**LAB MANUAL**



**ROLLNO:AV.SC.U4CSE24121**

**NAME: RITHWIK G**

**SECTION: CSE-B**

**WEEK-1:**

**Aim:** How to install jdk and first program on

printing student details*.*

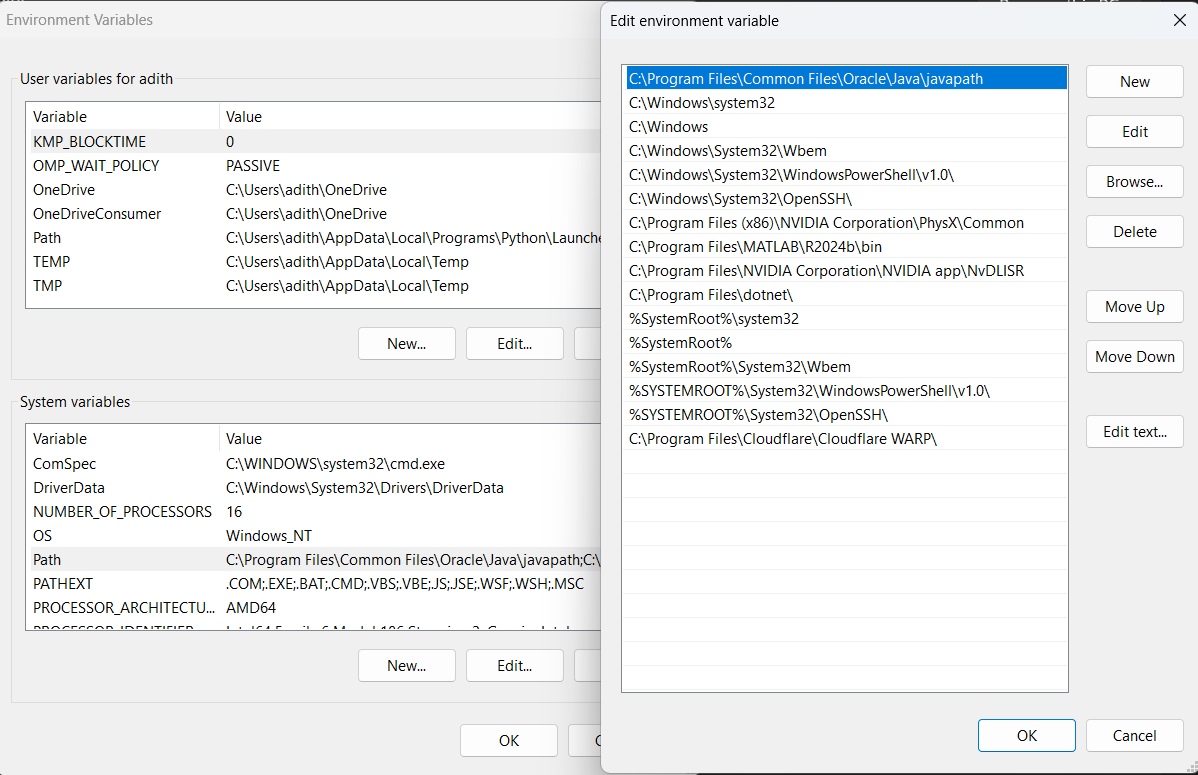
**Step-1:** Download JDK-21 from oracle website

**

**Step-2:**Install the JDK-21 with accepting terms and

conditions according to the respective windows.

**Step-3**:Setting up environmental variables.



\*Windows c -> C-drive -> program files ->Java -

>JDK-21->select bin

\*Select and open environmental variable in search

bar-> either select system variables or user

variables-> select path-> click edit->New-> paste

the bin-> finish the setup(apply the changes).

~for verifying the installed version

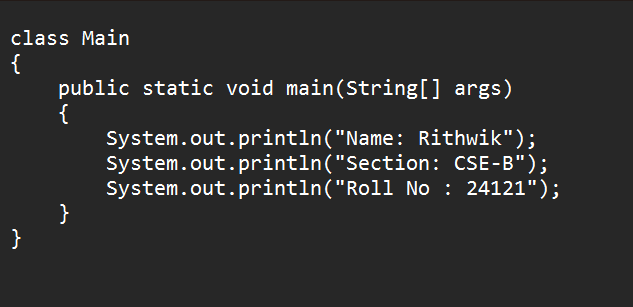
Open cmd-> type java --version

~command propt

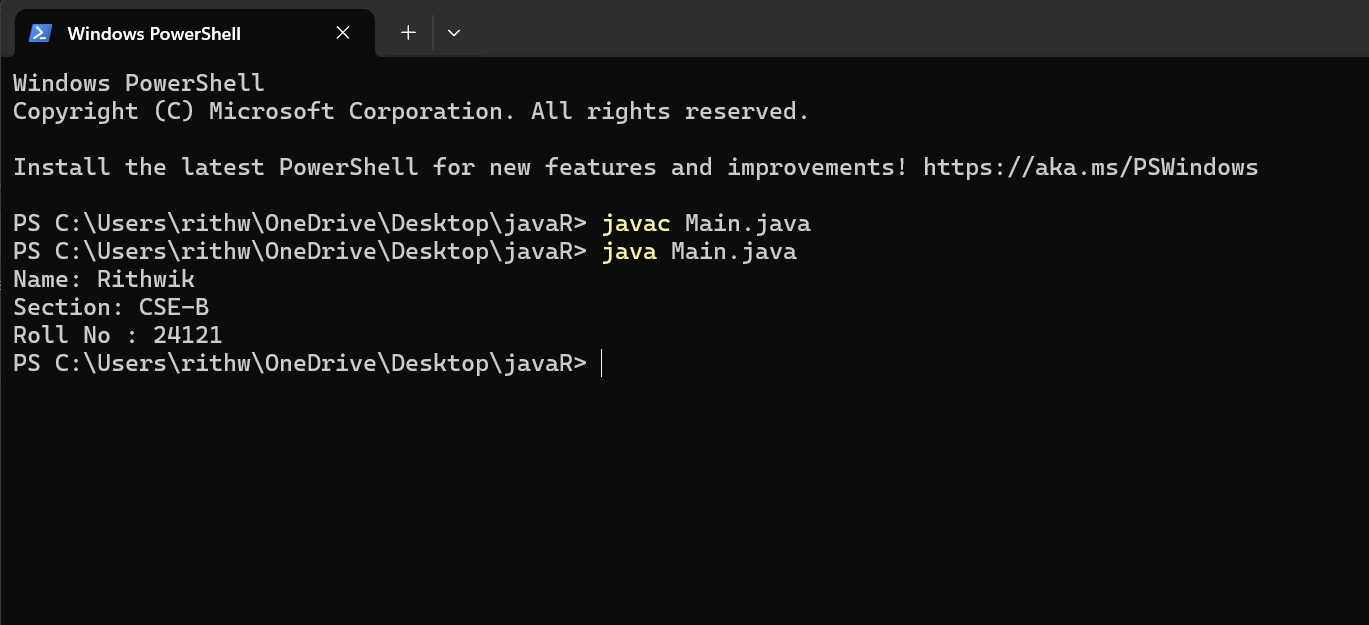
Javac filename.java ->compiling.

