**23CSE111**

**OBJECT ORIENTED PROGRAMMING**

**LAB REPORT**



**Department of Computer Science Engineering**

**Amrita School of Computing**

**Amrita Vishwa Vidyapeetham, Amaravati Campus**

**Verified By: Name: T.V. THARUN**

**Roll No: AV.SC.U4CSE24328**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S.No.** | **Experiment**  **Name** | **Page No.** | **Remarks** | **Signature** |
| **1** | **Installation Process of JDK** | **3** |  |  |
| **2** | **Simple Java Program for printing basic details of student** | **5** |  |  |
| **LAB 2** | **Simple Java programs** |  |  |  |
| **1.** | **SI java program** |  |  |  |
| **2.** | **Factorial of a no** |  |  |  |
| **3.** | **Convert temperature from Celsius to fahrenheit** |  |  |  |
| **4.** | **Convert temperature from Fahrenheit to Celsius** |  |  |  |
| **5.** | **Area of rectangle** |  |  |  |
| **6.** | **Program for area of rectangle using herons formula** |  |  |  |
| **7.** | **Program for Fibanocci series** |  |  |  |
| **Lab 3** | **Creating the classes** |  |  |  |
| **1.** | **Creating a class with name car with 4 attributes ,3 methods and 3 objects** |  |  |  |
| **2.** | **Create a class name bank account with methods deposit and withdrawl** |  |  |  |
| **Lab 4** |  |  |  |  |
| **1.** | **java program with class named book the class should contain various attributes such as title author year of publication** |  |  |  |
| **2.** | **java program with class name my class with static variable count of int type initialize to zero and constant variable pi of double initialize to 3.14** |  |  |  |
| **Lab 5** | **Create a calculator using operations including addition, subtraction, multiplication, division using multilevel inheritance and displayed the desired output.** |  |  |  |
|  | **A vechile rental company wants to develop a system that maintains information about different types of vechiles available for rent. The company rents out cars and bikes and they need a program to store details about each vechile such as brand and speed** |  |  |  |
| **Lab 6** | **Write a java program to create a vechile class with method display Info override this method in the car subclass provide specific information about car(car model, company, price,color).** |  |  |  |
|  | **Create a shape class with method calculate area that overloaded for different shapes (square, rectangle ) and create a subclass circle that overrides the calculate area method for circle.** |  |  |  |
|  | **Create a calculator class with overloaded methods to perform addition of 2 integers, addition of 2 doubles, addition of 3 integrs.** |  |  |  |
|  |  |  |  |  |

**WEEK-1**

**Task- 1: Explain the process of Installing JDK (Java Development Kit)**

**Steps:**

1. Download JDK(Java Development Kit)
2. Run the Installer
3. Configure Environment Variables

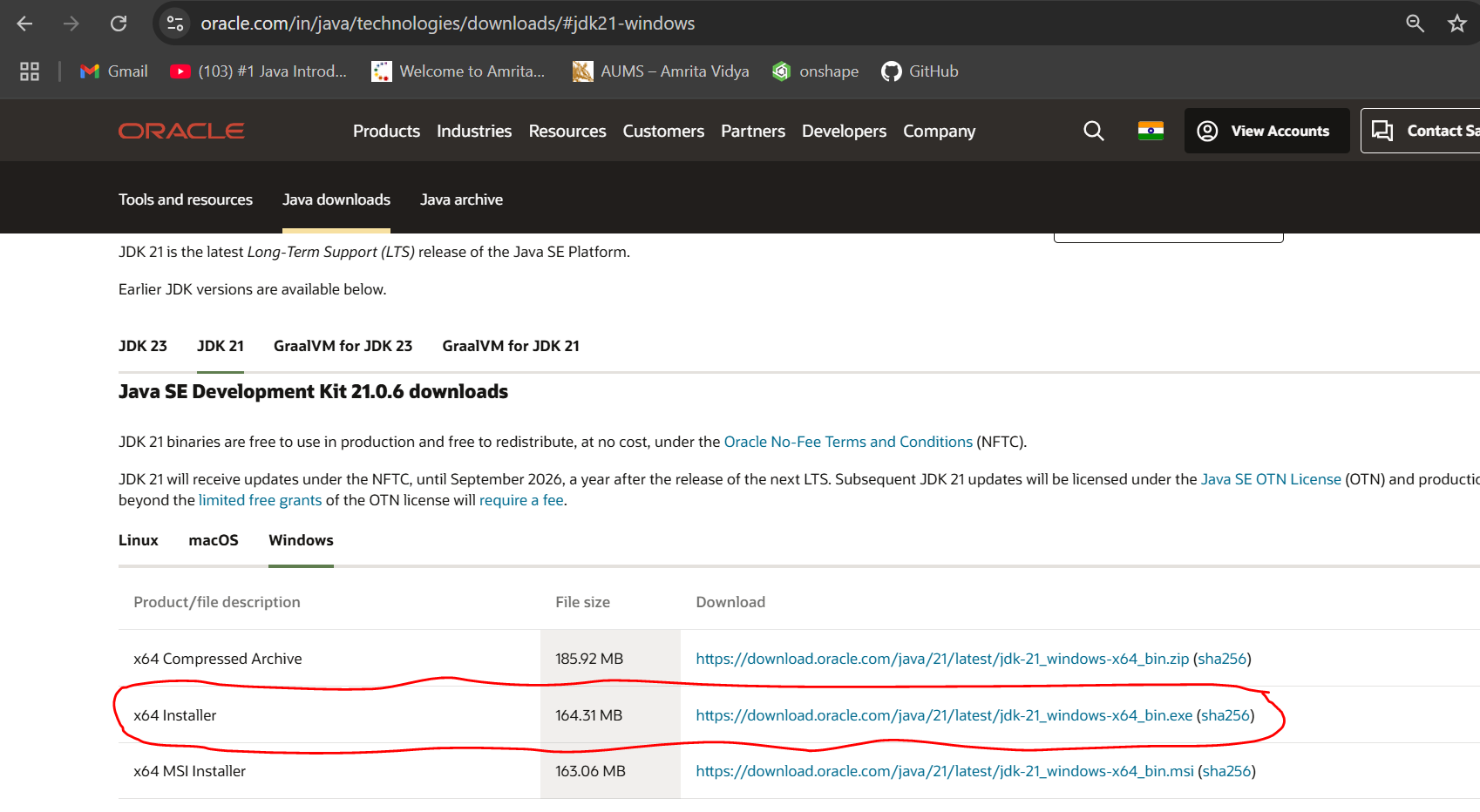
4. Verify Installation

**Step 1: Download JDK**

1. Go to the Oracle JDK download page in your web browser and click on JDK-21 version which is Long term support (LTS) version.

Click on the download link for your operating system (Windows, macOS, or Linux).

