**23CSE111**

OOPS

(Object Oriented Programming System)

**LAB MANUAL**

A logo with pink letters

Description automatically generated

**Department of CSE**

**Amrita School of Engineering**

**Amrita Vishwa Vidyapeetham, Amaravati Campus**

**Verified By : Name: D. Ganesh Reddy**

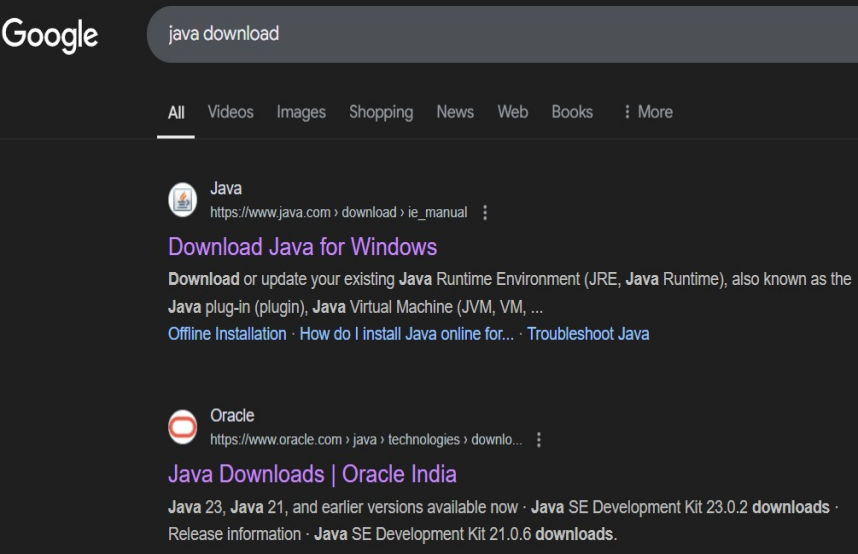
**Rajkumar Sir Class: CSE-A**

**Roll No: 24042**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| S.NO | Programs | Date | Pg:No | Signature |
| 1 | 1. Download and Install Java Software. 2. Write a java program to print message “Welcome to java programming”. 3. Write a java program that prints name, roll number, section of a student. |  |  |  |
| 2 | 1. java program to calculate the area of a rectangle 2. java program to convert temp from Celsius to   Fahrenheit and vice versa.   1. java program to calculate the simple interest 2. java program to find the largest of three numbers using ternary operators. 3. java program to find the factorial of the given number. |  |  |  |
| 3 | 1. Write a java program with the following instructions.  a) Create a class with name car.  b) Create four attributes named car\_colour,car\_brand,fuel  type,top\_speed.  c) Create three method named  “Start\_Racing”,”End\_Race”.{ }  d) Create three objects named Car1,Car2,Car3.  e) Create a constructor which should print “Welcome to car  garage”.  2. Write a class by writing java program named Bank Account  with two methods “deposits and withdraw”.  a) In deposit method whenever an amount is deposited it  has to be updated with current amount (logic C.A+D.A).  b) With draw amount whenever an amount is being  withdraw it has to be less than the current amount less  than the amount else print “Insufficient funds”. |  |  |  |
| 4 | 1.Write a java program with class named |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

Week-1

* Program : 1
* Aim : Download and Install Java Software.
* Step 1 : Visit chrome and search “ java download”.And select Oracle website.



* Step 2 : Now open Oracle website scroll down and now select “JDK 21” for

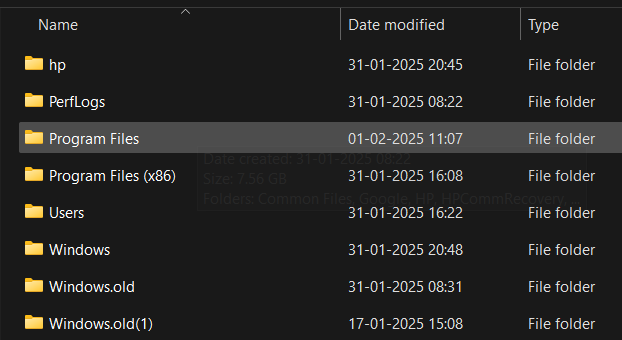
Windows and select “X64 installer” and download it.

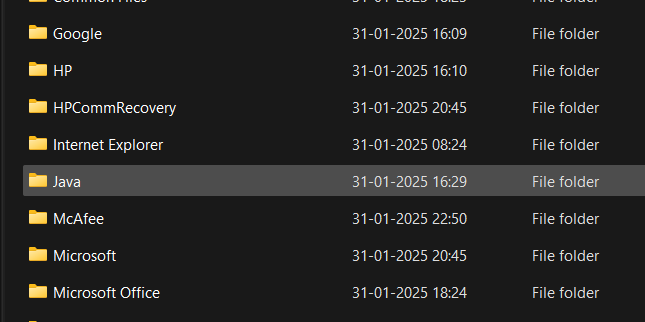
A screenshot of a computer

Description automatically generated

* Step 3 : After downloading open “this pc” in our laptop and open “program

files”,open “java”,open “JDK 21”



 A black screen with white numbers

Description automatically generated

A screenshot of a computer

Description automatically generated

* Step 4 : In the task bar search and open “environment variables of

system”,after opening environment variables, go to the system

variables and see for java if there leave it. Or click path and add

“JAVA” in ‘variable name’ and copy link in ‘variable value’

A screenshot of a computer

Description automatically generated

Screens screenshot of a computer

Description automatically generated

* Step 5 : Verifying Installation of Java. Again open task bar and search “cmd”,

open it ant type “java –version” and press enter. It will show the

version of installation of java.

A screenshot of a computer program

Description automatically generated

Successfully Java is installed and it will show the version otherwise it will show error and command is not recognized.

