**Q.1: Write a java program to print the welcome message.**

class HelloWorld{

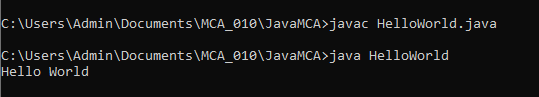
public static void main(String[] args){

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

}

}

**Output** :



**Q.2: Write a java program to perform arithmetical operations using static & Scanner class and Command Line Argument as input data**

**// Using Command Line Arguments**

class Arithmetic{

public static void main(String[] args){

int a, b,c;

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

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

c = a+b;

System.out.println("Addition is : " + c);

c = a-b;

System.out.println("Subtraction is : " + c);

c = a\*b;

System.out.println("Multiplication is : " + c);

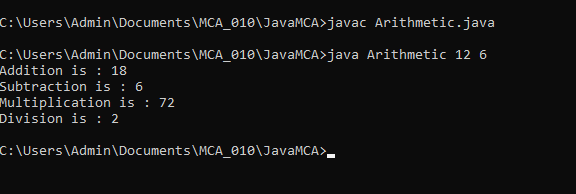
c = a/b;

System.out.println("Division is : " + c);

}

}

**Output :**



**// Using Scanner (User Input)**

import java.util.Scanner;

public class Arithmetic {

    public static void main(String[] *args*) {

Scanner sc = new Scanner(System.in);

        int num1, num2;

        System.out.print("Enter the first Num : ");

        num1 = sc.nextInt();

        System.out.print("Enter the second Num : ");

        num2 = sc.nextInt();

        int num3 = num1 + num2;

        System.out.println("The Addition is : " + num3);

        num3 = num1 - num2;

        System.out.println("The Subtraction is : " + num3);

        num3 = num1 \* num2;

        System.out.println("The Multiplication is : " + num3);

        num3 = num1 / num2;

        System.out.println("The Division is : " + num3);

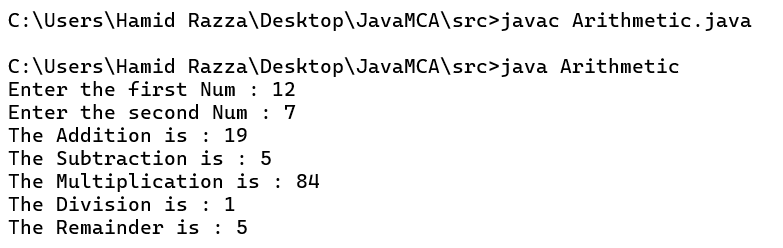
        num3 = num1 % num2;

        System.out.println("The Remainder is : " + num3);

    }

}

**Output :**



**// Using Static way**

public class Arithmetic {

    public static void main(String[] *args*) {

int num1 = 11, num2 = 6;

        int num3 = num1 + num2;

        System.out.println("The Addition is : " + num3);

        num3 = num1 - num2;

        System.out.println("The Subtraction is : " + num3);

        num3 = num1 \* num2;

        System.out.println("The Multiplication is : " + num3);

        num3 = num1 / num2;

        System.out.println("The Division is : " + num3);

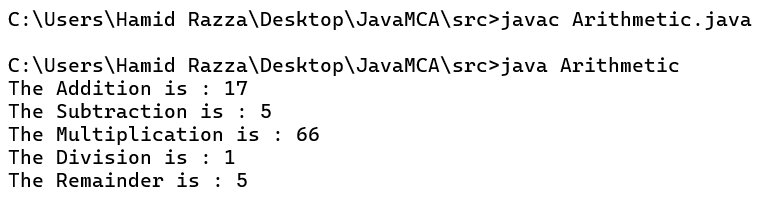
        num3 = num1 % num2;

        System.out.println("The Remainder is : " + num3);

    }

}

**Output :**

****

**Q.3: Write a java program to find the greatest among the 3 numbers using command line arguments**

class GreatestThree{

public static void main(String[] args){

System.out.println("Program to Find Greatest using Command Line");

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

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

int c = Integer.parseInt(args[2]);

if((a > b) && (a>c)){

System.out.println(a + " is greatest.");

}

else if((b > a) && (b > c)){

System.out.println(b + " is greatest.");

