23CSE111

OBJECT ORIENTED PROGRAMMING LAB MANUAL

Department of Computer and Science Engineering

Amrita School of Engineering Amrita Vishwa Vidyapeetham,
Amaravati Campus.

NAME:- A.Rishitha

ROLL.NO:-AV.SC.U4.CSE24005

Verified by:

Sno	programs	date	Pg no	Signature
WEEK 1		27-01-2025		
1.	Write the steps to download and install Java.			
2.	Write a java program to printthe message“Welcome to java programming”.			
3.	Write a java program that prints name, roll number and section of a student.			
WEEK-2		03-02-2025		
1.	Write a java program to calculate the area of a rectangle.			
2.a)	Write a program to convert temperature from Celsius to Fahrenheit .			
2.b)	Write a java program to convert temperature from Fahrenheit to Celsius.			
3.	Write a java program to calculate the simple interest			

4.	Write a java program to find the largest of three numbers using ternary operator.			
5.	Write a java program to find the factorial of a number.			
WEEK-3		11-02-2025		
1.	Create a java program with following instructions a)create a class with name car b) Create 4 attributes name car color, car brand, fuel type, milage. c) Create 3 methods named start, stop, services d) Create 3objects named car1, car2, car3. e) Create a constructor which should print “welcome to car garage”			
2.	Write a java program to create a class BackAccount with two methods deposit( ) and withdraw() a) In deposit( ) whenever an amount is deposited it has to be updated with current amount b) In withdraw( ) whenever an amount is			

	withdrawn it has to be less than current amount else print “Insufficient funds”			
WEEK-4		02-03-2025		
1.	Write a java program with class named “Book”. The class should contain various attributes such as “Title of the book , author , year of publication “. It should also contain a constructor with parameters details of the book. i.e. “ Title of the book, author and year of publication”. Display the details of two books by creating two objects.			
2.	To create a java program with class named Myclass with a staticvariable “Count” of “int type”, Initialized to 0 and a constant variable “pi” of type double initialized to 3.1415 as attributes of that class Now, define a constructor for “Myclass” that increments the “Count” variable each that an object of Myclass is created. Finally , print the final values of “Count” and “pi”variables			
WEEK-5		09-03-2025		

1.	Create a calc using the operations including add, sub, mul, div using multilevel inheritance and display the desired output			
2.	Creating a Rental Sysytem			
WEEK-6		16-03-2025		
1.	Write a java program to create a Vehicle class with displayInfo() method , overridden in Car subclass to provide info about carcompany , model , price ,seating and petrol.			
2.	An automated admission system that verifies student eligibility for UG and PG with different criteria. 1.UG requires minimum of 60% 2.PG requires minimum of 70%			
3.	Create a calculator class with overloaded methods to perform additions 1.add two integers. 2.add two double values 3.add three integers.			
4.	Create a shape class with method calculateArea() that is overloaded for different shapes (eg:			

	square, rectangle).Then create a subclass Circle that overrides calculateArea() method for Circle.			
WEEK-7:		14-04-2025		
1.	Write a Java program to create an abstract class Animal with an abstract method called sound(). Create subclasses Lion and Tiger that extend the Animal class and implement the sound() method to make a specific sound for each animal.			
2.	To write a Java program to create an abstract class Shape3D with abstract methods calculateVolume() and calculateSurfaceArea(). Create subclasses Sphere and Cube that extend the Shape3D class and implement the respective methods to calculate the volume and surface area of each shape.			
3.	Write a java program to create a abstract class named patternprinter with an abstract method printpattern(int n) and a			

	concrete method to display the pattern title.			

WEEK-1:

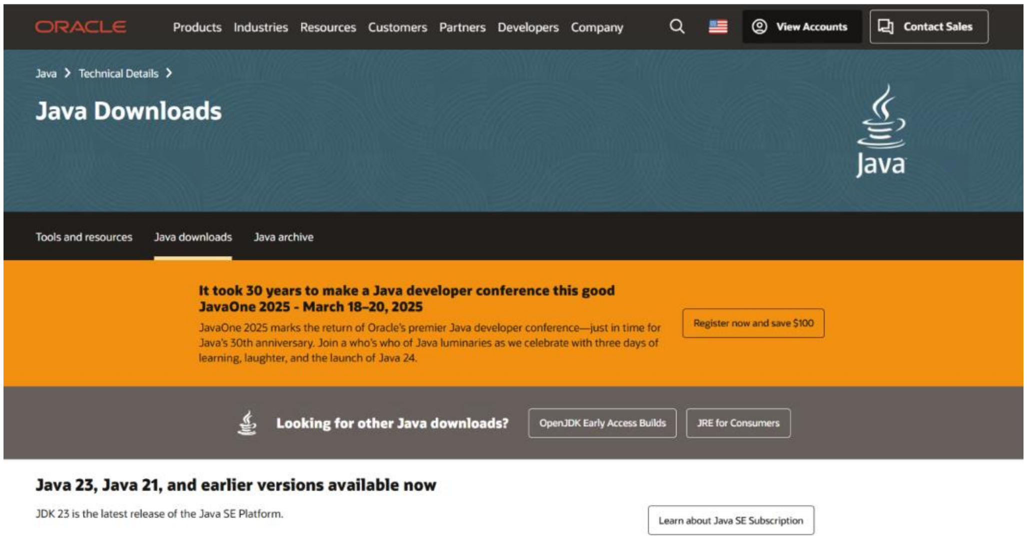
Program-1:

Aim:- Downloading and installing Java software

Procedure:-

Step1:- Open Google chrome and search for java downloads

Step2:- Download java development kit(JDK) from oracle official website.



Java Downloads

It took 30 years to make a Java developer conference this good
JavaOne 2025 - March 18–20, 2025

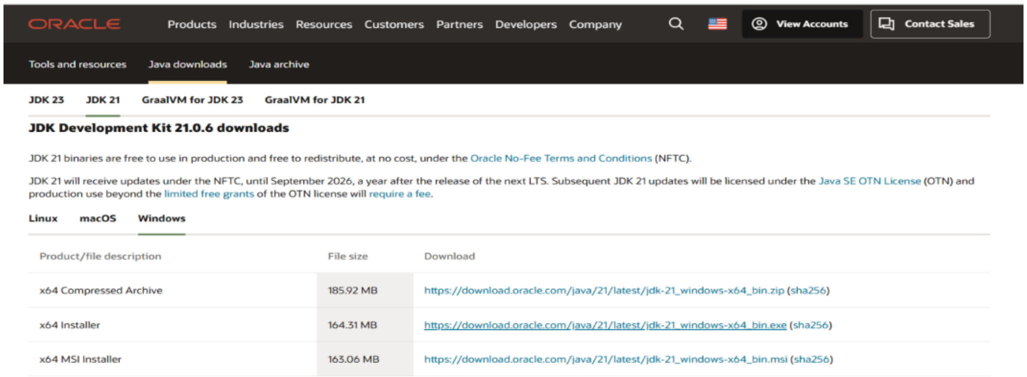
JavaOne 2025 marks the return of Oracle's premier Java developer conference—just in time for Java's 30th anniversary. Join a who's who of Java luminaries as we celebrate with three days of learning, laughter, and the launch of Java 24.

Looking for other Java downloads?

Java 23, Java 21, and earlier versions available now

JDK 23 is the latest release of the Java SE Platform.

Step3:- Download JDK21.0.6 x64 Installer Windows version.



JDK Development Kit 21.0.6 downloads

JDK 21 binaries are free to use in production and free to redistribute, at no cost, under the Oracle No-Fee Terms and Conditions (NFTC).

JDK 21 will receive updates under the NFTC, until September 2026, a year after the release of the next LTS. Subsequent JDK 21 updates will be licensed under the Java SE OTN License (OTN) and production use beyond the limited free grants of the OTN license will require a fee.

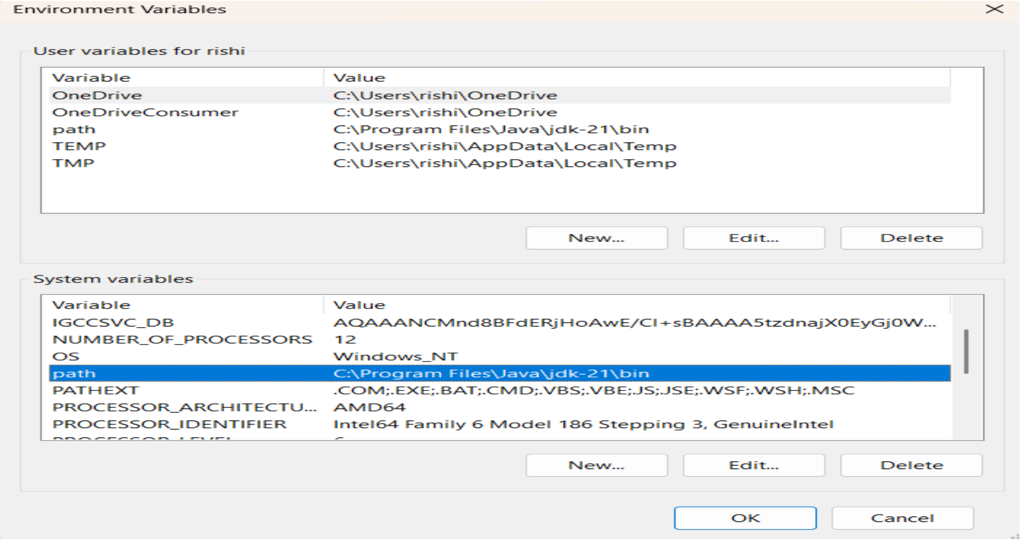
Product/file description	File size	Download
x64 Compressed Archive	185.92 MB	https://download.oracle.com/java/21/latest/jdk-21_windows-x64_bin.zip (sha256)
x64 Installer	164.31 MB	https://download.oracle.com/java/21/latest/jdk-21_windows-x64_bin.exe (sha256)
x64 MSI Installer	163.06 MB	https://download.oracle.com/java/21/latest/jdk-21_windows-x64_bin.msi (sha256)

Step-4:-Click on the link provided to download. After downloading install the software into your device.

Step-5:- After installation we need to set environment set variable path of installation.

Step-6:- To set variable path open files then windows c file then program files then java then jdk-21. Select bin and copy the path.

Step-7:- Now open environmental variables then open system variable. Here we have to paste the copied path in path variable.



User variables for rishi

Variable	Value
OneDrive	C:\Users\rishi\OneDrive
OneDriveConsumer	C:\Users\rishi\OneDrive
path	C:\Program Files\Java\jdk-21\bin
TEMP	C:\Users\rishi\AppData\Local\Temp
TMP	C:\Users\rishi\AppData\Local\Temp

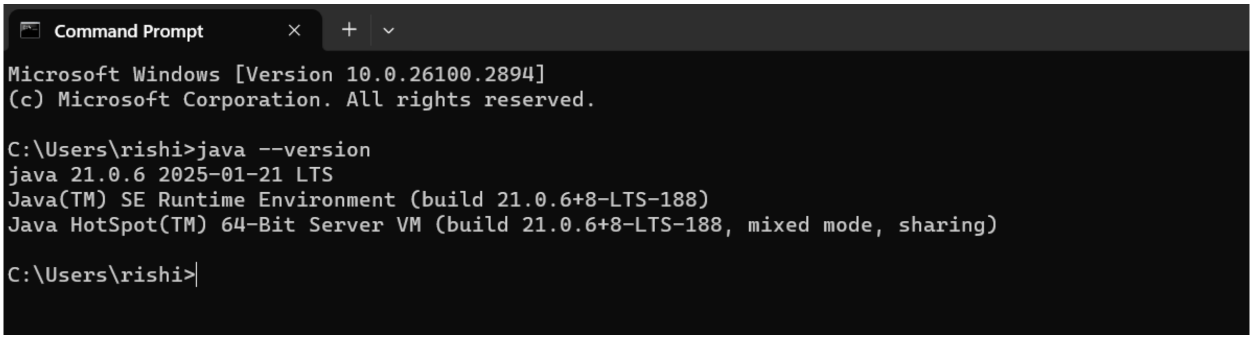
System variables

Variable	Value
IGCCSVC_DB	AQAAANCMnd8BFdERjHoAwE/Cl+sBAAAA5tzdnajX0EyGj0W...
NUMBER_OF_PROCESSORS	12
OS	Windows_NT
path	C:\Program Files\Java\jdk-21\bin
PATHEXT	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC
PROCESSOR_ARCHITECTU...	AMD64
PROCESSOR_IDENTIFIER	Intel64 Family 6 Model 186 Stepping 3, GenuineIntel

Step-8:- Now java software is installed in your device. Now to check the installed version. Open command prompt