* Program : 2

Q) Write a java program to print the message “welcome to java program”.

class Main{

public static void main(String[] args){

System.out.println("welcome to java programming");

}

}

OUTPUT

A screenshot of a computer program

Description automatically generated

* Program : 3

Q) Write a java program that prints name,roll number,section of a student.

public class My\_Profile

{

public static void main(String args[]) {

System.out.println("Name: A.Vijay Kumar");

System.out.println("Class: CSE 'A'");

System.out.println("Roll No: 24002");

}

}

OUTPUT

A computer screen shot of a black screen

Description automatically generated

Week-2

A) write a java program to calculate the area of the rectangle

CODE:

import java.util.Scanner;

class arearect{

public static void main(String[]args){

Scanner input=new Scanner(System.in);

System.out.println("enter the length");

int len=input.nextInt();

System.out.println("enter the bredth");

int bred=input.nextInt();

int area=len\*bred;

System.out.println("THE AREA OF THE RECTANGLE IS:"+area);

}}

ERROR TABLE

|  |  |  |
| --- | --- | --- |
| S.No | Expected Error | Reason |
| 1 | ; | ; is expected at the end |
| 2 | area | Declaration of int type variable |

GIVEN INPUT:

Length=10;

Breadth=8;

OUTPUT:

A computer screen with white text

Description automatically generated

B) write a java program to convert the temperature from temperature in Celsius to

Fahrenheit and vice versa.

🡪Celsius to Fahrenheit :

CODE:

import java.util.Scanner;

class temperature{

public static void main(String[]args){

Scanner input =new Scanner(System.in);

System.out.println("enter the temperature in Celsius:");

double deg=input.nextDouble();

System.out.println("the temperature in Fahrenheit"+((deg\*9/5)+32));

}

}

ERROR TABLE:

|  |  |  |
| --- | --- | --- |
| S.No | Expected Error | Reason |
| 1 | ; | ; is expected at the end |
| 2 | Input .close(); | The input is expected to be closed |

GIVEN INPUT:

32 degrees of Celsius

OUTPUT:

A computer screen with white text

Description automatically generated

🡪 Fahrenheit to Celsius :

CODE:

import java.util.Scanner;

class ftoc{

public static void main(String[]args){

Scanner input =new Scanner(System.in);

System.out.println("enter the the temperature in degrees:");

double deg=input.nextDouble();

System.out.println("the temperatuer in fahrenheit"+((deg-32)\*5/9));

}

}

ERROR TABLE:

|  |  |  |
| --- | --- | --- |
| S.No | Expected Error | Reason |
| 1 | string | String is defined wrongly |
| 2 | system | System is defined wrongly |

GIVEN INPUT:

99 degrees of fahrenheit

OUTPUT:

A computer screen with white text

Description automatically generated

C) write a java program to calculate the simple interest

CODE:

import java.util.Scanner;

class simpleintrest{

public static void main(String[]args){

Scanner input=new Scanner(System.in);

System.out.println("enter the p value");

int p=input.nextInt();

System.out.println("enter the t value");

int t=input.nextInt();

System.out.println("enter the r value");

int r=input.nextInt();

float si=(p\*t\*r)/100;

System.out.println(si);

}}

ERROR TABLE:

|  |  |  |
| --- | --- | --- |
| S.No | Expected Error | Reason |
| 1 | ; | ; is expected at the end |
| 2 | Int t | Without declaration of “t” the computer cannot execute the program |

GIVEN INPUT:

P=1000;T=3;R=5

OUTPUT:

A computer screen with white text

Description automatically generated

D) write a java program to find the largest of the three numbers using ternary opperators

CODE:

import java.util.Scanner;

class ternary{

public static void main(String[]args){

Scanner input=new Scanner(System.in);

System.out.println("enter value of A");

int a=input.nextInt();

System.out.println("enter value of B");

int b=input.nextInt();

System.out.println("enter value of C");

int c=input.nextInt();

int large=(a>b)?((a>c)?a:c):((b>c)?b:c);

System.out.println(large);

}}

ERROR TABLE:

|  |  |  |
| --- | --- | --- |
| S.No | Expected Error | Reason |
| 1 | ? | Checks the condition |
| 2 | : | Comparison between two variables |

GIVEN INPUT:

A=34;B=56;C=42;

OUTPUT:

A computer screen with white text

Description automatically generated

E) write a java program to find the factorial of the given number

CODE:

import java.util.Scanner;

class factorial{

public static void main(String[]args){

Scanner input=new Scanner(System.in);

System.out.println("enter the number to find its factorial");

int n=input.nextInt();

int sum=1;

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

sum=sum\*i;}

System.out.println(sum);}}

ERROR TABLE:

|  |  |  |
| --- | --- | --- |
| S.No | Expected Error | Reason |
| 1 | } | To close for loop |
| 2 | System.out.pritnln(); | Spelling in the given program is incorrect  (System.out.println();) |

GIVEN INPUT:

Entered number=10;

OUTPUT:

A computer screen with white text

Description automatically generated

Week-3

A) Write a java program with the following instructions.

Create a class with name car.

Create four attributes named car\_colour,car\_brand,fuel\_type,top\_speed.

Create three method named “Start\_Racing”,”End\_Race”.{ }

Create three objects named Car1,Car2,Car3.

Create a constructor which should print “Welcome to Garage”.