Java filename.java ->displaying

**PROGRAM-1(Rectified):**

******

**Output:**



***IMPORTANT POINTS:***

1. ***When printing the statements, everything should be inside double quotes.***

**WEEK-2:**

**PROGRAM-1:**

**Aim:**Write a java program for SI

**

**Output:**

******

**ERROR TABLE*:***

|  |  |
| --- | --- |
| **Code Error** | **Code rectification** |
| 1.Giving space between next and Double.  2.Not giving parenthesis after closing the input. | 1.Should not give space between next and Double.  2.We must put parenthesis after closing the input. |

***IMPORTANT POINTS:***

1. ***Simple interest formula is: (p\*t\*r)/100, where:***

***P: Principal amount***

***R: Rate of interest***

***T: Time period***

1. ***The data type double indicates the floating points in the integers.***
2. ***The line “import java.util.Scanner” indicates:***

***Import: tells the java compiler that we want to use a specific class or package in your code.***

***Java.util : This is the package that contains utility classes for Java programming, including the “Scanner” class.***

***Scanner: this is the class that allows you to read input from the keyboard.***

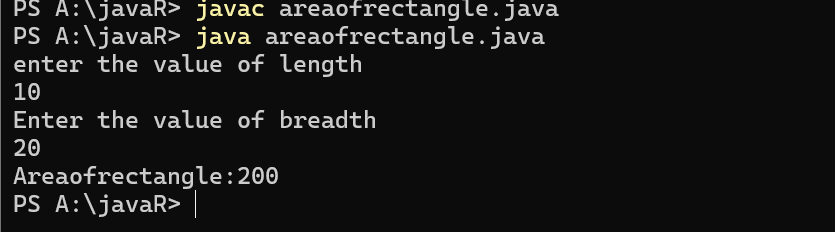
**PROGRAM-2:**

**Aim:**Write a program in java for area of

rectangle.

**

**Output:**

******

**ERROR TABLE:**

|  |  |
| --- | --- |
| **Code Error** | **Code rectification** |
| 1.While using for iteration, not giving the conditions correctly.  2.Declaring the data type as double instead of int. | 1.We should give iterative statements correctly.  2.We should give the data type as int for integers. |

IMPORTANT POINTS:

1. Area of a rectangle is area = l\*b, where

L = length of a side of the rectangle,

B= breadth of a side of the rectangle.

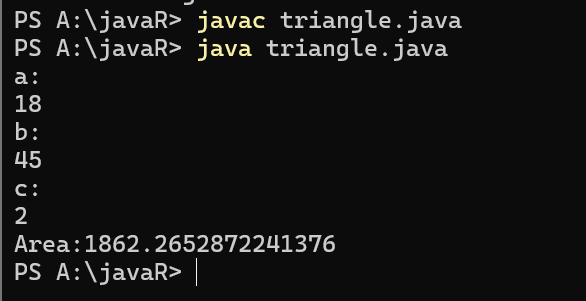
1. Here, we must be sure that all the expressions/conditions inside for the for loop must be given correctly.

**PROGRAM-3:**

**Aim:**Write a program in java for area of triangle using heron’s formula.



**Output:**

******

ERROR TABLE:

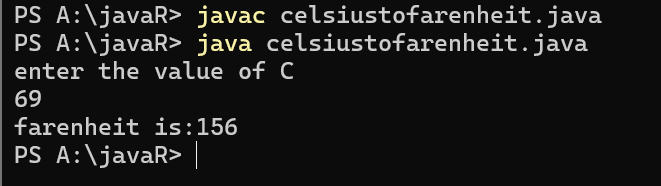
|  |  |
| --- | --- |
| **Code Error** | Code rectification |
| 1.While printing the variable not giving + sign.  2.Not closing the scanner. | 1.We should give correct indentation.  2.Closing the scanner is must. |

**PROGRAM-4(a):**

**Aim:**Write a program in java for converting temperature from celsius to fahrenite.



OUTPUT:

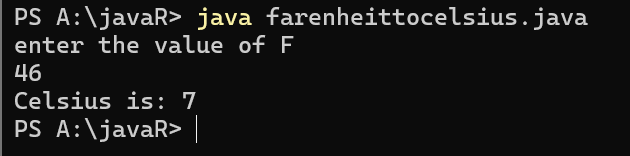


**PROGRAM-4(b):**

**Aim:**Write a program in java for converting temperature from fahrenite to celsius.

******

**Output:**

******

**ERROR TABLE:**

|  |  |
| --- | --- |
| **Code Error** | **Code rectification** |
| 1.While printing the variable not giving + sign.  2.Not closing the scanner. | 1.We should give correct indentation.  2.Closing the scanner is must. |

IMPORTANT POINTS:

1. Area of a rectangle is area = l\*b, where

L = length of a side of the rectangle,

B= breadth of a side of the rectangle.

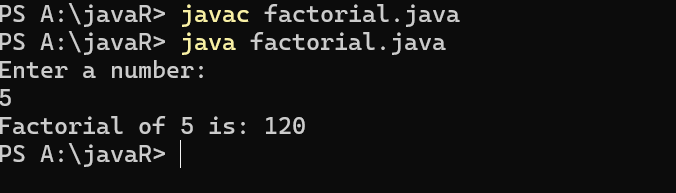
1. Here, we must be sure that all the expressions/conditions inside for the for loop must be given correctly.

**PROGRAM-5:**

**Aim:**Write a program in java for factorial of a number.

******

OUTPUT:



IMPORTANT POINTS:

1. While the for loop the data inside the parenthesis indicates the Initial expression

Test expression and

Update expression.

1. Here “factorial\*=I” means factorial = factorial\*I.
2. Here we are using the data type “int” just to calculate the integer values and it doesn’t support floating points.

ERROR TABLE:

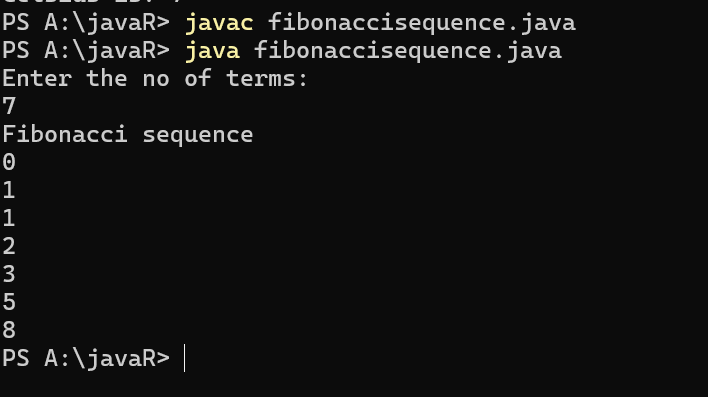
|  |  |
| --- | --- |
| **Code Error** | **Code rectification** |
| 1.While using for iteration, not giving the conditions correctly.  2.Declaring the data type as double instead of int. | 1.We should give iterative statements correctly.  2.We should give the data type as int for integers. |

**PROGRAM-6:**

**Aim:**Write a program in java for fibonacci series.



OUTPUT:



ERROR TABLE:

|  |  |
| --- | --- |
| **Code Error** | **Code rectification** |
| 1.Giving space between next and Double.  2.Not giving parenthesis after closing the input. | 1.Should not give space between next and Double.  2.We must put parenthesis after closing the input. |

**WEEK -3:**

**PROGRAM-1:**

**AIM:** To create java program with following instructions :

1.Create a class with name Car

2.Create four attributes named car\_color,car\_brand, fuel\_type, mileage

3.Create these methods named start(),stop(),service()

4.Create the objects named car, car1,car2

**CODE:**

public class Car {

private String car\_color;

private String car\_brand;

private String fuel\_type;

private String mileage;

public void start() {

System.out.println("car is started");

}

public void stop() {

System.out.println("car is stopped");

}

public void service() {

System.out.println("car is for service");

}

public static void main(String args[]) {

Car car = new Car();

car.car\_color = "white";

car.car\_brand = "audi";

car.fuel\_type = "petrol";

car.mileage = "20";

car.start();