Step-9:-in command prompt type java –version then click on enter

Step-10:-The version is displayed

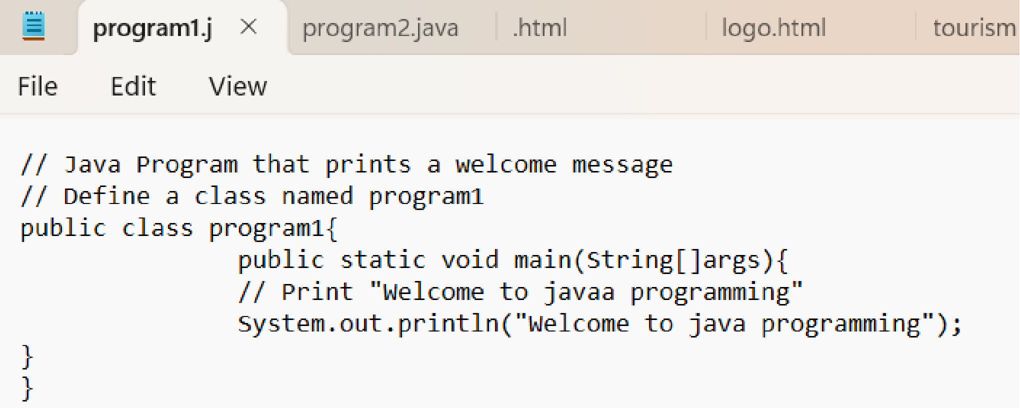


```
Microsoft Windows [Version 10.0.26100.2894]
(c) Microsoft Corporation. All rights reserved.

C:\Users\rishi>java --version
java 21.0.6 2025-01-21 LTS
Java(TM) SE Runtime Environment (build 21.0.6+8-LTS-188)
Java HotSpot(TM) 64-Bit Server VM (build 21.0.6+8-LTS-188, mixed mode, sharing)

C:\Users\rishi>
```

PROGRAM-2:

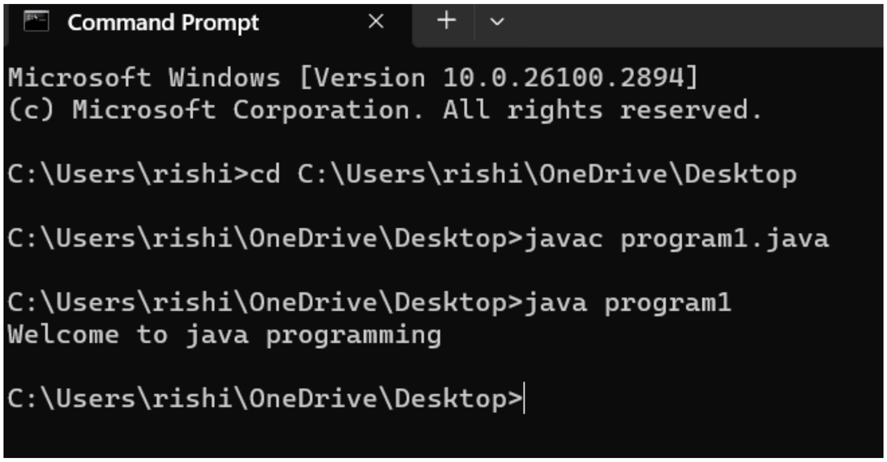
AIM: Write a java program to print the message "Welcome to java programming”

INPUT:



```
// Java Program that prints a welcome message
// Define a class named program1
public class program1{
               public static void main(String[]args){
               // Print "Welcome to javaa programming"
               System.out.println("Welcome to java programming");
}
}
```

OUTPUT:

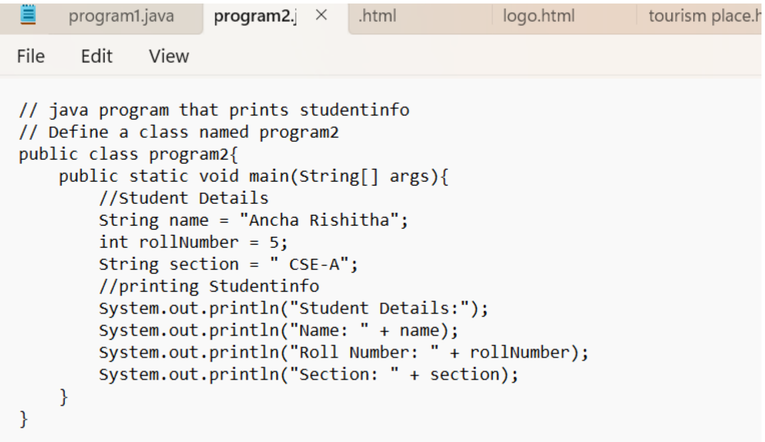


```
Microsoft Windows [Version 10.0.26100.2894]
(c) Microsoft Corporation. All rights reserved.

C:\Users\rishi>cd C:\Users\rishi\OneDrive\Desktop

C:\Users\rishi\OneDrive\Desktop>javac program1.java

C:\Users\rishi\OneDrive\Desktop>java program1
Welcome to java programming

C:\Users\rishi\OneDrive\Desktop>
```

PROGRAM-3:

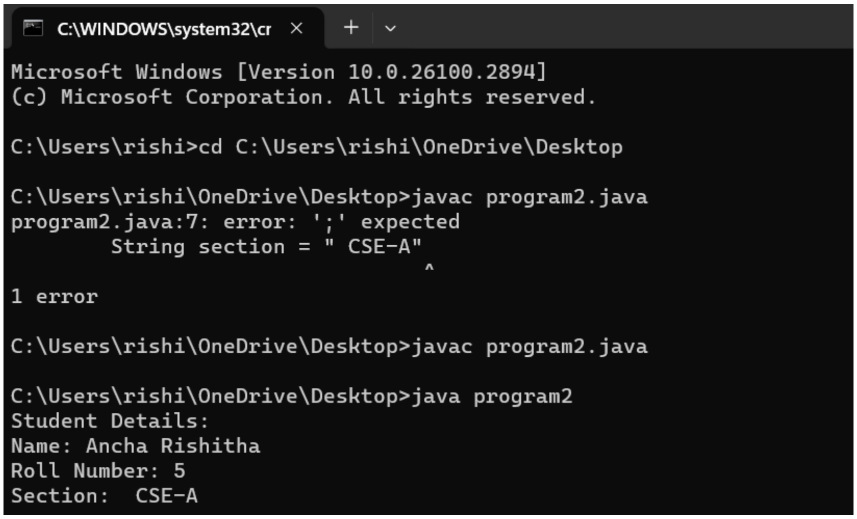
AIM:- Write a java program that prints name, roll number, section of a student

INPUT:



```
// java program that prints studentinfo
// Define a class named program2
public class program2{
    public static void main(String[] args){
        //Student Details
        String name = "Ancha Rishitha";
        int rollNumber = 5;
        String section = " CSE-A";
        //printing Studentinfo
        System.out.println("Student Details:");
        System.out.println("Name: " + name);
        System.out.println("Roll Number: " + rollNumber);
        System.out.println("Section: " + section);
    }
}
```

OUTPUT:



```
Microsoft Windows [Version 10.0.26100.2894]
(c) Microsoft Corporation. All rights reserved.

C:\Users\rishi>cd C:\Users\rishi\OneDrive\Desktop

C:\Users\rishi\OneDrive\Desktop>javac program2.java
program2.java:7: error: ';' expected
        String section = " CSE-A"
                                 ^
1 error

C:\Users\rishi\OneDrive\Desktop>javac program2.java

C:\Users\rishi\OneDrive\Desktop>java program2
Student Details:
Name: Ancha Rishitha
Roll Number: 5
Section:  CSE-A
```

ERRORS:

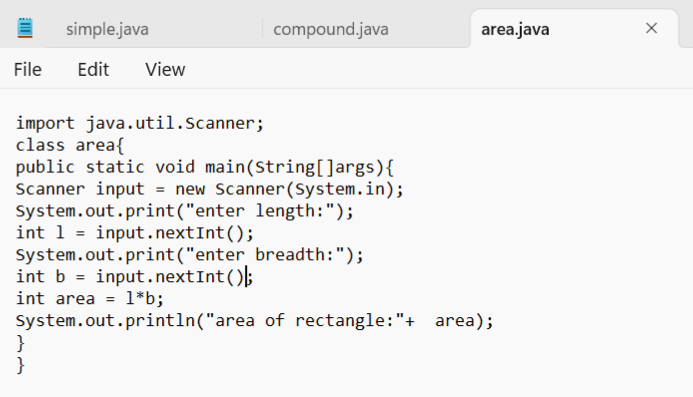
S.NO	ERROR MESSAGE	ERROR RECTIFICATION
1.	‘;’ expected	Placing ‘;’ symbol

Week-2:

PROGRAM-1:

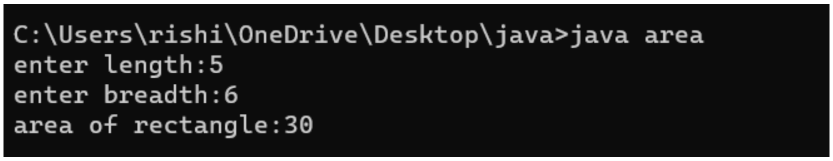
Aim: To write a java program for calculating area of rectangle.

Input Code:



```java
import java.util.Scanner;
class area{
public static void main(String[]args){
Scanner input = new Scanner(System.in);
System.out.print("enter length:");
int l = input.nextInt();
System.out.print("enter breadth:");
int b = input.nextInt();
int area = l*b;
System.out.println("area of rectangle:"+  area);
}
}
```

Output:



```
C:\Users\rishi\OneDrive\Desktop\java>java area
enter length:5
enter breadth:6
area of rectangle:30
```

ERRORS:

S.NO	*ERROR MESSAGE*	*ERROR RECTIFICATION*
1.	Error:";" expected	Inserted ;
2.	Error:"string"small letter case sensitive	"String"

Steps:

1.import java.util.Scanner -step to import library.

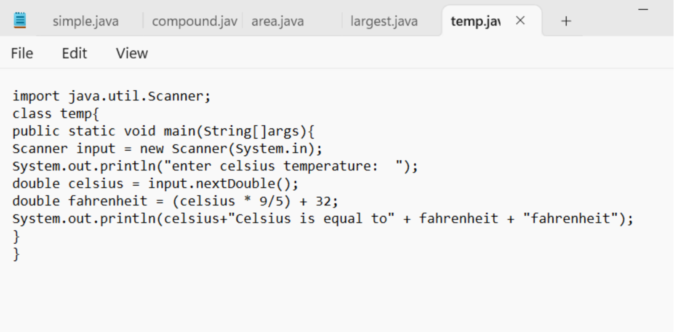
2. used Scanner library to get input from user in run time.

3. "Scanner input=new Scanner(System.in);"-step to use the scanner .

PROGRAM-2:

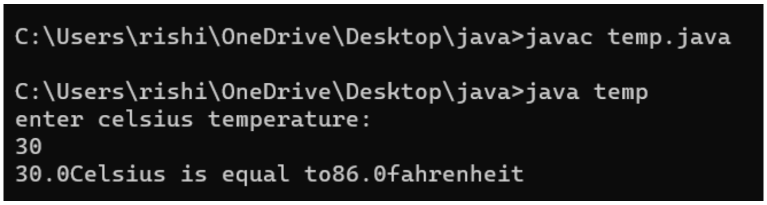
Aim:To write a java program for converting Celsius to Fahrenheit.

Code:



```
import java.util.Scanner;
class temp{
public static void main(String[]args){
Scanner input = new Scanner(System.in);
System.out.println("enter celsius temperature:  ");
double celsius = input.nextDouble();
double fahrenheit = (celsius * 9/5) + 32;
System.out.println(celsius+"Celsius is equal to" + fahrenheit + "fahrenheit");
}
}
```

Output:



```
C:\Users\rishi\OneDrive\Desktop\java>javac temp.java

C:\Users\rishi\OneDrive\Desktop\java>java temp
enter celsius temperature:
30
30.0Celsius is equal to86.0fahrenheit
```

Errors:

S.NO	*ERROR MESSAGE*	*ERROR RECTIFICATION*
1.	Error:Temp “file not found”	“temp”
2.	Error: scanner input “Capital letter case sensitive”	“Scanner input”

Steps:

1.import java.util.Scanner -step to import library.

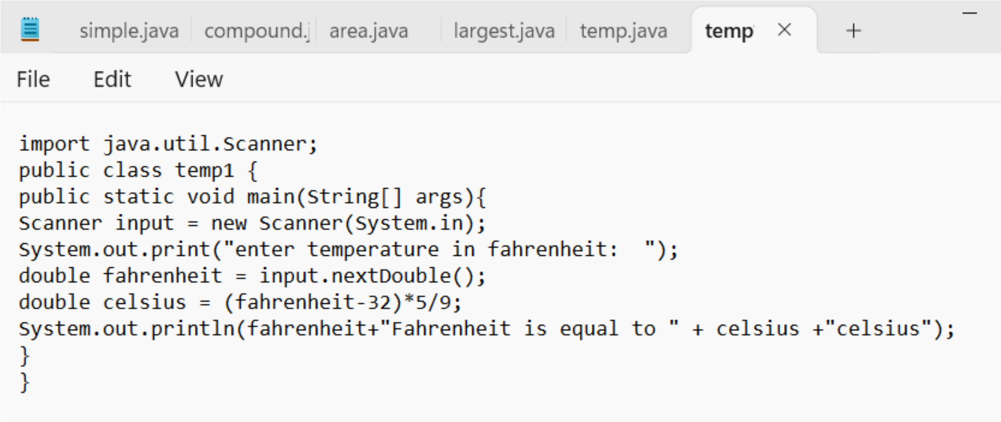
2. used Scanner library to get input from user in run time.

3. “Scanner input=new Scanner(System.in);”-step to use the scanner .

