not access until the first will free it. Implement both the ways of synchronization..**

**>|** class Maths implements Runnable{

Thread T;

Maths(){

T= new Thread(this);

T.start();

}

public void run(){

System.out.println("Initiated....");

}

**synchronized** void add(int a,int b){

System.out.print("Addition of "+ a +" & "+b+": ");

try{T.sleep(2000);}

catch(InterruptedException e){System.out.println("<->");}

System.out.println((a+b)+"   >");

}

void multiply(int m,int n){

System.out.print("Multiplication of "+ m +" & "+n+": ");

try{T.sleep(3000);}

catch(InterruptedException e){System.out.println("<->");}

int t= m\*n;

System.out.println(t +"   >");

}

}

class Synchronized{

public static void main(String args[]){

System.out.println("Name: Gaurav Bhardwaj \t Roll no.: 181030304");

Maths m1 = new Maths();

Maths m2 = new Maths();

m1.add(10,20);

**synchronized**(m2){ m2.multiply(77,56); }

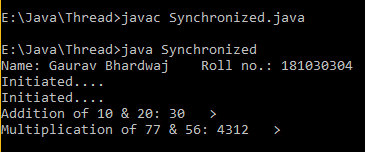
try{m1.T.join();m2.T.join();}

catch(InterruptedException e){System.out.println("..........");}

}

}

**Output:**

****

1. **Write a java program to implement producer and consumer problem. Where the producer thread is putting data and consumer thread is consuming data to/from a shared queue of size 5. Both the threads are running at variable speed (value passed into sleep() needs to be modified every time). After putting 5 values into the queue if no item is consumed by the consumer thread (queue is full) then it needs to wait. Similarly, if there is no data to consume (queue is empty) then the consumer thread needs to wait until the producer does not put data into the queue.**

**>|** class Queue{

int[] Q = {0,0,0,0,0};

int full=0,empty=5;

synchronized void put(int item){

if(empty == 0){System.out.println("Queue full!!");}

else{

Q[full] = item;

full  += 1;

empty -= 1;

System.out.println("Put: "+item);

}

}

synchronized void get(int item){

if(full == 0){System.out.println("Queue empty!!");}

else{

System.out.println("Got: "+Q[item]);

Q[item] = 0;

full  -= 1;

empty += 1;

}

}

}

class Producer implements Runnable{

Thread T;

Queue Q;

Producer(Queue p){

Q=p;

T = new Thread(this);

T.start();

}

public void run(){

for(int j=0;j<10;j++){

if(Q.full != 5){

Q.put((j+1)%5);

try{T.sleep(1000);}

catch(InterruptedException e){System.out.println("..........");}

}

else{

try{Q.wait(2000);}

catch(InterruptedException e){System.out.println("..........");}

}

}

}

}

class Consumer implements Runnable{

Thread T;

Queue Q;

Consumer(Queue c){

this.Q=c;

T = new Thread(this);

T.start();

}

public void run(){

for(int j=0;j<10;j++){

if(Q.full != 0){

Q.get(j%5);

try{T.sleep(3000);}

catch(InterruptedException e){System.out.println("..........");}

}

else{

try{Q.wait(2000);}

catch(InterruptedException e){System.out.println("..........");}

}

}

}

}

class Producer\_Consumer{

public static void main(String args[]){

System.out.println("Name: Gaurav Bhardwaj \t Roll no.: 18103034");

Queue obj = new Queue();

new Producer(obj);

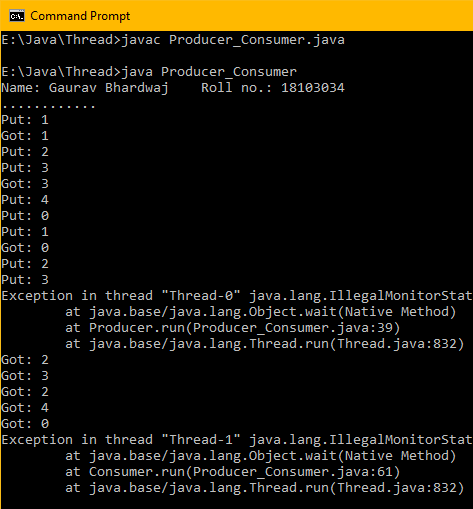
new Consumer(obj);

