Program1. // Java Program to Demonstrate Char Primitive Data Type

Class GFG {

Public static void main(String args[])

{

Char a = ‘G’;

Int I = 89;

Byte b = 4;

Short s = 56;

Double d = 4.355453532;

Float f = 4.7333434f;

Long l = 121212;

System.out.println(“Atul Sharma\n0827CI221039\n”);

System.out.println(“char: “ + a);

System.out.println(“integer: “+ i);

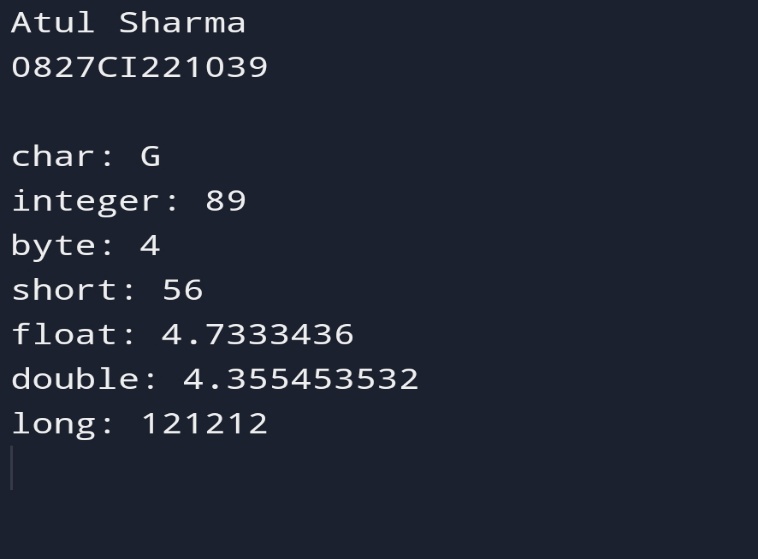
System.out.println(“byte: “ + b);

System.out.println(“short: “ + s);

System.out.println(“float: “ + f);

System.out.println(“double: “+ d);

System.out.println(“long: “ + l);}

}

Program2- //String Data type

Class Main {

Public static void main(String[] args) {

System.out.println(“Atul Sharma\n0827CI221039\n”);

String first = “Java”;

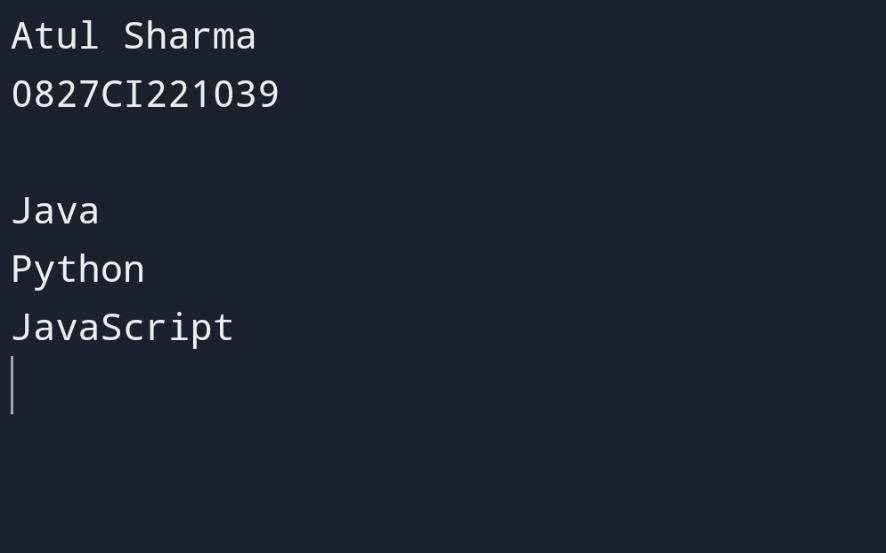
String second = “Python”;

String third = “JavaScript”;

System.out.println(first);

System.out.println(second);

System.out.println(third);



Program3.

// Java Program to Illustrate Automatic Type Conversion

Class Conversion {

Public static void main(String[] args)

{

System.out.println(“Atul Sharma\n0827CI221039\n”);

Int I = 100;

// Automatic type conversion

// Integer to long type

Long l = I;

// Automatic type conversion

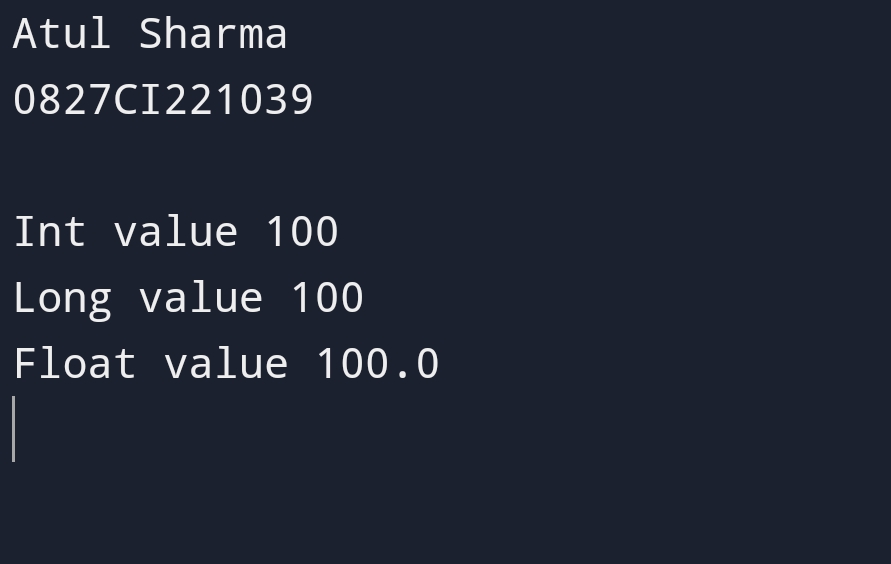
// long to float type

Float f = l;

System.out.println(“Int value “ + i);

System.out.println(“Long value “ + l);

System.out.println(“Float value “ + f);

 }}

Program4.// Java program to Illustrate Explicit Type Conversion

Public class GFG {

Public static void main(String[] args)

{

System.out.println(“Atul Sharma\n0827CI221039\n”);

// Double datatype

Double d = 100.04;

// Explicit type casting by forcefully getting

// data from long datatype to integer type

Long l = (long)d;

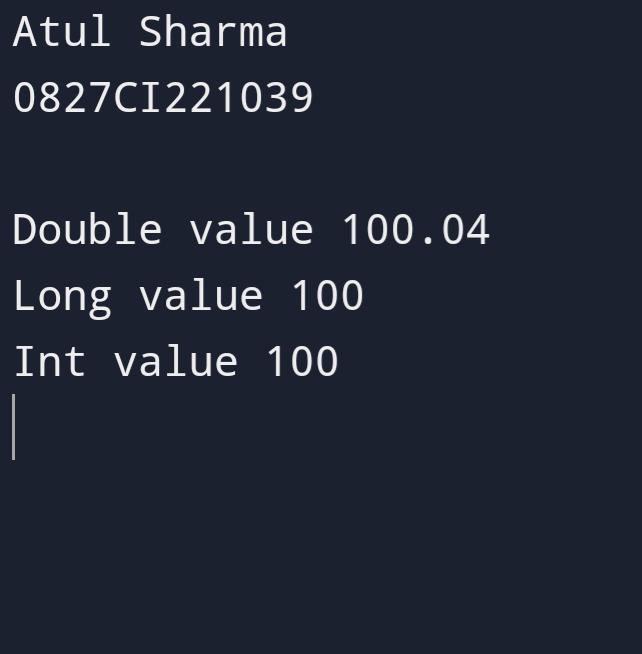
// Explicit type casting

Int I = (int)l;

System.out.println(“Double value “ + d);

System.out.println(“Long value “ + l);

System.out.println(“Int value “ + i);

 }}.

Program 5 .//Program using Class and object

// print the integer enter by user?

Import java.util.Scanner;

Public class A

{

Public static void main(String[]args)

{

System.out.println(“Atul Sharma\n0827CI221039\n”);

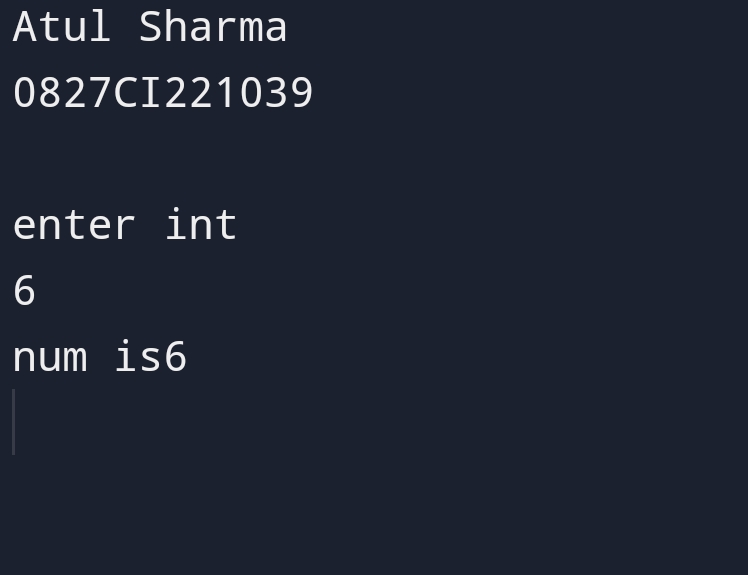
Scanner sc=new Scanner(System.in);

System.out.println(“enter int”);

Int n=sc.nextInt();

System.out.println(“num is”+n);

}



Program 6. //Write a program To use class and object.

//2)To check whether given input number is even or odd?

Import java.util.Scanner;

Public class A

{

Public static void main(String[]args)

{

System.out.println(“Atul Sharma\n0827CI221039\n”);

Scanner sc=new Scanner(System.in);

System.out.println(“enter int”);

Int n=sc.nextInt();

If(n%2==0)

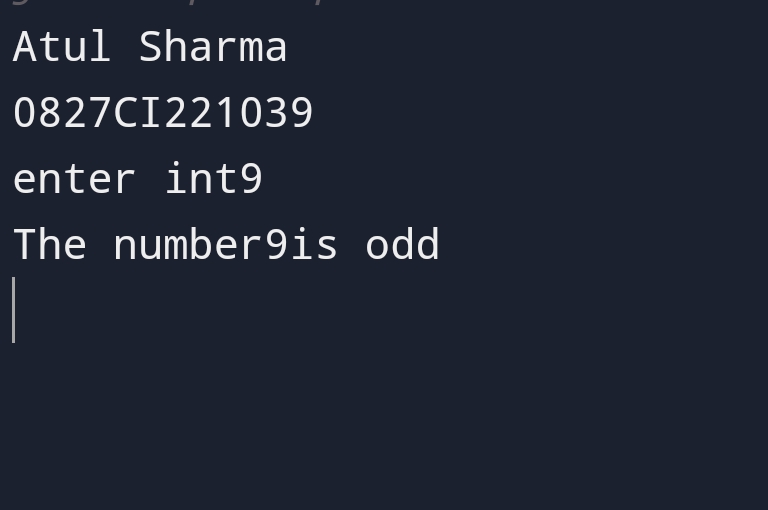
System.out.println(“The number”+n+”is even”);

Else

System.out.println(“The number”+n+”is odd”);

}

}

Program 7. //Static Variable Program

Public class swap

{ static int a;

Static int b;

Public static void set(int I,int j)

{A=I;

B=j;}

Public static void swap()

{Int c;

C=a;A=b;B=c;}

Public static void main(String args[])

{System.out.println(“Atul Sharma\n0827CI221039\n”);

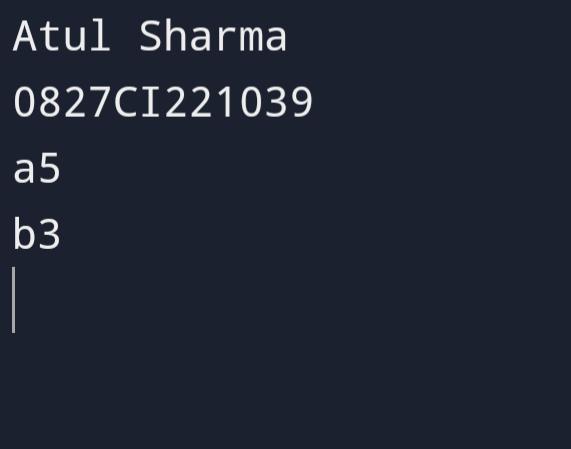
Set(3,5);

Swap();

System.out.println(“a”+a);

System.out.println(“b”+b);

}

}

Program 8 . //Java program to demonstrate that

// The static method does not have

// access to the instance variable

Import java.util.Scanner;

Public class SV {

// static variable

Static int a = 40;

// instance variable

Int b = 50;

Void simpleDisplay()

{ System.out.println(a);

System.out.println(b); }

// Declaration of a static method.

Static void staticDisplay()

{ System.out.println(a); }

// main method

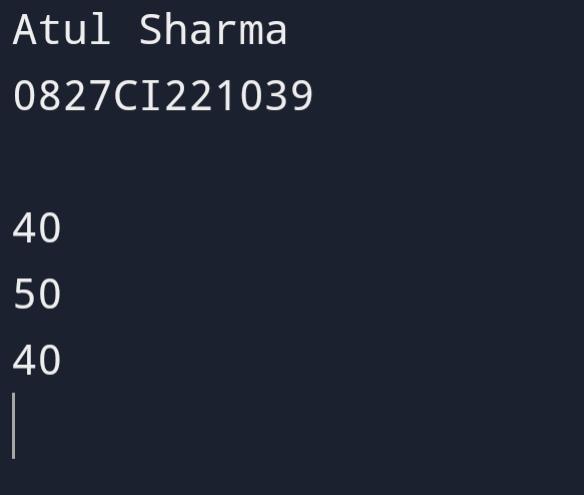
Public static void main(String[] args)

{ System.out.println(“Atul Sharma\n0827CI221039\n”);

SV obj = new SV();

Obj.simpleDisplay();

// Calling static method.

 staticDisplay(); } }

Program 9.//Declare primitive data type with some value convert it into object using wrapper class?