PROGRAM-3:

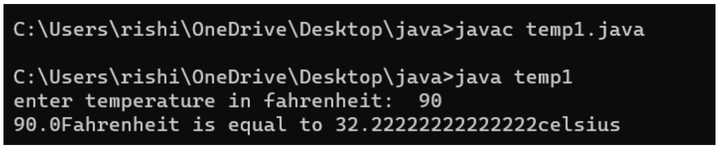
Aim: To write a java program for converting Fahrenheit to Celsius.

Code:



```
import java.util.Scanner;
public class temp1 {
public static void main(String[] args){
Scanner input = new Scanner(System.in);
System.out.print("enter temperature in fahrenheit:  ");
double fahrenheit = input.nextDouble();
double celsius = (fahrenheit-32)*5/9;
System.out.println(fahrenheit+"Fahrenheit is equal to " + celsius +"celsius");
}
}
```

Output:



```
C:\Users\rishi\OneDrive\Desktop\java>javac temp1.java

C:\Users\rishi\OneDrive\Desktop\java>java temp1
enter temperature in fahrenheit:  90
90.0Fahrenheit is equal to 32.22222222222222celsius
```

Errors:

S.no	*Error message*	*Error Rectification*
1.	Error:File not found	“temp1 “
2.	Error:”;” unexpected	Inserted ;
3.	Error: incorrect formula	Rewrite formula

STEPS:

1.import java.util.Scanner -step to import library.

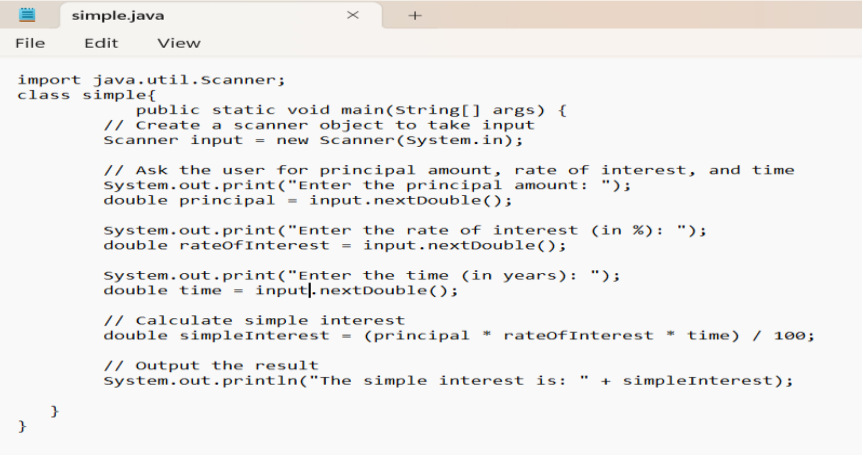
2. used Scanner library to get input from user in run time.

3. “Scanner input=new Scanner(System.in);”-step to use the scanner .

PROGRAM-4:-

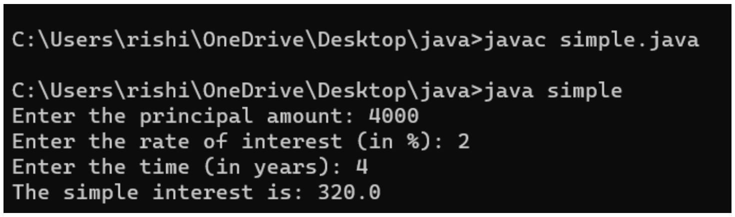
Aim: To write a java program for calculating simple interest.

Code:



```
import java.util.Scanner;
class simple{
        public static void main(String[] args) {
        // Create a scanner object to take input
        Scanner input = new Scanner(System.in);

        // Ask the user for principal amount, rate of interest, and time
        System.out.print("Enter the principal amount: ");
        double principal = input.nextDouble();

        System.out.print("Enter the rate of interest (in %): ");
        double rateOfInterest = input.nextDouble();

        System.out.print("Enter the time (in years): ");
        double time = input.nextDouble();

        // Calculate simple interest
        double simpleInterest = (principal * rateOfInterest * time) / 100;

        // Output the result
        System.out.println("The simple interest is: " + simpleInterest);

    }
}
```

Output:



```
C:\Users\rishi\OneDrive\Desktop\java>javac simple.java

C:\Users\rishi\OneDrive\Desktop\java>java simple
Enter the principal amount: 4000
Enter the rate of interest (in %): 2
Enter the time (in years): 4
The simple interest is: 320.0
```

ERROR:

S.no	*Error Message*	*Error Rectification*
1.	Error: input.nextdouble[] “parenthesis”	“input.nextdouble()”
2.	Error: “Access Modifier for class “	Public class simple

STEPS:

1.import java.util.Scanner -step to import library.

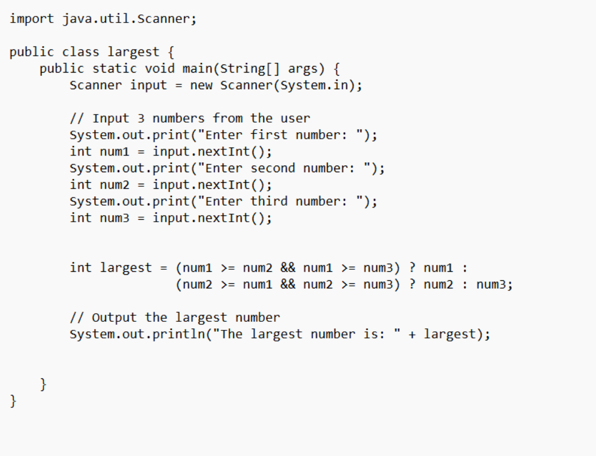
2. used Scanner library to get input from user in run time.

3. “Scanner input=new Scanner(System.in);”-step to use the scanner .

PROGRAM-5:-

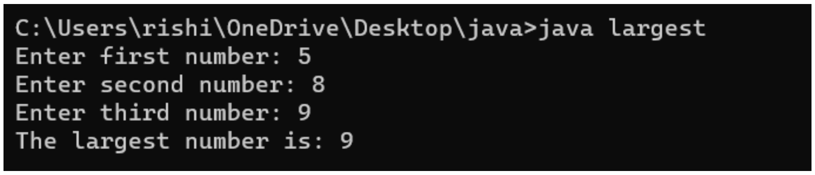
Aim: To write a java program for finding largest among 3 numbers using ternary operator.

Input code:



```
import java.util.Scanner;

public class largest {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);

        // Input 3 numbers from the user
        System.out.print("Enter first number: ");
        int num1 = input.nextInt();
        System.out.print("Enter second number: ");
        int num2 = input.nextInt();
        System.out.print("Enter third number: ");
        int num3 = input.nextInt();


        int largest = (num1 >= num2 && num1 >= num3) ? num1 :
                      (num2 >= num1 && num2 >= num3) ? num2 : num3;

        // Output the largest number
        System.out.println("The largest number is: " + largest);


    }
}
```

Output:



```
C:\Users\rishi\OneDrive\Desktop\java>java largest
Enter first number: 5
Enter second number: 8
Enter third number: 9
The largest number is: 9
```

ERROR:

S.no	*Error Message*	*Error Rectification*
1.	Error: input.nextInt[] “parenthesis”	“input.nextInt()”
2.	Error: “Scanner Object not Closed“	Placed parenthesis ()

STEPS:

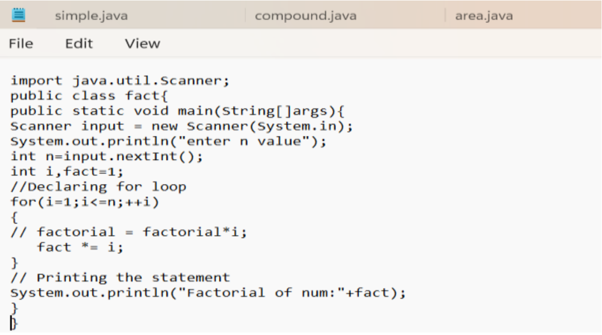
1.import java.util.Scanner -step to import library.

2. used Scanner library to get input from user in run time.

3. “Scanner input=new Scanner(System.in);”-step to use the scanner .

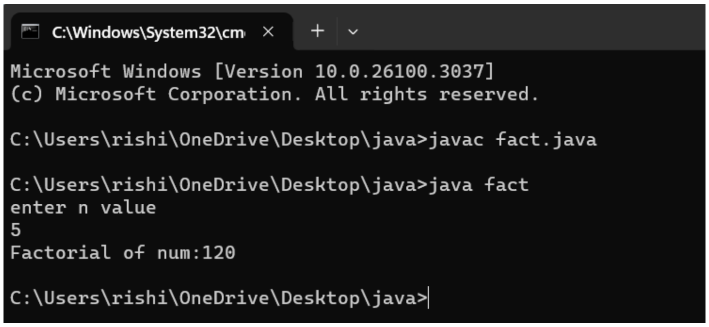
5)Aim: Writing a java program for finding factorial of a number.

Code:



```
import java.util.Scanner;
public class fact{
public static void main(String[]args){
Scanner input = new Scanner(System.in);
System.out.println("enter n value");
int n=input.nextInt();
int i,fact=1;
//Declaring for loop
for(i=1;i<=n;++i)
{
// factorial = factorial*i;
   fact *= i;
}
// Printing the statement
System.out.println("Factorial of num:"+fact);
}
}
```

Output:



```
Microsoft Windows [Version 10.0.26100.3037]
(c) Microsoft Corporation. All rights reserved.

C:\Users\rishi\OneDrive\Desktop\java>javac fact.java

C:\Users\rishi\OneDrive\Desktop\java>java fact
enter n value
5
Factorial of num:120

C:\Users\rishi\OneDrive\Desktop\java>
```

ERROR:

S.no	*Error Message*	*Error Rectification*
1.	Error: ‘file not found’Fact	‘fact’
2.	Error: “Variable i not declared properly“	Declared inside the loop
3.	Error: expected”;”	Inserting ‘;’

STEPS:

1.import java.util.Scanner -step to import library.

2. used Scanner library to get input from user in run time.

3. “Scanner input=new Scanner(System.in);”-step to use the scanner .

Week-3:

PROGRAM-1:

Aim: To write a java program that defines a Car class with attributes, a constructor and methods to perform actions like starting,stopping and servicing a car

Class Diagram:

Car

- Name:String
- Color: String
- Brand: String
- Mileage: double

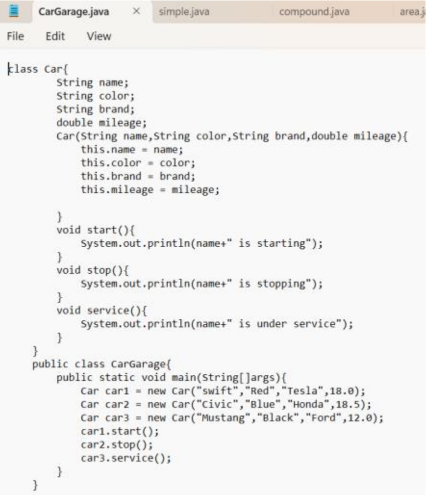
+ Car(name,color,brand.mileage)

+start():void

+stop():void

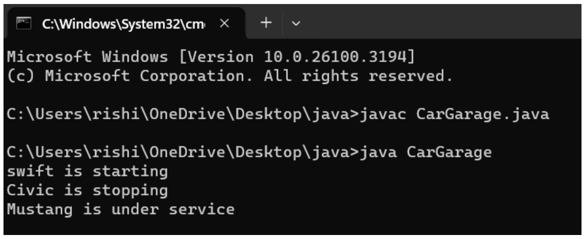
+service():void

Input Code:



```
class Car{
        String name;
        String color;
        String brand;
        double mileage;
        Car(String name,String color,String brand,double mileage){
            this.name = name;
            this.color = color;
            this.brand = brand;
            this.mileage = mileage;

        }
        void start(){
            System.out.println(name+" is starting");
        }
        void stop(){
            System.out.println(name+" is stopping");
        }
        void service(){
            System.out.println(name+" is under service");
        }
    }
    public class CarGarage{
        public static void main(String[]args){
            Car car1 = new Car("swift","Red","Tesla",18.0);
            Car car2 = new Car("Civic","Blue","Honda",18.5);
            Car car3 = new Car("Mustang","Black","Ford",12.0);
            car1.start();
            car2.stop();
            car3.service();
        }
    }
```

Output:



```
Microsoft Windows [Version 10.0.26100.3194]
(c) Microsoft Corporation. All rights reserved.

C:\Users\rishi\OneDrive\Desktop\java>javac CarGarage.java

C:\Users\rishi\OneDrive\Desktop\java>java CarGarage
swift is starting
Civic is stopping
Mustang is under service
```

ERROR:

S.no	*Error Message*	*Error Rectification*
1.	Error: expected () paranthesis	Start()

2.	Error: expected '}'	Placed '}'

STEPS:

1.Defines a car class with attributed and initializes them using a constructor.

2.Implements start(),stop() and service() methods to print car actions.

3. Creates Car objects and calls their methods to display messages.

PROGRAM-2:

Aim: To write a java program that allows user to create a bank account,deposit money and withdraw money.