2. Choose x64 MSI Installer on the windows tab and click on download link in JDK(21) kit



**Step 2: Run the Installer**

1.Now, go to your downloads folder and run the installer you just downloaded. You will be prompted to the screen installation wizard for java SE development kit.

2. This wizard will guide you through the installation process for java SE development kit so that click on next option.

**Step 3: Configure Environment Variables**

1**. Locate JDK Path**: Navigate through your file explorer to reach the JDK installation directory. Normally, it is located at C:\Program Files\Java\jdk-22\bin. Copy this path.

2.**Access Environment Variables**: Search environment variable on the terminal. In system properties, click on environment variables. You will be prompted to the screen below.



3.**Update the Path Variable:**

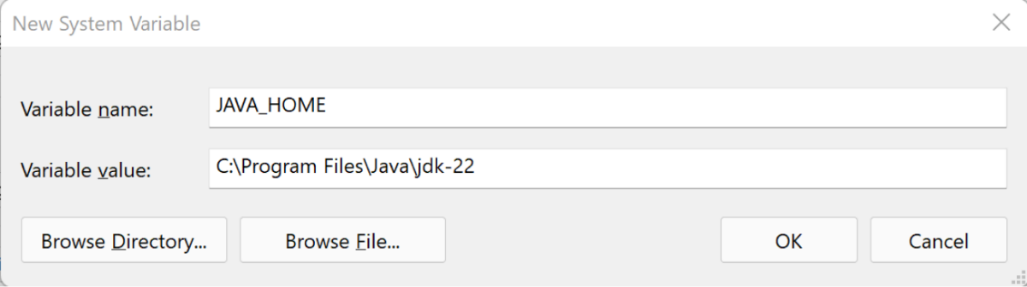
Find the Path variable in the System variables section and click on Edit. Then, click New and paste your JDK bin path (i.e. C:\Program Files\Java\jdk-22\bin).

Finally, click Ok to close each window.

4**.Set JAVA\_HOME Variable:**

Back in the environment variables window, under the system variables section, click New to create a new variable.

Now, name the variable JAVA\_HOME and set its value to the path of the JDK folder directory (i.e.C:\Program Files\Java\jdk-22).

****

Set JAVA\_HOME Variable.Close all the dialogues with the Ok button.

**Step 4: Verify your Installation**

1.After the installation, you can verify whether Java is installed by using the (java –version) in the command prompt.

**Check Version:**

Type java --version and press Enter. Type javac --version and press Enter.



**Task-2:**

**Implement a Java Program to print basic details of a Student**

Write your code in Notepad and execute in cmd prompt

**Important Points**:

* + - * **Understand the syntax of a class.**

class Main {

public static void main(String[] args) {

System.out.println(" ");

}

}

**CODE:**

**The file was saved with Main.java**

class Main

{

public static void main(String[] args)

{

System.out.println("Name: Tharun");

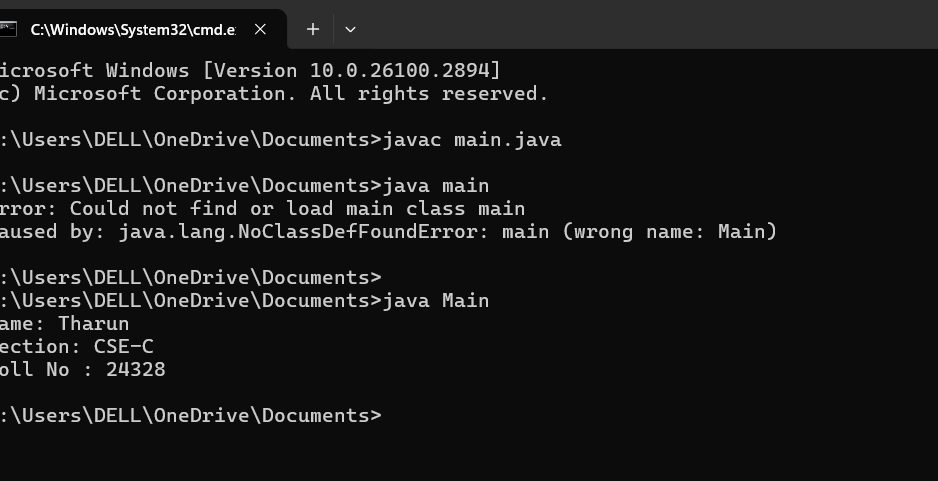
System.out.println("Section: CSE-C");

System.out.println("Roll No : 24328");

}

}

**Execute:**

****

**Errors:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Error Name** | **Error Rectification** |
| **1** | **Syntax/**  **Compilation Error** | **Absence of Semicolon** |
| **2** | **Closing Brackets** | **Need to Close the brackets** |

**LAB2**

**1. Program for SI calculation**

**AIM:** calculation of simple interest

**CODE:**

import java.util.Scanner;

class simpleinterest{

public static void main(String[]args){

Scanner input=new Scanner(System.in);

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

double principal=input.nextDouble();

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

double rateofinterest=input.nextDouble();

System.out.print("Enter the time in years:");

double time=input.nextDouble();

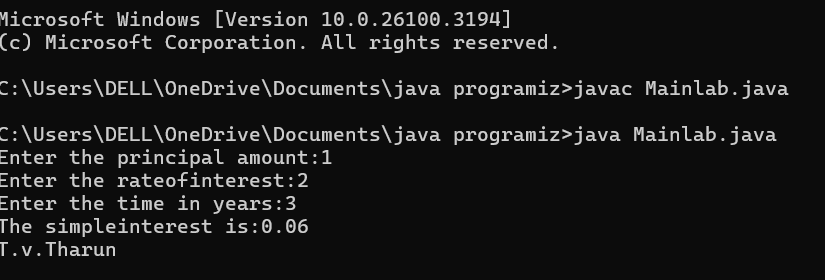
double simpleinterest=(principal\*rateofinterest\*time)/100;

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

System.out.println("T.v.Tharun");

}

}

****

**Errors:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Error Name** | **Error Rectification** |
| **1** | **Syntax/**  **Compilation Error** | **Absence of Semicolon** |
| **2** | **calling a object** | **'Scanner' is a class that parses primitive types and strings** |

**Important Points**:

public static void main(String[]args){

Scanner input=new Scanner(System.in);

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

double principal=input.nextDouble();

**2. Program for Factorial of a no**

**AIM:** calculation pf factorial of a number

**CODE:**

import java.util.Scanner;

class Factorial{

public static void main(String[]args){

Scanner input=new Scanner(System.in);

System.out.print("Enter a number:");

int number= input.nextInt();

System.out.println("The factorial of"+ number+" is:"+factorial(number));

}

public static factorial(int n){

fact=1;

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

fact\*=i;