Class Diagram:

|  |
| --- |
| **Car** |
| * carColor: String |
| * carBrand: String |
| * fuelType: String |
| * topSpeed: int |
| + Car(String,String,String,int) |
| + startRacing() |
| + endRace() |

**CODE:**

// Car.java

public class Car {

// Attributes

private String carColour;

private String carBrand;

private String fuelType;

private int topSpeed;

// Constructor

public Car(String carColour, String carBrand, String fuelType, int

topSpeed) {

this.carColour = carColour;

this.carBrand = carBrand;

this.fuelType = fuelType;

this.topSpeed = topSpeed;

System.out.println("Welcome to car garage");

}//End of the constructor

// Method to start racing

public void startRacing() {

System.out.println(carBrand + " (" + carColour + ") is starting the

race with a top speed of " + topSpeed + " km/h and runs on " +

fuelType + "!");

}//End of Method

// Method to end race

public void endRace() {

System.out.println(carBrand + " (" + carColour + ") has finished

the race!");

}

//End of the Method

// Main method to create objects and demonstrate functionality

public static void main(String[] args) {

// Creating three objects

Car car1 = new Car("Red", "Ferrari", "Petrol", 200);

Car car2 = new Car("Blue", "Tesla", "Electric", 250);

Car car3 = new Car("Black", "BMW", "Diesel", 220);

// Starting and ending races

car1.startRacing();

car1.endRace();

car2.startRacing();

car2.endRace();

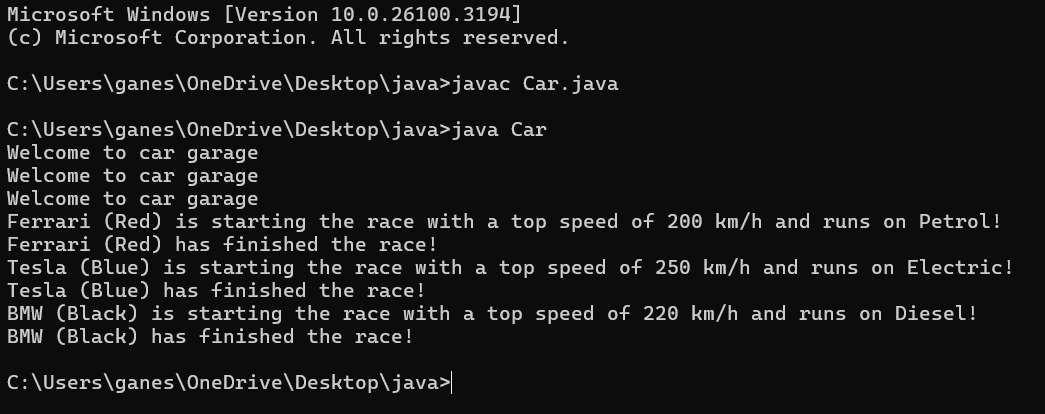
car3.startRacing();

car3.endRace();

}

}

Output:



**Errorc Table:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Errors** | **Rectification** |
| 1 | } | To close for loop |
| 2 | System.out.print(); | If we place the print statement inside the for loop it will print the each i value everytime but to print only the final value we must place it outside the for loop. |

B ) Write a class by writing java program named Bank Account

with two methods “deposits and withdraw”.

a) In deposit method whenever an amount is deposited it

has to be updated with current amount (logic C.A+D.A).

b) With draw amount whenever an amount is being

withdraw it has to be less than the current amount less

than the amount else print “Insufficient funds”.

**Class Diagram:**

|  |
| --- |
| Bank Account |
| * currentAmount: double |
| + BankAccount(initialAmount:double) |
| + deposit(amount: double):void |
| + withdraw(amount: double):void |
| + getCurrentAmount():double |

**Program:**

import java.util.Scanner;

class BankAccount {

String name;

int accountNumber;

int currentBalance;

// Constructor to initialize the bank account

BankAccount(String name, int accountNumber, int currentBalance) {

this.name = name;

this.accountNumber = accountNumber;

this.currentBalance = currentBalance;

System.out.println("Customer Details: " + name + ", Account Number: " + accountNumber + ", Current Balance: " + currentBalance);

}

// Method to withdraw an amount

public void withdraw(int withdrawAmount) {

if (withdrawAmount <= currentBalance) {

currentBalance -= withdrawAmount;

System.out.println("Withdrawn: " + withdrawAmount);

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

}

else {

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

}

}

// Method to deposit an amount

public int deposit(int depositAmount) {

currentBalance += depositAmount;

System.out.println("Deposited: " + depositAmount);

return currentBalance;

}

// Main method to run the program

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

// Input for account details

System.out.print("Enter your name: ");

String name = scanner.nextLine();

System.out.print("Enter your account number: ");

int accountNumber = scanner.nextInt();

System.out.print("Enter your initial balance: ");

int initialBalance = scanner.nextInt();

// Create a new bank account

BankAccount account = new BankAccount(name,

accountNumber, initialBalance);

// Input for withdrawal and deposit

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

int withdrawAmount = scanner.nextInt();

account.withdraw(withdrawAmount);

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

int depositAmount = scanner.nextInt();

account.deposit(depositAmount);

// Final balance

System.out.println("Final Amount: " + account.currentBalance);

// Close the scanner

scanner.close();

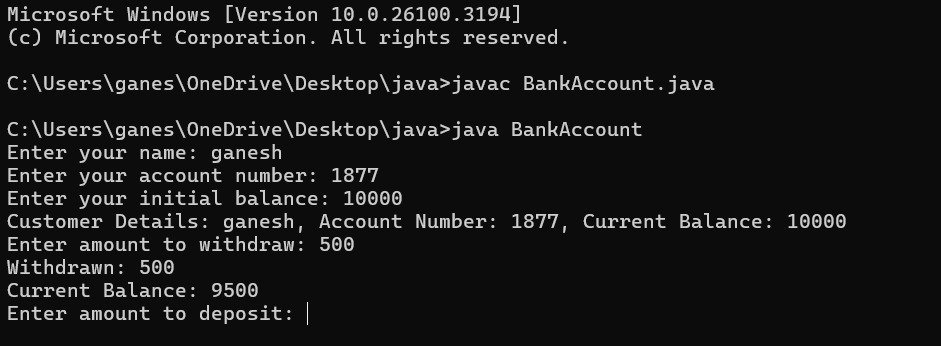
}

}

Given Input:

* Name: ganesh.
* Account number: 1877.
* Initial balance: 10000.
* Amount to withdraw: 500.