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

}

}

**Output:**

****

# **Lab 9**

**1. Write a java program to create a new string from two given strings in such a way that each character of new string come alternatively from two strings for example string1=”abcdef” string2=”xyzp” then the new string will be “axbyczdpef”.**

**>|**

import java.io.\*;

import java.util.\*;

class ConcatStrings{

public static void main(String args[]){

System.out.println("Name: Gaurav Bhardwaj \t Roll no.: 18103034");

System.out.print("Enter string 1:");

Scanner s = new Scanner(System.in);

String s1 = s.nextLine();

System.out.print("Enter String 2:");

String s2 = s.nextLine();

String S = " ";

int j=0;

int len1 = s1.length(),len2=s2.length();

for(int i=0;i<len1;i++){

S = S + s1.charAt(i);

if(j != len2){

S = S +s2.charAt(j) ;

j++;

}

}

if(len2 > len1){

while(j != len2){

S = S + s2.charAt(j);

j += 1;

}

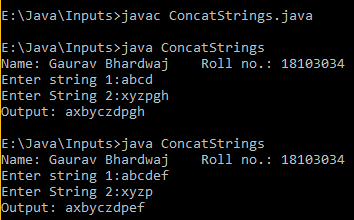
}

System.out.println("Output: " + S);

}

}

**Output:**

****

**2. Write a java program that takes full name (containing first, middle and last name) from the user as input and displays the abbreviations of the first and middle names while the last name is displayed as it is. For example Krishna Pal Sharma should print as K P Sharma.**

**>|** import java.io.\*;

import java.util.\*;

class Form{

public static void main(String args[]){

System.out.println("Name: Gaurav Bhardwaj \t Roll no.: 18103034");

System.out.print("Enter First Name:");

Scanner s = new Scanner(System.in);

String F\_n = s.nextLine();

System.out.print("Enter Last name:");

String L\_n = s.nextLine();

String S="";

int d;

System.out.print("1. Enter Middle Name \t 2. Skip:");

d = Integer.parseInt(s.nextLine());

if(d == 1){

System.out.print("Enter Middle name:");

String M\_n = s.nextLine();

S = S + F\_n.charAt(0) +" "+ M\_n.charAt(0) + " " + L\_n;

}

else{

S = S + F\_n.charAt(0) +" "+ L\_n;

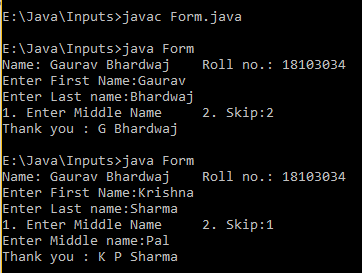
}

System.out.println("Thank you : " + S);

}

}

**Output:**

****

**3. Write a java program to input a string of alphabets and find out the number of occurrences of all alphabets in that string. Also, find out the alphabet with maximum occurrence.**

**>|**

import java.io.\*;

import java.util.\*;

class CharOccurences{

public void collect(char ch[],String s,int len){

if(len == 0){

ch[len] = s.charAt(0);

len += 1;

}

else{

for(int i=0;i<s.length();i++){

int flag=0;

for(char a: ch){

if(s.charAt(i) == a){

flag=1;

break;

}

}

if(flag == 0){

ch[len]=s.charAt(i);

len += 1;

}

}

}

}

public void counter(char ch[],int len,String s){

int arr[]= new int[len+1];

int i=0;

for(int k=0;k<len;k++){

int cnt=0;

for(int j=0;j<s.length();j++)

if(s.charAt(j) == ch[k])

cnt += 1;

arr[i]=cnt;

i += 1;

}

for(int l=0;l<len;l++){

if(arr[l] != 0)

System.out.println("'"+ch[l]+"'"+" : "+ arr[l]);

}

}

public static void main(String args[]){

System.out.println("Name: Gaurav Bhardwaj \t Roll no.: 18103034");

System.out.print("Enter String:");

Scanner S = new Scanner(System.in);

String s = S.nextLine();

int len = s.length();

char ch[]=new char[len+1];

int leng=0;