}

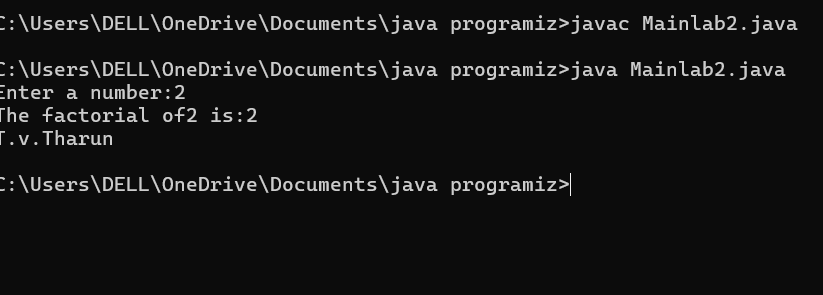
return fact;

System.out.println("T.v.Tharun");

}

}

**Execution:**



**Errors:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Error Name** | **Error Rectification** |
| **1** | **Syntax/**  **Compilation Error** | **Absence of Semicolon** |
| **2** | **Closing Brackets** | **Need to Close the brackets** |

**Important Points**:

public static factorial(int n){

fact=1;

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

fact\*=i;

}

**3. Convert temperature from Celsius to Fahrenheit**

**AIM:** calculation of temperature from celsius to fahrenheit

**CODE:**

import java.util.Scanner;

class CelsiusToFahrenheit{

public static void main(String[]args){

Scanner input=new Scanner(System.in);

System.out.print("Enter temperature in celsius:");

double Celsius = input.nextDouble();

double fahrenheit =(Celsius\*9/5)+32;

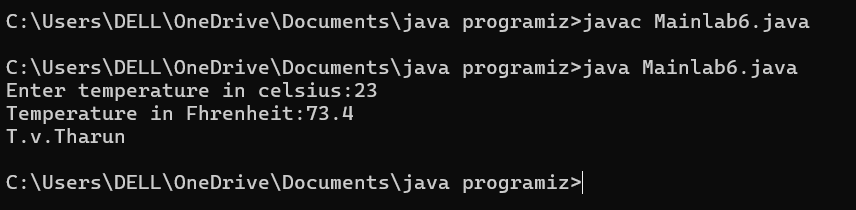
System.out.println("Temperature in Fhrenheit:"+fahrenheit);

System.out.println("T.v.Tharun");

}

}

**Execution:**

****

**Errors:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Error Name** | **Error Rectification** |
| **1** | **Syntax/**  **Compilation Error** | **Absence of dot** |
| **2** | **Double the variable** | **We should use capital d in “nextDouble”** |

**Important Points**:

Scanner input=new Scanner(System.in);

System.out.print("Enter temperature in celsius:");

double Celsius = input.nextDouble();

double fahrenheit =(Celsius\*9/5)+32;

**4. Convert temperature from Fahrenheit to Celsius**

**AIM:** calculation of temperature from Fahrenheit to Celsius

**CODE:**

import java.util.Scanner;

class FahrenheitToCelsius{

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;

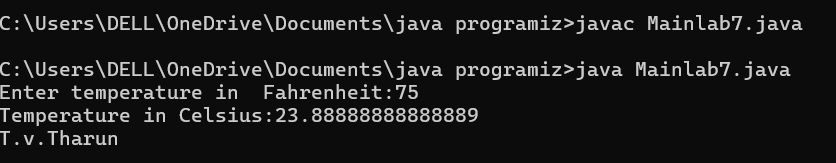
System.out.println("Temperature in Celsius:"+Celsius);

System.out.println("T.v.Tharun");

}

}

**Execution:**



**Errors:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Error Name** | **Error Rectification** |
| **1** | **Syntax/**  **Compilation Error** | **Absence of dot** |
| **2.** | **Closing Brackets** | **Need to Close the brackets** |

**Important Points**:

Scanner input=new Scanner(System.in);

System.out.print("Enter temperature in Fahrenheit:");

double Fahrenheit = input.nextDouble();

double Celsius =( Fahrenheit-32)\*5/9;

**5. Program for Area of rectangle**

**AIM:** calculation of area of rectangle

**CODE:**

import java.util.Scanner;

class RectangleArea{

public static void main(String[]args){

Scanner input=new Scanner(System.in);

System.out.println("Enter the length of rectangle:");

double length=input.nextDouble();

System.out.println("Enter the Width of rectangle:");

double Width=input.nextDouble();

double area=length\*Width;

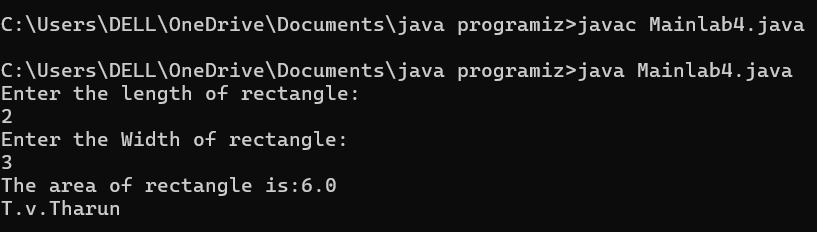
System.out.println("The area of rectangle is:"+ area);

System.out.println("T.v.Tharun");

}

}

**Execution:**

****

**Errors:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Error Name** | **Error Rectification** |
| **1** | **Syntax/**  **Compilation Error** | **Absence of semicolon and dot** |

**Important Points**:

public static void main(String[]args){

Scanner input=new Scanner(System.in);

System.out.println("Enter the length of rectangle:");

double length=input.nextDouble();

System.out.println("Enter the Width of rectangle:");

double Width=input.nextDouble();

double area=length\*Width;

**6.Program for Area of triangle**

**AIM:** calculation of area of triangle using heron’s formula

**CODE:**

import java.util.Scanner;

class TriangleArea{

public static void main(String[]args){

Scanner input=new Scanner(System.in);

System.out.print("Enter the length of side a :");

double a= input.nextDouble();

System.out.print("Enter the length of side b :");

double b= input.nextDouble();

System.out.print("Enter the length of side c:");

double c= input.nextDouble();

double s=(a+b+c)/2;

double area= Math.sqrt(s\*(s-a)\*(s-b)\*(s-c));

System.out.println("T.v.Tharun");

if(Double.isNaN(area)) {

System.out.println("The entered sides do not form a valid triangle.");

}else {

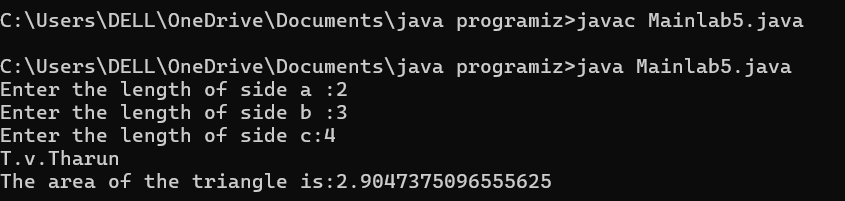
System.out.println("The area of the triangle is:"+area);