}

else{

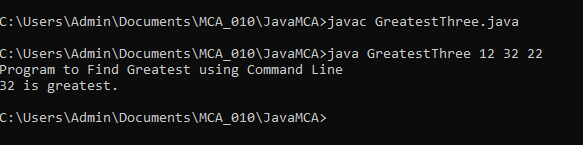
System.out.println(c + " is greatest.");

}

}

}

**Output :**



**Q.4: Write a java program to find the ODD or EVEN number**

import java.util.Scanner;

class OddEven{

public static void main(String[] args){

System.out.println("Program to find ODD or EVEN");

Scanner sc = new Scanner(System.in);

System.out.print("Enter the no: ");

int num = sc.nextInt();

if(num%2 == 0){

System.out.println(num + " is an EVEN no");

}else{

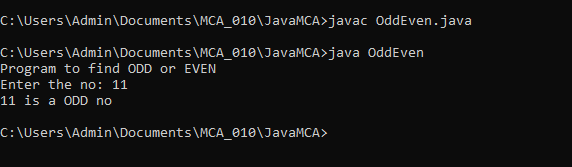
System.out.println(num + " is a ODD no");

}

}

}

**Output :**

****

**Q.5 : Develop a java application for Banking transaction : Balance\_Check, Deposit\_Amount, Withdraw\_Amount and Exit.**

import java.util.Scanner;

public class Bank {

// Making this program like the Banking system.

static double balance = 2000;

public static void main(String[] args) {

boolean repeat = true;

Scanner in = new Scanner(System.in);

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

System.out.println("| WELCOME TO THE BANK |");

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

System.out.println("| 1. CHECK BALANCE |");

System.out.println("| 2. WITHDRAW |");

System.out.println("| 3. DEPOSIT |");

System.out.println("| 0. EXIT |");

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

while (repeat) {

System.out.print("Enter an option : ");

int choice = in.nextInt();

switch (choice) {

case 0 :

repeat = false;

break;

case 1 :

checkVal();

break;

case 2 :

withdraw();

break;

case 3 :

deposit();

break;

default :

System.out.println("INVALID OPTION !!");

}

}

System.out.println("THANKS FOR VISITING, VISIT AGAIN !!!");

in.close();

}

public static void checkVal() {

System.out.println("Your balance is : Rs."+balance);

}

public static void withdraw() {

Scanner sc = new Scanner(System.in);

System.out.print("Enter the amount : ");

double amount = sc.nextDouble();

if(amount > balance){

System.out.println("Insufficient Balance !!");

}

else if(amount >= 50000){

System.out.println("To withdraw 50000 or more than 50000");

System.out.print("You need to enter PAN NO : ");

String panNo = sc.next();

if(panNo.length() == 10) {

System.out.println("PAN NO VERIFIED !!");

System.out.println("WITHDRAW SUCCESSFUL !!!");

balance -= amount;

System.out.println("Your current balance is : Rs." + (balance));

}

else{

System.out.println("Wrong PAN No. !!!");

}

}

else{

System.out.println("WITHDRAW SUCCESSFUL !!!");

balance -= amount;

System.out.println("Your current balance is : Rs." + (balance));

}

}

public static void deposit() {

Scanner sc = new Scanner(System.in);

System.out.print("Enter the amount : ");

double amount = sc.nextDouble();

if(amount >= 50000){

System.out.println("To Deposit 50000 or more than 50000");

System.out.print("You need to enter PAN NO : ");

String panNo = sc.next();

if(panNo.length() == 10) {

System.out.println("PAN NO VERIFIED !!");

System.out.println("DEPOSIT SUCCESSFUL !!!");

balance += amount;

System.out.println("Your current balance is : Rs." + (balance));

}

else{

System.out.println("Wrong PAN No. !!!");

}

}

else{

System.out.println("DEPOSIT SUCCESSFUL !!!");

balance += amount;