System.out.println("String :" +"'"+s+"'");

System.out.println("---Summary---");

CharOccurences x = new CharOccurences();

for(int i=0;i<len;i++){

x.collect(ch,s,i);

leng += 1;

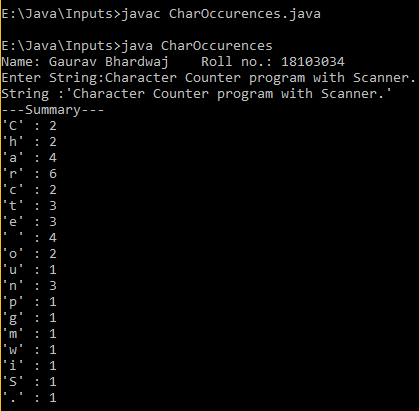
}

x.counter(ch,leng,s);

}

}

**Output:**

****

**4. Write a java program to find the number of vowels, consonants, digits and white space characters in a string.**

**>|**

import java.io.\*;

import java.util.\*;

class StringDiff{

public static void main(String args[]){

System.out.println("Name: Gaurav Bhardwaj \t Roll no.: 18103034");

System.out.print("Enter String:");

Scanner S = new Scanner(System.in);

String s = S.nextLine();

int len = s.length(),v =0,w=0,c=0,d=0;

for(int i=0;i<len;i++){

if(s.charAt(i) == 'a' || s.charAt(i) == 'e' || s.charAt(i) == 'i' || s.charAt(i) == 'o' || s.charAt(i) == 'u' || s.charAt(i) == 'A' || s.charAt(i) == 'E' || s.charAt(i) == 'I' || s.charAt(i) == 'O' || s.charAt(i) == 'U')

v += 1;

else if(s.charAt(i) == '0' || s.charAt(i) == '1' || s.charAt(i) == '2' || s.charAt(i) == '3' || s.charAt(i) == '4' || s.charAt(i) == '5' || s.charAt(i) == '6' || s.charAt(i) == '7' || s.charAt(i) == '8' || s.charAt(i) == '9')

d += 1;

else if(s.charAt(i) == ' ')

w += 1;

else

c += 1;

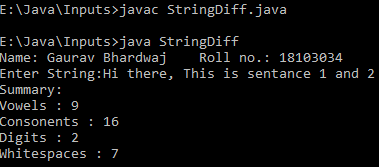
}

System.out.println("Summary:\nVowels : " + v +"\nConsonents : "+ c +"\nDigits : "+d+"\nWhitespaces : "+ w);

}

}

**Output:**

****

**5. Write a java program to find some tokens (keywords, variables and constants) in a java file. Copy all these tokens in a different file and print them with their counting.**

**>|**

import java.io.\*;

import java.util.\*;

class FileInput{

public static void main(String args[]) throws FileNotFoundException{

System.out.println("Name: Gaurav Bhardwaj \t Roll no.: 18103034");

System.out.println("File Contents:");

try{

File file = new File("FileTxt.txt");

Scanner S = new Scanner(file);

while(S.hasNextLine()){

String s = S.nextLine();

System.out.println(s);

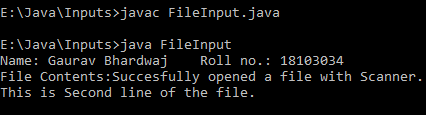
}

}catch(Exception e){}

}

}

**Output:**

****

# **Lab 10**

**1. Write a java program to handle all the events generated by the mouse.**

**>|** import java.awt.\*;

import java.awt.event.\*;

public class MouseHandler extends Frame implements MouseListener{

String msg = " ";

int x,y;

public void MouseHandler(){

addWindowListener(new WindowAdapter(){public void windowClosing(WindowEvent we){System.exit(0);}});

addMouseListener(this);

}

public void paint(Graphics g){

g.drawString(msg, x,y);

}

public void mouseClicked(MouseEvent me){

x=me.getX();

y=me.getY();

msg = "Mouse Clicked";

repaint();

}

public void mouseEntered(MouseEvent me){

x=me.getX();

y=me.getY();

msg = "Mouse Entered ";

repaint();