Class Diagram:

Bank Account

- balance: double

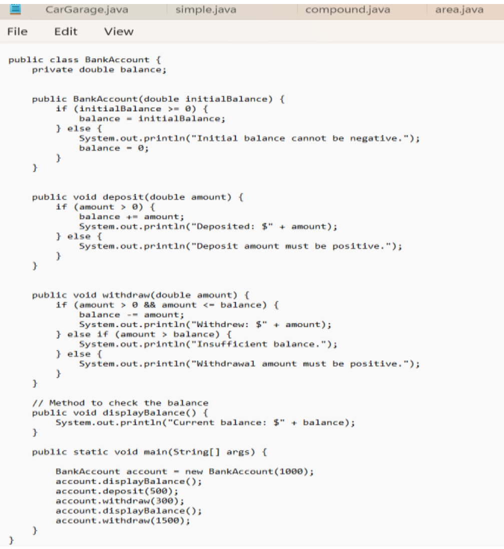
+ BankAccount(initialBalance:double)

+ deposit(amount:double):void

+withdraw(amount:double):void

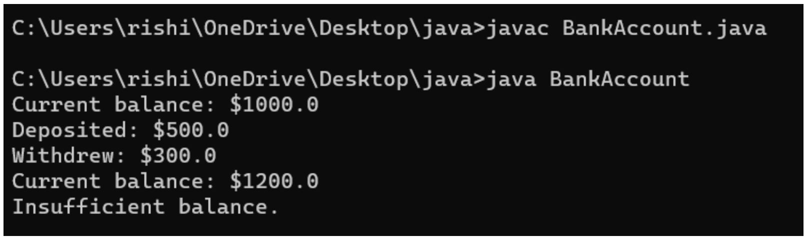
+ displayBalance():void

Input Code:



```
public class BankAccount {
    private double balance;


    public BankAccount(double initialBalance) {
        if (initialBalance >= 0) {
            balance = initialBalance;
        } else {
            System.out.println("Initial balance cannot be negative.");
            balance = 0;
        }
    }


    public void deposit(double amount) {
        if (amount > 0) {
            balance += amount;
            System.out.println("Deposited: $" + amount);
        } else {
            System.out.println("Deposit amount must be positive.");
        }
    }


    public void withdraw(double amount) {
        if (amount > 0 && amount <= balance) {
            balance -= amount;
            System.out.println("Withdrew: $" + amount);
        } else if (amount > balance) {
            System.out.println("Insufficient balance.");
        } else {
            System.out.println("Withdrawal amount must be positive.");
        }
    }

    // Method to check the balance
    public void displayBalance() {
        System.out.println("Current balance: $" + balance);
    }

    public static void main(String[] args) {

        BankAccount account = new BankAccount(1000);
        account.displayBalance();
        account.deposit(500);
        account.withdraw(300);
        account.displayBalance();
        account.withdraw(1500);
    }
}
```

Output:



```
C:\Users\rishi\OneDrive\Desktop\java>javac BankAccount.java

C:\Users\rishi\OneDrive\Desktop\java>java BankAccount
Current balance: $1000.0
Deposited: $500.0
Withdrew: $300.0
Current balance: $1200.0
Insufficient balance.
```

ERROR:

S.no	*Error Message*	*Error Rectification*
1.	Error: incorrect method call	Inserted ()
2.	Error: “;” expected	Placed ‘;’

STEPS:

1. Define the BankAccount class with a constructor to initialize the balance.

2. Create deposit(), withdraw(), and displayBalance() methods for account operations.

3. Create a BankAccount object, perform transactions, and display the balance.

WEEK-4:

PROGRAM1:

AIM:Write a java program that prints the details of the book

CLASS DIAGRAM:

Book

Attributes

-title: String

- author:String

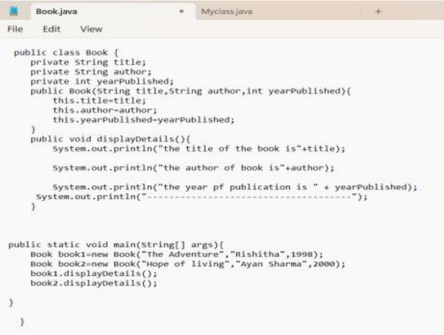
-yearPublished:int

Methods

+ Book(title,author,yearPublished)

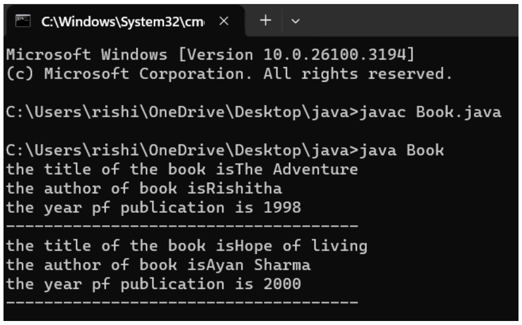
+displayDetails():void

INPUT CODE:



```
 public class Book {
    private String title;
    private String author;
    private int yearPublished;
    public Book(String title,String author,int yearPublished){
        this.title=title;
        this.author=author;
        this.yearPublished=yearPublished;
    }
    public void displayDetails(){
        System.out.println("the title of the book is"+title);

        System.out.println("the author of book is"+author);

        System.out.println("the year pf publication is " + yearPublished);
     System.out.println("-------------------------------------");
    }


public static void main(String[] args){
    Book book1=new Book("The Adventure","Rishitha",1998);
    Book book2=new Book("Hope of living","Ayan Sharma",2000);
    book1.displayDetails();
    book2.displayDetails();

}

  }
```

OUTPUT:



```
Microsoft Windows [Version 10.0.26100.3194]
(c) Microsoft Corporation. All rights reserved.

C:\Users\rishi\OneDrive\Desktop\java>javac Book.java

C:\Users\rishi\OneDrive\Desktop\java>java Book
the title of the book isThe Adventure
the author of book isRishitha
the year pf publication is 1998
-------------------------------------
the title of the book isHope of living
the author of book isAyan Sharma
the year pf publication is 2000
-------------------------------------
```

ERROR:

S.no	*Error Message*	*Error Rectification*
1.	Error: incorrect method call	Inserted ()
2.	Error: “;” expected	Placed ‘;’

Steps:

1. Create a Book class with attributes (`title`, `author`, `yearPublished`) and a constructor to initialize them.

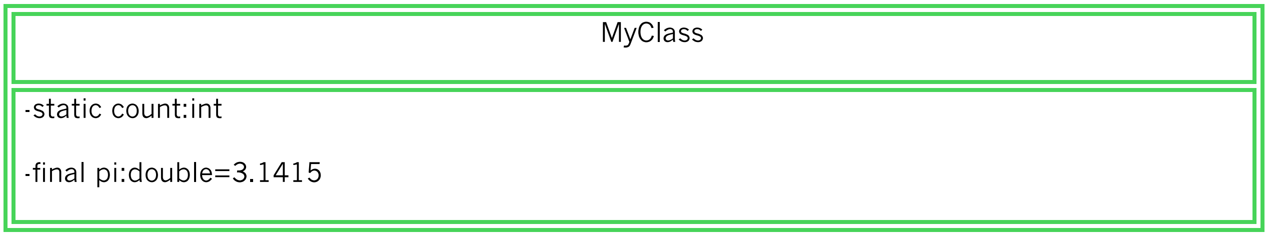
2. Write method (`displayDetails()`)

3. Instantiate Book objects in `main()`, then call the display methods to output their details.

PROGRAM2:

AIM:Write a java program that increments the count variable

CLASS DIAGRAM:



MyClass

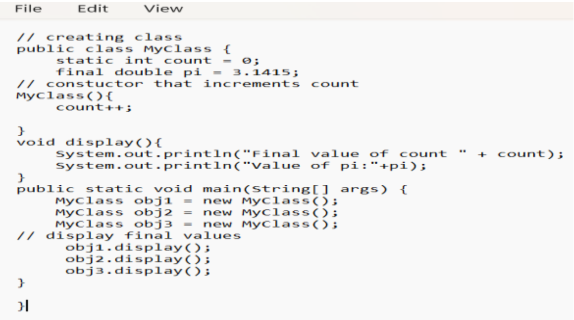
-static count:int

-final pi:double=3.1415

+MyClass()

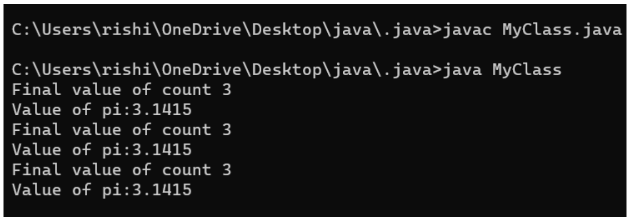
+display():void

INPUT CODE:



```
// creating class
public class MyClass {
     static int count = 0;
     final double pi = 3.1415;
// constuctor that increments count
MyClass(){
     count++;

}
void display(){
     System.out.println("Final value of count " + count);
     System.out.println("Value of pi:"+pi);
}
public static void main(String[] args) {
     MyClass obj1 = new MyClass();
     MyClass obj2 = new MyClass();
     MyClass obj3 = new MyClass();
// display final values
      obj1.display();
      obj2.display();
      obj3.display();
}

}
```

OUTPUT:



```
C:\Users\rishi\OneDrive\Desktop\java\.java>javac MyClass.java

C:\Users\rishi\OneDrive\Desktop\java\.java>java MyClass
Final value of count 3
Value of pi:3.1415
Final value of count 3
Value of pi:3.1415
Final value of count 3
Value of pi:3.1415
```

ERROR:

S.no	*Error Message*	*Error Rectification*
1.	Error: unresolved compilation problem	Removed pi++
2.	Error: incorrect object name pnj1	Corrected spelling obj1

STEPS:

1.Create a `MyClass` with a static variable `count` and a final constant `pi`.

2.The constructor increments `count`, and `display()` prints `count` and `pi`.

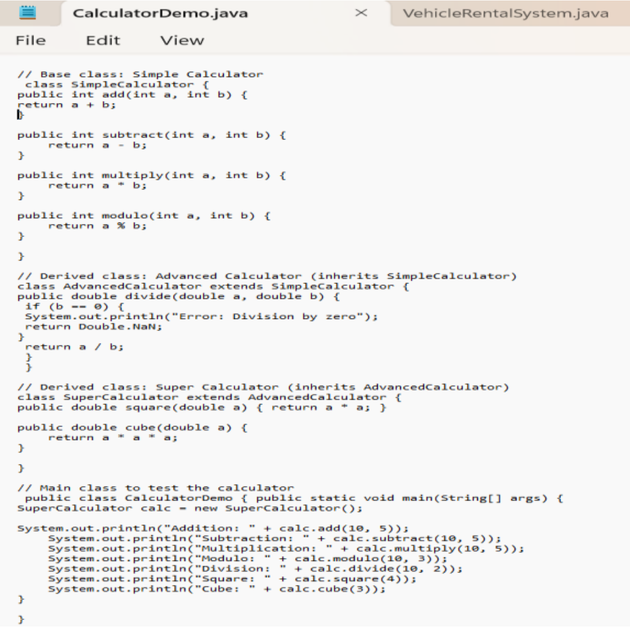
3.Instantiate `MyClass` objects in `main()` and call `display()` to observe `count` changes.

WEEK-5:

Program-1:

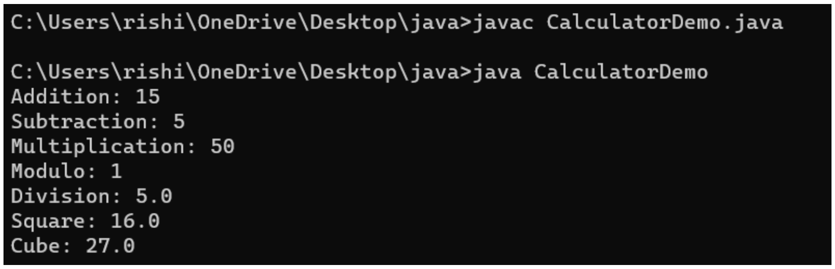
Aim:Write a java program to create a calculator using multilevel inheritance

Input code:



```java
// Base class: Simple Calculator
 class SimpleCalculator {
public int add(int a, int b) {
return a + b;
}

public int subtract(int a, int b) {
    return a - b;
}

public int multiply(int a, int b) {
    return a * b;
}

public int modulo(int a, int b) {
    return a % b;
}

}

// Derived class: Advanced Calculator (inherits SimpleCalculator)
class AdvancedCalculator extends SimpleCalculator {
public double divide(double a, double b) {
 if (b == 0) {
 System.out.println("Error: Division by zero");
 return Double.NaN;
}
 return a / b;
 }
 }

// Derived class: Super Calculator (inherits AdvancedCalculator)
class SuperCalculator extends AdvancedCalculator {
public double square(double a) { return a * a; }

public double cube(double a) {
    return a * a * a;
}

}

// Main class to test the calculator
 public class CalculatorDemo { public static void main(String[] args) {
SuperCalculator calc = new SuperCalculator();

System.out.println("Addition: " + calc.add(10, 5));
    System.out.println("Subtraction: " + calc.subtract(10, 5));
    System.out.println("Multiplication: " + calc.multiply(10, 5));
    System.out.println("Modulo: " + calc.modulo(10, 3));
    System.out.println("Division: " + calc.divide(10, 2));
    System.out.println("Square: " + calc.square(4));
    System.out.println("Cube: " + calc.cube(3));
}

}
```

Output code:



```
C:\Users\rishi\OneDrive\Desktop\java>javac CalculatorDemo.java

C:\Users\rishi\OneDrive\Desktop\java>java CalculatorDemo
Addition: 15
Subtraction: 5
Multiplication: 50
Modulo: 1
Division: 5.0
Square: 16.0
Cube: 27.0
```

Error:

S.NO	ERROR MESSAGE	ERROR RECTIFICATION
1.	‘;’ expected	Placing ‘;’ symbol

Steps:

1.Create the SimpleCalculator Class

2.Create subclasses(AdvancedCalculator,SuperCalculator)

3.Implement the main method

Program-2:

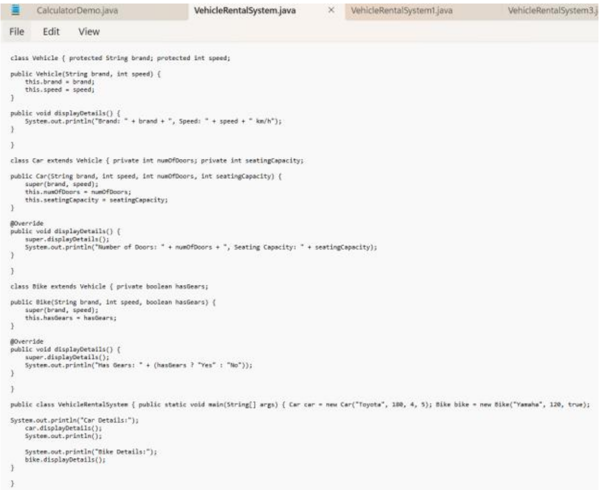
Aim: Write a java program for vehicle renter company to store details about each vehicle such as brand and speed.