Output:



**Error Table:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Errors** | **Rectification** |
| 1 | ; | ; is expected at end |
| 2 | Int t | Without declaring the compiler cannot execute the  program. |

#### WEEK-4

1. Program : 1

Q)  **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 which initializes**

**“ Title of the book, author, year of publication”. Create a method which displays the**

**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.**

Class Diagram:

|  |
| --- |
| **Book** |
| - title: String  - author: String  - yearOfPublication: int |
| + Book(title: String, author: String,  yearOfPublication: int)  + displayDetails(): void |

**Program:**

class Book {

// beginning of the class Book

public String title; // Changed Title to title for consistency

private String author;

public int yearOfPublication;

// beginning of constructor

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

this.title = title; // Changed Title to title for consistency

this.author = author;

this.yearOfPublication = yearOfPublication;

}

// constructor ends here

// method display starts here

public void display() {

System.out.println("Title of the book is: " + title +

", The name of the author is: " + author +

", The year of publication is: " + yearOfPublication);

}

// method display ends here

// creating objects

public static void main(String[] args) {

Book book1 = new Book("Harry Potter", "J.K. Rowling", 1993);

Book book2 = new Book("Someone Like You", "Nikitha Singh", 2010);

book1.display();

book2.display();

}

}

// class ends here

Output:



**Error Table:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Errors** | **Rectification** |
| 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. |

1. Program : 2

Q)  **To create a java program with class named Myclass with a static variable “Count” of**

**“inttype”, 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 .**

Class Diagram:

|  |
| --- |
| **MyClass** |
| - Count: int  + pi: double |
| + MyClass()  + getCount(): int |

**Program:**

class Myclass{

// class starts here

static int Count = 0;

final double pi = 3.1415;

// the constructor starts here

Myclass(){

Count++;

}

// the constructor ends here

public static void main(String[] args){

Myclass c1 = new Myclass();

Myclass c2 = new Myclass();

System.out.println("Count : " + c1.Count);

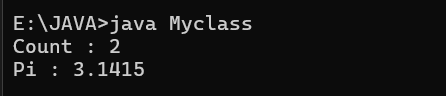
System.out.println("Pi : " + c1.pi);

}

}

// class ends here

Output:



**Error Table:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Errors** | **Rectification** |
| 1 | .variable | We must mention variable name to call the variable. |
| 2 | static | Static variables contain only one value. |

**WEEK-5**

1. Create a calc using the operations including add, sub, mul, div using multilevel inheritance and display the desired output.

Class Diagram:

|  |
| --- |
| **Basic Operations** |
| + add (a,b)  +subtract (a,b) |

|  |
| --- |
| **Multiplication** |
| +Multiply (a,b) |

|  |
| --- |
| **Division** |
| + Divide (a,b) |

|  |  |
| --- | --- |
| **Subtraction** | |
| + subtraction(a,b) | |
|  | |  | |

|  |
| --- |
| **Calculator** |
| +calculate (op,a,b) |

**Program:**

class bcalc {

int a, b;

int sum, diff;

bcalc(int a, int b) {

this.a = a;

this.b = b;

}

public void add()

{ diff = a - b;

sum = a + b;

System.out.println("Difference: " + diff);

System.out.println("Sum: " + sum);

}

}

class acalc extends bcalc {

int mul; acalc(int a, int b) {

super(a, b);

}

public void mult() {

mul = a \* b;

System.out.println("Multiplication: " + mul);

}

}

class aacalc extends acalc {

float div;

aacalc(int a, int b) {

super(a, b);

}

public void divi()

{

if (b != 0) { // Check to avoid division by zero

div = (float) a / b;

System.out.println("Division: " + div);

}

else {

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

}

}

}

class ocalc {

public static void main(String[] args) {

aacalc c = new aacalc(10, 2);

c.divi();

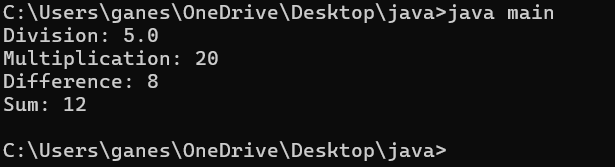
c.mult();

c.add();

}

}

Output:



Error table:

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Errors** | **Rectification** |
| 1 | .variable | We must mention variable name to call the variable. |
| 2 | static | Static variables contain only one value. |

B.Program : 2

Q)  **A Vechile 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 a need a program to store details about each vehicle such as brand and speed .**

* **Cars should have an additional properties .**
* **“Number of doors “ seating capacity.**
* **Bikes should have a property indicating whether they have gears are not ?**
* **The system should also include a fuction to display details about each vehicle and indicate when a vechicle is starting .**
* **If the company describes to add a new type of vechile ‘truck’ how would you modify above program.**
* **Truck should include an addition property capacity ‘in tons’.**
* **Create a show truck details method to display the trucks capacity.**
* **Write a constructor for truck that initializes all properties.**
* **Implement the truck class and update the main method to create a truck object and also create an object and also create an object car and bike subclass find display it details.**

Class Diagram:

|  |
| --- |
| Vechile |
| * Brand: String * speed: int |
| + Vechile(String,b int)  + Start()  + DisplayDetails() |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | Car | | * numberofdoors: int * seatingCapacity: int | | + car(String,int,int,int)  + displaydetails() | | |  | | --- | | Bike | | -hasGears: boolean | | + Bike(String,int,Boolean)  +displayetails() | |

|  |
| --- |
| Truck |
| -capacity: double |
| + truck(String,int,double)  +showtruckdetails()  +displaydetails() |