}

public void mouseExited(MouseEvent me){

x=me.getX();

y=me.getY();

msg = "Mouse Exited ";

repaint();

}

public void mousePressed(MouseEvent me){

x=me.getX();

y=me.getY();

msg = "Mouse Pressed ";

repaint();

}

public void mouseReleased(MouseEvent me){

x=me.getX();

y=me.getY();

msg = "Mouse Released ";

repaint();

}

public static void main(String args[]){

MouseHandler appwin=new MouseHandler();

appwin.setSize(new Dimension(350,350));

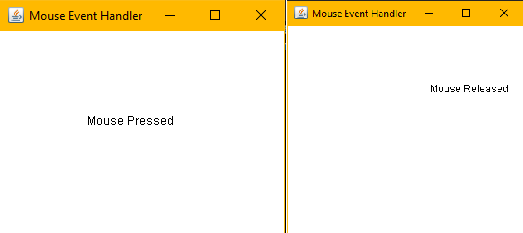
appwin.setTitle("Mouse Event Handler");

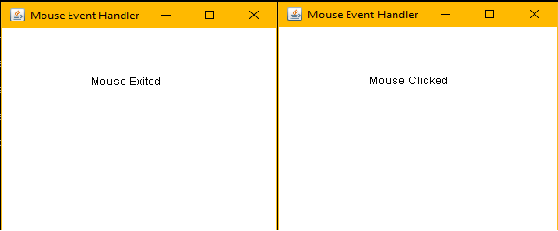
appwin.setVisible(true);

}

}

**Output:**

****

****

**2. Write a java program to handle all the events generated by the keyboard.**

**>|**import java.awt.\*;

import java.awt.event.\*;

public class KeyEvents extends Frame implements KeyListener{

String msg = "";

public KeyEvents(){

addKeyListener(this);

addWindowListener(new WindowAdapter(){public void windowClosing(WindowEvent we){System.exit(0);}});

}

public void keyPressed(KeyEvent ke){msg="Key Pressed";repaint();}

public void keyReleased(KeyEvent ke){msg="Key Released";repaint();}

public void keyTyped(KeyEvent ke){msg="Typing";repaint();}

public void paint(Graphics g) {

g.drawString(msg, 100, 150);

}

public static void main(String args[]) {

KeyEvents appwin=new KeyEvents();

appwin.setSize(new Dimension(300,300));

appwin.setTitle("Key Event Handler");

appwin.setVisible(true);

}

}

**Output:**

****

**3. Write a java program to handle mouse move event and mouse dragged event using anonymous classes.**

**>|** import java.awt.\*;

import java.awt.event.\*;

public class MouseEvents extends Frame{

String msg = "";

int mouseX = 0, mouseY = 0; // coordinates of mouse

public MouseEvents() **{**

addMouseListener(new MyMouseAdapter(this));

addMouseMotionListener(new MyMouseAdapter(this));

addWindowListener(new WindowAdapter(){public void windowClosing(WindowEvent we){System.exit(0);}});

**}**

class MyMouseAdapter extends MouseAdapter{

MouseEvents M;

public MyMouseAdapter(MouseEvents m){M = m;}

// Mouse Click

public void mouseClicked(MouseEvent me) {

M.mouseX = 100;

M.mouseY = 100;

M.msg = "Mouse Clicked";

M.repaint();

}

// Mouse Enter

public void mouseEntered(MouseEvent me) {

M.mouseX = 100;

M.mouseY = 100;

M.msg = "Mouse Entered";

M.repaint();

}

// Mouse Exit

public void mouseExited(MouseEvent me) {

M.mouseX = 100;

M.mouseY = 100;

M.msg = "Mouse Exited";

M.repaint();

}

// Button Press

public void mousePressed(MouseEvent me) {

M.mouseX = me.getX();

M.mouseY = me.getY();

M.msg = "Mouse Pressed";

M.repaint();

}

// Button Release

public void mouseReleased(MouseEvent me) {

M.mouseX = me.getX();

M.mouseY = me.getY();

M.msg = "Mouse Released";

M.repaint();

}

// Mouse Drag

public void mouseDragged(MouseEvent me) {

M.mouseX = me.getX();

M.mouseY = me.getY();