Public class Wrapper

{

Public static void main(String args[])

{

System.out.println(“Atul Sharma\n0827CI221039\n”);

Int i=16;

Integer I=I;

Char c=’a’;

Character C=c;

Short s=8;

Short S=s;

Long l=135;

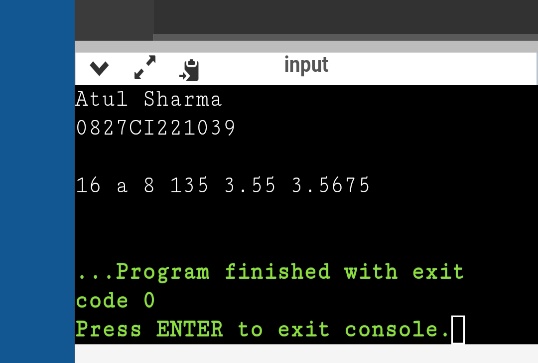
Long L=l;

Float f=3.55F;

Float F=f;

Double d=3.5675;

Double D=d;

 System.out.println(I+” “+C+” “+S+” “+L+” “+F+” “+D+” “);}}

Program 10. //Java Command Line Argument

Public class MyClass {

Static int sum=0;

Public static void main(String args[]) {

System.out.println(args.length);

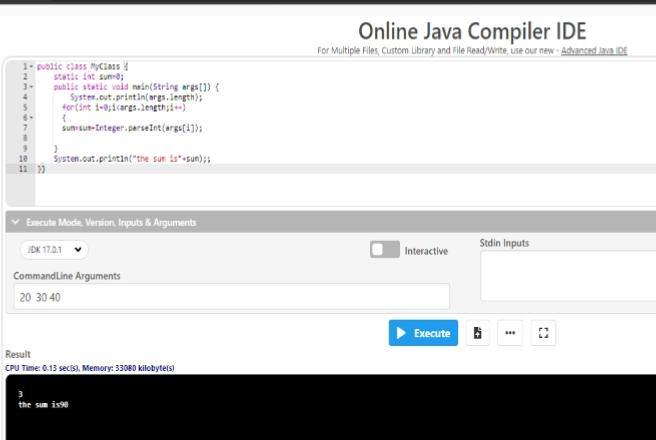
For(int i=0;i<args.length;i++)

{

Sum=sum+Integer.parseInt(args[i]);

}

System.out.println(“the sum is”+sum);;

}}

Program 11. //Java Command line argument

Public class MyClass {

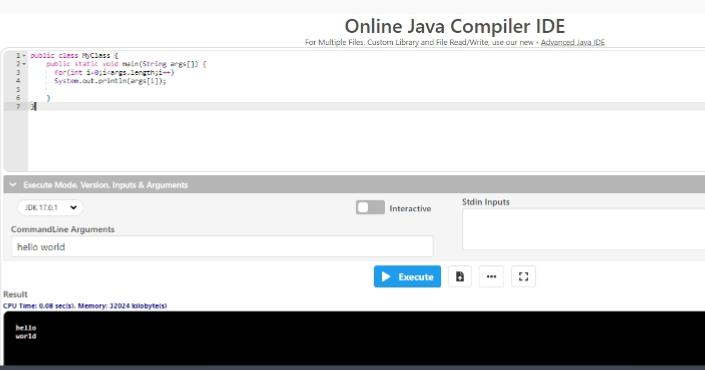
Public static void main(String args[]) {

For(int i=0;i<args.length;i++)

System.out.println(args[i]);

}

}



Program 12. //Wrapper Class Example Autoboxing

Public class wrapper

{

Public static void main(String args[])

{

System.out.println(“Atul Sharma\n0827CI221039\n”);

Int a=20;

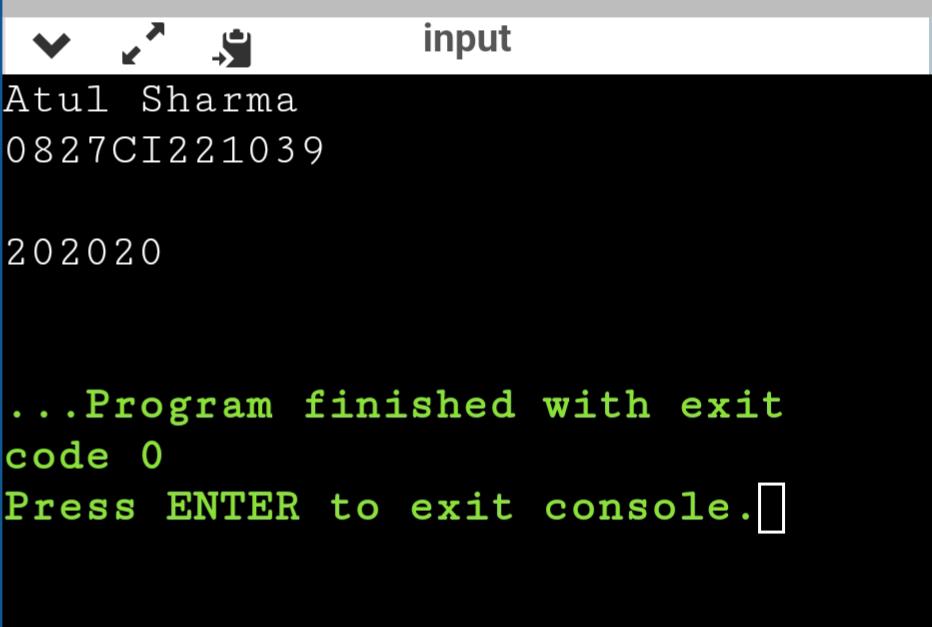
Integer i=Integer.valueOf(a);

Integer j=a;

System.out.println(a+””+i+””+j);

}

}

Program 13. //Wrapper Class Example Unboxing

Public class wrapper

{

Public static void main(String args[])

{

System.out.println(“Atul Sharma\n0827CI221039\n”);

Integer a = new Integer(3);

Int i=a.intValue();

Int j=a;

System.out.println(a+””+i+””+j);

}

}



Program 14. //Protected Access Modifier

Class A

{

Protected void display()

{

System.out.println(“Java Is Best”);

}

}

Class B extends A {}

Class C extends B {}

Class Main{

Public static void main(String args[])

{

System.out.println(“Atul Sharma\n0827CI221039\n”);

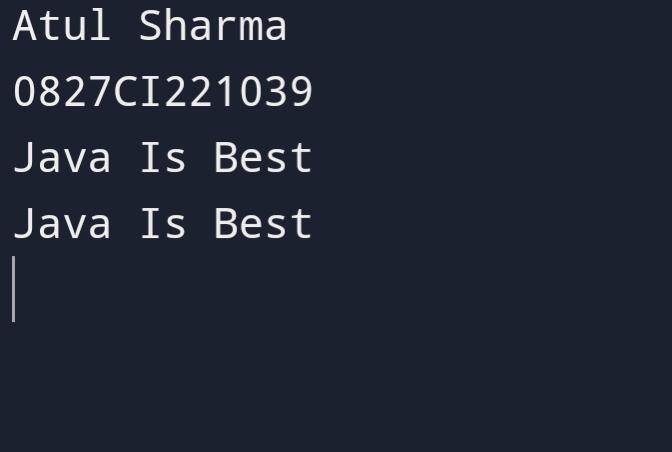
B obj = new B(); B

Obj.display();

C cObj = new C();

cObj.display ();

}

}

Program 15.//Private Access Modifier

Public class Main {

Private String fname = “Atul”;

Private String lname = “Sharma”;

Private String roll = “0827CI221039”;

Private int age = 19;

Public static void main(String[] args) {

Main myObj = new Main();

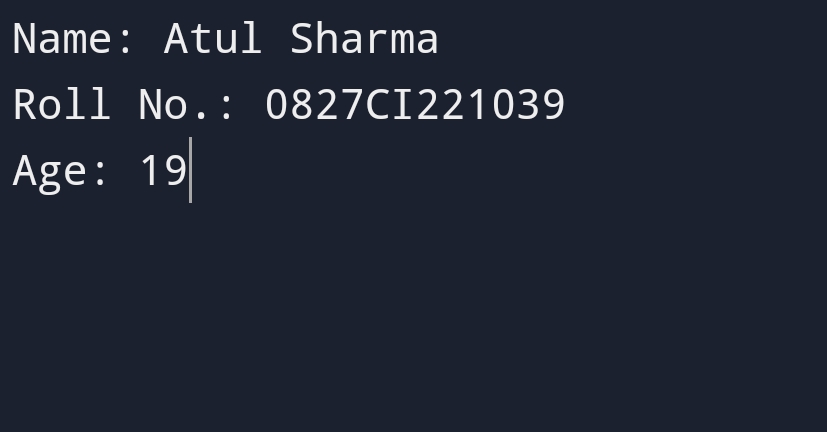
System.out.println(“Name: “ + myObj.fname + “ “ + myObj.lname);

System.out.println(“Roll No.: “ + myObj.roll);

System.out.println(“Age: “ + myObj.age);

}

}



Program 16. //Constructor Program

Class Main {

Private String name;

Main() {

System.out.println(“Constructor Called:\n”);

Name = “Atul Sharma\n0827CI221039\n”;

}

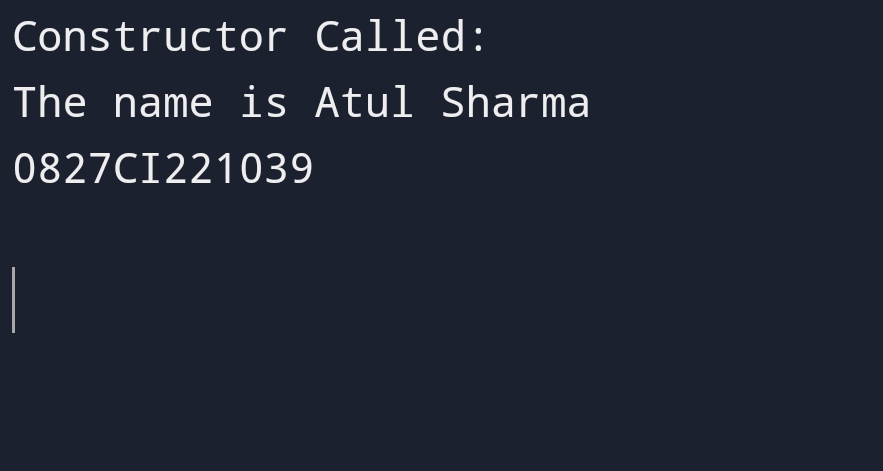
Public static void main(String[] args) {

Main obj = new Main();

System.out.println(“The name is “ + obj.name);

}

}



Program 17. //Constructor Program

Class Main {

String languages;

Main(String lang) {

Languages = lang;

System.out.println(languages + “ Programming Language”);

}

Public static void main(String[] args) {

System.out.println(“Atul Sharma\n0827CI221039\n”);

// call constructor by passing a single value

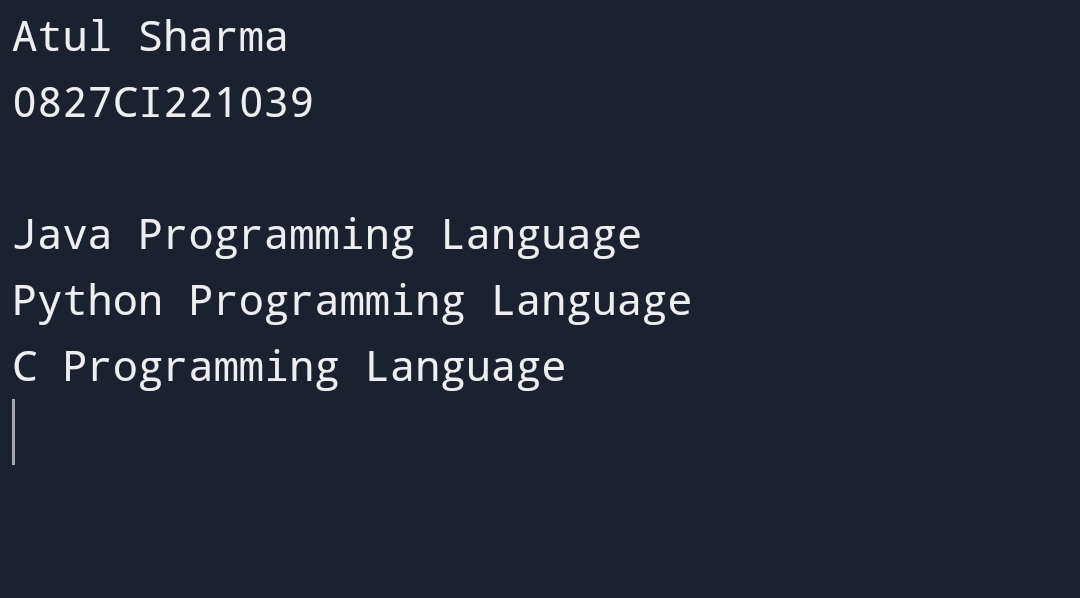
Main obj1 = new Main(“Java”);

Main obj2 = new Main(“Python”);

Main obj3 = new Main(“C”);

}

}



Program 18. //Inheritance Program

Class Employee{

Float salary=40000;

}

Class Programmer extends Employee{

Int bonus=10000;