// Base class for Vehicle

class Vehicle {

protected String brand;

protected int speed;

public Vehicle(String brand, int speed) {

this.brand = brand;

this.speed = speed;

}

public void start() {

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

}

public void displayDetails() {

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

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

}

}

// Car class that extends Vehicle

class Car extends Vehicle {

private int numberOfDoors;

private int seatingCapacity;

public Car(String brand, int speed, int numberOfDoors, int seatingCapacity) {

super(brand, speed);

this.numberOfDoors = numberOfDoors;

this.seatingCapacity = seatingCapacity;

}

@Override

public void displayDetails() {

super.displayDetails();

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

System.out.println("Seating Capacity: " + seatingCapacity);

}

}

// Bike class that extends Vehicle

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"));

}

}

// Truck class that extends Vehicle

class Truck extends Vehicle {

private double capacity; // in tons

public Truck(String brand, int speed, double capacity) {

super(brand, speed);

this.capacity = capacity;

}

public void showTruckDetails() {

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

}

@Override

public void displayDetails() {

super.displayDetails();

showTruckDetails();

}

}

// Main class to test the implementation

public class Main {

public static void main(String[] args) {

// Create a Car object

Car car = new Car("Toyota", 180, 4, 5);

car.start();

car.displayDetails();

System.out.println();

// Create a Bike object

Bike bike = new Bike("Yamaha", 120, true);

bike.start();

bike.displayDetails();

System.out.println();

// Create a Truck object

Truck truck = new Truck("Volvo", 100, 10.5);

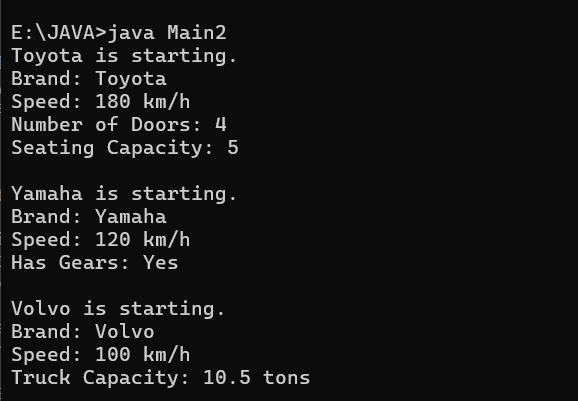
truck.start();

truck.displayDetails();

}

}

Output:



|  |  |  |
| --- | --- | --- |
| **S.NO** | **Errors** | **Rectification** |
| 1 | .variable | We must mention variable name to call the variable. |
| 2 | static | Static variables contain only one value. |

**WEEK-6**

1. Write a java program to create a vechiles class with a method displayinfo() override this method in the car subclass to provide specific information about a car

* Company
* Model
* Price
* Seating capacity
* Petrol or not

Pogram:

public class vehicles {

void displayinfo() {

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

}

}

class Car extends vehicles {

String Name, Model, Capacity;

int Price;

boolean Petrol;

Car(String Name, String Model, String Capacity, int Price, boolean Petrol) {

this.Name = Name;

this.Model = Model;

this.Capacity = Capacity;

this.Price = Price;

this.Petrol = Petrol;

}

@Override

public void displayinfo() {

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

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

System.out.println("The price of the car is: " + Price);

System.out.println("The seating capacity of the car is: " + Capacity);

System.out.println("Is it petrol? " + Petrol);

}

}

class Main {

public static void main(String[] args) {

Car c1 = new Car("Toyota", "Sedan", "5-Seater", 4500, true);

c1.displayinfo();

}

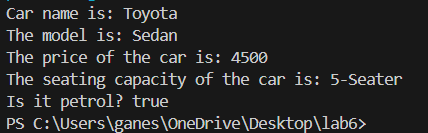
}

**Class Diagram:**

|  |
| --- |
| Vehicle |
| +displayInfo(): void |

|  |
| --- |
| car |
| +displayInfo(): void |

**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) A college is developing automated admission system that verifies students eligibility for UG and PG programs .Each program has different eligibility criteria based on the students percentage in their previous qualification.

* UG admission require minimum 60%
* PG admission require minimum 70%

**Program:**

import java.util.Scanner;

class College {

String name;

int qualification;

int percentage;

// Constructor

College(String name, int qualification, int percentage) {

this.name = name;

this.qualification = qualification;

this.percentage = percentage;

}

// Default Eligibility method

public void Eligibility() {

System.out.println("Name: " + name + ", Qualification: " + qualification + ", Percentage: " + percentage);

System.out.println("The candidate is a fluke");

}

}

class UG extends College {

UG(String name, int qualification, int percentage) {

super(name, qualification, percentage);

}

@Override

public void Eligibility() {

System.out.println("Name: " + name + ", Qualification: " + qualification + ", Percentage: " + percentage);

System.out.println("The candidate is eligible for UG");

}

}

class PG extends College {

PG(String name, int qualification, int percentage) {

super(name, qualification, percentage);

}

@Override

public void Eligibility() {

System.out.println("Name: " + name + ", Qualification: " + qualification + ", Percentage: " + percentage);

System.out.println("The candidate is eligible for PG");

}

}

public class Main {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

// Taking inputs

System.out.println("Enter your name:");

String name = input.nextLine();

System.out.println("Enter your qualification (e.g., 12 for high school, 10 for 10th, etc.):");

int qualification = input.nextInt();

System.out.println("Enter your percentage:");

int percentage = input.nextInt();

// Close scanner

input.close();

// Logic to check eligibility

College candidate;

if (percentage >= 70) {

candidate = new PG(name, qualification, percentage);

} else if (percentage >= 60) {

candidate = new UG(name, qualification, percentage);

} else {

candidate = new College(name, qualification, percentage);