M.msg = "Dragging Mouse";

M.repaint();

}

// Mouse Move

public void mouseMoved(MouseEvent me) {

M.msg="Moving Mouse";

M.repaint();

}

}

// Display msg in applet window at current X,Y location.

public void paint(Graphics g) {

g.drawString(msg, mouseX, mouseY);

}

public static void main(String args[]) {

MouseEvents appwin=new MouseEvents();

appwin.setSize(new Dimension(300,300));

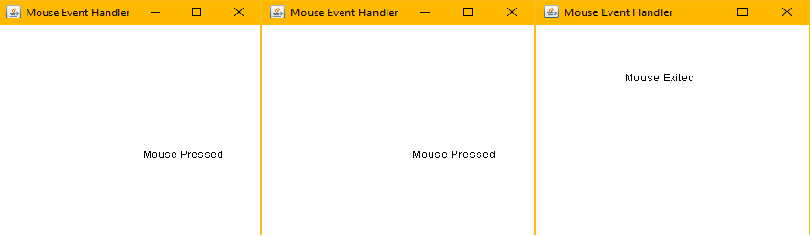
appwin.setTitle("Mouse Event Handler");

appwin.setVisible(true);

}

}

**Output:**

****

**4. Write a java program to add two numbers using a GUI based frame where two numbers are passed in text fields and sum is displayed in text field on pressing the button sum. Also add one button to clear all fields.**

**>|** import java.awt.\*;

import java.awt.event.\*;

import javax.swing.\*;

public class SimpleCalculator extends Frame implements ActionListener{

String sum=" ";

TextField tf1,tf2,tf3;

Label L1,L3,L2;

public SimpleCalculator(){

//Layout

setLayout(new GridLayout(4,2));

//InputBox

tf1 = new TextField("0",1);

tf2 = new TextField("0",1);

tf3 = new TextField(" ",1);

//OutputScreen

L1 = new Label("Number 1:",Label.CENTER);

L2 = new Label("Number 2:",Label.CENTER);

L3 = new Label("Sum:",Label.CENTER);

//Buttons

Button btn\_sum = new Button(" Sum ");

Button btn\_clear = new Button("Clear");

add(L1);

add(tf1);

add(L2);

add(tf2);

add(L3);

add(tf3);

add(btn\_sum);

add(btn\_clear);

//Listener

btn\_sum.addActionListener(this);

btn\_clear.addActionListener(this);

addWindowListener(new WindowAdapter(){public void windowClosing(WindowEvent we){System.exit(0);}});

}

public void actionPerformed(ActionEvent e){

String s = e.getActionCommand();

if(s.equals(" Sum ")){

float a = Float.parseFloat(tf1.getText());

float b = Float.parseFloat(tf2.getText());

sum = Float.toString(a+b);

tf3.setText(sum);

}

else if(s.equals("Clear")){

tf1.setText(" ");

tf2.setText(" ");

tf3.setText(" ");

}

}

public static void main(String args[]){

SimpleCalculator appwin=new SimpleCalculator();

appwin.setSize(new Dimension(350,350));

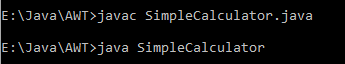
appwin.setTitle("Simple Calculator");

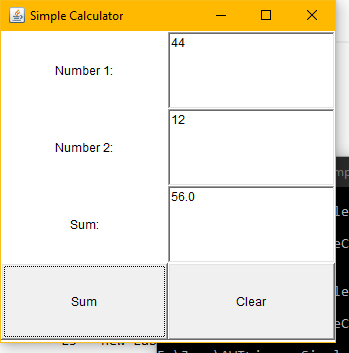
appwin.setVisible(true);

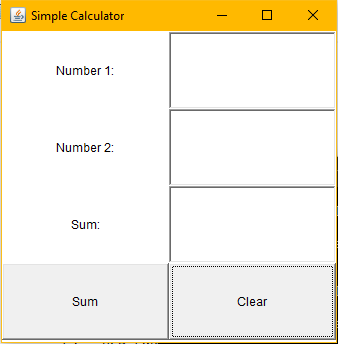
}

}

**Output:**







# Scientific Calculator

**Write a program to make a scientific calculator in java:**

**>|** import java.awt.\*;

import java.awt.event.\*;

import javax.swing.\*;

public class ScientifiCalculator extends Frame implements ActionListener{

String txt="";