Class Diagram:

```
                         |     Vehicle       |

               | - brand: String   |

              | - speed: int       |

              | + displayDetails(): void |

                                        ▲

                                        |

 -----------------------------------------------------------------------------------------------------

  |                                |                                         |

+-----------+                  +-----------+                            +-----------+

|    Car     |                 |   Bike     |                            |   Truck    |

+-----------+                   +-----------+                             +-----------+

| - numOfDoors: int  |

- hasGears: boolean |

| - capacityInTons: double |

| - seatingCapacity: int |

+-----------+                       +-----------+                  +-----------+

| + displayDetails(): void  |

+ displayDetails(): void  |

 + displayDetails(): void  |
```

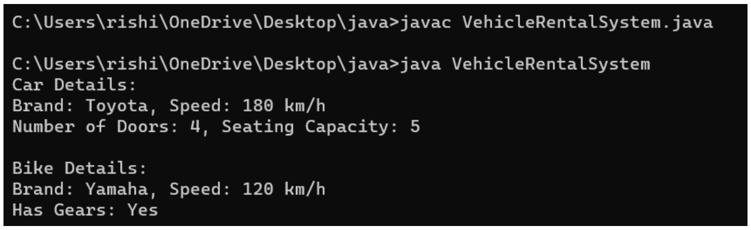
+-----------+   +-----------+   +-----------+

Input code:



```java
class Vehicle { protected String brand; protected int speed;

public Vehicle(String brand, int speed) {
    this.brand = brand;
    this.speed = speed;
}

public void displayDetails() {
    System.out.println("Brand: " + brand + ", Speed: " + speed + " km/h");
}

}

class Car extends Vehicle { private int numOfDoors; private int seatingCapacity;

public Car(String brand, int speed, int numOfDoors, int seatingCapacity) {
    super(brand, speed);
    this.numOfDoors = numOfDoors;
    this.seatingCapacity = seatingCapacity;
}

@Override
public void displayDetails() {
    super.displayDetails();
    System.out.println("Number of Doors: " + numOfDoors + ", Seating Capacity: " + seatingCapacity);
}

}

class Bike extends Vehicle { private boolean hasGears;

public Bike(String brand, int speed, boolean hasGears) {
    super(brand, speed);
    this.hasGears = hasGears;
}

@Override
public void displayDetails() {
    super.displayDetails();
    System.out.println("Has Gears: " + (hasGears ? "Yes" : "No"));
}

}

public class VehicleRentalSystem { public static void main(String[] args) { Car car = new Car("Toyota", 180, 4, 5); Bike bike = new Bike("Yamaha", 120, true);

System.out.println("Car Details:");
    car.displayDetails();
    System.out.println();

    System.out.println("Bike Details:");
    bike.displayDetails();
}

}
```

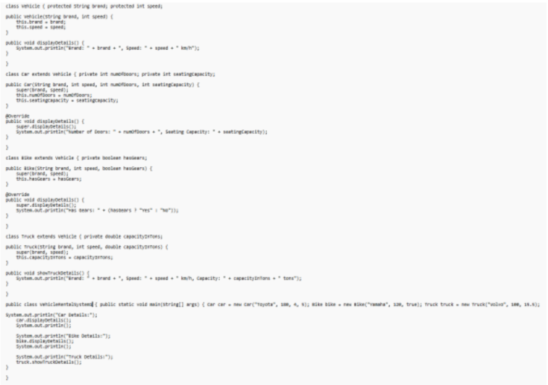
Output code:



```
C:\Users\rishi\OneDrive\Desktop\java>javac VehicleRentalSystem.java

C:\Users\rishi\OneDrive\Desktop\java>java VehicleRentalSystem
Car Details:
Brand: Toyota, Speed: 180 km/h
Number of Doors: 4, Seating Capacity: 5

Bike Details:
Brand: Yamaha, Speed: 120 km/h
Has Gears: Yes
```

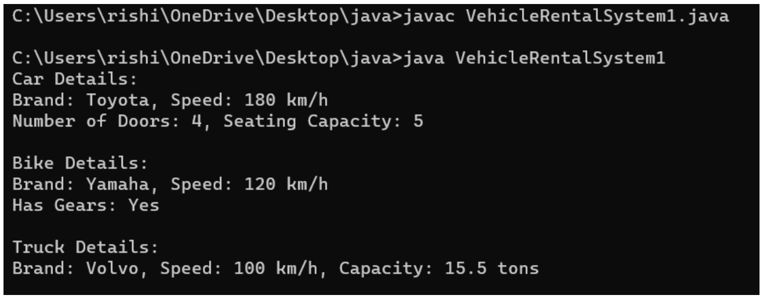
2b)

Input code:



```java
class Vehicle { protected String brand; protected int speed;

public Vehicle(String brand, int speed) {
    this.brand = brand;
    this.speed = speed;
}

public void displayDetails() {
    System.out.println("Brand: " + brand + ", Speed: " + speed + " km/h");
}

}

class Car extends Vehicle { private int numOfDoors; private int seatingCapacity;

public Car(String brand, int speed, int numOfDoors, int seatingCapacity) {
    super(brand, speed);
    this.numOfDoors = numOfDoors;
    this.seatingCapacity = seatingCapacity;
}

@Override
public void displayDetails() {
    super.displayDetails();
    System.out.println("Number of Doors: " + numOfDoors + ", Seating Capacity: " + seatingCapacity);
}

}

class Bike extends Vehicle { private boolean hasGears;

public Bike(String brand, int speed, boolean hasGears) {
    super(brand, speed);
    this.hasGears = hasGears;
}

@Override
public void displayDetails() {
    super.displayDetails();
    System.out.println("Has Gears: " + (hasGears ? "Yes" : "No"));
}

}

class Truck extends Vehicle { private double capacityInTons;

public Truck(String brand, int speed, double capacityInTons) {
    super(brand, speed);
    this.capacityInTons = capacityInTons;
}

public void showTruckDetails() {
    System.out.println("Brand: " + brand + ", Speed: " + speed + " km/h, Capacity: " + capacityInTons + " tons");
}

}

public class VehicleRentalSystem3 { public static void main(String[] args) { Car car = new Car("Toyota", 180, 4, 5); Bike bike = new Bike("Yamaha", 120, true); Truck truck = new Truck("Volvo", 100, 15.5);

System.out.println("Car Details:");
    car.displayDetails();
    System.out.println();

    System.out.println("Bike Details:");
    bike.displayDetails();
    System.out.println();

    System.out.println("Truck Details:");
    truck.showTruckDetails();
}

}
```

Output code:

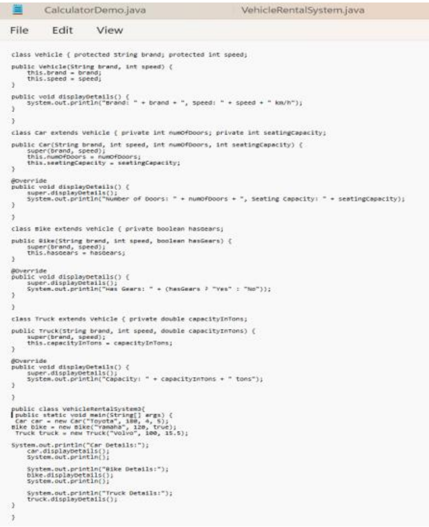


```
C:\Users\rishi\OneDrive\Desktop\java>javac VehicleRentalSystem1.java

C:\Users\rishi\OneDrive\Desktop\java>java VehicleRentalSystem1
Car Details:
Brand: Toyota, Speed: 180 km/h
Number of Doors: 4, Seating Capacity: 5

Bike Details:
Brand: Yamaha, Speed: 120 km/h
Has Gears: Yes

Truck Details:
Brand: Volvo, Speed: 100 km/h, Capacity: 15.5 tons
```

Program-3:

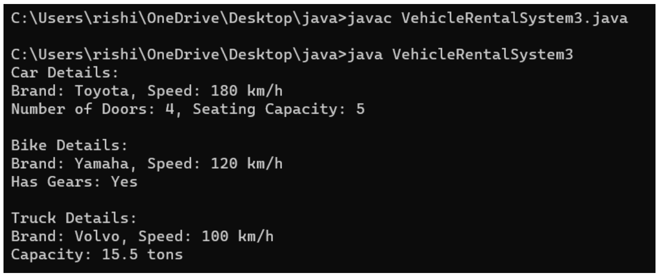
Aim:Implement the truck class and update the main method to create a truck object and also create an object for car and bike subclasses and display its details.

Input Code:



```
class Vehicle { protected String brand; protected int speed;

public Vehicle(String brand, int speed) {
    this.brand = brand;
    this.speed = speed;
}

public void displayDetails() {
    System.out.println("Brand: " + brand + ", Speed: " + speed + " km/h");
}

}

class Car extends Vehicle { private int numOfDoors; private int seatingCapacity;

public Car(String brand, int speed, int numOfDoors, int seatingCapacity) {
    super(brand, speed);
    this.numOfDoors = numOfDoors;
    this.seatingCapacity = seatingCapacity;
}

@Override
public void displayDetails() {
    super.displayDetails();
    System.out.println("Number of Doors: " + numOfDoors + ", Seating Capacity: " + seatingCapacity);
}

}

class Bike extends Vehicle { private boolean hasGears;

public Bike(String brand, int speed, boolean hasGears) {
    super(brand, speed);
    this.hasGears = hasGears;
}

@Override
public void displayDetails() {
    super.displayDetails();
    System.out.println("Has Gears: " + (hasGears ? "Yes" : "No"));
}

}

class Truck extends Vehicle { private double capacityInTons;

public Truck(String brand, int speed, double capacityInTons) {
    super(brand, speed);
    this.capacityInTons = capacityInTons;
}

@Override
public void displayDetails() {
    super.displayDetails();
    System.out.println("Capacity: " + capacityInTons + " tons");
}

}

public class VehicleRentalSystem3{
 public static void main(String[] args) {
 Car car = new Car("Toyota", 180, 4, 5);
Bike bike = new Bike("Yamaha", 120, true);
 Truck truck = new Truck("Volvo", 100, 15.5);

System.out.println("Car Details:");
    car.displayDetails();
    System.out.println();

    System.out.println("Bike Details:");
    bike.displayDetails();
    System.out.println();

    System.out.println("Truck Details:");
    truck.displayDetails();
}

}
```

Ouput code:



```
C:\Users\rishi\OneDrive\Desktop\java>javac VehicleRentalSystem3.java

C:\Users\rishi\OneDrive\Desktop\java>java VehicleRentalSystem3
Car Details:
Brand: Toyota, Speed: 180 km/h
Number of Doors: 4, Seating Capacity: 5

Bike Details:
Brand: Yamaha, Speed: 120 km/h
Has Gears: Yes

Truck Details:
Brand: Volvo, Speed: 100 km/h
Capacity: 15.5 tons
```

Errors:

S.no	*Error Message*	*Error Rectification*
1.	Error: ‘file not found’vehivleRentalSystem	‘VehicleRentalSystem'

Steps:

1.Create the VehicleBase Class

2.Create subclasses(Car,Bike and Truck)

3.Implement the main method

WEEK-6:

PROGRAM-1:

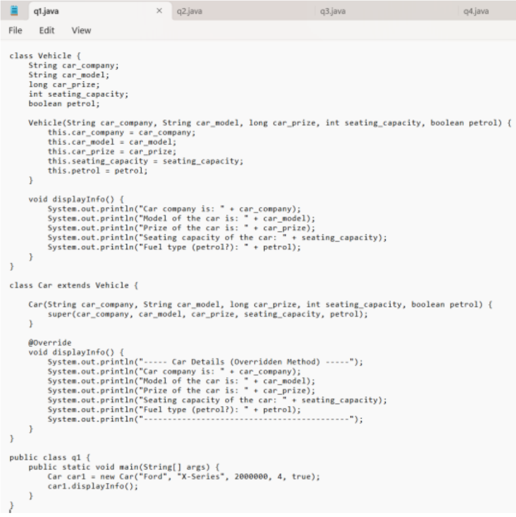
AIM:Write a java program to create a vehicle class with a method displayinfo() override this method in the car subclass to provide specific instancrs,about a car.