}

}

}

**EXECUTION:**

****

**Errors:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Error Name** | **Error Rectification** |
| **1** | **Syntax/**  **Compilation Error** | **Absence of dot** |
| **2** | **Absence of Math.sqrt** | **We should use Math.sqrt** |

**Important Points**:

double s=(a+b+c)/2;

double area= Math.sqrt(s\*(s-a)\*(s-b)\*(s-c));

System.out.println("T.v.Tharun");

if(Double.isNaN(area)) {

System.out.println("The entered sides do not form a valid triangle.");

}else {

System.out.println("The area of the triangle is:"+area);

**7. Program for Fibanocci series**

**AIM:** program for fibanocci series

**CODE:**

class FibanocciSeries{

public static void main(String[]args){

int n=100;

int first=0,second=1;

System.out.println("Fibanocci Series up to "+n+" terms:");

System.out.println("T.v.Tharun");

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

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

int next=first + second;

first= second;

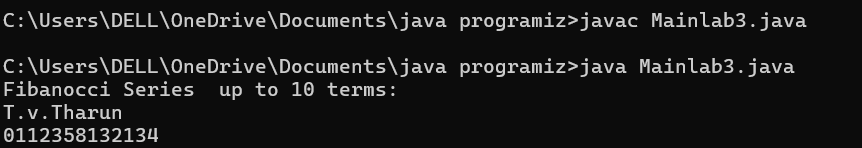
second= next;

}

}

}

**EXECUTION:**

****

**Errors:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Error Name** | **Error Rectification** |
| **1** | **Syntax/**  **Compilation Error** | **Absence of dot** |
| **2** | **Absence of semicolon in output** | **We need to keep the semicolon** |

**Important Points**:

int n=100;

int first=0,second=1;

System.out.println("Fibanocci Series up to "+n+" terms:");

System.out.println("T.v.Tharun");

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

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

int next=first + second;

first= second;

}

}

**LAB3**

**AIM:** Creating a class with name car with 4 attributes ,3 methods and 3 objects

**CODE:**

class Car {

private String color;

private String brand;

private String fuelType;

private int mileage;

public Car(String color, String brand, String fuelType, int mileage) {

this.color = color;

this.brand = brand;

this.fuelType = fuelType;

this.mileage = mileage;

}

public void start() {

System.out.println("The " + color + " " + brand + " is starting.");

}

public void stop() {

System.out.println("The " + color + " " + brand + " is stopping.");

} public void service() {

System.out.println("The " + color + " " + brand + " requires service.");

}

public static void main(String[] args) {

Car car1 = new Car("Red", "Toyota", "Petrol", 15000);

Car car2 = new Car("Blue", "Honda", "Diesel", 20000);

Car car3 = new Car("White", "Tesla", "Electric", 25000);

car1.start();

car1.stop();

car1.service();

car2.start();

car2.stop();

car2.service();

car3.start();

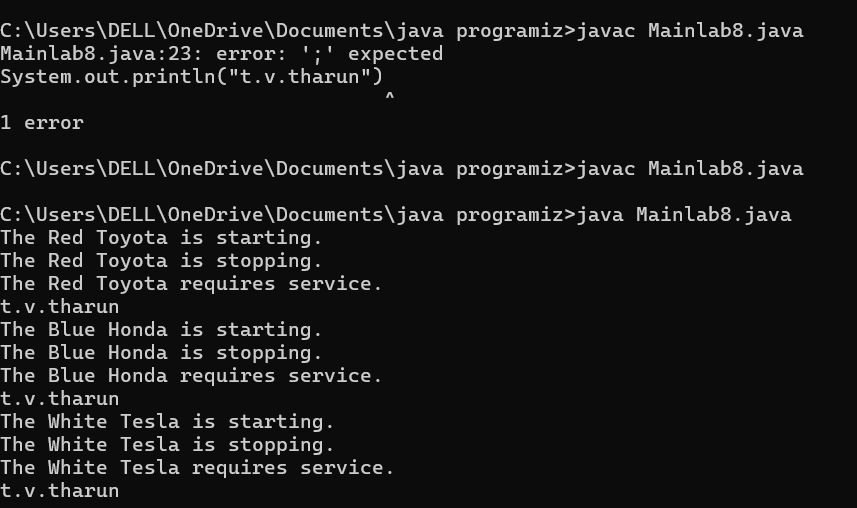
car3.stop();

car3.service();

}

}

**Execution:**



**Errors:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Error Name** | **Error Rectification** |
| **1** | **Error:duplicate**  **class:car** | **There are two classes of same name so that change the other class name to different name** |
| **2** | **Absence of semicolon in output** | **We need to keep the semicolon** |

**Important points:**

public Car(String color, String brand, String fuelType, int mileage) {

this.color = color;

this.brand = brand;

this.fuelType = fuelType;

this.mileage = mileage;

}

public void start() {

System.out.println("The " + color + " " + brand + " is starting.");

}

public void stop() {

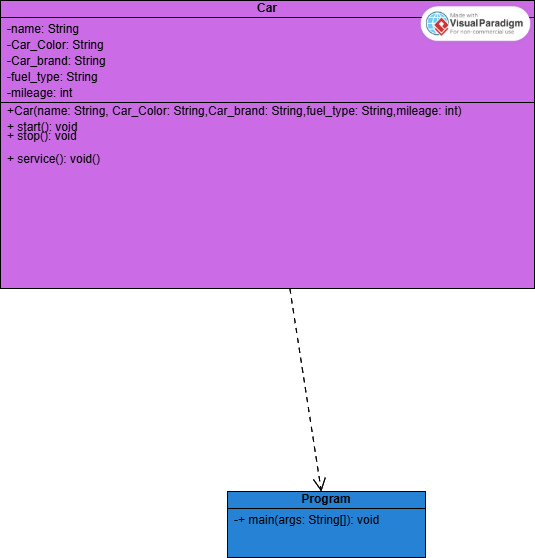
System.out.println("The " + color + " " + brand + " is stopping.");

} public void service() {

System.out.println("The " + color + " " + brand + " requires service.");

}

**Class diagram:**

****

**2. AIM:** Create a class name bank account with methods deposit and withdrawl

**CODE:**

import java.util.Scanner;

class BankAccount {

private int accountNumber;

private String accountHolder;

private float currentBalance;

public BankAccount(int accountNumber, String accountHolder, float currentBalance) {

this.accountNumber = accountNumber;

this.accountHolder = accountHolder;

this.currentBalance = currentBalance;