Public static void main(String args[]){

System.out.println(“Atul Sharma\n0827CI221039\n”);

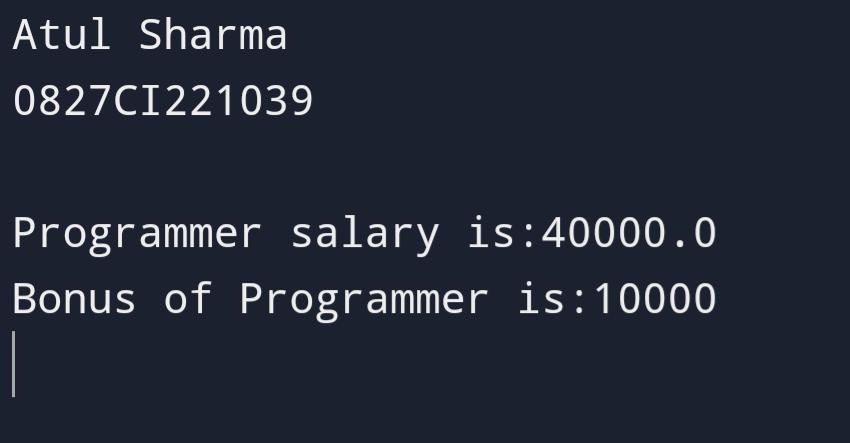
Programmer p=new Programmer();

System.out.println(“Programmer salary is:”+p.salary);

System.out.println(“Bonus of Programmer is:”+p.bonus);

}

}



Program 19. //Inheritance(Hierarchical)Program

Class Animal{

Void eat(){System.out.println(“eating…”);}

}

Class Dog extends Animal{

Void bark(){System.out.println(“barking…”);}

}

Class Cat extends Animal{

Void meow(){System.out.println(“meowing…”);}

}

Class TestInheritance3{

Public static void main(String args[]){

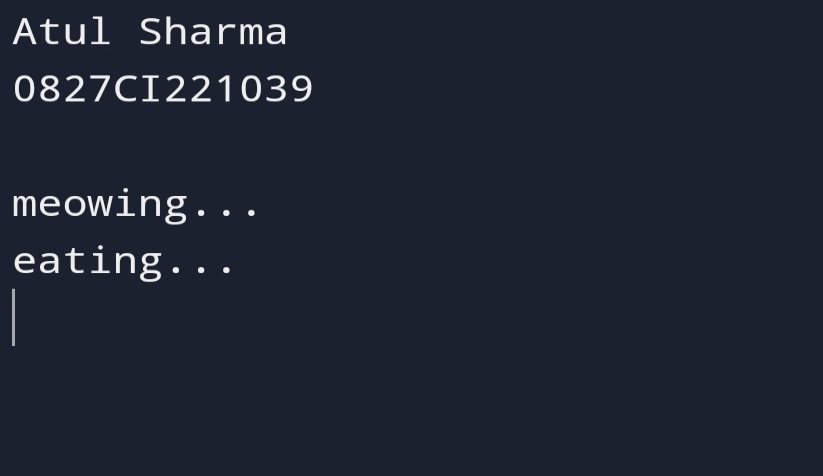
System.out.println(“Atul Sharma\n0827CI221039\n”);

Cat c=new Cat();

c.meow();

c.eat();

}}



Program 20. //Overloading in Java Program 1

Class MethodOverloading {

Private static void display(int a){

System.out.println(“Value of a: “ + a);

}

Private static void display(int a, int b){

System.out.println(“Value of a and b: “ + a + “ and “ + b);

}

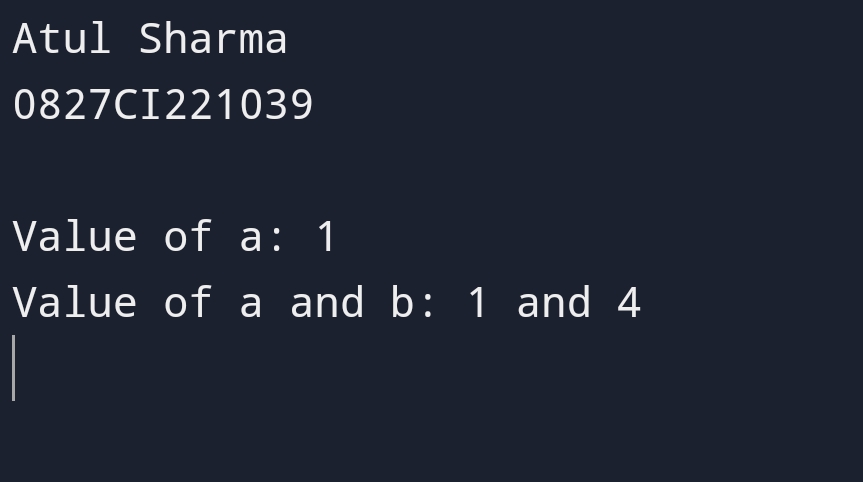
Public static void main(String[] args) {

System.out.println(“Atul Sharma\n0827CI221039\n”);

Display(1);

Display(1, 4);

}

}

Program 21. //Overloading Program 2

Class MethodOverloading {

Private static void display(int a){

System.out.println(“Got Integer data.”);

}

Private static void display(String a){

System.out.println(“Got String object.”);

}

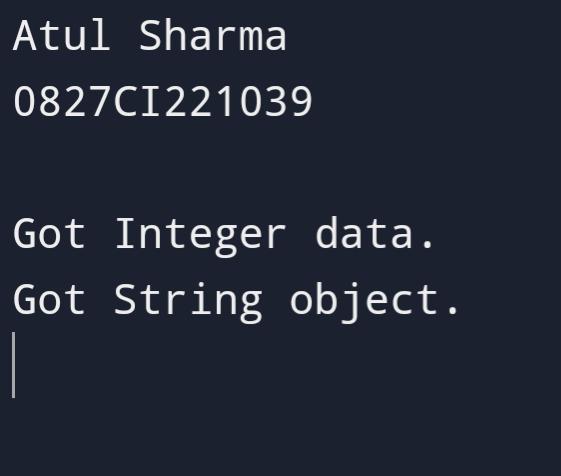
Public static void main(String[] args) {

System.out.println(“Atul Sharma\n0827CI221039\n”);

Display(1);

Display(“Hello”);

}

}

Program 22. //Overriding Program 1

Class Animal {

Public void displayInfo() {

System.out.println(“I am an animal.”);

}

}

Class Dog extends Animal {

Public void displayInfo() {

System.out.println(“I am a dog.”);

}

}

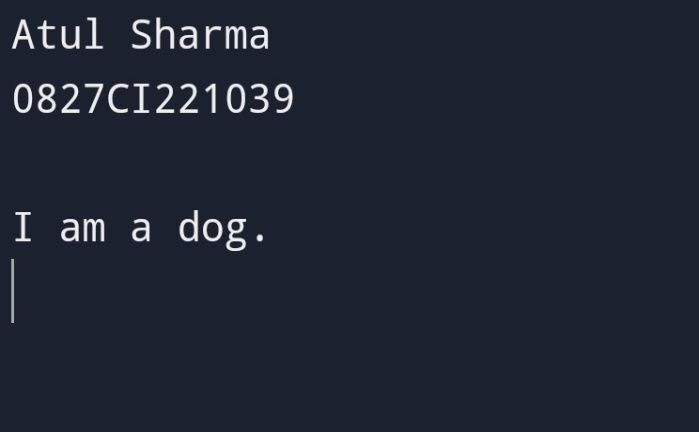
Class Main {

Public static void main(String[] args) {

System.out.println(“Atul Sharma\n0827CI221039\n”);

Dog d1 = new Dog();

D1.displayInfo();

 }}

Program 23. //Overriding Program 2

Class Name {

Public void displayInfo() {

System.out.println(“I am an Atul Sharma”);

}

}

Class Roll extends Name {

Public void displayInfo() {

System.out.println(“My name is Atul Sharma\nRoll no is 0827CI221039\n”);

}

}

Class Main {

Public static void main(String[] args) {

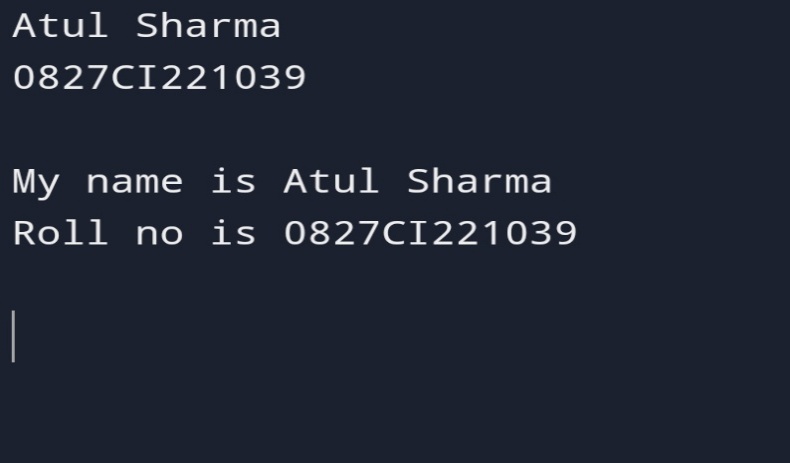
System.out.println(“Atul Sharma\n0827CI221039\n”);

Roll d1 = new Roll();

D1.displayInfo();

}

}



Program 24. // Java program to demonstrate local final variable

Class variable {

Public static void main(String args[])

{

System.out.println(“Atul Sharma\n0827CI221039\n”);

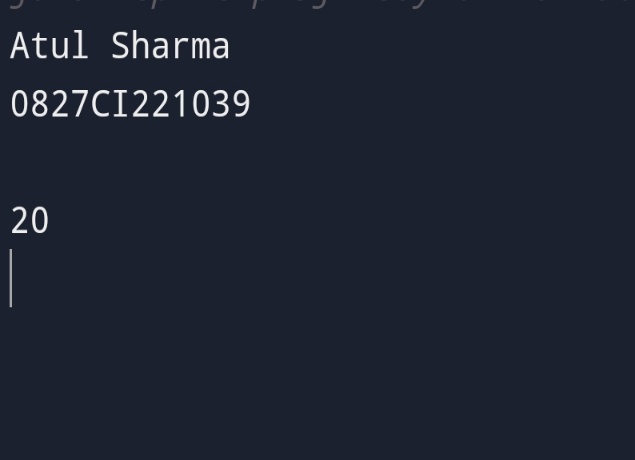
Final int I;

I = 20;

System.out.println(i);

}

}



Program 25. // this Keyword Program

Class Student{

Int rollno;

String name;

Float fee;

Student(int rollno,String name,float fee){

This.rollno=rollno;

This.name=name;

This.fee=fee;

}

Void display(){System.out.println(rollno+” “+name+” “+fee);}

}

Class TestThis2{

Public static void main(String args[]){

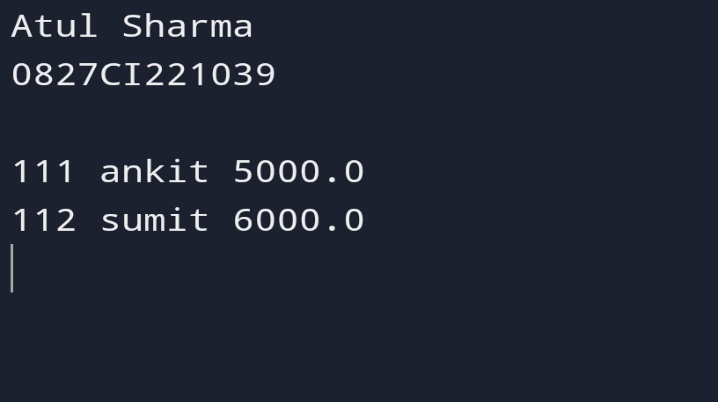
System.out.println(“Atul Sharma\n0827CI221039\n”);

Student s1=new Student(111,”ankit”,5000f);

Student s2=new Student(112,”sumit”,6000f);

S1.display();

S2.display();}}



Program 26. //This Variable Program 2

Class Main {

Int age;

Main(int age){

This.age = age;

}

Public static void main(String[] args) {

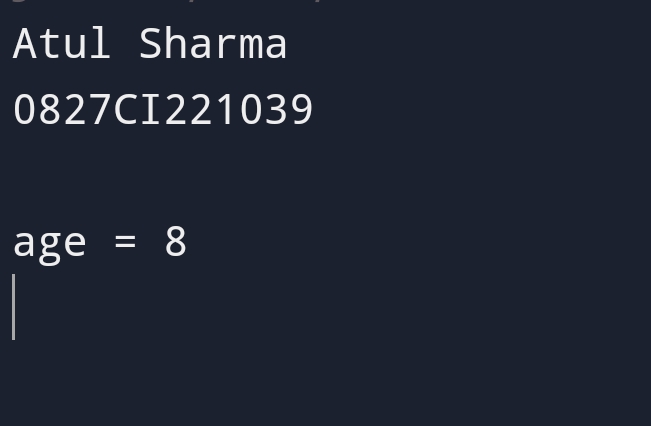
System.out.println(“Atul Sharma\n0827CI221039\n”);

Main obj = new Main(8);

System.out.println(“age = “ + obj.age);

}

}



Program 27. // Super keyword program 1 to call superclass method

Class Animal {

Public void display(){

System.out.println(“I am an animal”);

}}

Class Dog extends Animal {

Public void display(){

System.out.println(“I am a dog”);

}

Public void printMessage(){

Display();

Super.display();