System.out.println("car\_color: " + car.car\_color + " car\_brand: " + car.car\_brand + " fuel\_type: " + car.fuel\_type + " mileage: " + car.mileage);

Car car1 = new Car();

car1.car\_color = "white";

car1.car\_brand = "audi";

car1.fuel\_type = "petrol";

car1.mileage = "20";

car1.stop();

System.out.println("car\_color: " + car1.car\_color + " car\_brand: " + car1.car\_brand + " fuel\_type: " + car1.fuel\_type + " mileage: " + car1.mileage);

Car car2 = new Car();

car2.car\_color = "white";

car2.car\_brand = "audi";

car2.fuel\_type = "petrol";

car2.mileage = "20";

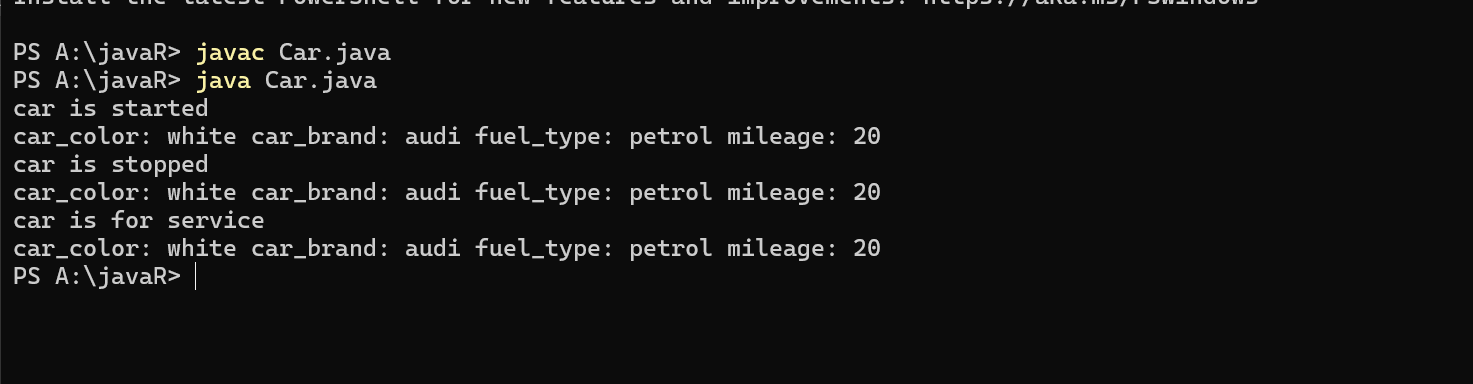
car2.service();

System.out.println("car\_color: " + car2.car\_color + " car\_brand: " + car2.car\_brand + " fuel\_type: " + car2.fuel\_type + " mileage: " + car2.mileage);

}

}

**OUTPUT:**

****

**Error table:**

|  |  |  |  |
| --- | --- | --- | --- |
| S.no | Error name | Cause of error | Rectification |
| 1 | Syntax Error | Missing ‘{‘ | ‘{‘ added |
| 2 | Compile time Error | Mispelled Variable call | Rectified with  Correct variable name |
| 3 | Case sensitive error | Uppercase and lowercase | rectified |

**Class diagram:**

|  |
| --- |
| **car**  **----------------------**-  -car\_color:string  -car\_brand:string  -fuel\_type:string  -milage:double  ----------------------  +start():void  +stop():void  +service():void |

IMPORTANT POINTS:

1. Before calling the function we should write the method properly.
2. Here, the “public void start( )” indicates that we are writing a method to call the function.
3. When we call a certain method, the process inside it will be printed as an output of the code.
4. Here the details inside the function are called objects, we can give any objects

**PROGRAM-2:**

**Aim:** To create a class BankAccount with methods deposit() and withdraw() . create two subclasses savingsaccount and checkingaccount override the withdraw () method in each subclass to impose different withdrawal limits and fees

public class BankAccount {

protected String accountHolder;

protected double balance;

protected int accountNumber;

public BankAccount(String accountHolder, int accountNumber, double balance) {

this.accountHolder = accountHolder;

this.accountNumber = accountNumber;

this.balance = balance;

}

public void withdrawal(double amount) {

if (amount <= balance) {

balance = balance - amount;

System.out.println("Current balance: " + balance);

} else {

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

}

}

public void deposit(double amount) {

balance = balance + amount;

System.out.println("Current balance: " + balance);

}

public static void main(String[] args) {

BankAccount BA = new BankAccount("Abdul", 24248, 1000);

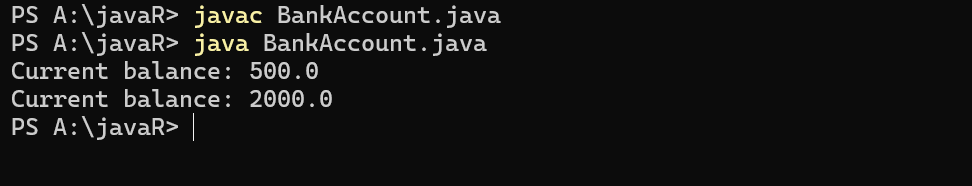
BA.withdrawal(500);

BA.deposit(1500);

}

}

**OUTPUT:**

****

**Error table:**

|  |  |  |  |
| --- | --- | --- | --- |
| S.no | Error name | Error name | Rectification |
| 1 | Name Error | Undefined name | Correct variable  Name replaced |
| 2 | Syntax Error | Missing Parenthesis | Parenthesis Added |
| 3 | Logical Error | Incorrect Condition | Condition Rectified |

**Class diagram:**

|  |
| --- |
| **BankAccount**  ----------------------------------------------------------  -balance: double  ----------------------------------------------------------  +BankAccount(intialBalance: double)  +deposit(amount: double):void  +withdraw(amount: double):void |

IMPORTANT POINTS:

1. The condition inside the if statement must be correct.
2. It explains that if the withdrawal money is less than the money in the bank account, then we can withdraw the amount.

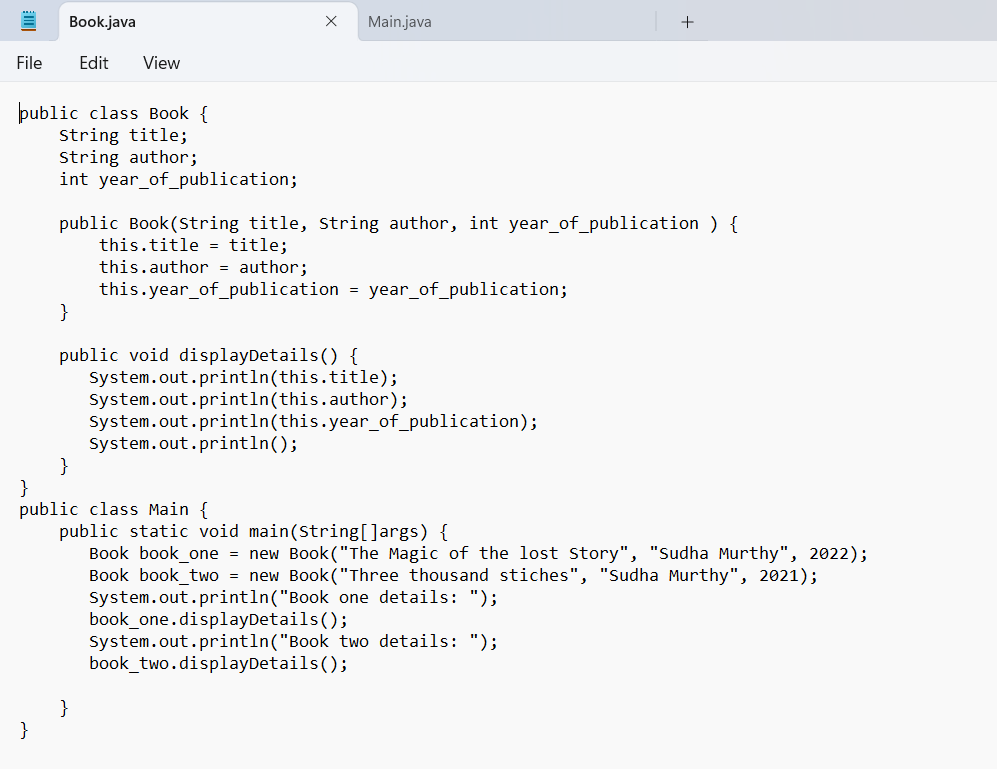
WEEK -4

PROGRAM – 1:

AIM: Write a java program with class named “book”, the class should contain various attributes such as title, author, year of publication it should also contain a constructor with parameters which initializes, title, author, and year of publication.