}

public void deposit(int amount) {

Scanner input = new Scanner(System.in);

System.out.println("Enter the amount to deposit:");

float deposit = input.nextFloat();

currentBalance += deposit;

System.out.println("Current Balance: " + currentBalance);

}

public void withdraw(int amount) {

Scanner input = new Scanner(System.in);

System.out.println("Enter the amount to withdraw:");

float withdrawal = input.nextFloat();

if (withdrawal <= currentBalance) {

currentBalance -= withdrawal;

System.out.println("Current Balance: " + currentBalance);

} else {

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

}

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

BankAccount BA = new BankAccount(1534210, "T.V.Tharun", 100000);

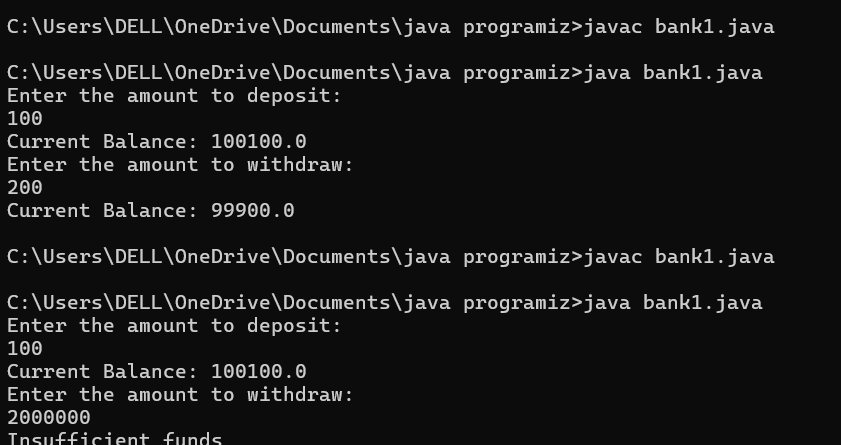
BA.deposit(5000);

BA.withdraw(3000);

}

}

**Execution:**

****

**Errors:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Error Name** | **Error Rectification** |
| **1** | **The scanner object is not closed** | **Close the scanner after use with input.close()** |
| **2.** | **The amount parameter in deposit and withdraw is not used** | **refactor the methods to directly use the amount parameter instead of tanking user input** |
| **3** | **Absence of semicolon in output** | **We need to keep the semicolon and no need of dot** |

**Important points:**

public void deposit(int amount) {

Scanner input = new Scanner(System.in);

System.out.println("Enter the amount to deposit:");

float deposit = input.nextFloat();

currentBalance += deposit;

System.out.println("Current Balance: " + currentBalance);

}

public void withdraw(int amount) {

Scanner input = new Scanner(System.in);

System.out.println("Enter the amount to withdraw:");

float withdrawal = input.nextFloat();

if (withdrawal <= currentBalance) {

currentBalance -= withdrawal;

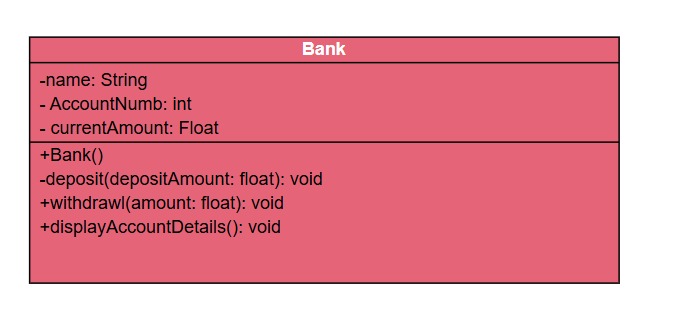
System.out.println("Current Balance: " + currentBalance);

} else {

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

}

**Class diagram:**



**LAB 4**

**1.AIM:** java program with class named book the class should contain various attributes such as title author year of publication

**CODE:**

class Book {

private String title;

private String author;

private int yearOfPublication;

public Book(String title, String author, int yearOfPublication) {

this.title = title;

this.author = author;

this.yearOfPublication = yearOfPublication;

}

public String getTitle() {

return title;

}

public String getAuthor() {

return author;

} public int getYearOfPublication() {

return yearOfPublication;

}

public void displayBookDetails() {

System.out.println("Title: " + title);

System.out.println("Author: " + author);

System.out.println("Year of Publication: " + yearOfPublication);

}

public static void main(String[] args) {

Book book1 = new Book("The Great Gatsby", "F. Scott Fitzgerald", 1925);

book1.displayBookDetails();

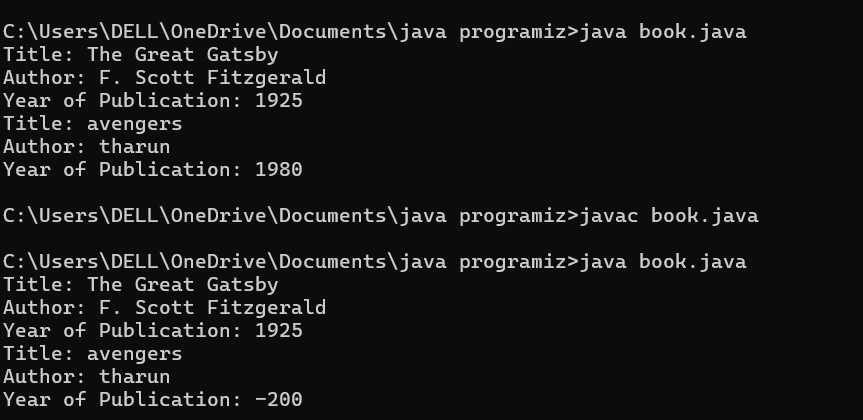
Book book2=new Book ("avengers","tharun",1980);

book2.displayBookDetails();

}

}

**Execution:**



**Errors:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Error Name** | **Error Rectification** |
| **1** | **Execption in thread “main” java.lang.Error:unresolved complication problem:**  **At Book.main(Book.java:20)** | **Not use space in the name variable** |
| **2** | **The constructor book(int,string,int) is not defined** | **We have to use same specified datatype as in constructor while making object** |

**IMPORTANT POINTS:**

public Book(String title, String author, int yearOfPublication) {

this.title = title;

this.author = author;

this.yearOfPublication = yearOfPublication;

}

public String getTitle() {

return title;

}

public String getAuthor() {

return author;

} public int getYearOfPublication() {

return yearOfPublication;

}

public void displayBookDetails() {

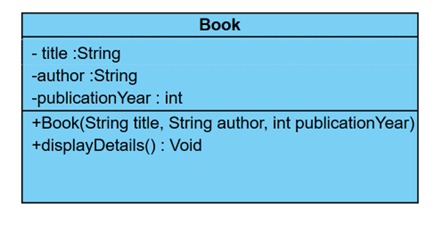
System.out.println("Title: " + title);