}}

Class Main {

Public static void main(String[] args) {

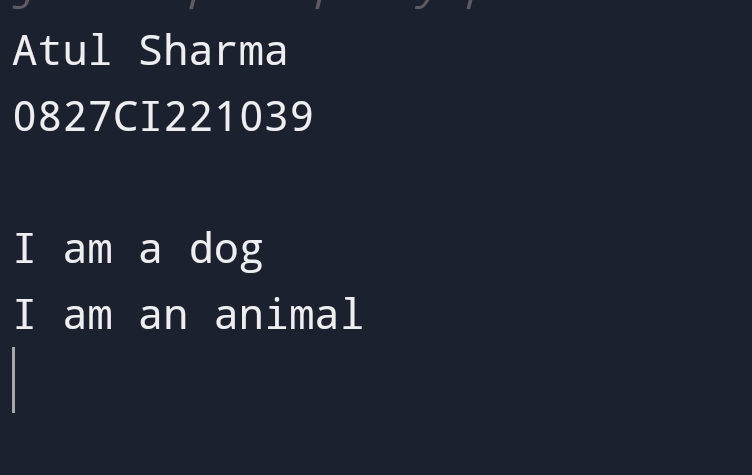
System.out.println(“Atul Sharma\n0827CI221039\n”);

Dog dog1 = new Dog();

Dog1.printMessage();

}

}



Program 28.//Super Keyword program to access superclass attributes

Class Animal {

Protected String type=”animal”;

}

Class Dog extends Animal {

Public String type=”mammal”;

Public void printType() {

System.out.println(“I am a “ + type);

System.out.println(“I am an “ + super.type);

}

}

Class Main {

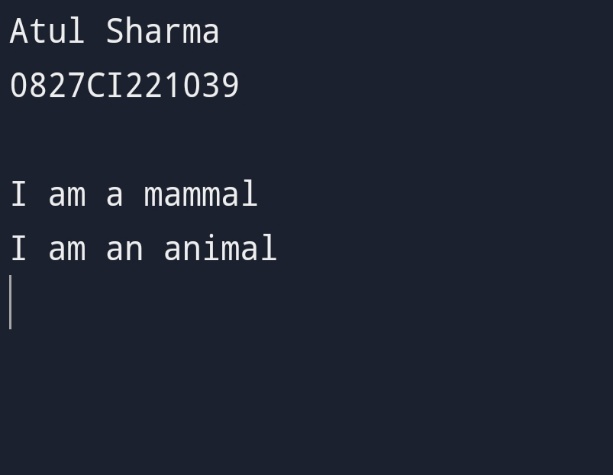
Public static void main(String[] args) {

System.out.println(“Atul Sharma\n0827CI221039\n”);

Dog dog1 = new Dog();

Dog1.printType();

}

}

Program 29.//Static Member in inheritance program

Class Parent {

Static int a = 1;

Static

{ System.out.println( “Inside static block of Parent class”);

System.out.println(“a = “ + a);

} }

Class Child extends Parent {

Static int b = 2;

Static

{ System.out.println( “Inside static block of Child class”);

System.out.println(“b = “ + b);

}

}

Public class Test {

Public static void main(String[] args)

{

System.out.println(“Atul Sharma\n0827CI221039\n”);

Child c = new Child();

 } }

.

Program 30.//Static member in inheritance program 2

Class Parent {

Static int a = 1;

Static {

System.out.println( “Inside static block of Parent class”);

System.out.println(“a = “ + a); }

}

Class Child extends Parent {

Static int b = 2;

Static

{

System.out.println( “Inside static block of Child class”);

System.out.println(“b = “ + b);

} }

Public class Test {

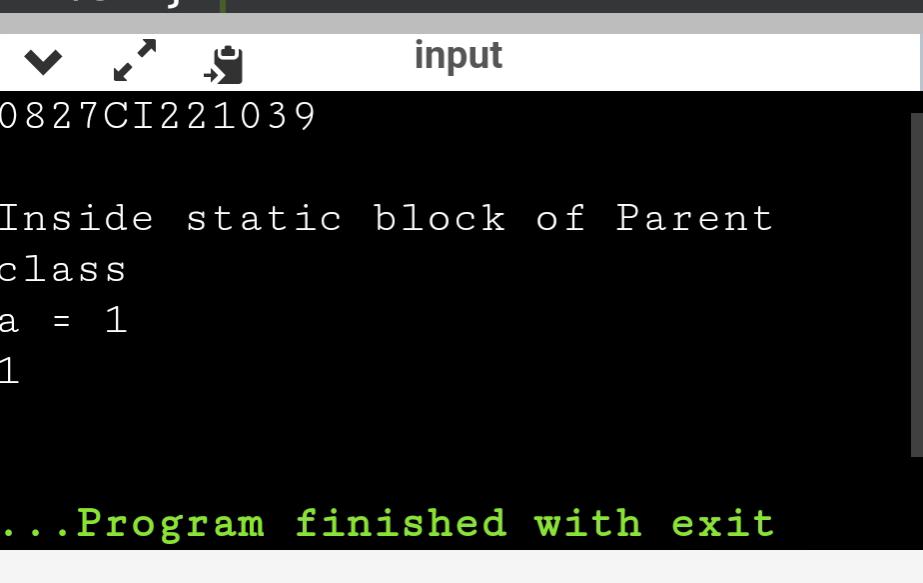
Public static void main(String[] args)

{

System.out.println(“Atul Sharma\n0827CI221039\n”);

System.out.println(Child.a);

}

}

Program 31// Java Program to Illustrate

// Invocation of Constructor

Class Base {

Base()

{

System.out.println(

“Base Class Constructor Called “);

}

}

Class Derived extends Base {

Derived()

{

System.out.println(

“Derived Class Constructor Called “); }

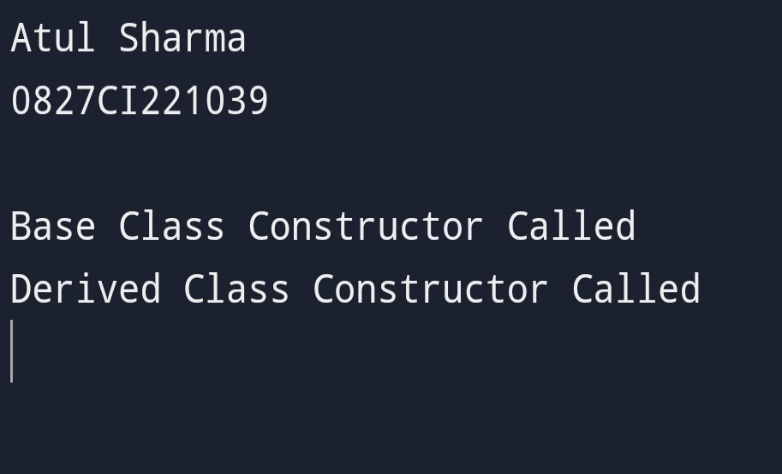
}

Class inheritance {

Public static void main(String[] args)

{

System.out.println(“Atul Sharma\n0827CI221039\n”);

 Derived d = new Derived();

}}

Program 32.// Java Program illustrating constructor in inheritance

Class College

{ College() {

System.out.println(“College constructor executed”);

} }

Class Department extends College

{

Department()

{ System.out.println(“Department constructor executed”);

}}

Class Student extends Department {

Student()

{ System.out.println(“Student constructor executed”);

} }

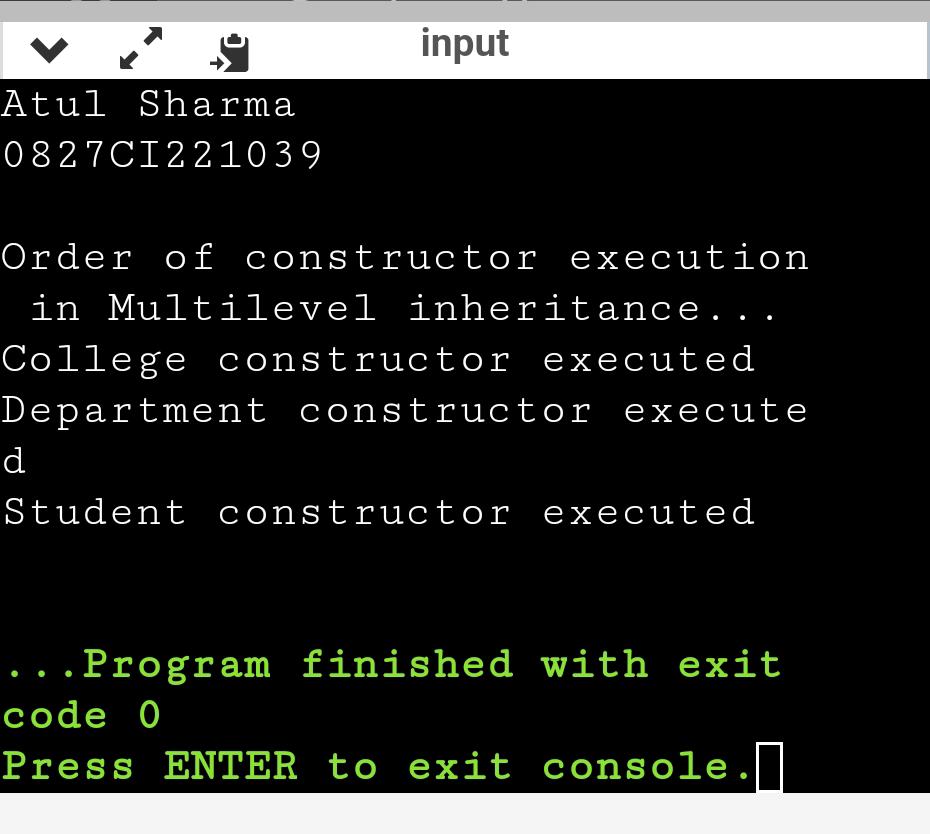
Public class OrderofExecution2

{ Public static void main(String ar[])

{ System.out.println(“Atul Sharma\n0827CI221039\n”);

System.out.println(“Order of constructor execution in Multilevel inheritance…”);

New Student(); }

}

Program 33. //Program Of Constructor Chaining

Public class Constructorchain{

Constructorchain()

{

This (“hello”);

System.out.println(“default constructor called”);

}

Constructorchain(String str)

{

System.out.println(“parameterized constructor called”);

}

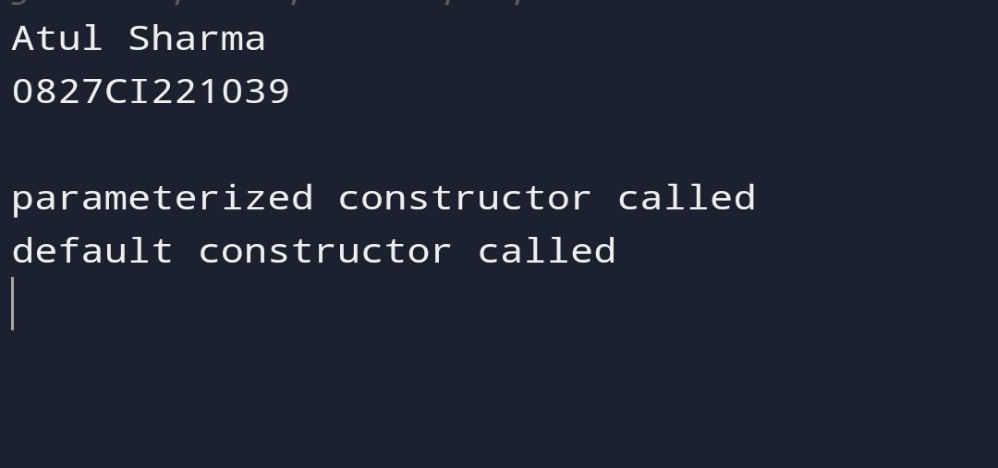
Public static void main(String[] args)

{

System.out.println(“Atul Sharma\n0827CI221039\n”);

Constructorchain cc=new Constructorchain();

}}



Program 34. // Java program to illustrate Constructor Chaining

Class Temp

{

Temp()

{

This(5);

System.out.println(“The Default constructor”);

}

Temp(int x)

{ This(5, 15);

System.out.println(x);

}

Temp(int x, int y)

{

System.out.println(x \* y);

}

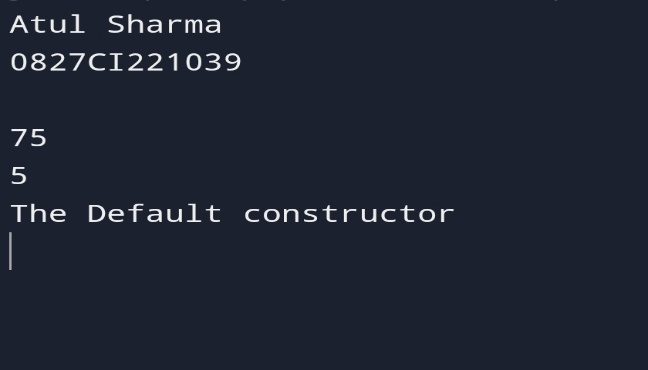
Public static void main(String args[])

{

System.out.println(“Atul Sharma\n0827CI221039\n”);

New Temp(); }

}



Program 35 . // Abstract Class Program

Abstract class Language {

// method of abstract class

Public void display() {

System.out.println(“This is Java Programming”);

}

}

Class Main extends Language {

Public static void main(String[] args) {

System.out.println(“Atul Sharma\n0827CI221039\n”);

// create an object of Main

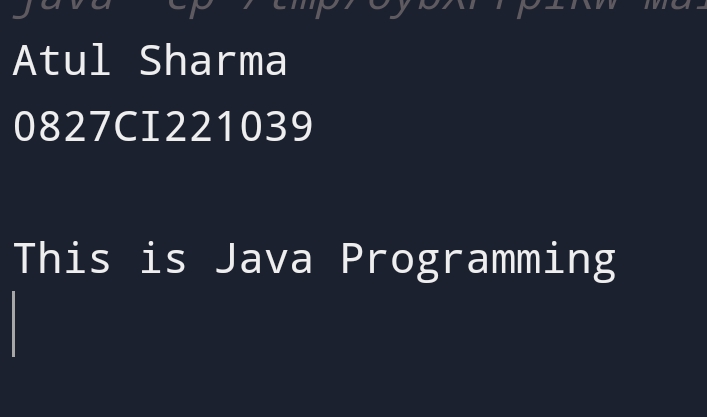
Main obj = new Main();