Create a method which displays the details of the book and display the details of two books.

CODE:



OUTPUT:



ERROR TABLE:

|  |  |
| --- | --- |
| **Code Error** | **Code rectification** |
| 1. Not defining the function in a file. 2. Two public class files should not be saved in the same file. | 1. To call the method we must define a function in a file. 2. Two public class files should be saved in different files. |

IMPORTANT POINTS:

1. While defining two classes for a code, we must be sure that we save both the classes in separate files.
2. While defining a method we should also define a function to call that method.

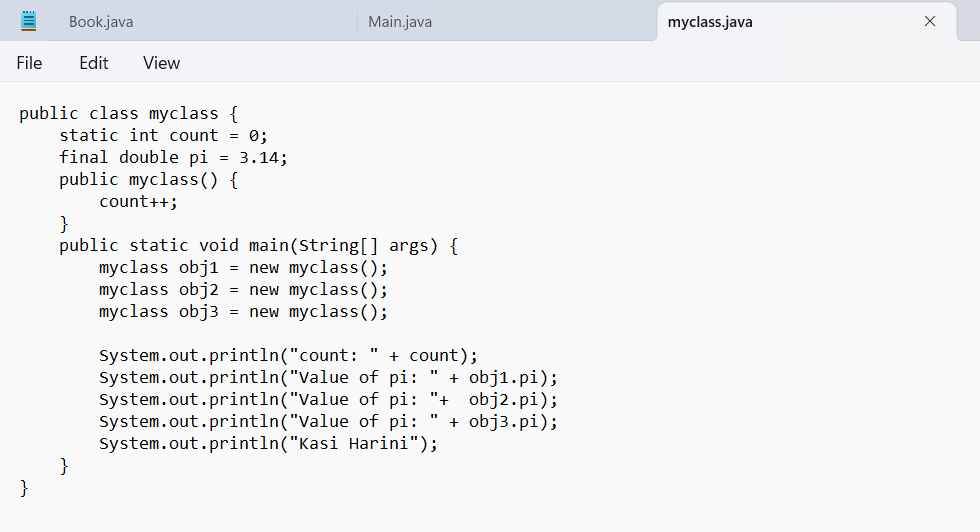
CLASS DIAGRAM:

|  |
| --- |
| Book   * Title: String * Author: String * Year of publication: int   + Book(title: String,  Author: String;  Year of publication: int  + displayDetails( ): void |

PROGRAM – 2:

AIM: Create a java Program with class named myclass with static variable count of int type, initialized to zero and a constant variable “pi” of type double initialized to 3.14 as attributes of the class, ow define a constructor for “myclass” that increments the count variable each time an object of my class is created (count++), finally print the final values of count and pi variables create three objects.

CODE:



OUTPUT:



ERROR TABLE:

|  |  |
| --- | --- |
| **Code Error** | **Code rectification** |
| 1. Not Putting the semi-colon after calling a function, 2. Not giving the indentation properly. | 1. Put the semi-colon after calling a function. 2. All the indentation must be correct to run the code correct. |

IMPORTANT POINTS:

1. We must declare the initial value of the variable before declaring the final one.
2. Here the main objective is to increase the count according to the number of objects we make, i.e the count increases when the no.of objects are increasing.

CLASS DIAGRAM:

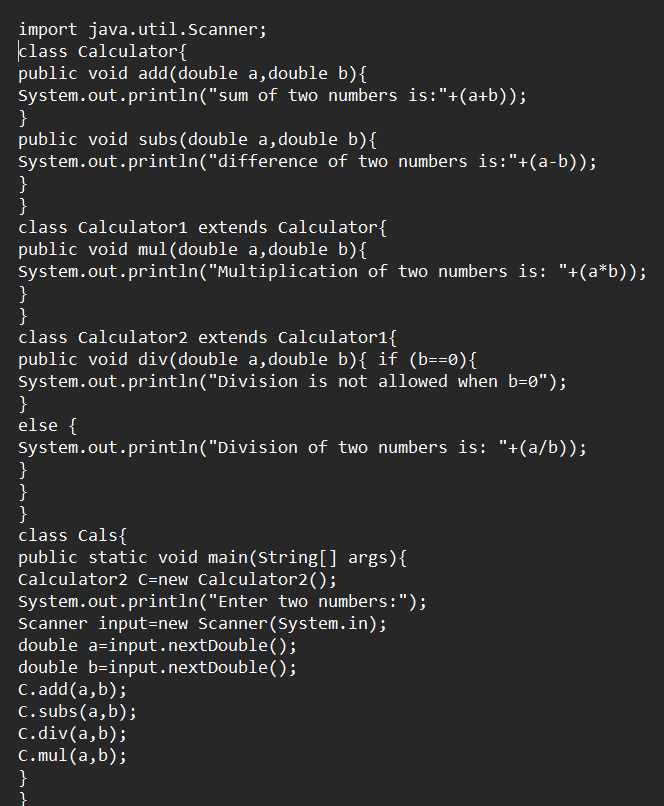
|  |
| --- |
| Myclass   * Count: int * Pi: double   + myclass( )  + main(args: String[]): void |

**Week-5**

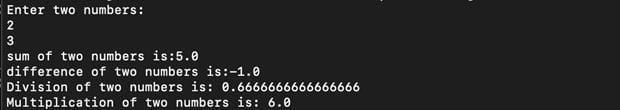
**Program 1:**

**Aim:** Create a calculator using the operations including addition, subtraction, multiplication and division using multilevel in heritance and display the desired output.