CLASS DIAGRAM:

```
+-----------------+

|    Vehicle      |

+-----------------+

| - car_company: String

| - car_model: String

| - car_prize: long

| - seating_capacity: int

| - petrol: boolean

+-----------------+

| + Vehicle(...)   |

| + displayInfo()  |

+-----------------+

      ▲
```

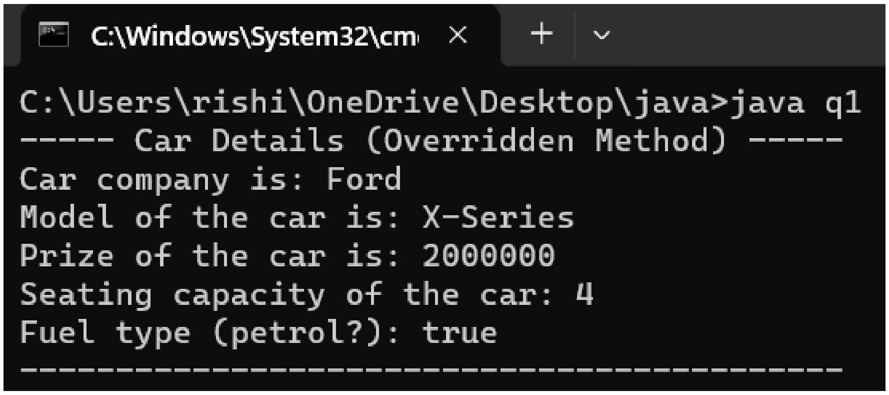
```
          |

          |

      +----------------+

      |     Car        |

      +----------------+

      | + Car(...)       |

      | + displayInfo()  |

      +----------------+
```

INPUT CODE:



```java
class Vehicle {
    String car_company;
    String car_model;
    long car_prize;
    int seating_capacity;
    boolean petrol;

    Vehicle(String car_company, String car_model, long car_prize, int seating_capacity, boolean petrol) {
        this.car_company = car_company;
        this.car_model = car_model;
        this.car_prize = car_prize;
        this.seating_capacity = seating_capacity;
        this.petrol = petrol;
    }

    void displayInfo() {
        System.out.println("Car company is: " + car_company);
        System.out.println("Model of the car is: " + car_model);
        System.out.println("Prize of the car is: " + car_prize);
        System.out.println("Seating capacity of the car: " + seating_capacity);
        System.out.println("Fuel type (petrol?): " + petrol);
    }
}

class Car extends Vehicle {

    Car(String car_company, String car_model, long car_prize, int seating_capacity, boolean petrol) {
        super(car_company, car_model, car_prize, seating_capacity, petrol);
    }

    @Override
    void displayInfo() {
        System.out.println("----- Car Details (Overridden Method) -----");
        System.out.println("Car company is: " + car_company);
        System.out.println("Model of the car is: " + car_model);
        System.out.println("Prize of the car is: " + car_prize);
        System.out.println("Seating capacity of the car: " + seating_capacity);
        System.out.println("Fuel type (petrol?): " + petrol);
        System.out.println("-------------------------------------------");
    }
}

public class q1 {
    public static void main(String[] args) {
        Car car1 = new Car("Ford", "X-Series", 2000000, 4, true);
        car1.displayInfo();
    }
}
```

OUTPUT CODE:



```
C:\Users\rishi\OneDrive\Desktop\java>java q1
----- Car Details (Overridden Method) -----
Car company is: Ford
Model of the car is: X-Series
Prize of the car is: 2000000
Seating capacity of the car: 4
Fuel type (petrol?): true
-------------------------------------------
```

ERRORS:

S.NO	ERROR MESSAGE	RECTIFICATION
1.	Method of the form(string,string,long,int,boolean)	Recorrected method car
2.	Incoorect method call public void vehicle()	Recorrected as vehicle()

IMPORTANT POINTS:

1. **Inheritance and Method Overriding**
   The class `Car` inherits from `Vehicle` and overrides the `displayInfo()` method to provide specific output formatting.
2. **Use of Constructor Chaining with `super()`**
   The `Car` constructor uses `super(...)` to call the constructor of the `Vehicle` class, ensuring proper initialization.
3. **Encapsulation (Basic Level)**
   Though variables are not marked as `private`, the structure shows basic encapsulation principles by initializing fields via constructor and displaying them using methods.

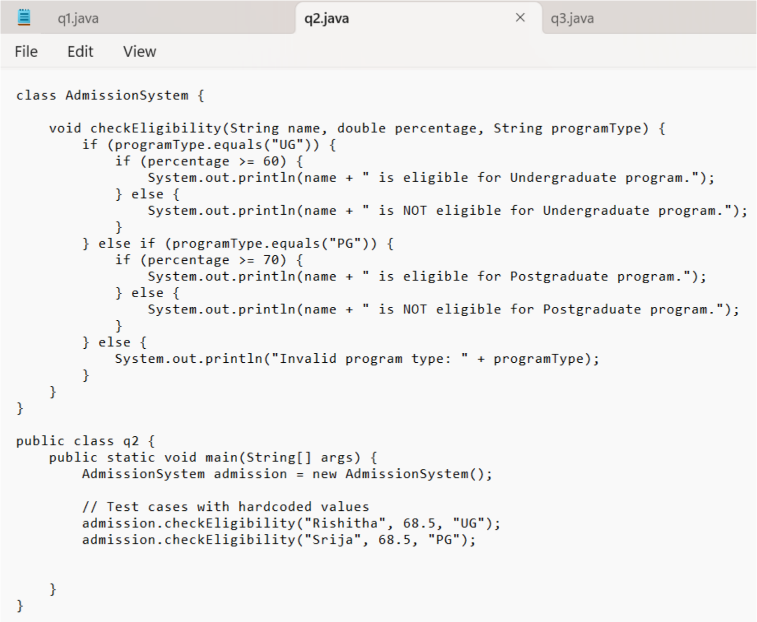
PROGRAM-2:

CLASS DIAGRAM:

```
+--------------------------+

|     AdmissionSystem      |

+--------------------------+

| + checkEligibility(name:  |

|   String, percentage: double, |

|   programType: String): void |

+--------------------------+



+--------------------------+

|          Q2          |

+--------------------------+
| + main(String[]): void     |
+--------------------------+
| Uses AdmissionSystem       |
+--------------------------+
```

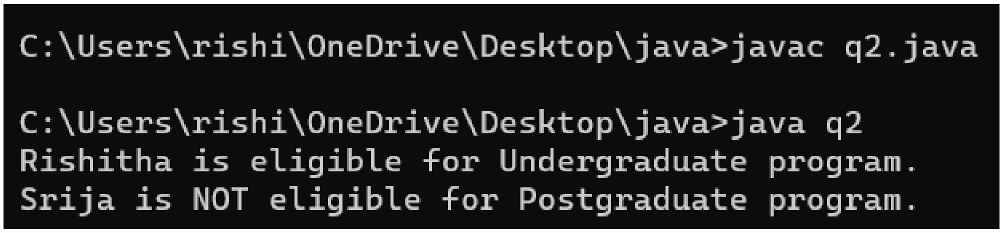
AIM: An automated admission system that verifies student eligibility for UG and PG with different criteria. 1.UG requires minimum of 60% 2.PG requires minimum of 70%

INPUT CODE:



```java
class AdmissionSystem {

    void checkEligibility(String name, double percentage, String programType) {
        if (programType.equals("UG")) {
            if (percentage >= 60) {
                System.out.println(name + " is eligible for Undergraduate program.");
            } else {
                System.out.println(name + " is NOT eligible for Undergraduate program.");
            }
        } else if (programType.equals("PG")) {
            if (percentage >= 70) {
                System.out.println(name + " is eligible for Postgraduate program.");
            } else {
                System.out.println(name + " is NOT eligible for Postgraduate program.");
            }
        } else {
            System.out.println("Invalid program type: " + programType);
        }
    }
}

public class q2 {
    public static void main(String[] args) {
        AdmissionSystem admission = new AdmissionSystem();

        // Test cases with hardcoded values
        admission.checkEligibility("Rishitha", 68.5, "UG");
        admission.checkEligibility("Srija", 68.5, "PG");


    }
}
```

OUTPUT CODE:



```
C:\Users\rishi\OneDrive\Desktop\java>javac q2.java

C:\Users\rishi\OneDrive\Desktop\java>java q2
Rishitha is eligible for Undergraduate program.
Srija is NOT eligible for Postgraduate program.
```

ERRORS :

S.NO	ERROR MESSAGE	RECTIFICATION
1.	Else without if	Recorrected if statement
2.	Hardcoded test values(Not scalable)	Accept user input scanner,or use array/loop structures for multiple students

IMPORTANT POINTS:

- **Program Type is Validated**
  The code includes validation for `UG`, `PG`, and prints a message for invalid program types.
- **Dynamic Message Generation**
  Eligibility messages are constructed using `System.out.println()` dynamically based on student name and condition.
- **Single Responsibility Principle Followed**
  The class is focused only on *admission eligibility checking*, keeping it clean and aligned with object-oriented design.

PROGRAM-3:

AIM: Create a calculator class with overloaded methods to perform additions

1.add two integers.

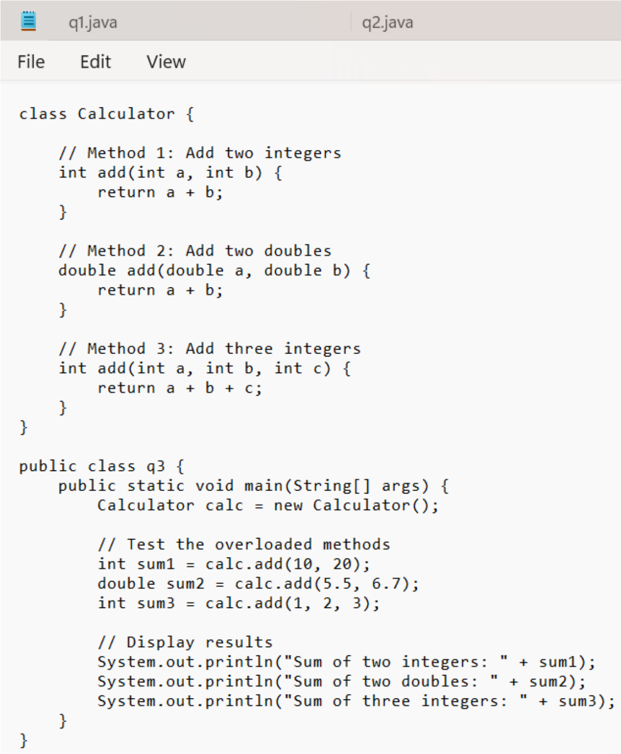
2.add two double values

3.add three integers.

CLASS DIAGRAM:

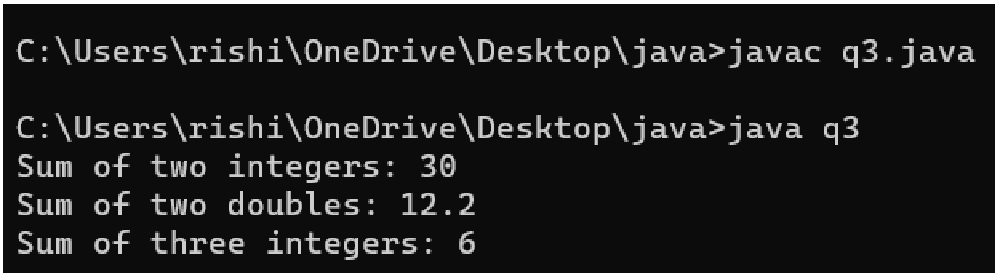
```
+--------------------+
|     Calculator       |
+--------------------+
| + add(int, int): int          |
| + add(double, double): double |
| + add(int, int, int): int     |
+--------------------+
+--------------------+
|        Q3          |
+--------------------+
| + main(String[]): void |
+--------------------+
| Uses Calculator      |
+--------------------+
```

INPUT CODE:



```
class Calculator {

    // Method 1: Add two integers
    int add(int a, int b) {
        return a + b;
    }

    // Method 2: Add two doubles
    double add(double a, double b) {
        return a + b;
    }

    // Method 3: Add three integers
    int add(int a, int b, int c) {
        return a + b + c;
    }
}

public class q3 {
    public static void main(String[] args) {
        Calculator calc = new Calculator();

        // Test the overloaded methods
        int sum1 = calc.add(10, 20);
        double sum2 = calc.add(5.5, 6.7);
        int sum3 = calc.add(1, 2, 3);

        // Display results
        System.out.println("Sum of two integers: " + sum1);
        System.out.println("Sum of two doubles: " + sum2);
        System.out.println("Sum of three integers: " + sum3);
    }
}
```

OUTPUT CODE:



```
C:\Users\rishi\OneDrive\Desktop\java>javac q3.java

C:\Users\rishi\OneDrive\Desktop\java>java q3
Sum of two integers: 30
Sum of two doubles: 12.2
Sum of three integers: 6
```

ERRORS:

S.NO	ERROR MESSAGE	RECTIFICATION
1.	method add(int, int) is already defined	Ensure method signatures differ in parameter **types or number**, not just return type.
2.	possible lossy conversion from double to int	Use correct data types during assignment: `double sum2 = calc.add(5.5, 6.7);`

IMPORTANT POINTS:

- **Method Overloading Demonstration**
  The `Calculator` class shows **method overloading**: same method name (`add`) but different parameter types/number.
- **Compile-Time Polymorphism**
  Java decides which `add()` method to call at **compile time**, making it an example of static (compile-time) polymorphism.
- **Strong Typing**
  The code enforces strong typing. The correct overloaded method is selected based on the **argument types** passed during method calls.