}

candidate.Eligibility();``

}

}

**Class Diagram:**

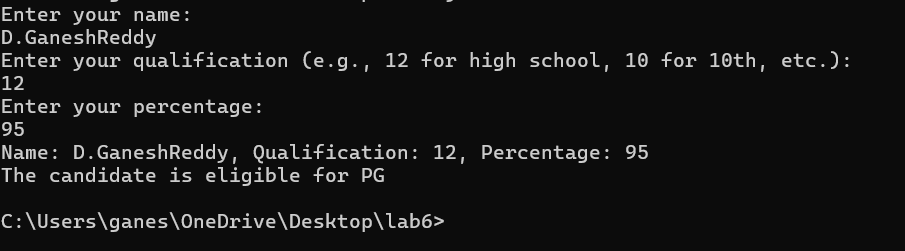
|  |
| --- |
| **adm** |
| elg():void |

|  |  |
| --- | --- |
| ug | pg |
| +elg():void | +elg():void |

**Input:**

* **Name:D.GaneshReddy**
* **Qualification: 12**
* **Percentage: 95**

**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)Create a calculator class with overloading methods to perform addition**

* **Add two doubles**
* **Add two integer**
* **Add three integer**

**Program:**

public class Calculator{

// Method to add two integers

public int add(int a, int b) {

return a + b;

}

// Method to add two tuples (represented as arrays)

public int add(int[] tuple1, int[] tuple2) {

int sum = 0;

for (int i = 0; i < tuple1.length; i++) {

sum += tuple1[i] + tuple2[i];

}

return sum;

}

// Method to add three integers

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

return a + b + c;

}

public static void main(String[] args) {

Calculator calc = new Calculator();

// Adding two integers

int result1 = calc.add(10, 20);

System.out.println("Addition of two integers: " + result1);

// Adding two tuples (arrays)

int[] tuple1 = {1, 2};

int[] tuple2 = {3, 4};

int result2 = calc.add(tuple1, tuple2);

System.out.println("Addition of two tuples: " + result2);

// Adding three integers

int result3 = calc.add(5, 10, 15);

System.out.println("Addition of three integers: " + result3);

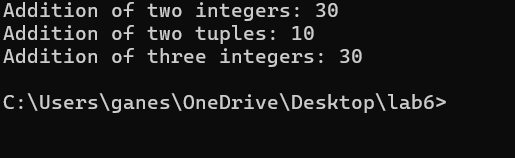
}

}

Class Diagram:

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

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 |

4Q)Create a shape class with a method calculate area that is overloaded for different shapes Square,Rectangle then create a sub class circle that overerides the calculate area methods for a circle.

**Program:**

public class Shape {

public double areaOfSquare(double side) {

return side \* side;

}

public double areaOfRectangle(double length, double width) {

return length \* width;

}

public double areaOfCircle(double radius) {

return 3.14 \* radius \* radius;

}

public static void main(String[] args) {

Shape shape = new Shape();

double square = shape.areaOfSquare(5);

System.out.println("Area of a square: " + square);

double rectangle = shape.areaOfRectangle(10, 20);

System.out.println("Area of a rectangle: " + rectangle);

double circle = shape.areaOfCircle(12);

System.out.println("Area of a circle: " + circle);

}

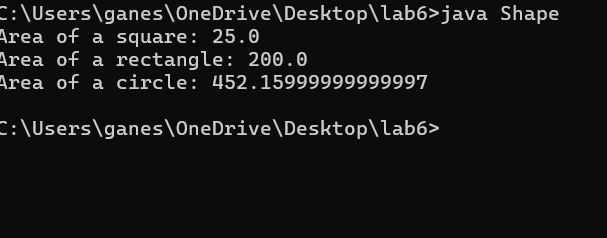
}

**Class Diagram:**

|  |
| --- |
| shape |
| +calarea(float side):float  +calarea(float l,float b):float  +calarea(float c):float |

|  |
| --- |
| Circle |
| +calarea(double r):double |

**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-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 Q1labw7 {

public static void main(String[] args) {

Animal lion = new Lion();

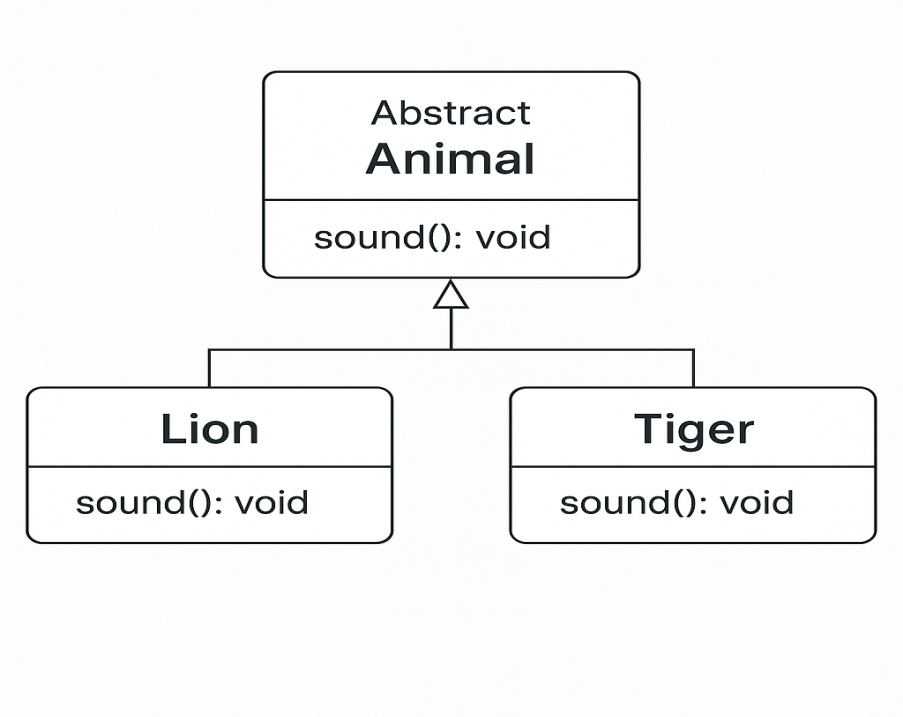
Animal tiger = new Tiger();

lion.sound();