**Program:**



**Output:**



**Errors:**

|  |  |  |
| --- | --- | --- |
| **Sl.No** | **Error name** | **Error Rectification** |

|  |  |  |
| --- | --- | --- |
| **1.** | **Semi colon (;)** | **Givethe semi colon (;) in each line where it is**  **required** |
| **2.** | **Syntax Error** | **Giving Capital ‘S’ in printing statements (System.out.println)** |

**Class Diagram:**

+add(doublea,doubleb): void

+subs(doublea,double b): void

calculator

+multiplication

(double a,double b): void

**Calculator1**

+div(doublea,double b): void

**Calculator2**

**Program 2:**

**Aim:** A vehicle rental company wants to develop a system that maintains information about different types of vehicles available for rent. The company rents out cars and bikes and they need a program to store details about each vehicle such as brand and speed**.**

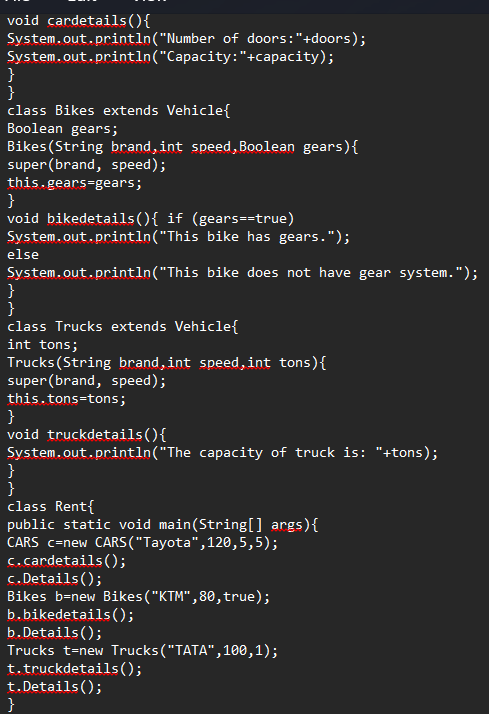
**Questions:**

1. Which OOPS concepts used in the above program? Explain why itis useful in this scenario.
2. If the company decides to add a new type of vehicle ‘Truck’, how would you modify the program?

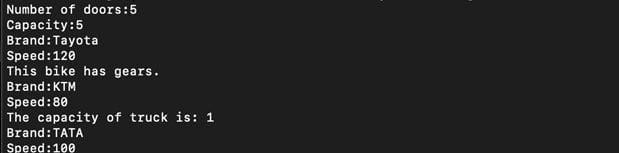
Truck should include and additionalpropertycapacity (in tons).

* + - 1. CreateashowTruck()methodtodisplaythetruck’s capacity.
      2. Write a constructor for truck that initializes all properties.
      3. Implementthetruck classandupdatethemainmethodto createaTruckobjectand alsocreateanobjectforcarand bikesubclasses. Finallydisplay the details

**Program:**



**Output:**

****

**Errors:**

|  |  |  |
| --- | --- | --- |
| **Sl.No** | **Error name** | **Error Rectification** |
| **1.** | **Semi colon (;)** | **Give the semi colon (;) in each line where it is required** |
| **2.** | **Syntax Error** | **Giving Capital ‘S’ in printing statements (System.out.println)** |

**Class Diagram:**

|  |
| --- |
| **Vehicle** |
| **brand: string speed: string** |
| **+Vehicle(String brand, int speed)**  **+Details(): void** |

|  |
| --- |
| **CARS** |
| **doors: int** |

|  |
| --- |
| **capacity: int** |
| **+ CARS (String brand, int speed, int doors, int capacity)**  **+cardetails(): void** |

|  |
| --- |
| **Bikes** |
| **gears: Boolean** |
| **+ Bikes(String brand, int speed, Boolean gears)**  **+bikedetails(): void** |

****

|  |
| --- |
| **Trucks** |
| **tons: int** |
| **+ Trucks(String brand,int speed,int tons)**  **+truckdetails(): void** |

**Important points:**

Multi-inheritance: It is one of the types of the inheritance where subclass 2 inherits subclass1 and subclass1 inherits superclass.

Here Vehicle is the super class or parent class and remaining cars, bikes, trucks are the subclasses or child classes

**Week 6:**

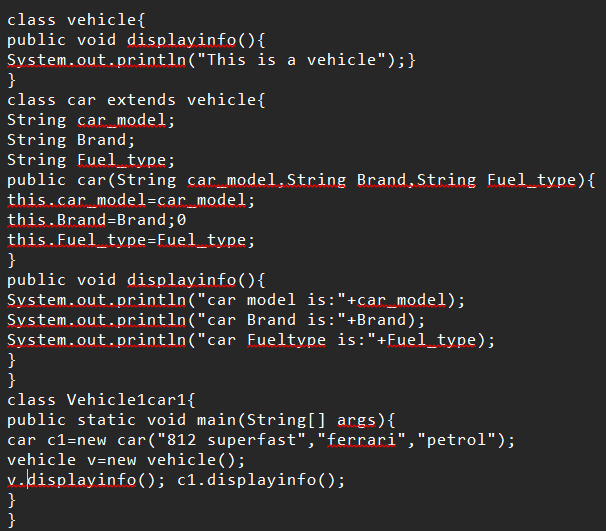
**Program 1:**

**Aim:** Write a Java program to create a vehicle class with a method displayInfo(). Override this method in the car subclass to provide specific information about a car, model, fuel type, and colour using the constructor

**Syntax:**

**Super class extends subclass**

Here extends is the main key word which represents the extending relation from parent class to child class.

**Program:**

Output:



**Errors:**

|  |  |  |
| --- | --- | --- |
| **Sl.No** | **Error name** | **Error Rectification** |
| 1. | Semi colon (;) | Give the semi colon (;) in each line where it is required |
| 2. | Syntax Error | Giving Capital ‘S’ in printing statements (System.out.println) |

**Class Diagram:**

|  |
| --- |
| **Vehicle** |
| **+displayinfo():void** |

|  |
| --- |
| **Car\_model:String Brand:String Fuel\_type:String** |
| **+ car(String car\_model,String Brand,String Fuel\_type)**  **+displayinfo(): void** |

**Important points:**

In order to do this, we have to use inheritance concept. Here we used the multi-inheritance concept**.**

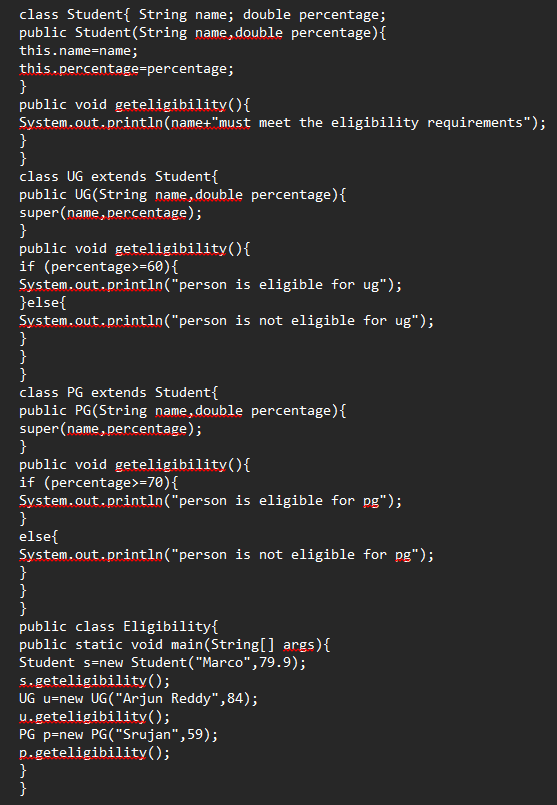
**Program 2:**

**Aim:** Create a Java program for the scenario.

A college is developing an automated admission system that verifies student eligibility for undergraduate (UG) and postgraduate(PG) programs. Each program has different eligibility criteria based on the student's percentage in their previous qualification.

1. UG admissions require aminimum of 60%
2. PG admissions require aminimum of 70%

**Program:**



**Output:**



**Errors:**

|  |  |  |
| --- | --- | --- |
| **Sl.No** | **Error name** | **Error Rectification** |
| **1.** | **Semi colon (;)** | **Give the semi colon (;) in each line where it is required** |
| **2.** | **Syntax Error** | **Giving Capital ‘S’ in printing statements (System.out.println)** |

**Important points:**

Super keyword is used take the method,variable,constructor from the super class.