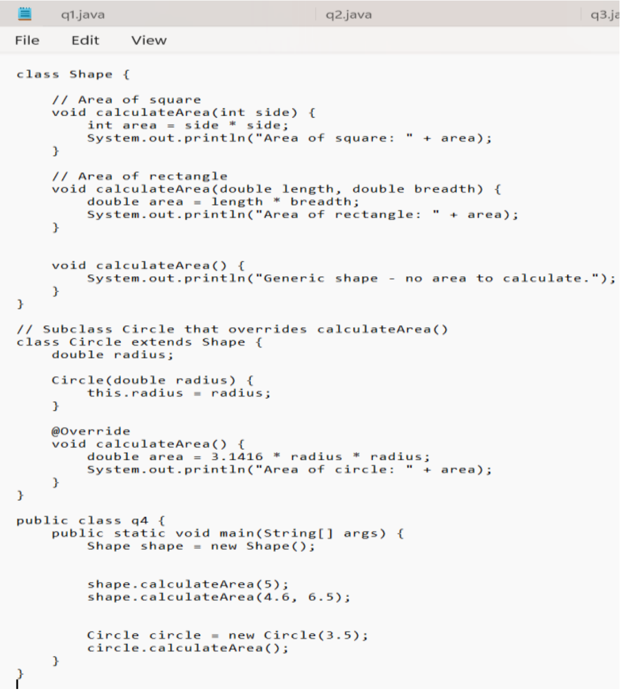
PROGRAM-4:

AIM: Create a shape class with method calculateArea() that is overloaded for different shapes (eg: square, rectangle).Then create a subclass Circle that overrides calculateArea() method for Circle.

CLASS DIAGRAM:

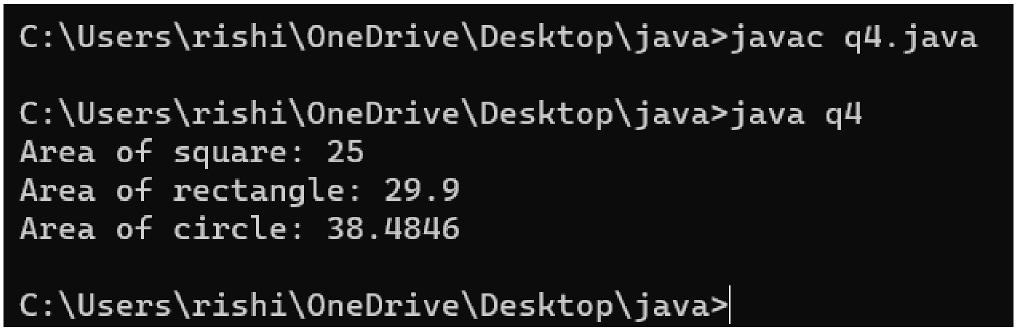
```
+------------------------+

|        Shape           |
+-----------------------+
| + calculateArea(): void              |
| + calculateArea(int): void           |
| + calculateArea(double, double): void|
+-----------------------+
          ▲
          |
+-----------------------+
|       Circle          |
+-----------------------+
| - radius: double      |
| + Circle(radius: double)|
| + calculateArea(): void |
+-----------------------+


+-----------------------+
|         Q4            |
+-----------------------+
| + main(String[]): void  |
+-----------------------+
```

INPUT CODE:



```java
class Shape {

    // Area of square
    void calculateArea(int side) {
        int area = side * side;
        System.out.println("Area of square: " + area);
    }

    // Area of rectangle
    void calculateArea(double length, double breadth) {
        double area = length * breadth;
        System.out.println("Area of rectangle: " + area);
    }


    void calculateArea() {
        System.out.println("Generic shape - no area to calculate.");
    }
}

// Subclass Circle that overrides calculateArea()
class Circle extends Shape {
    double radius;

    Circle(double radius) {
        this.radius = radius;
    }

    @Override
    void calculateArea() {
        double area = 3.1416 * radius * radius;
        System.out.println("Area of circle: " + area);
    }
}

public class q4 {
    public static void main(String[] args) {
        Shape shape = new Shape();


        shape.calculateArea(5);
        shape.calculateArea(4.6, 6.5);


        Circle circle = new Circle(3.5);
        circle.calculateArea();
    }
}
```

OUTPUT CODE:



```
C:\Users\rishi\OneDrive\Desktop\java>javac q4.java

C:\Users\rishi\OneDrive\Desktop\java>java q4
Area of square: 25
Area of rectangle: 29.9
Area of circle: 38.4846

C:\Users\rishi\OneDrive\Desktop\java>
```

ERRORS:

S.NO	ERROR MESSAGE	RECTIFICATION
1.	method calculateArea(double) is undefined for the type Shape	Add an overloaded method: `void calculateArea(double side)` to handle squares with double sides, or cast to int: `shape.calculateArea((int)5.0);`

IMPORTANT POINTS;

- **Method Overloading**:
 `Shape` class uses method overloading to define multiple versions of `calculateArea()` with different parameters.
- **Method Overriding with `@Override`**:
 `Circle` overrides the no-arg `calculateArea()` method of `Shape` to provide its own specific implementation.
- **Flexible Design for Extension**:
 The base class `Shape` can be easily extended by new shapes like `Triangle`, `Hexagon`, etc., demonstrating polymorphism and scalability.

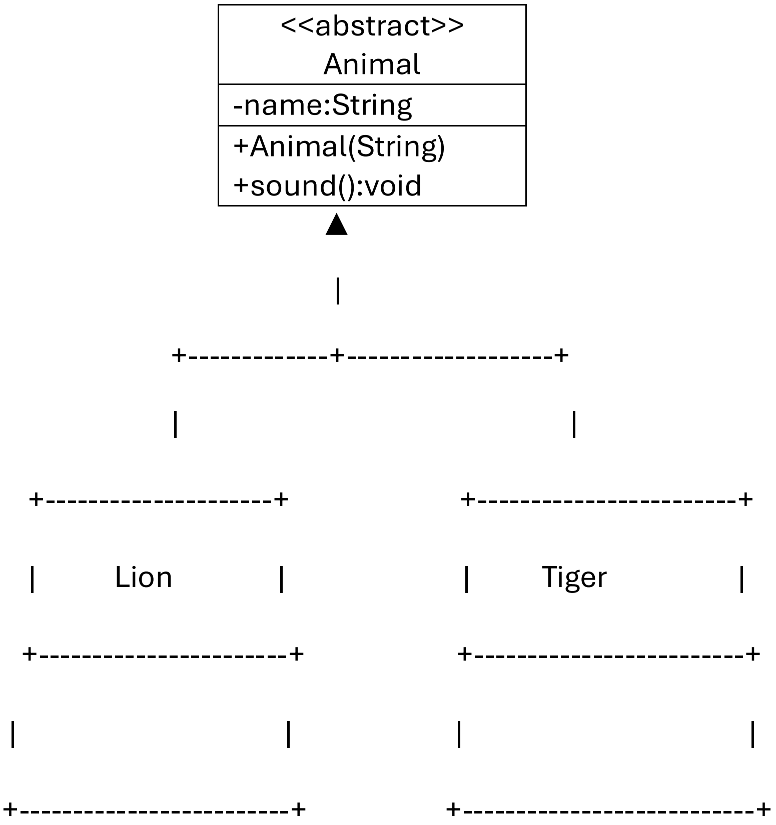
WEEK-7

PROGRAM-1:

AIM:

Write a Java program to create an abstract class Animal with an abstract method called sound(). Create subclasses Lion and Tiger that extend the Animal class and implement the sound() method to make a specific sound for each animal.

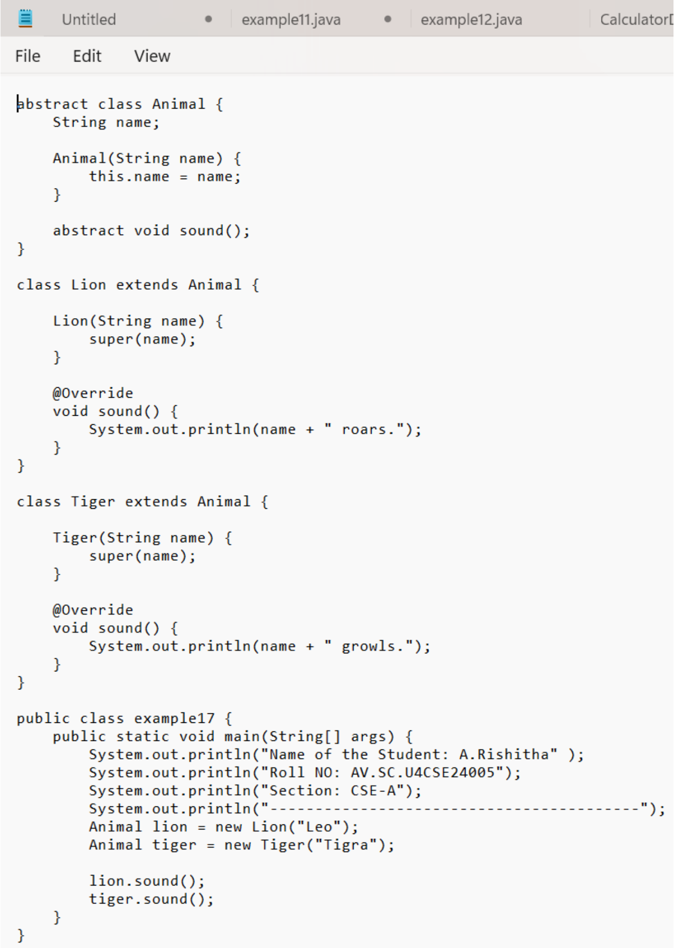
CLASS DIAGRAM:



| +Lion(String) | |+Tiger(String) |

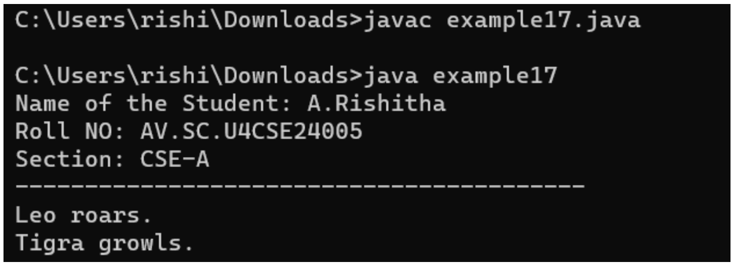
| +sound(): void | | +sound(): void |

INPUT CODE:



```
abstract class Animal {
    String name;

    Animal(String name) {
        this.name = name;
    }

    abstract void sound();
}

class Lion extends Animal {

    Lion(String name) {
        super(name);
    }

    @Override
    void sound() {
        System.out.println(name + " roars.");
    }
}

class Tiger extends Animal {

    Tiger(String name) {
        super(name);
    }

    @Override
    void sound() {
        System.out.println(name + " growls.");
    }
}

public class example17 {
    public static void main(String[] args) {
        System.out.println("Name of the Student: A.Rishitha" );
        System.out.println("Roll NO: AV.SC.U4CSE24005");
        System.out.println("Section: CSE-A");
        System.out.println("---------------------------------------");
        Animal lion = new Lion("Leo");
        Animal tiger = new Tiger("Tigra");

        lion.sound();
        tiger.sound();
    }
}
```

OUTPUT CODE:



```
C:\Users\rishi\Downloads>javac example17.java

C:\Users\rishi\Downloads>java example17
Name of the Student: A.Rishitha
Roll NO: AV.SC.U4CSE24005
Section: CSE-A
-----------------------------------------
Leo roars.
Tigra growls.
```

ERRORS:

S.NO	ERROR IDENTIFIED	RECTIFICATION

1.	Missing method	Removed print statement for private variable and method
2.	Animal is abstarct,cannot be instantiated	Use subclass

IMPORTANT POINTS:

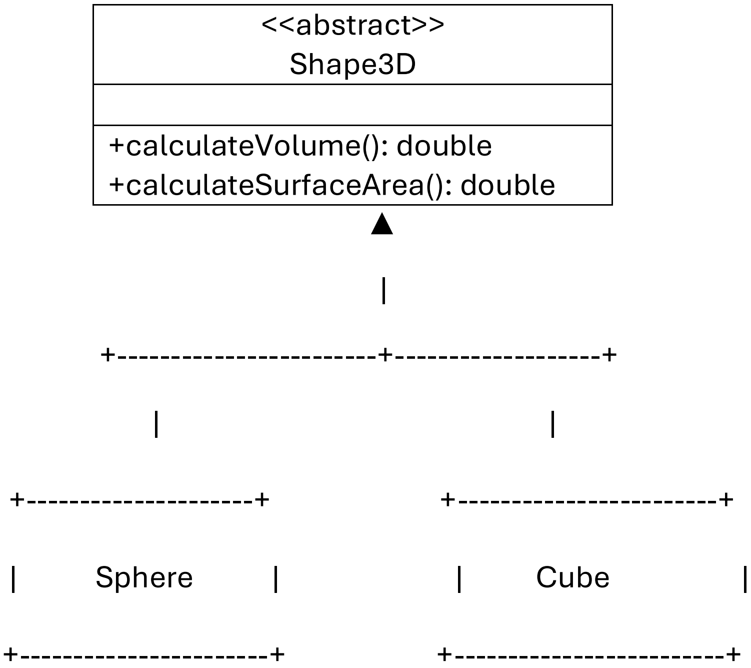
- **Abstract Class**: `Animal` is an abstract class, meaning it cannot be instantiated directly, and must be subclassed (e.g., `Lion`, `Tiger`).
- **Abstract Method**: The `sound()` method is abstract in the `Animal` class and must be overridden in each subclass (`Lion` and `Tiger`).
- **Polymorphism**: The `Animal` reference variable is used to hold objects of different types (`Lion`, `Tiger`), demonstrating polymorphism when calling the `sound()` method.