// access method of abstract class

// using object of Main class

Obj.display();

}

}

Program 36. // Abstract class Program

Abstract class Animal {

Abstract void makeSound();

Public void eat() {

System.out.println(“I can eat.”);

}

}

Class Dog extends Animal {

// provide implementation of abstract method

Public void makeSound() {

System.out.println(“Bark bark”); }}

Class Main {

Public static void main(String[] args) {

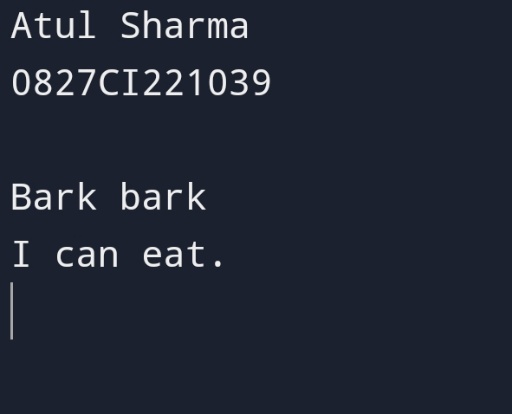
System.out.println(“Atul Sharma\n0827CI221039\n”);

// create an object of Dog class

Dog d1 = new Dog();

D1.makeSound();

D1.eat();

 }

}

Program 37 . //Final Variable Program

Class Main {

Public static void main(String[] args) {

System.out.println(“Atul Sharma\n0827CI221039\n”);

// create a final variable

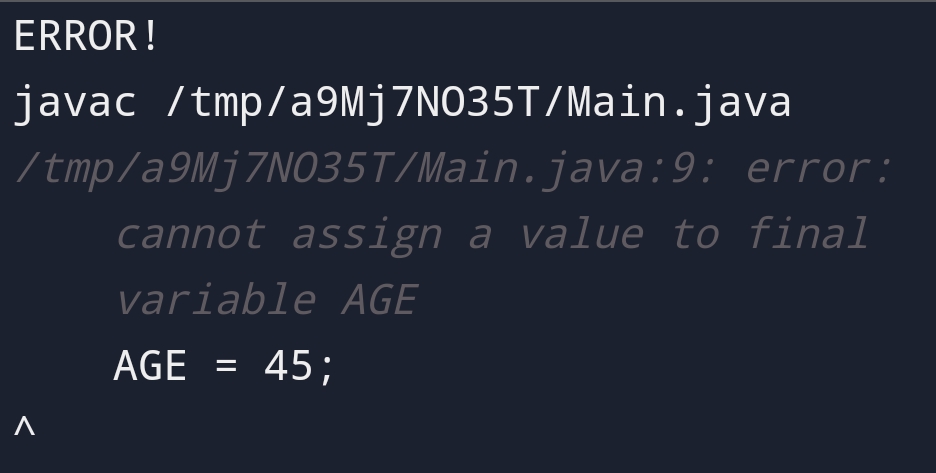
Final int AGE = 32;

// try to change the final variable

AGE = 45;

System.out.println(“Age: “ + AGE);

}

}

Program 38. //Reverse a number(Importing packages)

Import java.util.Scanner;

Public class Main{

Public static void main(String[] args){

System.out.println(“Atul Sharma\n0827CI221039\n”);

Scanner sc = new Scanner(System.in);

System.out.println(“enter a number:”);

Int a = sc.nextInt();

Int rev = 0,q;

While(a!=0){

Q=a%10;

Rev=(rev\*10)+q;

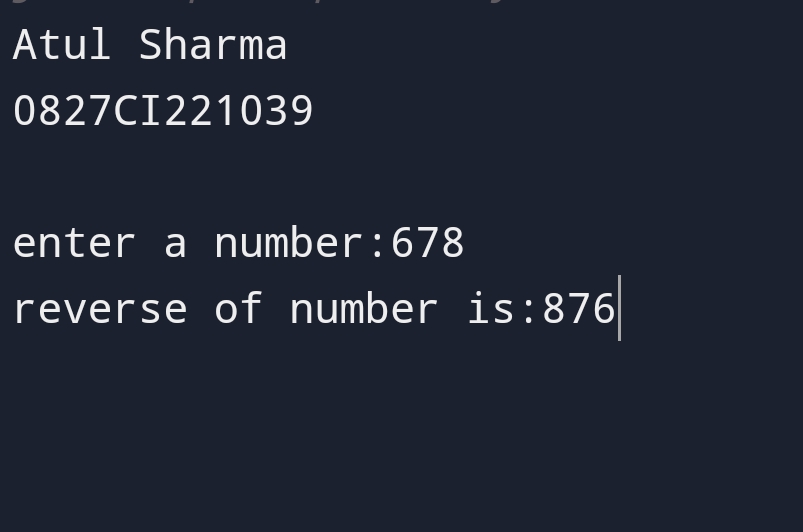
A=a/10;

}

System.out.println(“reverse of number is:”+rev);

}

}



Program 39.//Use of scanner class in Java (packages )

Import java.util.Scanner;

Public class A{

Public static void main(String[] args){

System.out.println(“Atul Sharma\n0827CI221039\n”);

Scanner sc=new Scanner(System.in);

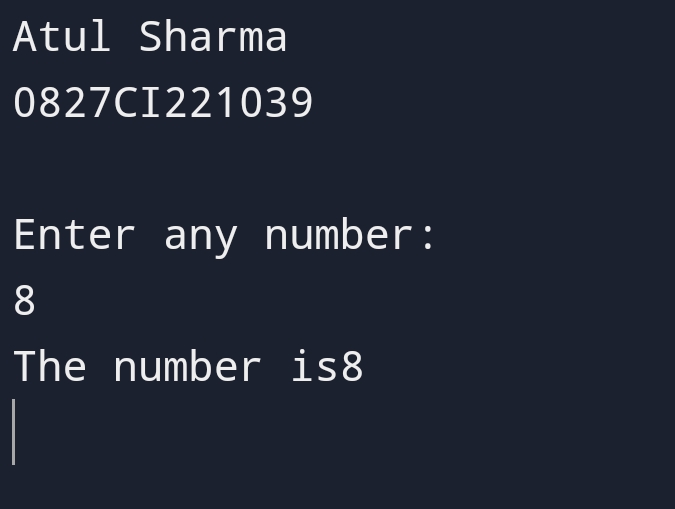
System.out.println(“Enter any number:”);

Int n=sc.nextInt();

System.out.println(“The number is”+n);

}

}



Program 40. //Interface program in java

Interface Polygon {

Void getArea(int length, int breadth);

}

// implement the Polygon interface

Class Rectangle implements Polygon {

// implementation of abstract method

Public void getArea(int length, int breadth) {

System.out.println(“The area of the rectangle is “ + (length \* breadth));

}

}

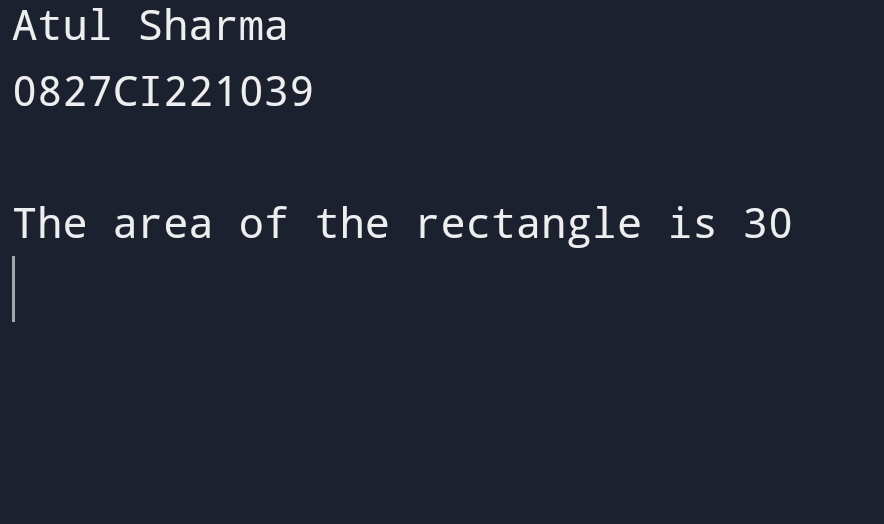
Class Main {

Public static void main(String[] args) {

System.out.println(“Atul Sharma\n0827CI221039\n”);

Rectangle r1 = new Rectangle();

R1.getArea(5, 6);

 }}

Program 41.//Interface program in java

Interface Language {

Void getName(String name);

}

// class implements interface

Class ProgrammingLanguage implements Language {

// implementation of abstract method

Public void getName(String name) {

System.out.println(“Programming Language: “ + name);

}

}

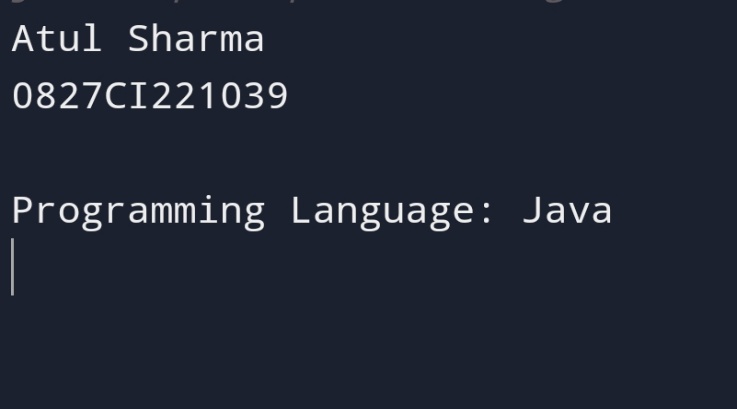
Class Main {

Public static void main(String[] args) {

System.out.println(“Atul Sharma\n0827CI221039\n”);

ProgrammingLanguage language = new ProgrammingLanguage();

Language.getName(“Java”);

 }

}

Program 42. // Program for difference between abstract class and interface in java

Interface A{

Void a();//bydefault, public and abstract

Void b(); Void c(); Void d(); }

//Creating abstract class that provides the implementation of one method of A interface

Abstract class B implements A{

Public void c(){System.out.println(“I am C”);} }

//Creating subclass of abstract class, now we need to provide the implementation of rest of the methods

Class M extends B{

Public void a(){System.out.println(“I am a”);}

Public void b(){System.out.println(“I am b”);}

Public void d(){System.out.println(“I am d”);}

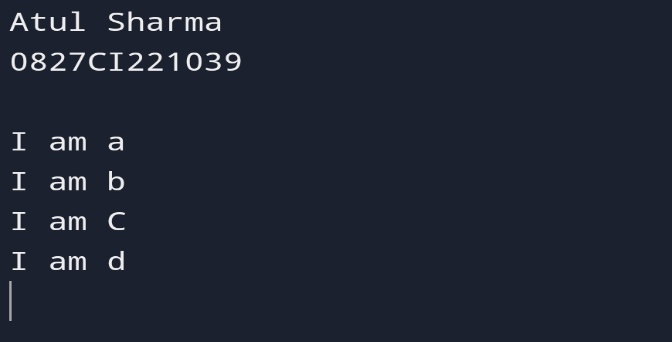
} //Creating a test class that calls the methods of A interface

Class Test{

Public static void main(String args[]){ System.out.println(“Atul Sharma\n0827CI221039\n”);

A a=new M();

a.a();

a.b();

a.c();

a.d();

}}

Program 43. //Program 2 for difference between abstract Class and Interface

Public class JavaTester {

Public static void main(String args[]) {

System.out.println(“Atul Sharma\n0827CI221039\n”);

Animal tiger = new Tiger();

Tiger.eat();

Cat lion = new Lion();

Lion.eat();

}}

Interface Animal {

Public void eat();

}

Class Tiger implements Animal {

Public void eat(){

System.out.println(“Tiger eats”);

}}

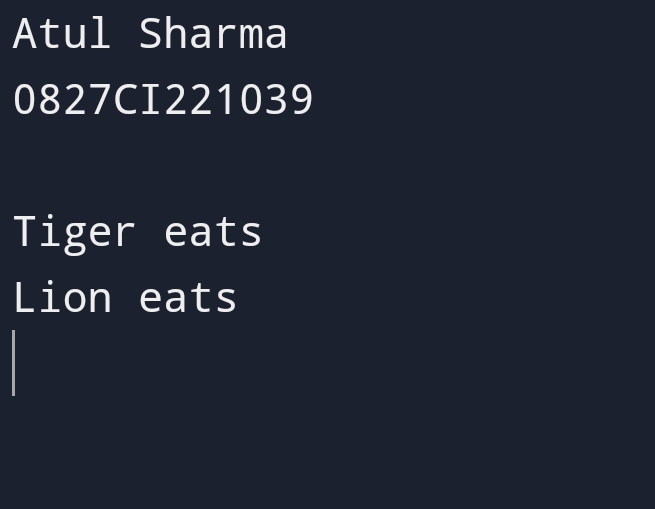
Abstract class Cat {

Abstract public void eat();}

Class Lion extends Cat{

Public void eat(){

System.out.println(“Lion eats”);

 }}

Program 44.//Array Program In Java

Class array {

Public static void main(String[] args)

{

System.out.println(“Atul Sharma\n0827CI221039\n”);

Int []age ={1,2,3};