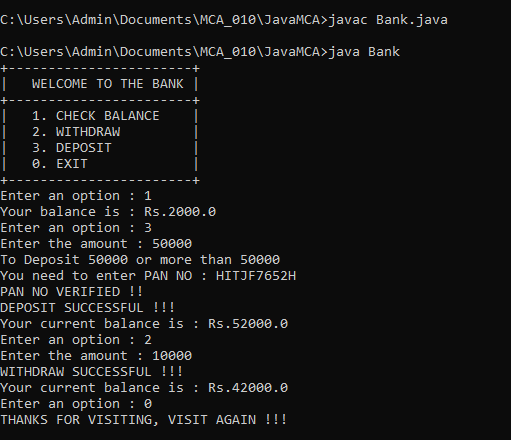
System.out.println("Your current balance is : Rs." + (balance));

}

}

}

**Output :**



**Q.6 : Generic Program for one-type parameter and two-type parameter.**

class GenericOne<T>{

T obj;

GenericOne(T obj){

this.obj = obj;

}

public T getObj(){

return this.obj;

}

}

class GenericTwo<T,E>{

T obj1;

E obj2;

GenericTwo(T obj1, E obj2){

this.obj1 = obj1;

this.obj2 = obj2;

}

public T getObj1(){

return this.obj1;

}

public E getObj2(){

return this.obj2;

}

}

class GenericDemo{

// Main method

public static void main(String[] args){

System.out.println("Single Type Generic Class");

GenericOne<Integer> first = new GenericOne<Integer>(22);

System.out.println("Object is : " + first.getObj());

System.out.println("Double Type Generic Class");

GenericTwo<Integer,String> second = new GenericTwo<Integer,String>(12, "Hamid");

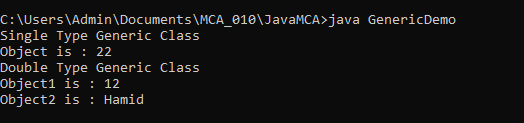
System.out.println("Object1 is : " + second.getObj1());

System.out.println("Object2 is : " + second.getObj2());

}

}

**Output :**



**Q.7 : Sorting of Numbers and Names.**

import java.util.Scanner;

class SortName{

public static void main(String[] args){

int n;

Scanner scan = new Scanner(System.in);

System.out.print("How many names you want to enter : ");

n = scan.nextInt();

String[] arr = new String[n];

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

System.out.print("Enter the " + (i+1) + "th element : ");

arr[i] = scan.next();

}

for(int i = 0; i < n-1; i++){

for(int j = i+1; j < n; j++){

if(arr[i].compareTo(arr[j]) > 0){

String temp = arr[i];

arr[i] = arr[j];

arr[j] = temp;

}

}

}

System.out.print("\nSorted Names are: ");

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

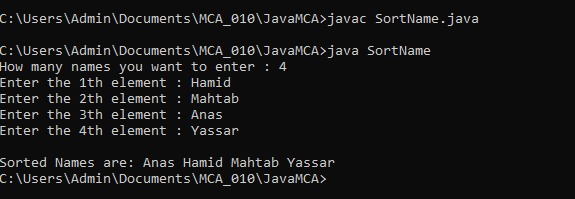
System.out.print(arr[i] + " ");

}

}

}

**Output :**



class SortNum{

public static void main(String[] args){

int[] arr = new int[] {5,1,7,2,3};

System.out.println("Original Array is given below");

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

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

}

for(int i = 0; i < arr.length-1; i++){

for(int j = i+1; j < arr.length; j++){

if(arr[i] > arr[j]){

int temp = arr[i];

arr[i] = arr[j];

arr[j] = temp;

}

}

}

System.out.println("\nSorted Array is: ");

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

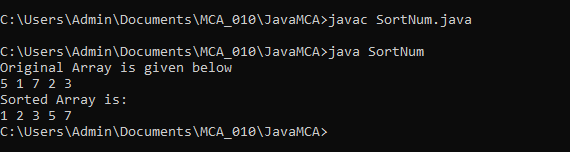
System.out.print(arr[i] + " ");

}

}

}

**Output :**



**Q.8 : Command Line program**

class CommandLine {

    public static void main(String[] *args*) {

        try{

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

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

            System.out.println(a + " is assigned using CommandLine Argument");

            System.out.println(b + " is also assigned using CommandLine Argument");