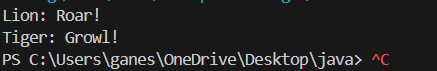
tiger.sound();

}

}

****

**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 Q2labw7 {

public static void main(String[] args) {

Shape3D sphere = new Sphere(5);

Shape3D cube = new Cube(3);

sphere.calculate\_surf\_a();

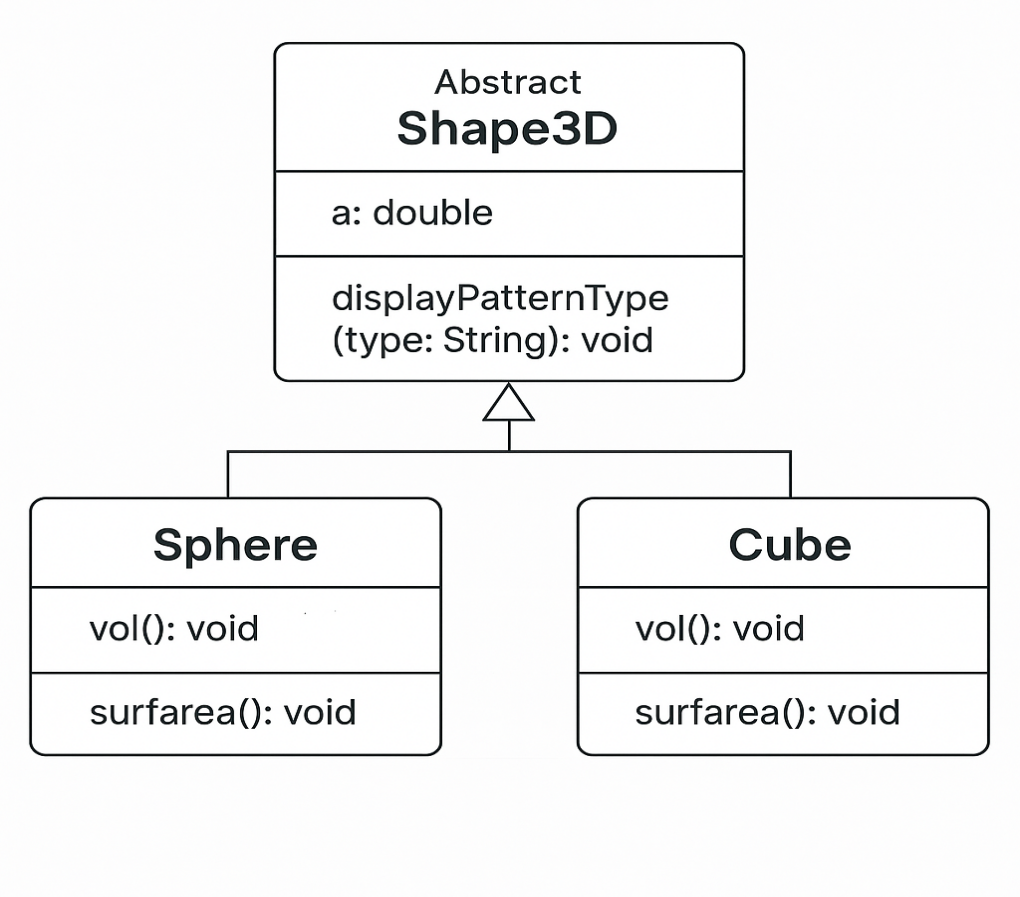
sphere.calculate\_volume();

cube.calculate\_surf\_a();

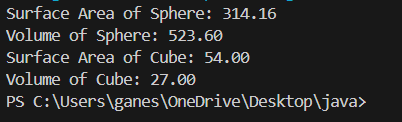
cube.calculate\_volume();

}

}



**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 {

public abstract void printPattern(int n);

}

class RightTrianglePattern extends Pattern {

@Override

public void printPattern(int n) {

System.out.println("Right Triangle Pattern:");

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

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

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

}

System.out.println();

}

}

}

class NumberPattern extends Pattern {

@Override

public void printPattern(int n) {

System.out.println("number pattern:");

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

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

System.out.print( j);

}

System.out.println();

}

}

}

public class Q3labw7 {

public static void main(String[] args) {

Scanner input= new Scanner(System.in);

System.out.println("enter the n value to select number of rows");

int n=input.nextInt();

Pattern rightTriangle = new RightTrianglePattern();