System.out.println("Author: " + author);

System.out.println("Year of Publication: " + yearOfPublication);

}

**Class diagram:**



**2. AIM:** java program with class name my class with static variable count of int type initialize to zero and constant variable pi of double initialize to 3.14

**CODE:**

class MyClass {

static int count = 0;

final double PI = 3.14159;

public MyClass() {

count++;

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

MyClass obj1 = new MyClass();

MyClass obj2 = new MyClass();

MyClass obj3 = new MyClass();

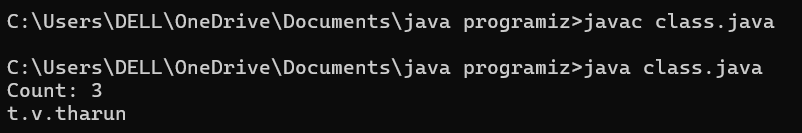
System.out.println("Count: " + count);

System.out.println("t.v.tharun");

}

}

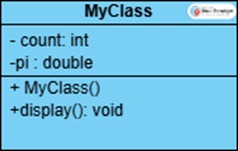
**Execution:**

****

**Errors:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Error Name** | **Error Rectification** |
| **1** | **The static int count variable is shared among all instances of my class** | **Count will be updated across all instances** |
| **2** | **Absence of semicolon in output** | **We need to keep the semicolon and no need of dot** |

Class diagram:



**Important points:**

public MyClass() {

count++;

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

MyClass obj1 = new MyClass();

MyClass obj2 = new MyClass();

MyClass obj3 = new MyClass();

System.out.println("Count: " + count);

System.out.println("t.v.tharun");

**LAB 5**

**AIM:** Create a calculator using operations including addition, subtraction, multiplication, division using multilevel inheritance and displayed the desired output.

**CODE:**

class Calculator {

public int add(int a, int b) {

return a + b;

}

}

class SubCalculator extends Calculator {

public int subtract(int a, int b) {

return a - b;

}

}

class MultiCalculator extends SubCalculator {

public int multiply(int a, int b) {

return a \* b;

}

}

class FinalCalculator extends MultiCalculator {

public double divide(int a, int b) {

if (b != 0) {

return (double) a / b;

} else {

System.out.println("Error: Division by zero!");

return Double.NaN;

}

}

}

class Calci{

public static void main(String[] args) {

FinalCalculator cal = new FinalCalculator();

int a = 15;

int b=3;

System.out.println("Addition: " + cal.add(a, b));

System.out.println("Subtraction: " + cal.subtract(a, b));

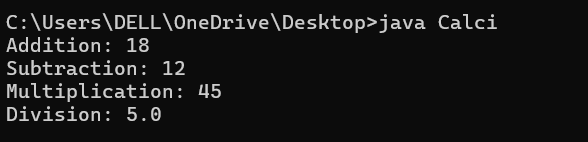
System.out.println("Multiplication: " + cal.multiply(a, b));

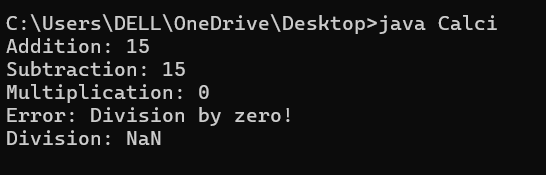
System.out.println("Division: " + cal.divide(a, b));

}

}

**EXECUTION:**

****

****

**ERRORS:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Error Name** | **Error Rectification** |
| **1** | **Compile-Time Error** | **Forgetting to cast int to double in division (integer division truncation).**  **Use (double) a / b** |
| **2** | **Absence of semicolon in output** | **We need to keep the semicolon and no need of dot** |

**IMPORTANT POINTS:**

class FinalCalculator extends MultiCalculator {

public double divide(int a, int b) {

if (b != 0) {

return (double) a / b;

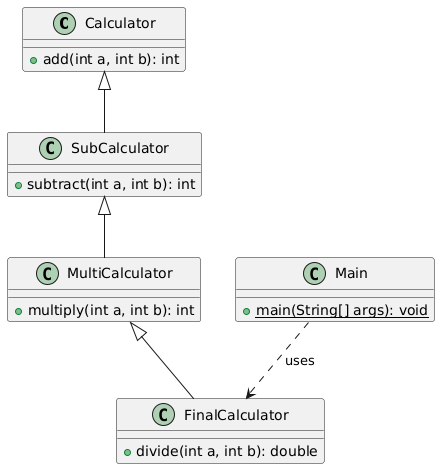
} else {

System.out.println("Error: Division by zero!");

return Double.NaN;

}

**CLASS DIAGRAM:**

****

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

**CODE:**

class Vehicle{

String brand;

int speed;

Vehicle(String brand,int speed){

this.brand=brand;

this.speed=speed;

}

void Details(){

System.out.println("Brand:"+brand);

System.out.println("Speed:"+speed);

}

}

class CARS extends Vehicle{

int doors;

int capacity;

public CARS(String brand,int speed,int doors,int capacity){

super(brand, speed);

this.doors=doors;

this.capacity=capacity;

}

void cardetails(){

System.out.println("Number of doors:"+doors);

System.out.println("Capacity:"+capacity);

}

}

class Bikes extends Vehicle{

Boolean gears;

Bikes(String brand,int speed,Boolean gears){

super(brand, speed);

this.gears=gears;

}

void bikedetails(){

if (gears==true)

System.out.println("This bike has gears.");

else

System.out.println("This bike does not have gear system.");

}

}

class Trucks extends Vehicle{

int tons;

Trucks(String brand,int speed,int tons){

super(brand, speed);

this.tons=tons;

}

void truckdetails(){

System.out.println("The capacity of truck is: "+tons);

}

}

class Rent{

public static void main(String[] args){

CARS c=new CARS("Tayota",120,5,5);

c.cardetails();

c.Details();

Bikes b=new Bikes("KTM",80,true);

b.bikedetails();

b.Details();

Trucks t=new Trucks("TATA",100,1);

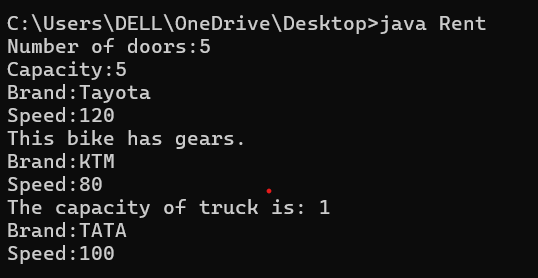
t.truckdetails();

t.Details();

}

}

**EXECUTION**:



**ERRORS:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Error Name** | **Error Rectification** |
| **1** | **Compile-Time Error** | **Using Boolean (object) instead of boolean (primitive) for hasGears.**  **Change to boolean** |
| **2** | **Logical Error** | **Incorrect speed unit assumption (e.g., mph vs. km/h).** **Clarify unit in output** |