        }

        catch(Exception *e*){

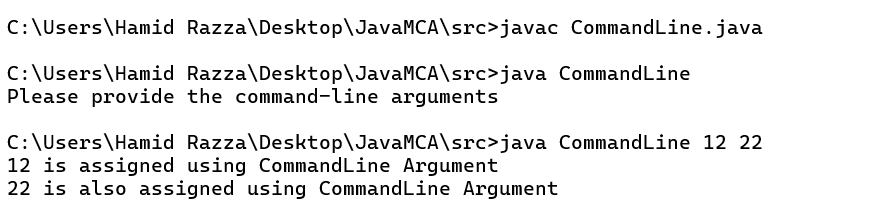
            System.out.println("Please provide the command-line arguments");

        }

    }

}

**Output :**

****

**Q.9 : BigInteger and BigDecimal Program**

import java.util.Scanner;

import java.math.BigInteger;

import java.math.BigDecimal;

class BigIntDecDemo{

public static void main(String[] args){

Scanner scan = new Scanner(System.in);

// BigInteger Demo

System.out.println("-------- Big Integer Example --------");

System.out.print("Enter the first element : ");

BigInteger b1 = scan.nextBigInteger();

System.out.print("Enter the second element : ");

BigInteger b2 = scan.nextBigInteger();

BigInteger b3 = b1.add(b2);

System.out.println("The addition of BigInteger : " + b3 + "\n");

// BigDecimal Demo

System.out.println("-------- Big Decimal Example --------");

System.out.print("Enter the first element : ");

BigDecimal d1 = scan.nextBigDecimal();

System.out.print("Enter the second element : ");

BigDecimal d2 = scan.nextBigDecimal();

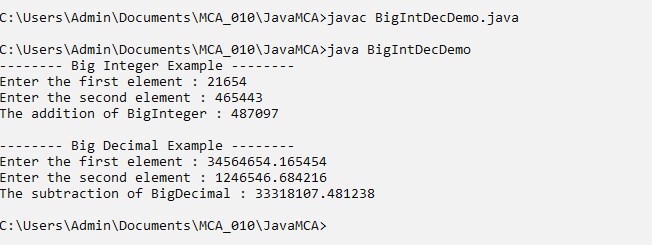
BigDecimal d3 = d1.subtract(d2);

System.out.println("The subtraction of BigDecimal : " + d3);

}

}

**Output :**

****

**Q.10 : Constructor Overloading Program**

public class ConstructorDemo{

private String name = "Unknown";

private int id = 0;

//Constructor : It invokes when the object is created.

ConstructorDemo(){

System.out.println("Name : " + name + " and id is : " + id);

}

//Constructor Overloading

ConstructorDemo(int id){

this.id = id;

System.out.println("Name : " + name + " and id is : " + id);

}

//Constructor Overloading

ConstructorDemo(int id, String name){

this.id = id;

this.name = name;

System.out.println("Name : " + name + " and id is : " + id);

}

public static void main(String[] args){

// Making the object of the current class

ConstructorDemo obj1 = new ConstructorDemo();

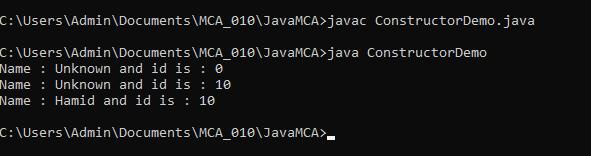
ConstructorDemo obj2 = new ConstructorDemo(10);

ConstructorDemo obj3 = new ConstructorDemo(10, "Hamid");

}

}

**Output :**

****

**Q.11 : Method Overloading program**

class MethodOver {

    public static void main(String[] *args*) {

        multiply(3,5);

        multiply(2.5,3.1);

    }

    public static void multiply(int *a*, int *b*) {

        System.out.println("Multiply is (int): " + (a\*b));

    }

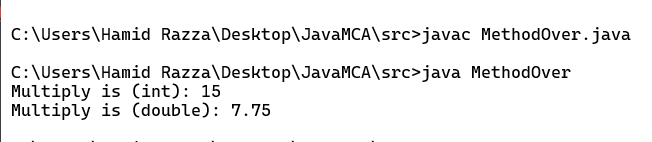
    public static void multiply(double *a*, double *b*) {

        System.out.println("Multiply is (double): " + (a\*b));

    }

}

**Output :**

****

**Q.12 : Method Overriding program**

class OverRiding extends Demo{

    public static void main(String[] *args*) {

        echo();

        Demo.echo();