**Class diagram:**

**Student**

name: String

percentage:double

+Student(String name,double percentage)

+geteligibility():void

+UG(String name,double

UG

**Program-3:**

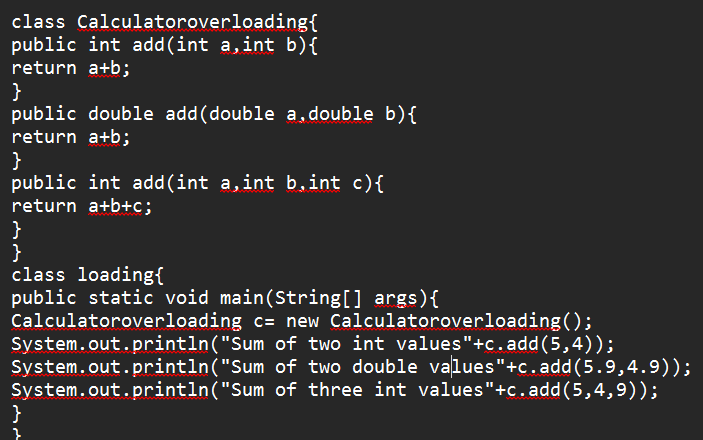
**Aim:** Write a Java Program to create a Calculator class with overloaded methods to perform addition: Take the integer values a and b from the user.

* 1. Addtwointegers
  2. Addtwodoubles
  3. Addthreeintegers

**Important points:**

We should carefully pass the double and integer and different types of input to an constructor when creating an object to access the different constructors based on the parameter.

**Program:**

****

**Output:**

****

**Class diagram:**

|  |
| --- |
| **Calculatoroverloading** |
| **+ add(int a,int b):int**  **+add(double a,double b):double**  **+ add(int a,int b,int c):int** |

**Errors:**

|  |  |  |
| --- | --- | --- |
| **Sl.No** | **Error name** | **Error Rectification** |
| **1.** | **Semi colon (;)** | **Give the semi colon (;) in each line where it is**  **required** |
| **2.** | **Syntax Error** | **Giving Capital ‘S’ in printing statements (System.out.println)** |

**Program 4:**

**Aim:** Write a Java Program to create a shape class with a method calculateArea() that is overloaded for different shapes(e.g., Square, Rectangle ). Then create a subclass Circle that overrides the

calculateArea() method for a circle.

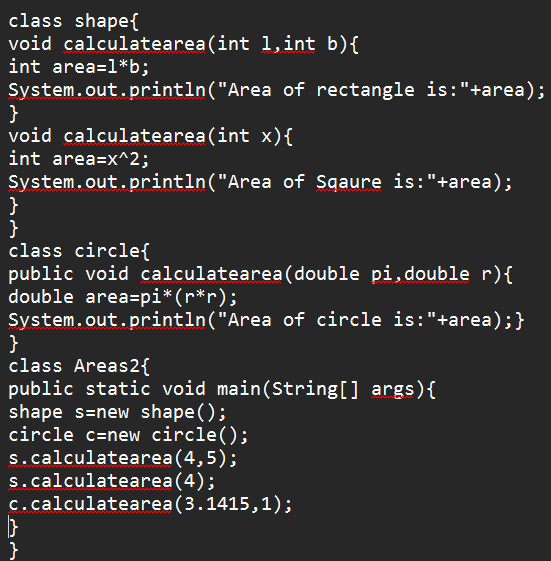
**Important points:**

In this program we use both method overloading and overriding to calculate area of different shapes**.**

**Class Diagram:**

|  |
| --- |
| **shape** |
| **+calculatearea(int l,intb):void**  **+calculatearea(int x):void** |

|  |
| --- |
| **circle** |
| **+void calculatearea(double pi,double r):void** |

**PROGRAM:**

**Output:**

****

**Errors:**

|  |  |  |
| --- | --- | --- |
| **Sl.No** | **Error name** | **Error Rectification** |
| **1.** | **Semi colon (;)** | **Give the semi colon (;) in each line where it is**  **required** |
| **2.** | **Syntax Error** | **Giving Capital ‘S’ in printing statements (System.out.println)** |

# WEEK-7

**1Q)Write a java program to create an abstract class Animal with an abstract method sound().Create Subclass Tiger and Lion extends the Animal class and implement the sound() method to make a specific sound for each animal**

**Program:**

abstract class Animal {

public abstract void sound();

}

class Lion extends Animal {

@Override

public void sound() {

System.out.println("Lion: Roar!");

}

}

class Tiger extends Animal {

@Override

public void sound() {

System.out.println("Tiger: Growl!");

}

}

public class week7 {

public static void main(String[] args) {

Animal lion = new Lion();

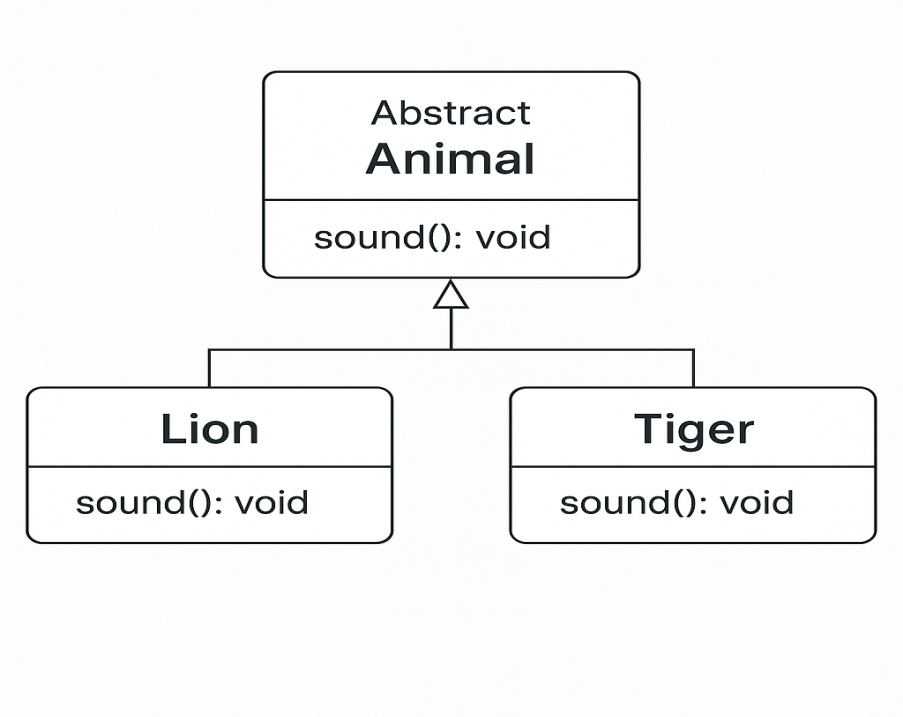
Animal tiger = new Tiger();

lion.sound();

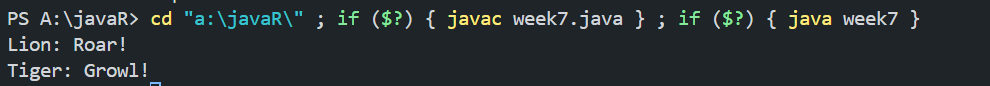
tiger.sound();

}

}

****

**Output:**

****

**Error Table:**

|  |  |  |
| --- | --- | --- |
| S.no | Expected Error | Reason |
| **1** | Setting the parameters inside the constructor | We cannot pass the values inside constructor without setting them first |
| **2** | } | Ending the class and main method is required |

**2Q)Write a java program to create an abstract class Shape3D with an abstract methods Calculate\_volume() and Calculate\_Surface\_area.Create Subclass Sphere and Cube extends the Shape3D class and implement the respective methods to calculate the volume and surface\_area of each shape.**