TextField tf;

Label L1;

int flag = 0,dflag =0;

Button btn\_DRG;

public static String eval(String tx){

int j=0;

int len = tx.length();

for(int i=0;i<len;i++){

char c= tx.charAt(i);

if(c == '+' || (c == '-' && i != 0) || c == '/' || c == 'x' ){

j=i;

break;

}

}

char ch = tx.charAt(j);

String t1= tx.substring(0,j);

String t2= tx.substring(j+1,len);

Double a = Double.parseDouble(t1);

Double b = Double.parseDouble(t2);

if(ch == '/'){

if(b != 0)

tx = Double.toString(a/b);

else

tx = "You just reach \u221E";

}

else if(ch == 'x')

tx = Double.toString(a\*b);

else if(ch == '+')

tx = Double.toString(a+b);

else if(ch == '-')

tx = Double.toString(a-b);

return tx;

}

public static Double Degree\_Radian\_Conversion(Double x,int fl){

// flag = 1 -> Degree to Radian

if(fl == 0){

//To Degree

x = (180/3.1415926)\*x;

//set Degree flag = 1;

fl=1;

}

else if(fl == 1){

//To Radian

x = (3.1415926/180)\*x;

//set Degree flag = 0;

fl =0;

}

return x;

}

public ScientifiCalculator(){

//Layout

setLayout(new BoxLayout(this,BoxLayout.Y\_AXIS));

Panel f = new Panel();

Panel temp = new Panel();

Panel f2 = new Panel();

temp.setLayout(new GridLayout(1,2));

f.setLayout(new GridLayout(5,4));

f2.setLayout(new GridLayout(5,3));

//InputBox

tf = new TextField("0");

//tf.setHorizontalAlignment(-1);

tf.setMinimumSize(new Dimension(350, 20));

tf.setPreferredSize(new Dimension(350, 30));

tf.setMaximumSize(new Dimension(500, 50));

//OutputScreen

L1 = new Label(txt,Label.RIGHT);

L1.setMinimumSize(new Dimension(350, 20));

L1.setPreferredSize(new Dimension(350, 30));

L1.setMaximumSize(new Dimension(350, 50));

//Buttons

Button btn\_cls = new Button("C");

Button btn\_bracket = new Button("( )");

Button btn\_prcnt = new Button("%");

Button btn\_1 = new Button("1");

Button btn\_2 = new Button("2");

Button btn\_3 = new Button("3");

Button btn\_add = new Button("+");

Button btn\_4 = new Button("4");

Button btn\_5 = new Button("5");

Button btn\_6 = new Button("6");

Button btn\_sub = new Button("-");

Button btn\_7 = new Button("7");

Button btn\_8 = new Button("8");

Button btn\_9 = new Button("9");

Button btn\_pro = new Button("x");

Button btn\_0 = new Button("0");

Button btn\_dec = new Button(".");

Button btn\_div = new Button("/");

Button btn\_ans = new Button("=");

Button btn\_rm = new Button("<");  // Remove the last element

btn\_DRG = new Button("R");

Button btn\_sqrt = new Button("\u221a");

Button btn\_fact = new Button("n!");

Button btn\_sin = new Button("Sin");

Button btn\_cos = new Button("Cos");

Button btn\_tan = new Button("Tan");

Button btn\_ln = new Button("ln");

Button btn\_log = new Button("log");

Button btn\_in = new Button("1/x");

Button btn\_ex = new Button("e^x");

Button btn\_x2 = new Button("x^2");

Button btn\_xy = new Button("x^y");

Button btn\_abs = new Button("|x|");

Button btn\_pi = new Button("\u03C0");

Button btn\_e = new Button("e");