    }

    static void echo(){

        System.out.println("Hello, from Base Class");

    }

}

class Demo{

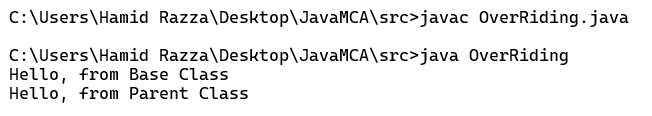
    static void echo(){

        System.out.println("Hello, from Parent Class");

    }

}

**Output :**

****

**Q.13 : Inheritance implementation (Single, Multiple, Multilevel, Hierarchical)**

**// Single Inheritance**

class Species {

    private int legs;

    private int hands;

    Species(int *legs*, int *hands*) {

*this*.legs = legs;

*this*.hands = hands;

    }

    public void pwdHands() {

        hands -= 1;

    }

    public void pwdLegs() {

        legs -= 1;

    }

    public String toString() {

        return ("No. of hands are " + hands + "\n" + "No. of legs are " + legs);

    }

}

class Humans extends Species {

    private String name;

    public Humans(String *name*, int *legs*, int *hands*) {

*super*(legs, hands);

*this*.name = name;

    }

    public void setName(String *newName*) {

*this*.name = newName;

    }

    @Override

    public String toString() {

        return ("Name is " + name + "\n" + *super*.toString());

    }

}

public class DemoInheritance {

    public static void main(String[] *args*) {

        Humans aman = new Humans("Aman", 2, 2);

        System.out.println(aman.toString());

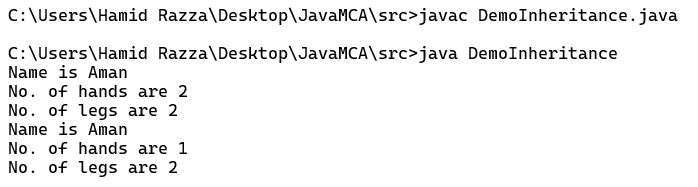
        aman.pwdHands();

        System.out.println(aman.toString());

    }

}

**Output :**



**//Hierarchical Inheritance**

class Species {

    private int legs;

    private int hands;

    Species(int *legs*, int *hands*) {

*this*.legs = legs;

*this*.hands = hands;

    }

    public void pwdHands() {

        hands -= 1;

    }

    public void pwdLegs() {

        legs -= 1;

    }

    public String toString() {

        return ("No. of hands are " + hands + "\n" + "No. of legs are " + legs);

    }

}

class Humans extends Species {

    private String name;

    public Humans(String *name*, int *legs*, int *hands*) {

*super*(legs, hands);

*this*.name = name;

    }

    public void setName(String *newName*) {

*this*.name = newName;

    }

    @Override

    public String toString() {

        return ("Name is " + name + "\n" + *super*.toString());

    }

}

class Insects extends Species{

    private String name;

    Insects(int *legs*, int *hands*){

*super*(legs, hands);

    }

    public void setName(String *newName*){

*this*.name = newName;

    }

    @Override

    public String toString(){

        return ("Insect name is " + name + "\n" + *super*.toString());

    }

}

public class DemoInheritance {

    public static void main(String[] *args*) {

        Humans aman = new Humans("Aman", 2, 2);

        System.out.println(aman.toString());

        Insects spider = new Insects(8,0);

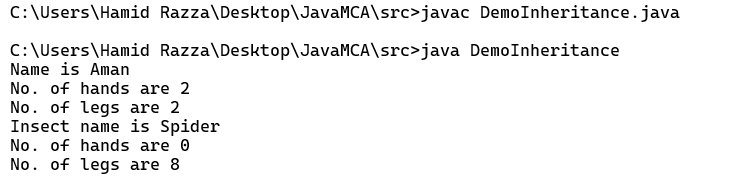
        spider.setName("Spider");

        System.out.println(spider.toString());

    }

}

**Output :**

****

**// Multi-level Inheritance**

class Species {

    private int legs;

    private int hands;

    Species(int *legs*, int *hands*) {

*this*.legs = legs;

*this*.hands = hands;

    }

    public void pwdHands() {

        hands -= 1;

    }

    public void pwdLegs() {

        legs -= 1;