**IMPORTANT POINTS:**

class Bikes extends Vehicle{

Boolean gears;

Bikes(String brand,int speed,Boolean gears){

super(brand, speed);

this.gears=gears;

}

void bikedetails(){

if (gears==true)

System.out.println("This bike has gears.");

else

System.out.println("This bike does not have gear system.");

}

}

class Trucks extends Vehicle{

int tons;

Trucks(String brand,int speed,int tons){

super(brand, speed);

this.tons=tons;

}

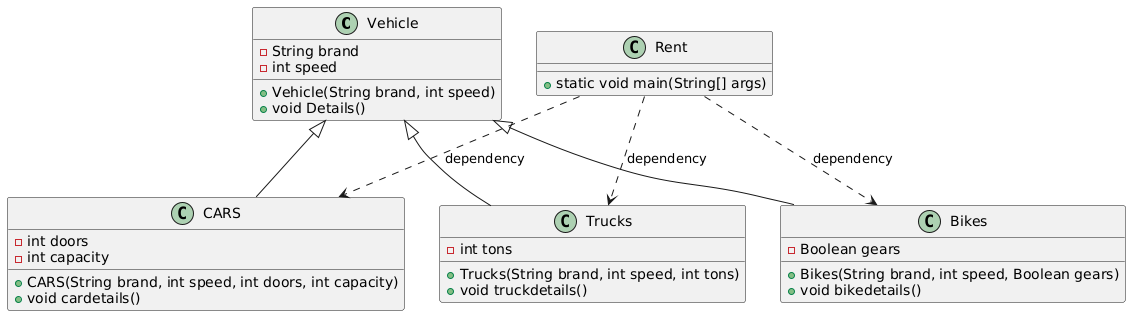
void truckdetails(){

System.out.println("The capacity of truck is: "+tons);

}

}

**CLASS DIAGRAM:**

****

**LAB 6**

**AIM:** Write a java program to create a vechile class with method display Info override this method in the car subclass provide specific information about car(car model, company, price,color).

**CODE:**

class Vehicle {

public void displayInfo() {

System.out.println("This is a generic vehicle");

}

}

class Car extends Vehicle {

private String model;

private String company;

private double price;

private String color;

public Car(String model, String company, double price, String color) {

this.model = model;

this.company = company;

this.price = price;

this.color = color;

}@Override

public void displayInfo() {

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

System.out.println("Model: " + model);

System.out.println("Company: " + company);

System.out.println("Price: $" + price);

System.out.println("Color: " + color);

}

}

class Vechile {

public static void main(String[] args) {

Vehicle genericVehicle = new Vehicle();

genericVehicle.displayInfo();

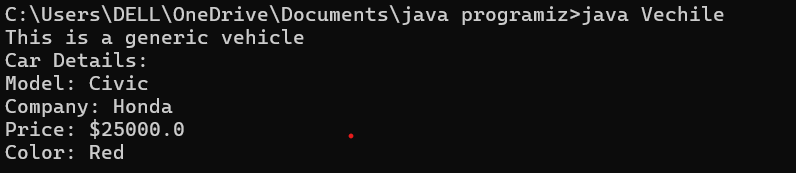
Car myCar = new Car("Civic", "Honda", 25000.0, "Red");

myCar.displayInfo();

}

}

**EXECUTION:**

****

**ERRORS:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Error Name** | **Error Rectification** |
| **1** | **Compilation Error** | **Main class is named "Vechile" while original class is "Vehicle".** **Rename main class to "Vehicle" to match filename or correct the spelling** |
| **2** | **Logical Error** | **Class "Car" follows proper case but "Vechile" doesn't.** **Maintain consistent naming** |

**IMPORTANT POINTS:**

public void displayInfo() {

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

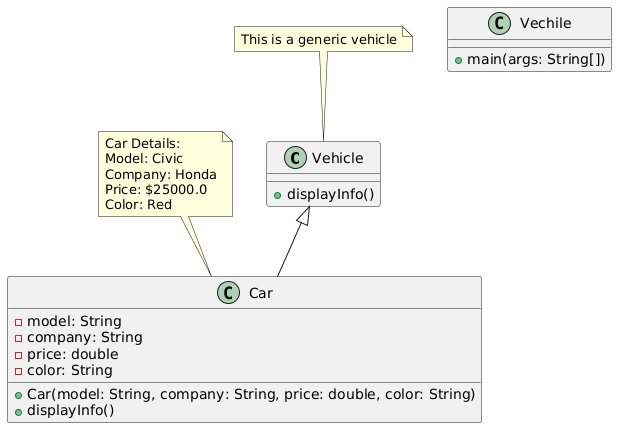
System.out.println("Model: " + model);

System.out.println("Company: " + company);

System.out.println("Price: $" + price);

System.out.println("Color: " + color);}

**CLASS DIAGRAM:**

****

**AIM:** Create a shape class with method calculate area that overloaded for different shapes (square, rectangle ) and create a subclass circle that overrides the calculate area method for circle.

**CODE:**

class Shape {

public double area(double side) {

return side \* side;

}

public double area(double length, double width) {

return length \* width;

}

public double area(double radius, String shape) {

if (shape.equalsIgnoreCase("circle")) {

return Math.PI \* radius \* radius;

}

return 0;

}

}

class Circle extends Shape {

private double radius;

public Circle(double radius) {

this.radius = radius;

}

@Override

public double area(double radius, String shape) {

return Math.PI \* radius \* radius;

}

public double circumference() {

return 2 \* Math.PI \* radius;

}

}

class Area {

public static void main(String[] args) {

Shape shape = new Shape();

System.out.println("Square area: " + shape.area(5));

System.out.println("Rectangle area: " + shape.area(4, 6));

System.out.println("Circle area: " + shape.area(3, "circle"));

Circle circle = new Circle(3);

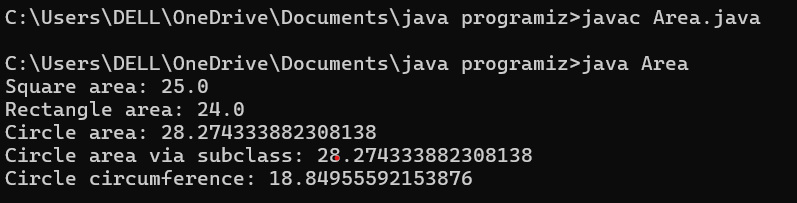
System.out.println("Circle area via subclass: " + circle.area(3, "circle"));