//adding widgets

add(tf);

add(L1);

add(temp);

temp.add(f2);

f2.add(btn\_DRG);

f2.add(btn\_sqrt);

f2.add(btn\_fact);

f2.add(btn\_sin);

f2.add(btn\_cos);

f2.add(btn\_tan);

f2.add(btn\_ln);

f2.add(btn\_log);

f2.add(btn\_in);

f2.add(btn\_ex);

f2.add(btn\_x2);

f2.add(btn\_xy);

f2.add(btn\_abs);

f2.add(btn\_pi);

f2.add(btn\_e);

temp.add(f);

f.add(btn\_cls);

f.add(btn\_bracket);

f.add(btn\_rm);

f.add(btn\_div);

f.add(btn\_1);

f.add(btn\_2);

f.add(btn\_3);

f.add(btn\_pro);

f.add(btn\_4);

f.add(btn\_5);

f.add(btn\_6);

f.add(btn\_sub);

f.add(btn\_7);

f.add(btn\_8);

f.add(btn\_9);

f.add(btn\_add);

f.add(btn\_0);

f.add(btn\_dec);

f.add(btn\_prcnt);

f.add(btn\_ans);

//Listener

tf.addActionListener(this);

btn\_DRG.addActionListener(this);

btn\_sqrt.addActionListener(this);

btn\_fact.addActionListener(this);

btn\_sin.addActionListener(this);

btn\_cos.addActionListener(this);

btn\_tan.addActionListener(this);

btn\_ln.addActionListener(this);

btn\_log.addActionListener(this);

btn\_in.addActionListener(this);

btn\_ex.addActionListener(this);

btn\_x2.addActionListener(this);

btn\_xy.addActionListener(this);

btn\_abs.addActionListener(this);

btn\_pi.addActionListener(this);

btn\_cls.addActionListener(this);

btn\_bracket.addActionListener(this);

btn\_prcnt.addActionListener(this);

btn\_1.addActionListener(this);

btn\_2.addActionListener(this);

btn\_3.addActionListener(this);

btn\_4.addActionListener(this);

btn\_5.addActionListener(this);

btn\_6.addActionListener(this);

btn\_7.addActionListener(this);

btn\_8.addActionListener(this);

btn\_9.addActionListener(this);

btn\_0.addActionListener(this);

btn\_add.addActionListener(this);

btn\_sub.addActionListener(this);

btn\_pro.addActionListener(this);

btn\_dec.addActionListener(this);

btn\_div.addActionListener(this);

btn\_ans.addActionListener(this);

btn\_rm.addActionListener(this);

addWindowListener(new WindowAdapter(){public void windowClosing(WindowEvent we){System.out.println(txt);System.exit(0);}});}

public void actionPerformed(ActionEvent e){

String s = e.getActionCommand();

int op\_indx=0,len=txt.length();

int dflag=0;

if(s.equals("1")){

txt = txt + "1";

}

else if(s.equals("2")){

txt = txt + "2";

}

else if(s.equals("3")){

txt = txt + "3";

}

else if(s.equals("4")){

txt = txt + "4";

}

else if(s.equals("5")){

txt = txt + "5";

}

else if(s.equals("6")){

txt = txt + "6";

}

else if(s.equals("7")){

txt = txt + "7";

}

else if(s.equals("8")){

txt = txt + "8";

}

else if(s.equals("9")){

txt = txt + "9";

}

else if(s.equals("0")){

txt = txt + "0";

}

else if(s.equals(".")){

int f=0;

for(int i=0;i<len;i++)

if(txt.charAt(i) == '+' ||(txt.charAt(i) == '-' &&  i != 0)||txt.charAt(i) == 'x'||txt.charAt(i) == '/')

f=0;

if(f == 0){

for(int i=0;i<len;i++)

if(txt.charAt(i) == '.'){

f=1;break;}

if(f==0)

txt =txt + ".";

}

if(f == 1)

txt = txt + ".";

}

else if(s.equals("+")){

if(flag == 1){

txt = eval(txt);

flag = 0;

}

txt = txt + "+";

flag++;

}

else if(s.equals("x")){

if(flag == 1){

txt = eval(txt);

flag = 0;

}

txt = txt + "x";

flag++;

}

else if(s.equals("-")){

if(flag == 1){

txt = eval(txt);

flag = 0;

}

txt = txt + "-";

flag++;

}

else if(s.equals("/")){

if(flag == 1){

txt = eval(txt);

flag = 0;