    }

    public String toString() {

        return ("No. of hands are " + hands + "\n" + "No. of legs are " + legs);

    }

}

class Humans extends Species {

    protected String name;

    public Humans(String *name*, int *legs*, int *hands*) {

*super*(legs, hands);

*this*.name = name;

    }

    public void setName(String *newName*) {

*this*.name = newName;

    }

    @Override

    public String toString() {

        return ("Name is " + name + "\n" + *super*.toString());

    }

}

class RichHumans extends Humans{

    private String car;

    private String business;

    RichHumans(String *name*, int *legs*, int *hands*){

*super*(name, legs, hands);

    }

    public void setCar(String *car*){

*this*.car = car;

    }

    public void setBusiness(String *businessName*){

*this*.business = businessName;

    }

    @Override

    public String toString(){

        return (*super*.toString() + "\nCar name is " + car + "\nBusiness is " + business);

    }

}

public class DemoInheritance {

    public static void main(String[] *args*) {

RichHumans demo = new RichHumans("sahil", 2, 2);

        demo.setCar("Lambo");

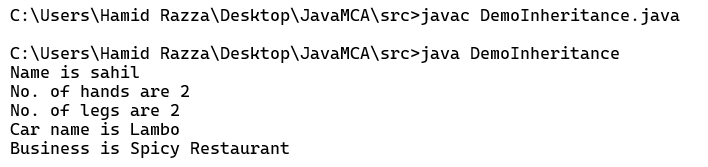
        demo.setBusiness("Spicy Restaurant");

        System.out.println(demo.toString());

    }

}

**Output :**

****

**// Multiple Inheritance**

public class MultipleInheritance extends Demo1 implements Demo2{

    private String name;

    @Override

    public void setName(String *name*){

*this*.name = name;

    }

    @Override

    public String getName(){

        return name;

    }

    public static void main(String[] *args*) {

        MultipleInheritance obj = new MultipleInheritance();

        obj.setName("Hamid");

        System.out.println("Name is : " + obj.getName());

        add(3,7 );

        sub(5,2);

    }

}

class Demo1{

    public static void add(int *a*, int *b*){

        System.out.println("Addition is : " + (a+b));

    }

    public static void sub(int *a*, int *b*){

        System.out.println("Subtraction is : " + (a-b));

    }

}

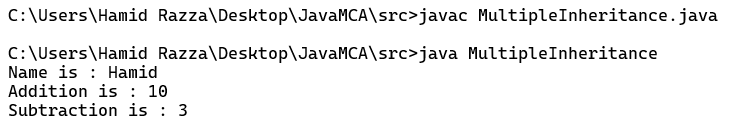
interface Demo2{

    public void setName(String *name*);

    public String getName();

}

**Output :**

****

**Q.14 : Abstract class implementation**

public class AbstractDemo extends Footwear{

    @Override

    public void setType(String *type*){

*this*.type = type;

    }

    public static void main(String[] *args*) {

        AbstractDemo demo = new AbstractDemo();

        demo.setType("Slippers");

        System.out.println("The type of footware is : " + demo.getType());

    }

}

abstract class Footwear{

    protected String type;

    public abstract void setType(String *type*);

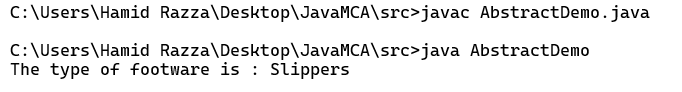
    public String getType(){

       return type;

    }

}

**Output :**



**Q.15 : Interface implementation**

public class InterfaceDemo implements Demo {

    @Override

    public int incrementNum(int *num*) {

        return (num += 1);

    }

    @Override

    public int decrementNum(int *num*) {

        return (num -= 1);

    }

    @Override

    public int doubleNum(int *num*) {

        return (num \*= num);

    }

    public static void main(String[] *args*) {

        InterfaceDemo obj = new InterfaceDemo();

        int num = 5;

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

        System.out.println("After increment the num = " + obj.incrementNum(num));

        System.out.println("After doubling the num = " + obj.doubleNum(num));

        System.out.println("After decrement the num = " + obj.decrementNum(num));