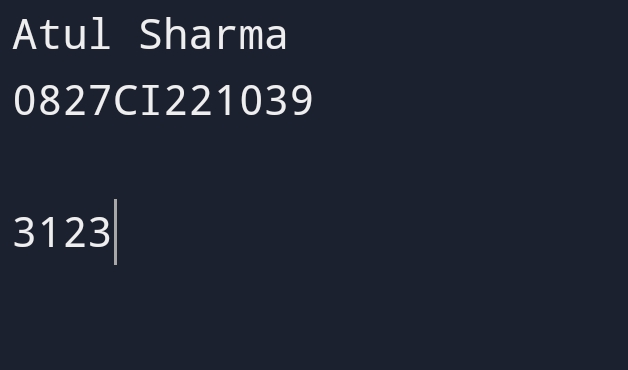
System.out.print(age.length );

For(int a:age){

System.out.print(a);

}

}}



Program 45. //Java Program To find sum and average using array

Class array {

Public static void main(String[] args)

{

System.out.println(“Atul Sharma\n0827CI221039\n”);

Int []num ={1,2,3,4,5,6,9,10};

Int sum=0;

Double avg=0;

System.out.println(num.length);

For(int a:num){

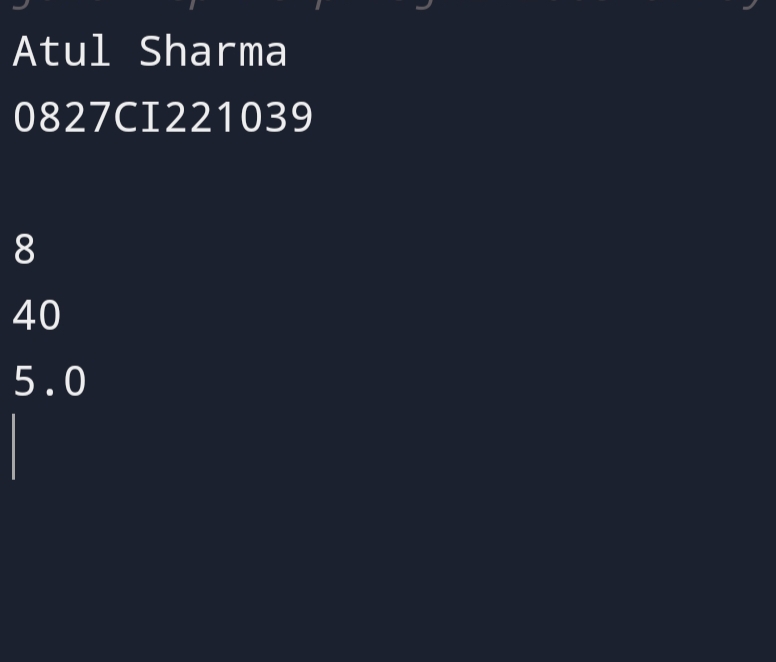
Sum=sum+a;}

System.out.println(sum);

Avg=sum/num.length;

System.out.println(avg);

}}



Program 46. // Two dimensional array program by user in java

Class TwodimentionalLoop

{

Public static void main(String args[])

{

System.out.println(“Atul Sharma\n0827CI221039\n”);

Int[][] a ={{10,20},{30,40},{50,60}};

System.out.println(“Two dimentional array elements are “);

For(int i=0;i<3;i++)

{

For(int j=0;j<2;j++){

System.out.print(a[i][j]+” “);

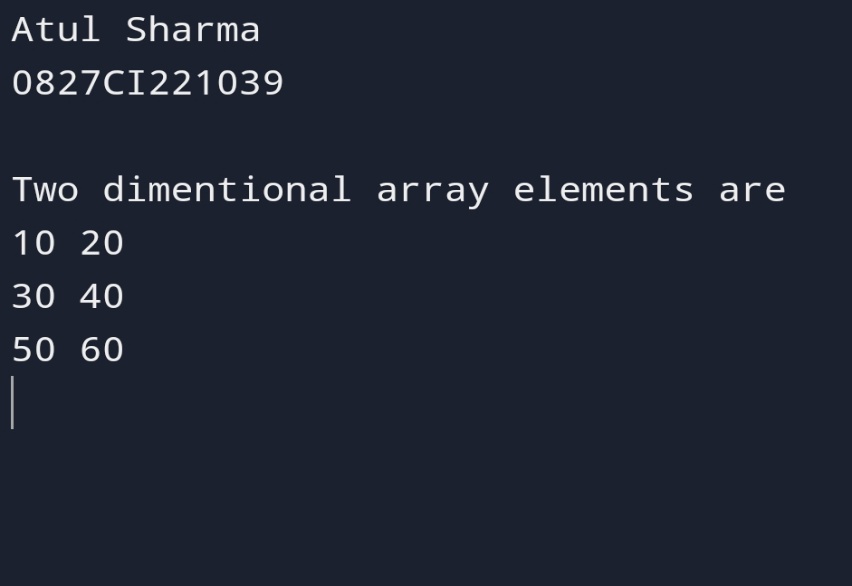
}

System.out.println();

}

}

}



Program 47. //Average program of two dimensional array in java

Public class Average

{

Public static double getAvg(int [][] a)

{

Double total = 0;

For(int[] i:a)

{

For(int val:i)

{

System.out.println(val);

Total=total+val;

} }

Return total/(a.length\*a[0].length);

} Public static void main(String args[])

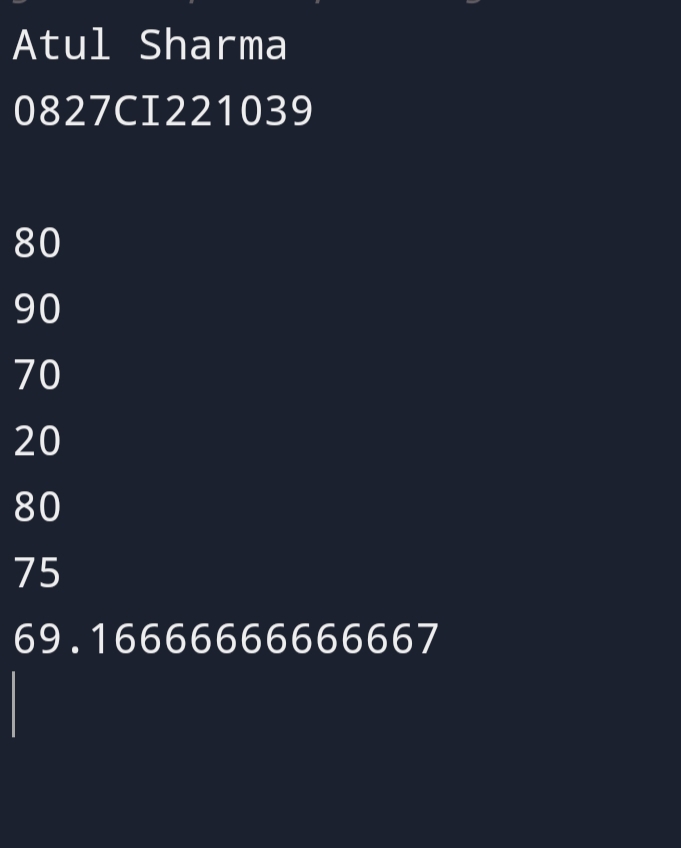
{System.out.println(“Atul Sharma\n0827CI221039\n”);

Int[][]theArray = {{80,90,70},{20,80,75}};

System.out.println(getAvg(theArray));

}

}



Program 48 . //String program in java

Class main

{ public static void main(String args[])

{

System.out.println(“Atul Sharma\n0827CI221039\n”);

String first=”java”;

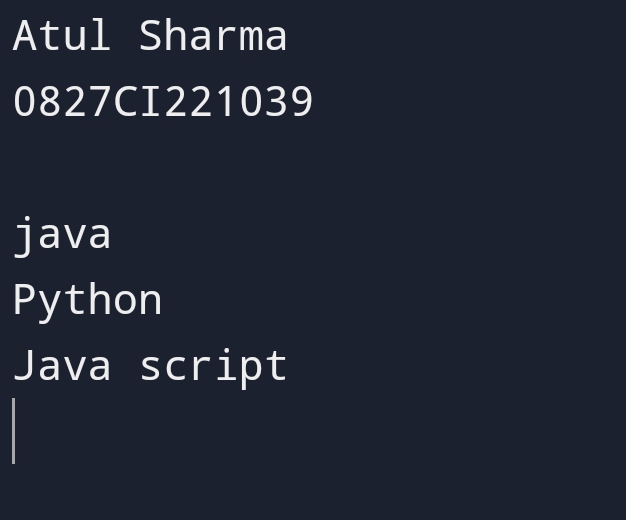
String second=”Python”;

String third=”Java script”;

System.out.println(first);

System.out.println(second);

System.out.println(third);

 }}

Progam 49. //String program 2 in java

Class main

{ public static void main(String args[])

{ //CREATE A STRING

System.out.println(“Atul Sharma\n0827CI221039\n”);

String greet=”ATUL SHARMA”;

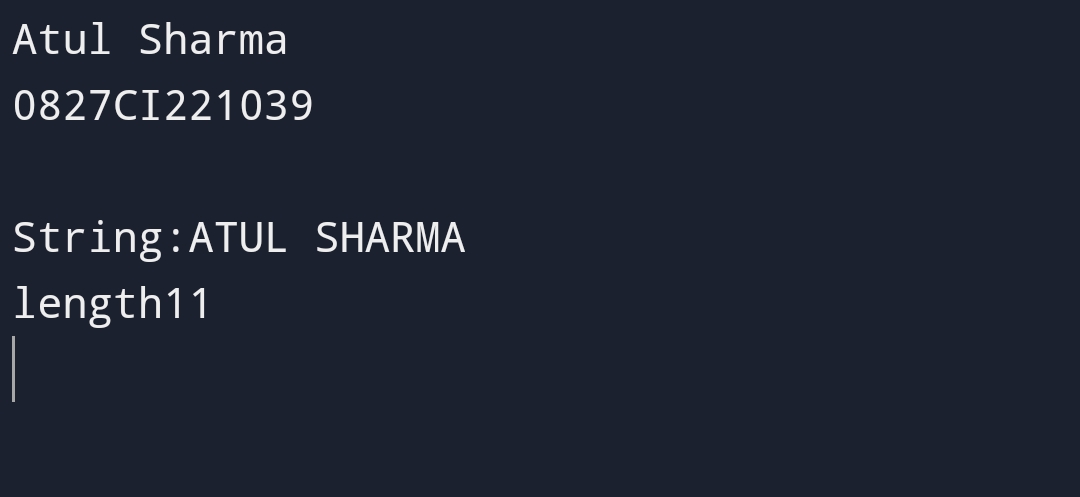
System.out.println(“String:”+greet);

//get the length of greet

Int length =greet.length();

System.out.println(“length”+length);

}

}

Program 50.//Exception handling program in java

Class Main {

Public static void main(String[] args) {

System.out.println(“Atul Sharma\n0827CI221039\n”);

Try {

// code that generate exception

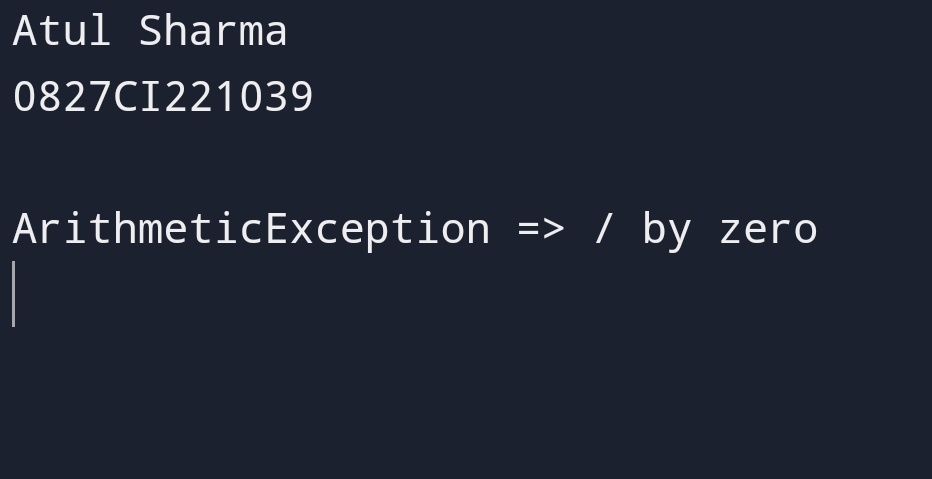
Int divideByZero = 5 / 0;

System.out.println(“Rest of code in try block”);

} Catch (ArithmeticException e) {

System.out.println(“ArithmeticException => “ + e.getMessage());

}

 }

}

Program 51. //Exception handling program 2 in java

Import java.util.Scanner;

Public class Test

{

Public static void main(String args[])

{

System.out.println(“Atul Sharma\n0827CI221039\n”);

Scanner scn = new Scanner(System.in);

Try

{ Int n = Integer.parseInt(scn.nextLine());

If (99%n == 0)

System.out.println(n + “ is a factor of 99”);

}

Catch (ArithmeticException ex)

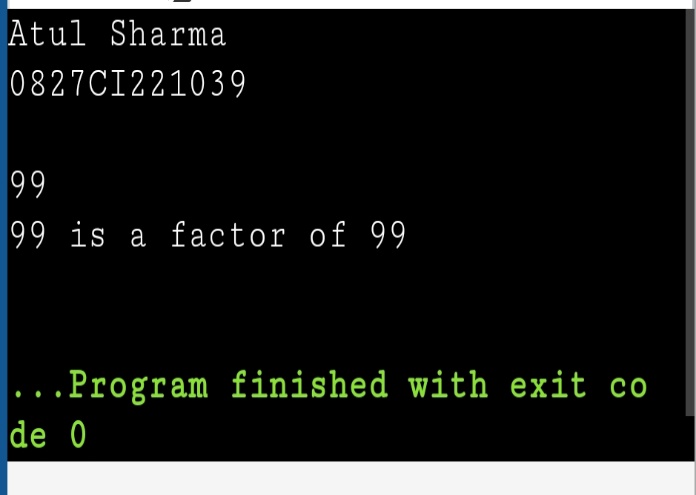
{System.out.println(“Arithmetic “ + ex); }