PROGRAM-2:

AIM:

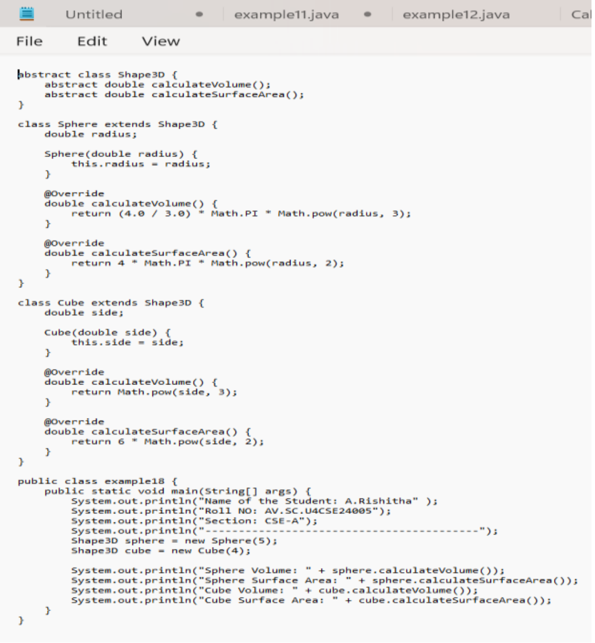
Write a Java program to create an abstract class Shape3D with abstract methods calculateVolume() and calculateSurfaceArea(). Create subclasses Sphere and Cube that extend the Shape3D class and implement the respective methods to calculate the volume and surface area of each shape.

CLASS DIAGRAM:



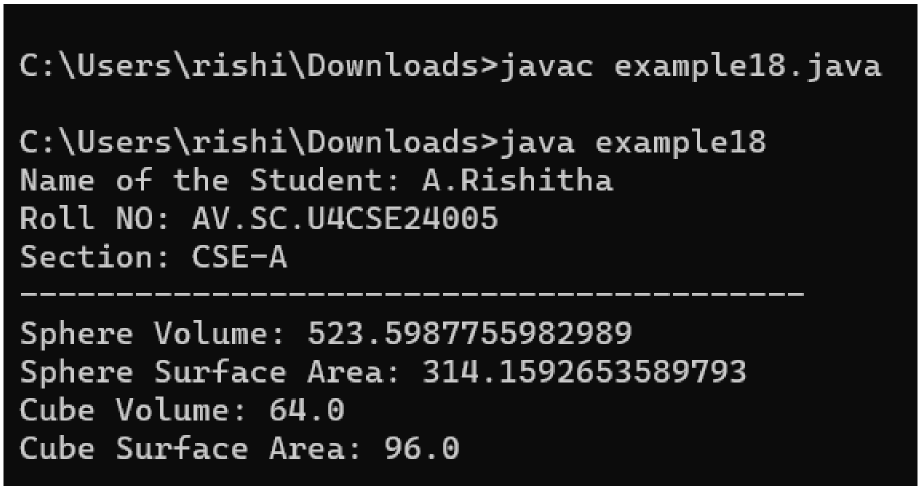
```
                 |  -radius:double      |                | -side:double           |

                +------------------------+                   +--------------------------+

                 | +Sphere(radius)        |                    |+Cube(side)             |

        | +calculateVolume():double   |              | +  calculateVolume():double|
| + calculateSurfaceArea(): double|                    | + calculateSurfaceArea():double|
+---------------------------------------------+            +----------------------------------------+
```

INPUT CODE:



```java
abstract class Shape3D {
    abstract double calculateVolume();
    abstract double calculateSurfaceArea();
}

class Sphere extends Shape3D {
    double radius;

    Sphere(double radius) {
        this.radius = radius;
    }

    @Override
    double calculateVolume() {
        return (4.0 / 3.0) * Math.PI * Math.pow(radius, 3);
    }

    @Override
    double calculateSurfaceArea() {
        return 4 * Math.PI * Math.pow(radius, 2);
    }
}

class Cube extends Shape3D {
    double side;

    Cube(double side) {
        this.side = side;
    }

    @Override
    double calculateVolume() {
        return Math.pow(side, 3);
    }

    @Override
    double calculateSurfaceArea() {
        return 6 * Math.pow(side, 2);
    }
}

public class example18 {
    public static void main(String[] args) {
        System.out.println("Name of the Student: A.Rishitha" );
        System.out.println("Roll NO: AV.SC.U4CSE24005");
        System.out.println("Section: CSE-A");
        System.out.println("----------------------------------------");
        Shape3D sphere = new Sphere(5);
        Shape3D cube = new Cube(4);

        System.out.println("Sphere Volume: " + sphere.calculateVolume());
        System.out.println("Sphere Surface Area: " + sphere.calculateSurfaceArea());
        System.out.println("Cube Volume: " + cube.calculateVolume());
        System.out.println("Cube Surface Area: " + cube.calculateSurfaceArea());
    }
}
```

OUTPUT CODE:



```
C:\Users\rishi\Downloads>javac example18.java

C:\Users\rishi\Downloads>java example18
Name of the Student: A.Rishitha
Roll NO: AV.SC.U4CSE24005
Section: CSE-A
-----------------------------------------
Sphere Volume: 523.5987755982989
Sphere Surface Area: 314.1592653589793
Cube Volume: 64.0
Cube Surface Area: 96.0
```

ERRORS:

S.NO	ERROR IDENTIFIED	RECTIFICATION
1.	CalculateSurfaceArea()	Implement method

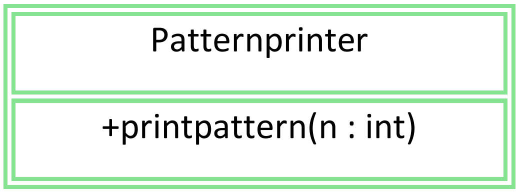
IMPORTANT POINTS:

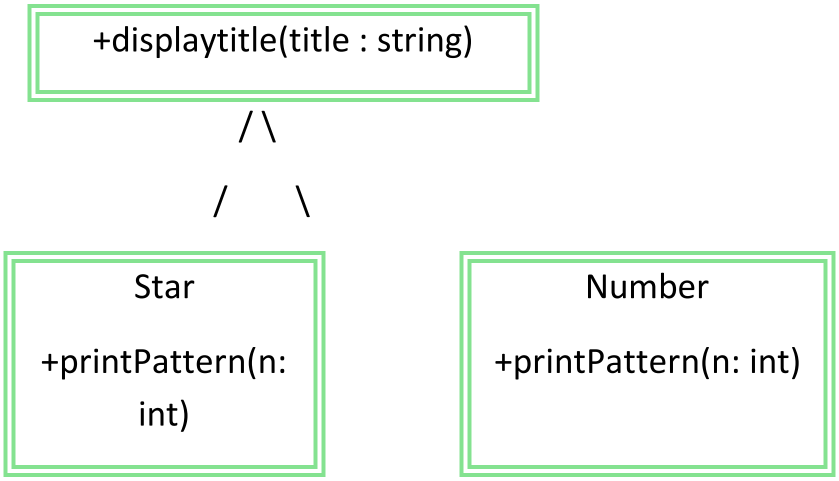
- **Abstract Class**: Shape3D is an abstract class, so it cannot be instantiated directly.
- **Method Overriding**: Sphere and Cube override the abstract methods `calculateVolume()` and `calculateSurfaceArea()`.
- **Polymorphism**: Using the Shape3D reference type allows calling overridden methods on Sphere and Cube.

PROGRAM-3:

AIM: Write a java program to create a abstract class named patternprinter with an abstract method printpattern(int n) and a concrete method to display the pattern title.

CLASS DIAGRAM:





+displaytitle(title : string)

/ \

/ \

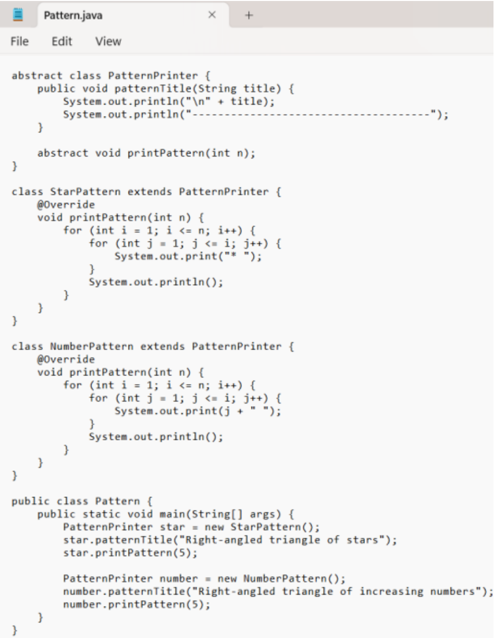
Star

+printPattern(n: int)

Number

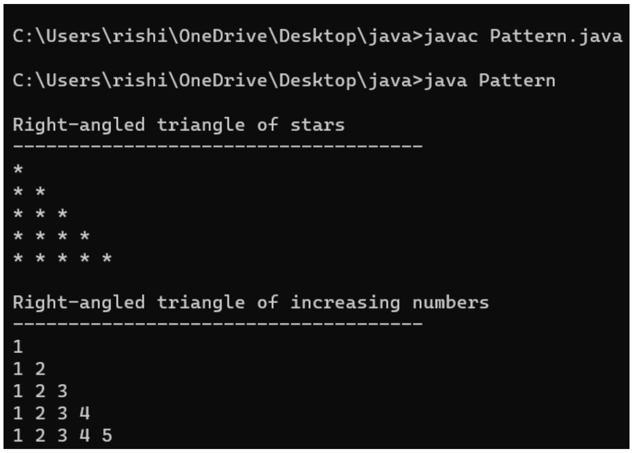
+printPattern(n: int)

INPUTCODE:



```
abstract class PatternPrinter {
    public void patternTitle(String title) {
        System.out.println("\n" + title);
        System.out.println("-------------------------------------");
    }

    abstract void printPattern(int n);
}

class StarPattern extends PatternPrinter {
    @Override
    void printPattern(int n) {
        for (int i = 1; i <= n; i++) {
            for (int j = 1; j <= i; j++) {
                System.out.print("* ");
            }
            System.out.println();
        }
    }
}

class NumberPattern extends PatternPrinter {
    @Override
    void printPattern(int n) {
        for (int i = 1; i <= n; i++) {
            for (int j = 1; j <= i; j++) {
                System.out.print(j + " ");
            }
            System.out.println();
        }
    }
}

public class Pattern {
    public static void main(String[] args) {
        PatternPrinter star = new StarPattern();
        star.patternTitle("Right-angled triangle of stars");
        star.printPattern(5);

        PatternPrinter number = new NumberPattern();
        number.patternTitle("Right-angled triangle of increasing numbers");
        number.printPattern(5);
    }
}
```

OUTPUTCODE:



```
C:\Users\rishi\OneDrive\Desktop\java>javac Pattern.java

C:\Users\rishi\OneDrive\Desktop\java>java Pattern

Right-angled triangle of stars
-------------------------------------
*
* *
* * *
* * * *
* * * * *

Right-angled triangle of increasing numbers
-------------------------------------
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
```

ERRORS:

S.NO	ERROR MESSAGES	RECTIFICATION
1.	Incorrect method call public String patterntitle();	Rectified as public void patterntitle(String title);
2.	Incorrect start of the loop	Overrided the method

IMPORTANT POINTS:

1. The abstract class `PatternPrinter` defines a common structure for different pattern types, enforcing implementation of `printPattern(int n)` in subclasses.
2. Polymorphism is used in the `main` method by referencing child objects (`StarPattern`, `NumberPattern`) through the parent class type.
3. The `patternTitle(String title)` method separates the title-printing logic from the pattern-printing logic, making the code more organized and readable.