    }

}

interface Demo {

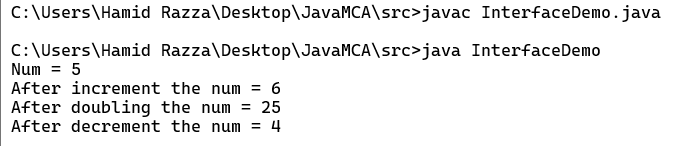
    int incrementNum(int *n*);

    int decrementNum(int *n*);

    int doubleNum(int *n*);

}

**Output :**



**Q.16 : Nested Classes (local, static, non-static, anonymous)**

**// Static Nested Class**

public class NestedClass {

    public static class InnerClass {

        private int num;

        public void setData(int *num*){

*this*.num = num;

        }

        public int getData(){

            return num;

        }

        public void square(int *num*){

*this*.num = (num\*num);

        }

    }

    public static void main(String[] *args*) {

        InnerClass obj = new InnerClass();

        obj.setData(12);

        System.out.println("The num is : " + obj.getData());

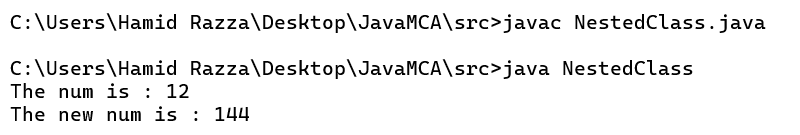
        obj.square(12);

        System.out.println("The new num is : " + obj.getData());

    }

}

**Output :**

****

**// Non-static Nested Class**

class OuterClass {

    static int num1 = 10;

    int num2 = 20;

    private int num3 = 30;

    class InnerClass{

        void display(){

            System.out.println("Static outer num1 : " + num1);

            System.out.println("Non-static outer num2 : " + num2);

            System.out.println("Private outer num3 : " + num3);

        }

    }

}

public class NestedDemo{

    public static void main(String[] *args*) {

        OuterClass outerObj = new OuterClass();

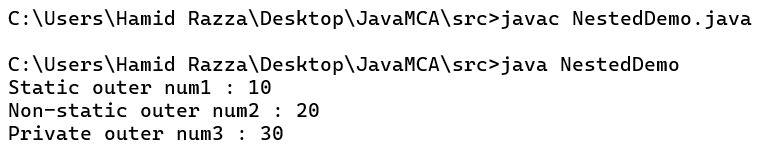
        OuterClass.InnerClass innerObj = outerObj.new InnerClass();

        innerObj.display();

    }

}

**Output :**



**// Local Nested Class**

class OuterClass {

    void display(){

        class LocalClass{

            public int num;

            LocalClass(){

*this*.num = 10;

            }

            private int getValue(){

                return num;

            }

            private int getSquare(){

                return (num\*num);

            }

        }

        LocalClass localObj = new LocalClass();

        System.out.println("The num is : " + localObj.num);

        System.out.println("The square num is : " + localObj.getSquare());

    }

}

public class NestedDemo{

    public static void main(String[] *args*) {

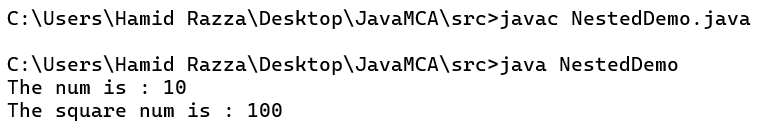
        OuterClass outerObj = new OuterClass();

        outerObj.display();

    }

}

**Output :**



**// Anonymous Nested Class**

abstract class OuterClass {

    int num = 15;

    abstract void getNum();

}

public class NestedDemo{

    public static void main(String[] *args*) {

        OuterClass obj = new OuterClass() {

            @Override

            void getNum(){

                System.out.println("The current num is : " + num);

            }

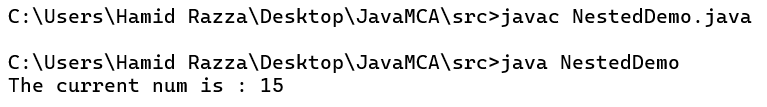
        };

        obj.getNum();

    }

}

**Output :**

****

**Q.17 : Package implementation program-user define package**

package com.hamid;

public class PackageDemo {

    public static void main(String[] *args*) {

        System.out.println("Just a demo of package fearture");

    }

}

**Output :**