**Program:**

abstract class Shape3D {

public abstract void calculate\_volume();

public abstract void calculate\_surf\_a();

}

class Sphere extends Shape3D {

private double radius;

public Sphere(double radius) {

this.radius = radius;

}

@Override

public void calculate\_surf\_a() {

double surfaceArea = 4 \* Math.PI \* Math.pow(radius, 2);

System.out.printf("Surface Area of Sphere: %.2f%n", surfaceArea);

}

@Override

public void calculate\_volume() {

double volume = (4.0 / 3) \* Math.PI \* Math.pow(radius, 3);

System.out.printf("Volume of Sphere: %.2f%n", volume);

}

}

class Cube extends Shape3D {

private double side;

public Cube(double side) {

this.side = side;

}

@Override

public void calculate\_surf\_a() {

double surfaceArea = 6 \* Math.pow(side, 2);

System.out.printf("Surface Area of Cube: %.2f%n", surfaceArea);

}

@Override

public void calculate\_volume() {

double volume = Math.pow(side, 3);

System.out.printf("Volume of Cube: %.2f%n", volume);

}

}

public class weekq2 {

public static void main(String[] args) {

Shape3D sphere = new Sphere(5);

Shape3D cube = new Cube(3);

sphere.calculate\_surf\_a();

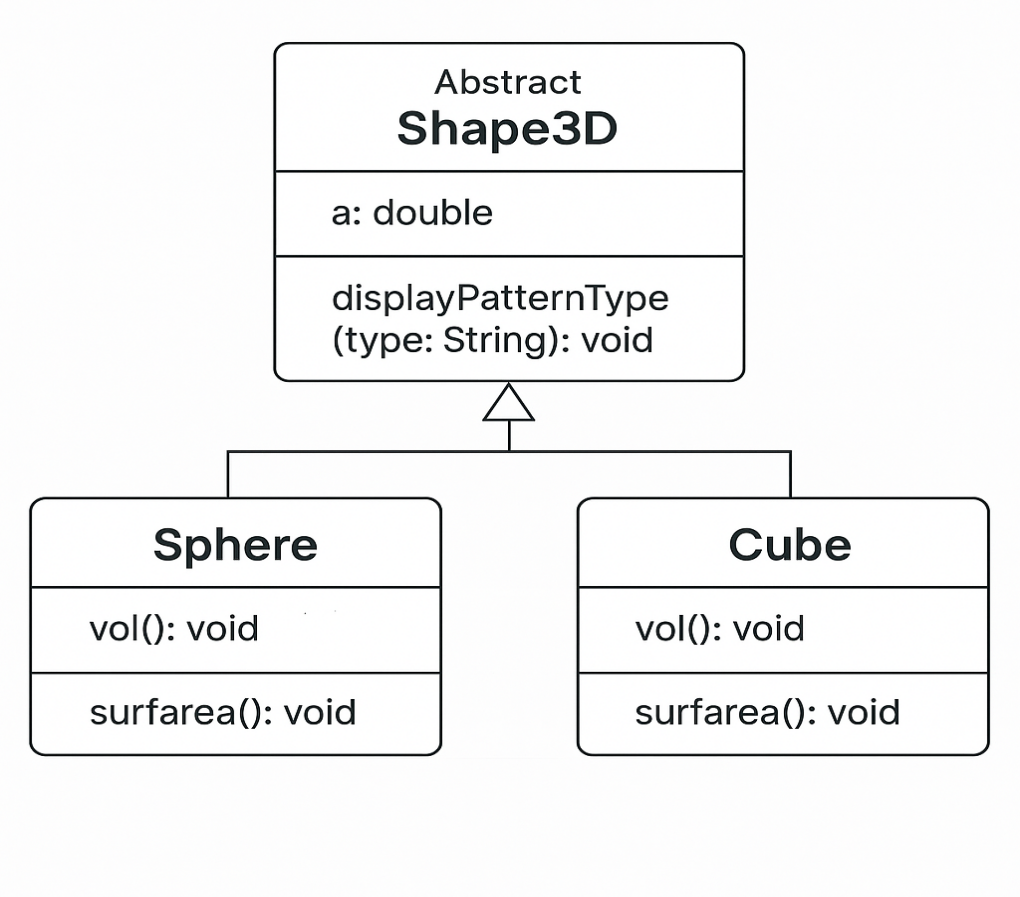
sphere.calculate\_volume();

cube.calculate\_surf\_a();

cube.calculate\_volume();

}

}



**Output:**

****

**Error Table:**

|  |  |  |
| --- | --- | --- |
| S.no | Expected Error | Reason |
| **1** | Setting the parameters inside the constructor | We cannot pass the values inside constructor without setting them first |
| **2** | } | Ending the class and main method is required |

**3Q)Write a java program using an abstract class to define a method for pattern printing**

**-->create an abstract class named patternprinting with an abstract method print pattern (int n) and a concrete method to display the pattern title**

**-->impletment two sub classes**

1. **star pattern**

**Prints a right angled triangle of stars**

1. **Number pattern**

**Prints a right angled triangle of increasing numbers**

**-->in the main()method create objects of both sub classes and print the patterns for a given number of rows**

**Program:**

import java.util.Scanner;

abstract class Pattern {

abstract void printPattern();

}

class DisplayPatterns extends Pattern {

int num;

Scanner sc = new Scanner(System.in);

DisplayPatterns() {

System.out.print("Enter the number of rows: ");

num = sc.nextInt();

}

void printPattern() {

System.out.println("Pattern 1:");

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

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

System.out.print("\* ");

}

System.out.println();

}

System.out.println("Pattern 2:");

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

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

System.out.print(i);

}

System.out.println();

}

}

}

public class patter1 {

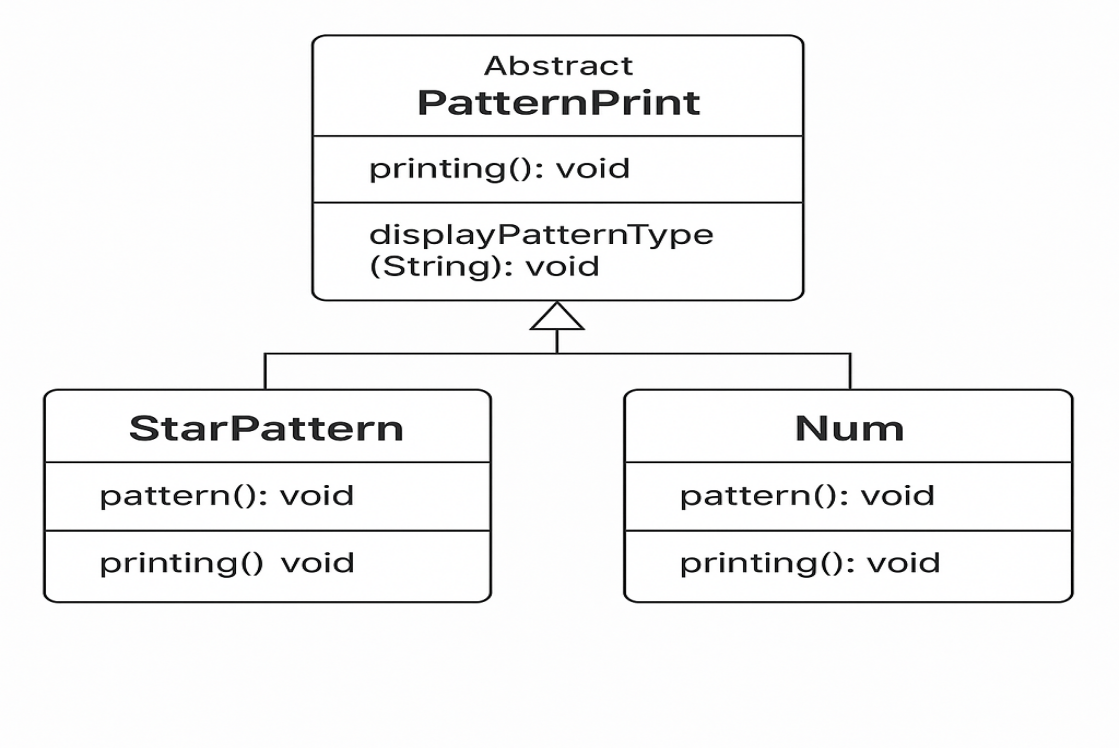
public static void main(String[] args) {