Catch (NumberFormatException ex)

{

System.out.println(“Number Format Exception “ + ex);

} }}



Program 52. //Unchecked exception handling Null pointer exception in java

Public class NassMultiCatch {

Public static void main(String[] args) {

System.out.println(“Atul Sharma\n0827CI221039\n”);

String s=null;

Int[] arr = {10, 20, 30};

Try {

System.out.println(s.length());

System.out.println(arr[2]);

} catch (ArrayIndexOutOfBoundsException el) {

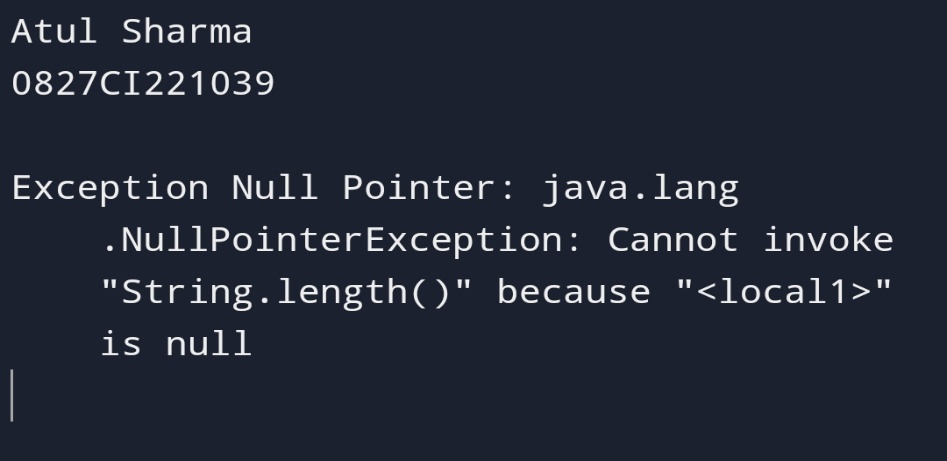
System.out.println(“Exception Index Out of Bounds: “ + el);

} catch (NullPointerException e2) {

System.out.println(“Exception Null Pointer: “ + e2);

}

}}



PROGRAM 53. // Unchecked Out of bounds exception program in java

Import java.util.Scanner;

Public class ex{

// Main Driver Method

Public static void main(String[] args) {

System.out.println(“Atul Sharma\n0827CI221039\n”);

// Inserting elements into Array

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

// Try block for exceptions

Try {

// Forcefully trying to access and print

// element/s beyond indexes of the array

System.out.println(a[5]); }

// Catch block for catching exceptions

Catch (ArrayIndexOutOfBoundsException e) {

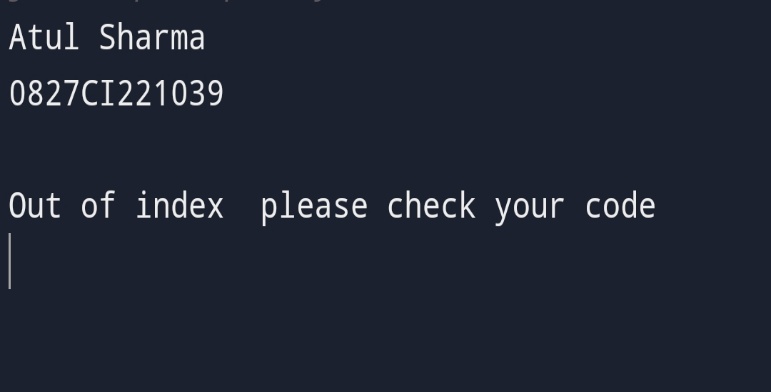
// Printing display message when index not

// present in a array is accessed

System.out.println(

“Out of index please check your code”);

}}}



Program 54. //Program of throw keyword in java

Import java.io.FileReader;

Import java.io.FileNotFoundException;

Public class ThrowsExample1 {

Static void readFile(String filePath) throws FileNotFoundException {

FileReader reader = new FileReader(filePath);

}

Public static void main(String[] args) {

System.out.println(“Atul Sharma\n0827CI221039\n”);

Try {

readFile(“nonexistentfile.txt”);

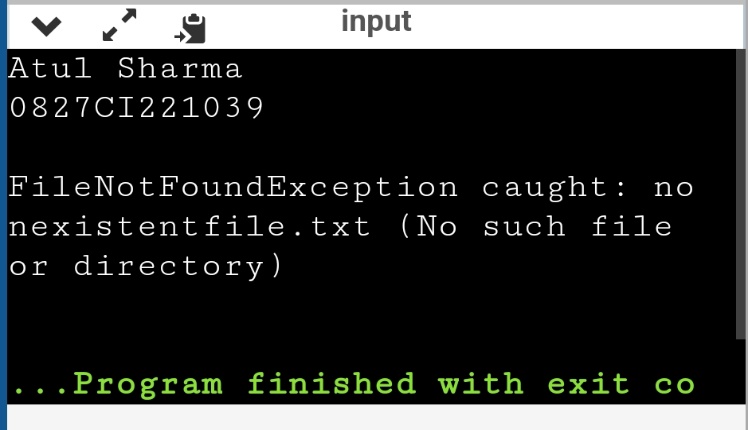
} catch (FileNotFoundException e) {

System.out.println(“FileNotFoundException caught: “ + e.getMessage());

}

}

}



Program 55. //Throws Program exception handling java

Import java.io.IOException;

Class Testthrows1{

Void m()throws IOException{

Throw new IOException(“device error”);//checked exception

}

Void n()throws IOException{

M();

}

Void p(){

Try{

N();

}catch(Exception e){System.out.println(“exception handled”);}

}

Public static void main(String args[]){

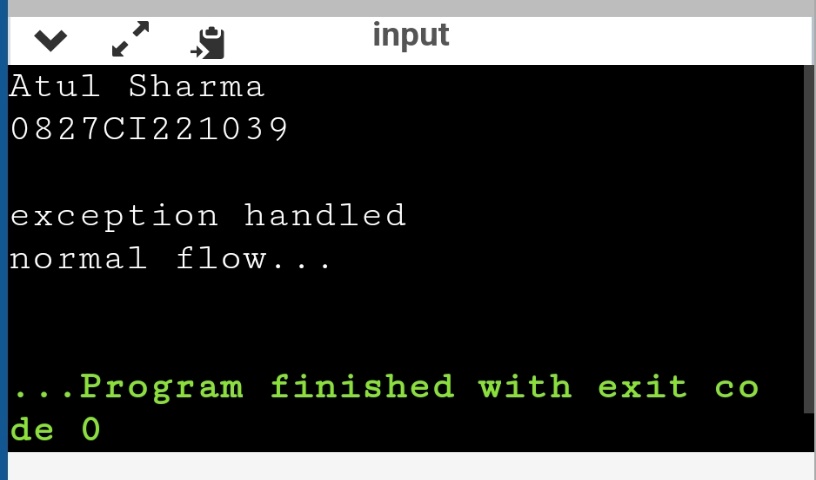
System.out.println(“Atul Sharma\n0827CI221039\n”);

Testthrows1 obj=new Testthrows1();

Obj.p();

System.out.println(“normal flow…”);

}

}

Program 56.//Program on introduction to threads in java

Class Main extends Thread{

Public void run(){

System.out.println(“This code is running in thread”);

}

Public static void main(String []args){

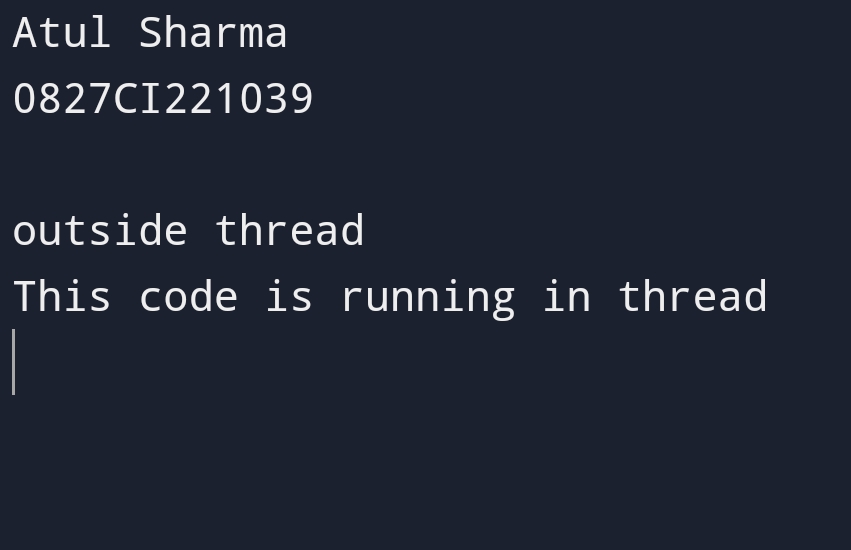
System.out.println(“Atul Sharma\n0827CI221039\n”);

Main thread=new Main();

Thread.start();

System.out.println(“outside thread”);

}

}

Program 57. //Program on introduction to Threads in java

Public class Main extends Thread {

Public void run() {

System.out.println(“Welcome To CSIT”);

}

Public static void main(String[] args) {

System.out.println(“Atul Sharma\n0827CI221039\n”);

Main t1 = new Main();

Main t2 = new Main();

Main t3 = new Main();

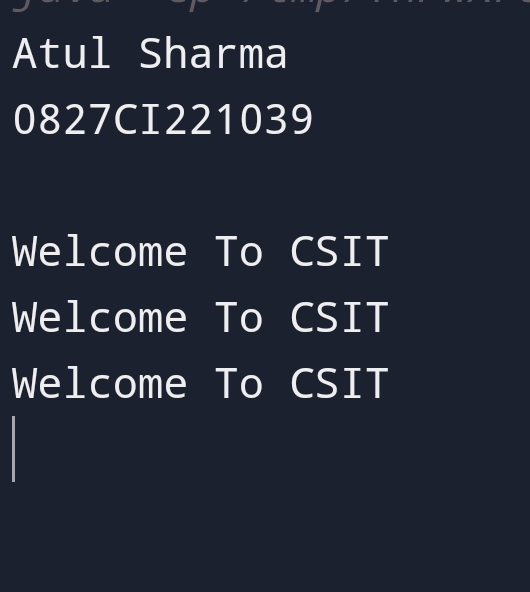
T1.start();

T2.start();

T3.start();

}

}



Program 58. //Java Threads using Runnable interface

Public class Main implements Runnable {

Public static void main(String[] args) {

System.out.println(“Atul Sharma\n0827CI221039\n”);

Main obj = new Main();

Thread thread = new Thread(obj);

Thread.start();

System.out.println(“This code is outside of the thread”);

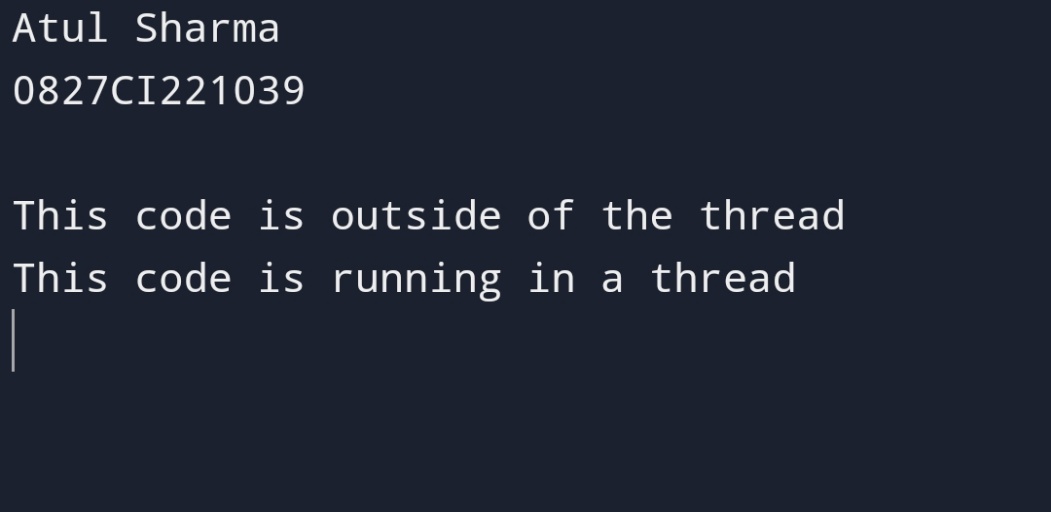
}

Public void run() {

System.out.println(“This code is running in a thread”);

}

}



Program 59. //Java Thread program Using Runnable interface

Public class Main implements Runnable{

Public void run() {

System.out.println(“Welcome To CSIT”);

}

Public static void main(String[] args) {

System.out.println(“Atul Sharma\n0827CI221039\n”);

Main t1 = new Main();

Thread n1=new Thread(t1);

Main t2 = new Main();

Thread n2=new Thread(t2);