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

}

}

**EXECUTION:**

****

**ERRORS:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Error Name** | **Error Rectification** |
| **1** | **Compilation Error** | **Typo in method name (e.g., Area instead of area).** **Ensure correct method name casing (area).** |
| **2** | **Logical Error** | **Using wrong formula (e.g., return radius \* radius for circle).** **Use correct formula (Math.PI \* radius \* radius).** |

**IMPORTANT POINTS:**

@Override

public double area(double radius, String shape) {

return Math.PI \* radius \* radius;

}

public double circumference() {

return 2 \* Math.PI \* radius;

}

}

class Area {

public static void main(String[] args) {

Shape shape = new Shape();

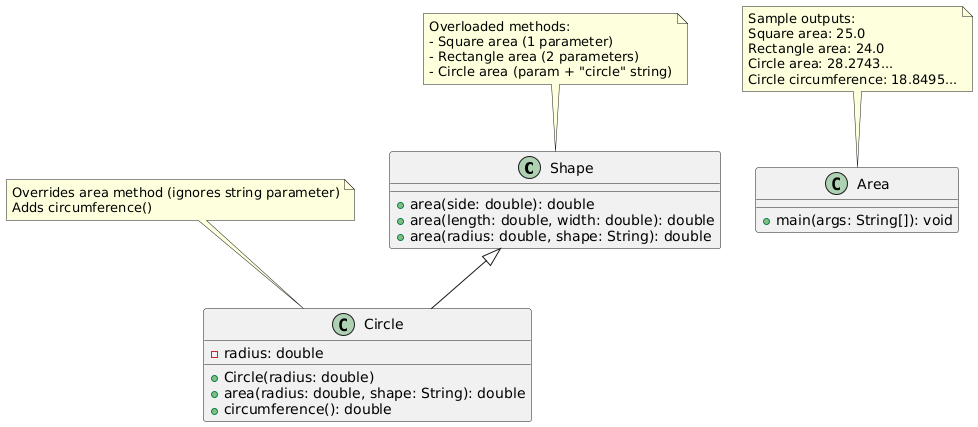
System.out.println("Square area: " + shape.area(5));

System.out.println("Rectangle area: " + shape.area(4, 6));

System.out.println("Circle area: " + shape.area(3, "circle"));

Circle circle = new Circle(3);

**CLASS DIAGRAM:**

****

**AIM:** Create a calculator class with overloaded methods to perform addition of 2 integers, addition of 2 doubles, addition of 3 integrs.

**CODE:**

class Calculator {

public int add(int a, int b) {

return a + b;

}

public double add(double a, double b) {

return a + b;

}

public int add(int a, int b, int c) {

return a + b + c;

}

public static void main(String[] args) {

Calculator calc = new Calculator();

System.out.println("Sum of 5 and 10 (integers): " + calc.add(5, 10));

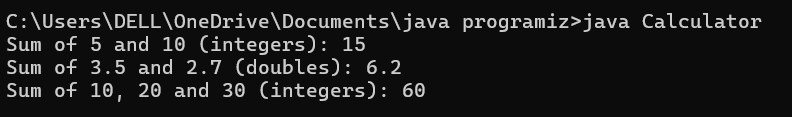
System.out.println("Sum of 3.5 and 2.7 (doubles): " + calc.add(3.5, 2.7));

System.out.println("Sum of 10, 20 and 30 (integers): " + calc.add(10, 20, 30));

}

}

**EXECUTION:**

****

**ERRORS:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Error Name** | **Error Rectification** |
| **1** | **Compilation Error** | **Typo in method name (e.g., Add instead of add).** **Ensure correct method name casing (add)** |
| **2** | **Logical Error** | **Misplaced parameters (e.g., add(b, a) instead of add(a, b)).** **Verify parameter order.**   |  | | --- | |  | |  |  |  |  | |

**IMPORTANT POINTS:**

Calculator calc = new Calculator();

System.out.println("Sum of 5 and 10 (integers): " + calc.add(5, 10));

System.out.println("Sum of 3.5 and 2.7 (doubles): " + calc.add(3.5, 2.7));

System.out.println("Sum of 10, 20 and 30 (integers): " + calc.add(10, 20, 30));

}

}

**CLASS DIAGRAM:**

**AIM:** A college is developing an automated admission system that verifies students eligibility for ug and pg programs. Each program has different eligibelity criteria based on student percentage in their previous qualification.UG admission require min of 60% and PG admissio require min of 70%

**CODE:**

class Student{

String name;

double percentage;

public Student(String name,double percentage){

this.name=name;

this.percentage=percentage;}

public void geteligibility(){

System.out.println(name+"must meet the eligibility requirements");}

}

class UG extends Student{

public UG(String name,double percentage){

super(name,percentage);}

public void geteligibility(){

if (percentage>=60){

System.out.println("person is eligible for ug");}

else{

System.out.println("person is not eligible for ug");}

}

}

class PG extends Student{

public PG(String name,double percentage){

super(name,percentage);}

public void geteligibility(){

if (percentage>=70){

System.out.println("person is eligible for pg");}

else{

System.out.println("person is not eligible for pg");}

}

}

public class Eligibility{

public static void main(String[] args){

Student s=new Student("Marco",79.9);

s.geteligibility();

UG u=new UG("praneeth",84);

u.geteligibility();

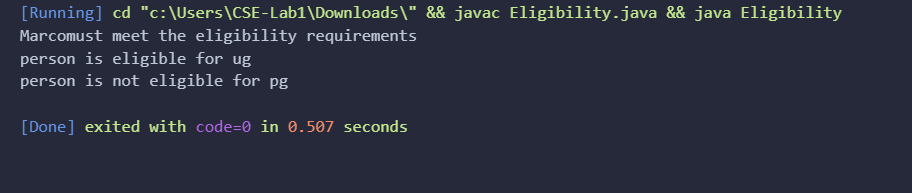
PG p=new PG("tharun",percentage:60);

p.geteligibility();

}

}

**EXECUTION:**



**ERRORS:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Error Name** | **Error Rectification** |
| **1** | **Naming Convention Violation** | **Method geteligibility() should follow camelCase (getEligibility()).Rename to getEligibility() for Java conventions.** |
| **2** | **Logical Error** | **Passing 60 (int) where double is expected, may cause loss of precision.** **Use 60.0 or (double)60 to ensure correct type.**   |  | | --- | |  | |

**IMPORTANT POINTS:**

class UG extends Student{

public UG(String name,double percentage){

super(name,percentage);}

public void geteligibility(){

if (percentage>=60){

System.out.println("person is eligible for ug");}

else{

System.out.println("person is not eligible for ug");}

}

}

class PG extends Student{

public PG(String name,double percentage){

super(name,percentage);}

public void geteligibility(){

if (percentage>=70){

System.out.println("person is eligible for pg");}

else{

System.out.println("person is not eligible for pg");}

}

}

**CLASS DIAGRAM:**