DisplayPatterns dp = new DisplayPatterns();

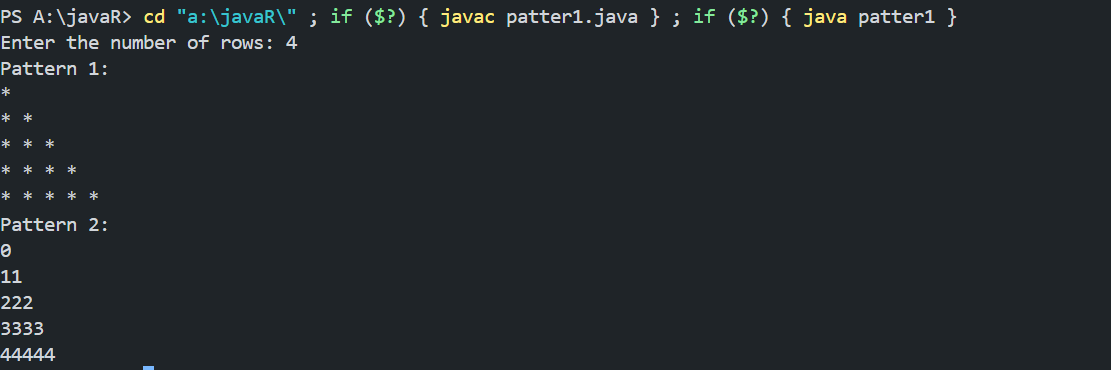
dp.printPattern();

}

}

****

**Output:**

****

**Error Table:**

|  |  |  |
| --- | --- | --- |
| S.no | Expected Error | Reason |
| **1** | Setting the parameters inside the constructor | We cannot pass the values inside constructor without setting them first |
| **2** | } | Ending the class and main method is required |

Week-8

**1Q)write a java program creating an interface Shape with the get perimeter method create 3 classes rectangle,triangleand circle that implements the shape interface ,implement the getperimeter method for each of the three classes**

**Program:**

interface Shape {

    double getPerimeter();

}

class Rectangle implements Shape {

    double length, width;

    Rectangle(double length, double width) {

        this.length = length;

        this.width = width;

    }

    public double getPerimeter() {

        return 2 \* (length + width);

    }

}

class Circle implements Shape {

    double radius;

    Circle(double radius) {

        this.radius = radius;

    }

    public double getPerimeter() {

        return 2 \* Math.PI \* radius;

    }

}

class Triangle implements Shape {

    double side1, side2, side3;

    Triangle(double side1, double side2, double side3) {

        this.side1 = side1;

        this.side2 = side2;

        this.side3 = side3;

    }

    public double getPerimeter() {

        return side1 + side2 + side3;

    }

}

// Main class

public class ass20{

    public static void main(String[] args) {

        System.out.println("Rithwik G, 24121, CSE--B");

        Shape rectangle = new Rectangle(10, 5);

        System.out.println("Rectangle Perimeter: " + rectangle.getPerimeter());

        Shape circle = new Circle(7);

        System.out.println("Circle Perimeter: " + circle.getPerimeter());

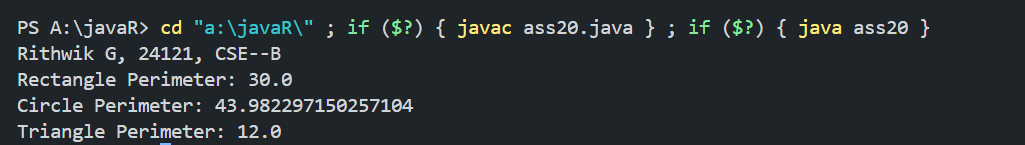
        Shape triangle = new Triangle(3, 4, 5);

        System.out.println("Triangle Perimeter: " + triangle.getPerimeter());

    }

}

**Output:**

****

**Class Diagram:**

**Error Table:**

|  |  |  |
| --- | --- | --- |
| S.no | Expected Error | Reason |
| **1** | Setting the parameters inside the constructor | We cannot pass the values inside constructor without setting them first |
| **2** | } | Ending the class and main method is required |

**2Q)write a java program to create an interface playable with a method play()**

**That takes no arguments and returns void create three classes football,volleyball and basketball that implements the playable and override the play method to play the respective sports**

**Program:**

interface playable{

void play();

}

class volleyball implements playable{

public void play(){

System.out.println("volleyball is a sport");

}

}

class basketball implements playable{

public void play(){

System.out.println("basketbal is also a sport");

}

}

class football implements playable{

public void play(){

System.out.println("football is a sport ");

}

}

public class weeka8 {

public static void main(String[] args) {

playable vb =new volleyball();

playable bb=new basketball();

playable fb=new football();

vb.play();

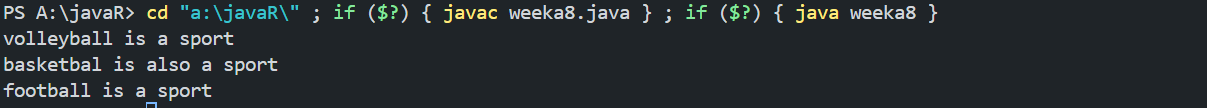
bb.play();

fb.play();

}

}

**Output:**

****

**Class Diagram:**

**Error Table:**

|  |  |  |
| --- | --- | --- |
| S.no | Expected Error | Reason |
| **1** | Setting the parameters inside the constructor | We cannot pass the values inside constructor without setting them first |
| **2** | } | Ending the class and main method is required |

**3Q)write a java program to implement a login system using interfaces**

**Program:**

interface LoginSystem {

boolean login(String id, String password);

}

class UniversityPortal implements LoginSystem {

@Override

public boolean login(String id, String password) {

if (id.equals("student123") && password.equals("pass123")) {

return true;

} else {

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

return false;

}

}

}

public class week8 {

public static void main(String[] args) {

UniversityPortal portal = new UniversityPortal();

boolean loginSuccess1 = portal.login("student123", "pass123");

System.out.println("Login successful: " + loginSuccess1);

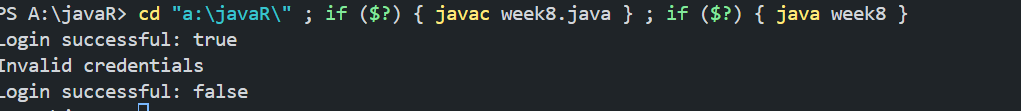
boolean loginSuccess2 = portal.login("student123", "wrong pass");

System.out.println("Login successful: " + loginSuccess2);

}

}

**Output:**

****

**Class Diagram:**

**Error Table:**

|  |  |  |
| --- | --- | --- |
| S.no | Expected Error | Reason |
| **1** | Setting the parameters inside the constructor | We cannot pass the values inside constructor without setting them first |
| **2** | } | Ending the class and main method is required |