}

txt = txt + "/";

flag++;

}

else if(s.equals("C")){

txt ="";flag=0;

}

else if (s.equals("<")){

if(len != 0)

txt = txt.substring(0,len-1);

flag=0;

}

else if(s.equals("( )")){

int c=0;

for(int i=0;i<txt.length();i++){

char ch = txt.charAt(i);

if(ch == '(')

c++;

else if(ch == ')')

c--;

}

if(c%2 != 0)

txt = txt + ")";

else

txt = txt + "(";

}

else if(s.equals("%")){

float a = Float.parseFloat(txt);

a = a/100;

txt = Float.toString(a);

}

else if(s.equals("R")){

Double x = Double.parseDouble(txt);

if(dflag == 0){

x = Degree\_Radian\_Conversion(x,dflag);

dflag=1;

}

else{

x = Degree\_Radian\_Conversion(x,dflag);

dflag=0;

}

btn\_DRG.setLabel("D");

}

else if(s.equals("\u221a")){

Double a= Double.parseDouble(txt);

a = Math.sqrt(a);

txt=Double.toString(a);

}

else if(s.equals("n!")){

//Factorial n -

float a = Float.parseFloat(txt);

int f=1;

for(int i=1;i<=a;i++)

f = f\*i;

txt=Integer.toString(f);

}

else if(s.equals("Sin")){

Double a= Double.parseDouble(txt);

if(dflag == 1)

a = Degree\_Radian\_Conversion(a,dflag);

a = Math.sin(a);

txt=Double.toString(a);

}

else if(s.equals("Cos")){

Double a= Double.parseDouble(txt);

a = Math.cos(a);

txt=Double.toString(a);

}

else if(s.equals("Tan")){

Double a= Double.parseDouble(txt);

a = Math.tan(a);

txt=Double.toString(a);

}

else if(s.equals("ln")){

Double a= Double.parseDouble(txt);

a = Math.log(a);

txt=Double.toString(a);

}

else if(s.equals("log")){

Double a= Double.parseDouble(txt);

a = Math.log10(a);

txt=Double.toString(a);

}

else if(s.equals("1/x")){

Float a = Float.parseFloat(txt);

a = 1/a;

txt = Float.toString(a);

}

else if(s.equals("e^x")){

Double a= Double.parseDouble(txt);

a = Math.exp(a);

txt=Double.toString(a);

}

else if(s.equals("x^2")){

Double a= Double.parseDouble(txt);

a = a\*a;

txt=Double.toString(a);

}

else if(s.equals("x^y")){

txt = txt + "^";

}

else if(s.equals("|x|")){

float a = Float.parseFloat(txt);

a = Math.abs(a);

txt = Float.toString(a);

}

else if(s.equals("\u03C0")){

if(len ==0)

txt = txt + "3.14152692x";

else

txt = txt +"x3.14152692";

flag=1;

}

else if(s.equals("e")){

txt = txt + "2.718";

}

else if(s.equals("=")){

int pow= 0;

for(int i=0;i<len;i++){

char c= txt.charAt(i);

if(c == '^'){

pow=i;break;

}

}

if(pow == 0){

txt = eval(txt);

System.out.println(txt);

flag=0;

}

else{

Double a = Double.parseDouble(txt.substring(0,pow));

Double b = Double.parseDouble(txt.substring(pow+1,len));

Double c =a;

for(Double i=b;i>1;i--)

c = a\*c;

txt = Double.toString(c);

}

}

tf.setText(txt);

L1.setText(txt);

}

public static void main(String args[]){

ScientifiCalculator appwin=new ScientifiCalculator();

appwin.setSize(new Dimension(350,350));

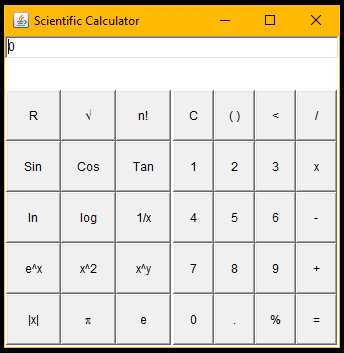
appwin.setTitle("Scientific Calculator");

appwin.setVisible(true);

}

}

**Output:**

****