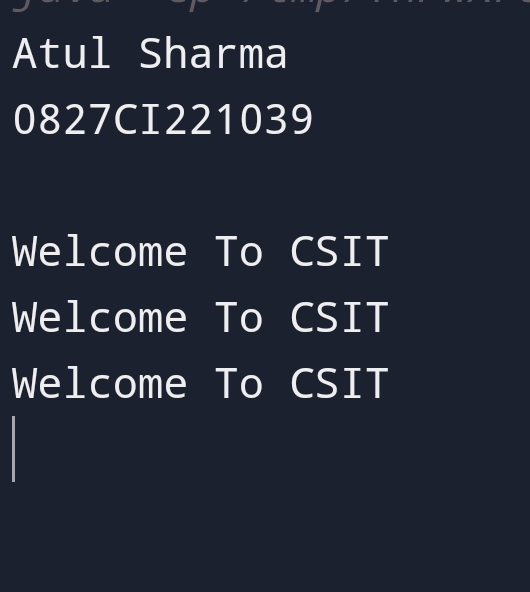
Main t3 = new Main();

Thread n3=new Thread(t3);

N1.start();

N2.start();

N3.start();

 }}

Program 60 .//Thread in java using thread class

Public class MyThread1

{

// Main method

Public static void main(String argvs[])

{

System.out.println(“Atul Sharma\n0827CI221039\n”);

// creating an object of the Thread class using the constructor Thread(String name)

Thread t= new Thread(“My first thread”);

// the start() method moves the thread to the active state

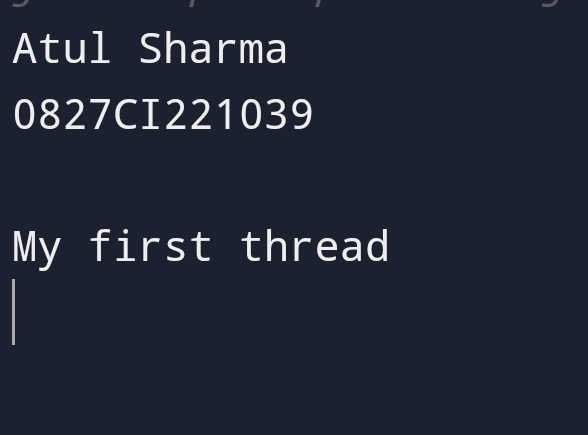
t.start();

// getting the thread name by invoking the getName() method

String str = t.getName();

System.out.println(str);

}

}

Program61. //Thread using Thread class in java

Public class MyThread2 implements Runnable

{

Public void run()

{

System.out.println(“Now the thread is running …”);

}

Public static void main(String argvs[])

{

System.out.println(“Atul Sharma\n0827CI221039\n”);

Runnable r1 = new MyThread2();

Thread th1 = new Thread(r1, “My new thread”);

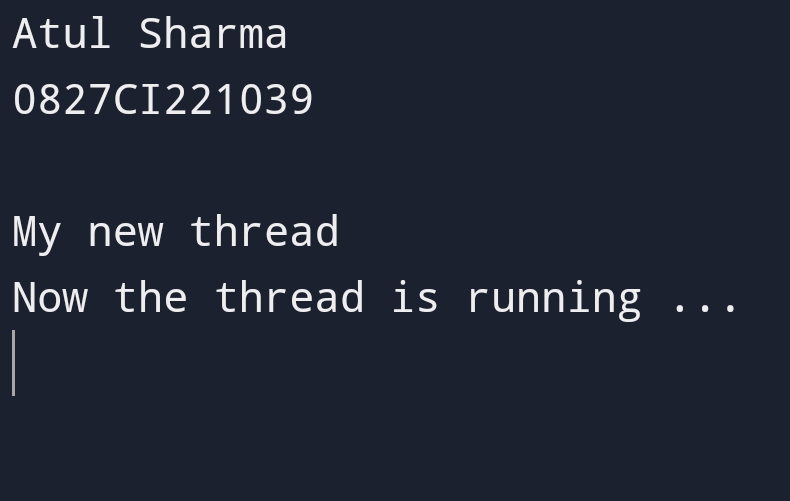
Th1.start();

String str = th1.getName();

System.out.println(str);

}

}



Program 62. //Java Program On File

Import java.io.File;

Class Main {

Public static void main(String[] args) {

System.out.println(“\nAtul Sharma\n0827CI221039”);

// create a file object for the current location

File file = new File(“newFile.txt”);

Try {

// trying to create a file based on the object

Boolean value = file.createNewFile();

If (value) {

System.out.println(“The new file is created.”);

}

Else {

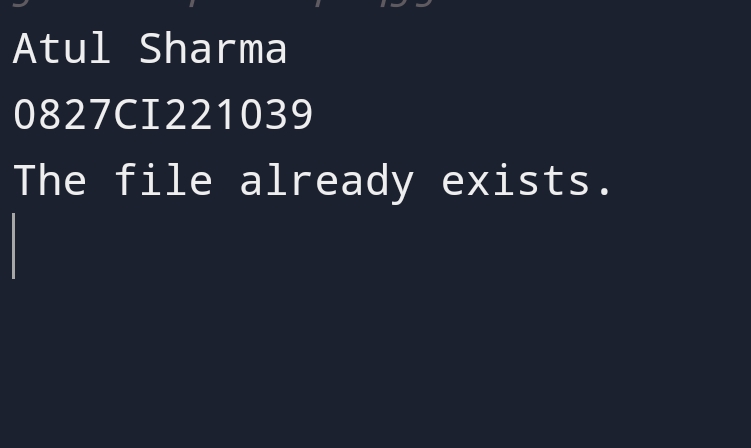
System.out.println(“The file already exists.”);

}

}

Catch(Exception e) {

e.getStackTrace();

 } }}

Program 63 .//Java Program To write data using outputstream

Import java.io.FileOutputStream;

Import java.io.IOException;

Public class FileOutputStreamExample1 {

Public static void main(String[] args) {

System.out.println(“Atul Sharma\n0827CI221039\n”);

String filePath = “output.txt”;

Try (FileOutputStream fos = new FileOutputStream(filePath)) {

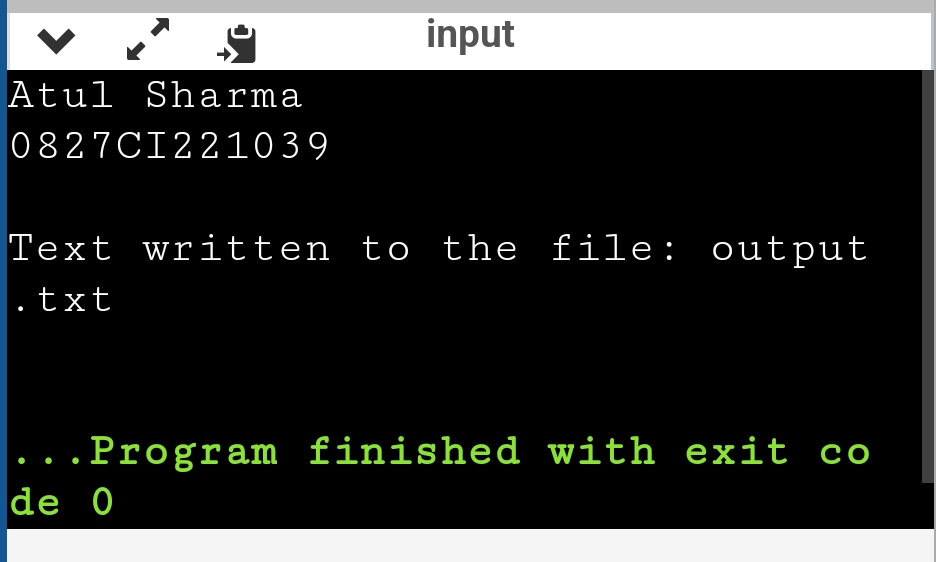
String text = “Hello, this is a sample text.\nWriting to a file in Java.”;

Fos.write(text.getBytes());

System.out.println(“Text written to the file: “ + filePath);

} catch (IOException e) {

e.printStackTrace(); } }

}

Program 64. //Java Program to write data using output.stream

Import java.io.FileOutputStream;

Import java.io.IOException;

Public class FileOutputStreamExample2 {

Public static void main(String[] args) {

System.out.println(“Atul Sharma\n0827CI221039\n”);

String filePath = “binaryoutput.dat”;

Try (FileOutputStream fos = new FileOutputStream(filePath)) {

Int data1 = 123;

Int data2 = 456;

Fos.write(intToBytes(data1));

Fos.write(intToBytes(data2));

System.out.println(“Binary data written to the file: “ + filePath);

} catch (IOException e) {

e.printStackTrace(); } }

Private static byte[] intToBytes(int value) {

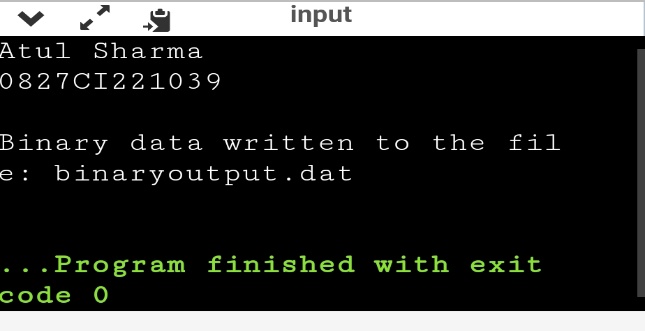
Return new byte[]{

(byte) (value >> 24),

(byte) (value >> 16),

(byte) (value >> 8),

(byte) value

 }; }}

Program 65. //Java Program to read data from file using inputstream

Import java.io.FileInputStream;

Import java.io.IOException;

Public class FileInputStreamExample1 {

Public static void main(String[] args) {

System.out.println(“Atul Sharma\n0827CI221039\n”);

String filePath = “input.txt”;

Try (FileInputStream fis = new FileInputStream(filePath)) {

Int data;

While ((data = fis.read()) != -1) {

System.out.print((char) data);

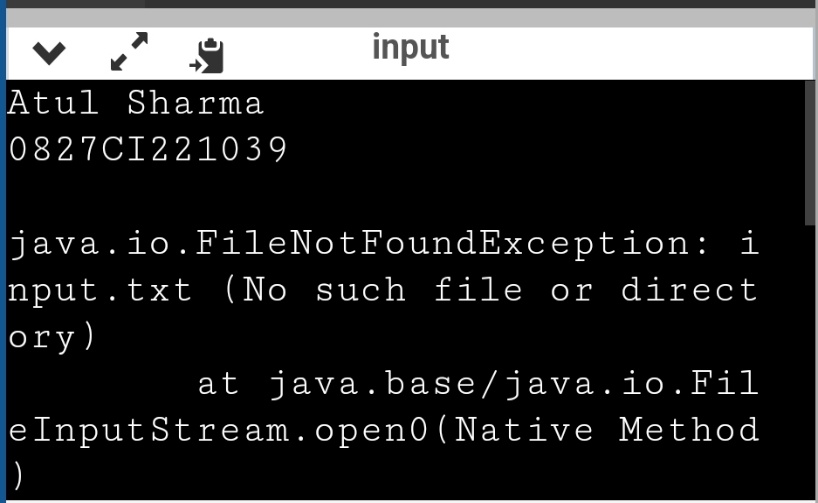
}

} catch (IOException e) {

e.printStackTrace();

}

}

}Program 66. //Java Program to read data using inputstream

Import java.io.FileInputStream;

Import java.io.IOException;

Public class FileInputStreamExample2 {

Public static void main(String[] args) {

System.out.println(“Atul Sharma\n0827CI221039\n”);

String filePath = “binaryinput.dat”;

Try (FileInputStream fis = new FileInputStream(filePath)) {

Byte[] buffer = new byte[4];

Int bytesRead = fis.read(buffer);

If (bytesRead == 4) {

Int data1 = bytesToInt(buffer);

System.out.println(“Read data 1: “ + data1); }

bytesRead = fis.read(buffer);

if (bytesRead == 4) {

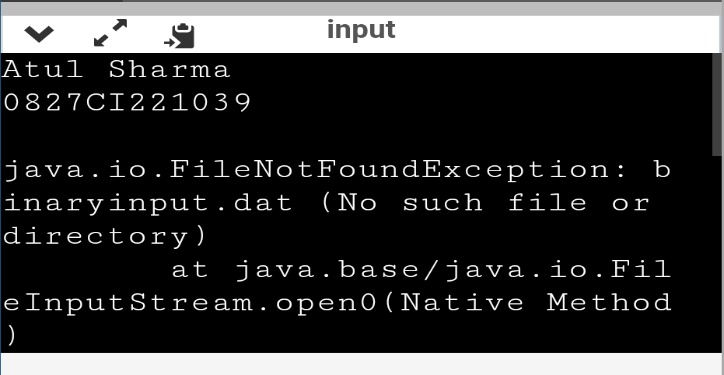
int data2 = bytesToInt(buffer);

System.out.println(“Read data 2: “ + data2); }

} catch (IOException e) {

e.printStackTrace() } }

Private static int bytesToInt(byte[] bytes) {

 Return (bytes[0] << 24) | ((bytes[1] & 0xFF) << 16) | ((bytes[2] & 0xFF) << 8) | (bytes[3] & 0xFF);

}}

Program 67.//Garbage Collection in java Program

Public class TestGarbage1{

Public void finalize(){System.out.println(“object is garbage collected”);}

Public static void main(String args[]){

System.out.println(“Atul Sharma\n0827CI221039\n”);

TestGarbage1 s1=new TestGarbage1();

TestGarbage1 s2=new TestGarbage1();

S1=null;

S2=null;

System.gc();

}

}



Program 68. //Java Finalize Method Program

Public class JavafinalizeExample1 {

Public static void main(String[] args)

{

System.out.println(“Atul Sharma\n0827CI221039\n”);

JavafinalizeExample1 obj = new JavafinalizeExample1();

System.out.println(obj.hashCode());

Obj = null;

// calling garbage collector

System.gc();

System.out.println(“end of garbage collection”);

}

Protected void finalize()

{

System.out.println(“finalize method called”);

}

}