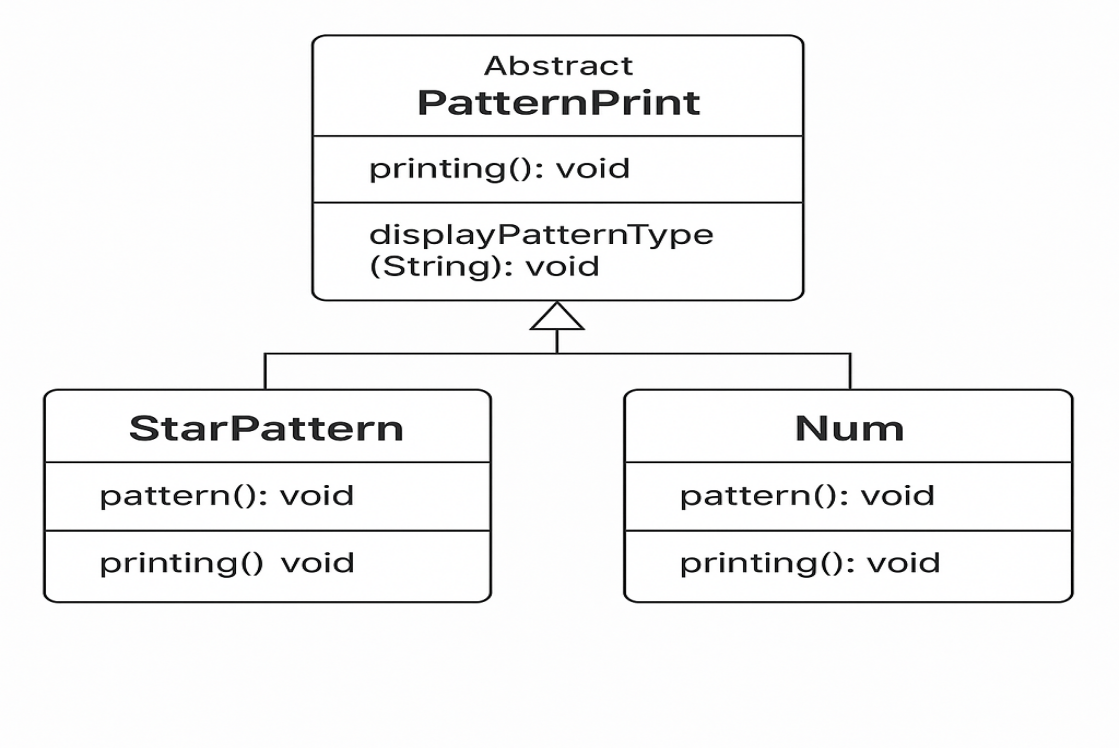
Pattern numberpattern = new NumberPattern();

rightTriangle.printPattern(n);

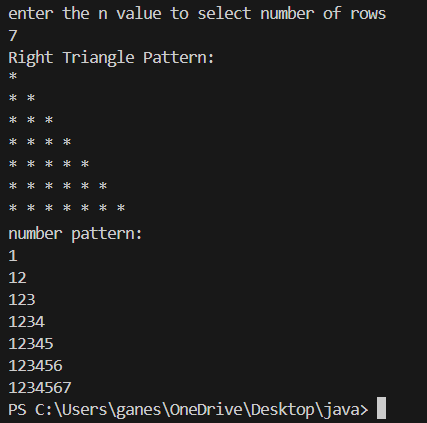
numberpattern.printPattern(n);

}

}

****

**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 {

int width, height;

Rectangle(int width, int height) {

this.width = width;

this.height = height;

}

public double getPerimeter() {

return 2 \* (width + height);

}

}

class Circle implements Shape {

int radius;

Circle(int radius) {

this.radius = radius;

}

public double getPerimeter() {

return 2 \* Math.PI \* radius;

}

}

class Triangle implements Shape {

int side1;

int side2;

int side3;

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

this.side1 = side1;

this.side2 = side2;

this.side3 = side3;

}

public double getPerimeter() {

return side1 + side2 + side3;

}

}

public class Q1labweek8{

public static void main(String[] args) { // Corrected brackets

Shape rectangle = new Rectangle(5, 10);

Shape circle = new Circle(7);

Shape triangle = new Triangle(1, 2, 3);

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

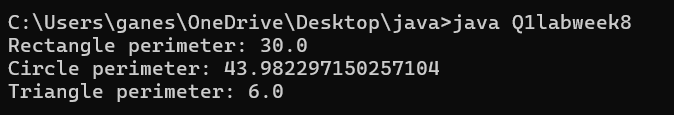
System.out.println("Circle perimeter: " + circle.getPerimeter());

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

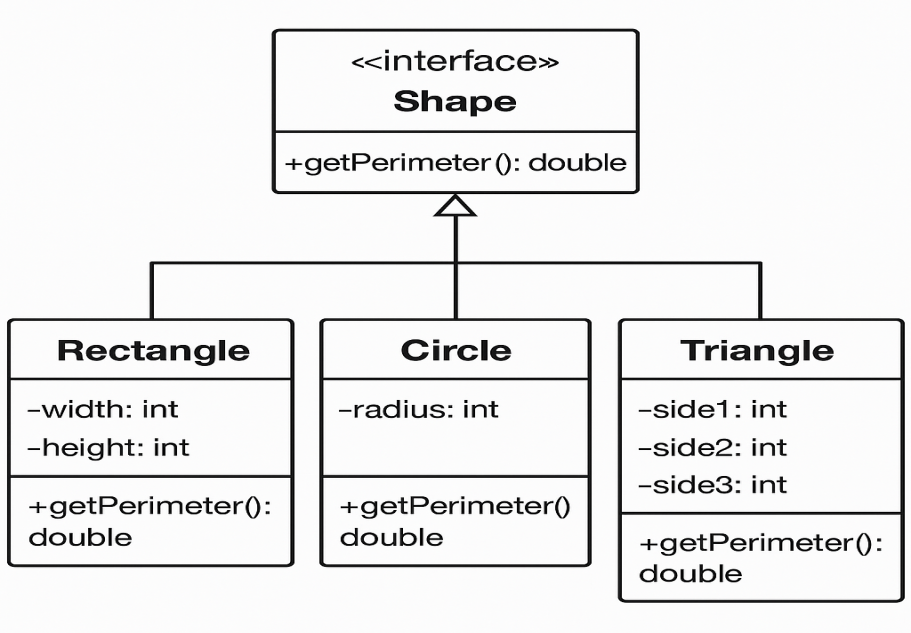
}

}

**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 Q2labweek8 {

public static void main(String[] args) {

playable vb =new volleyball();

playable bb=new basketball();

playable fb=new football();

vb.play();

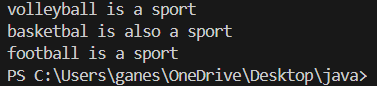
bb.play();

fb.play();

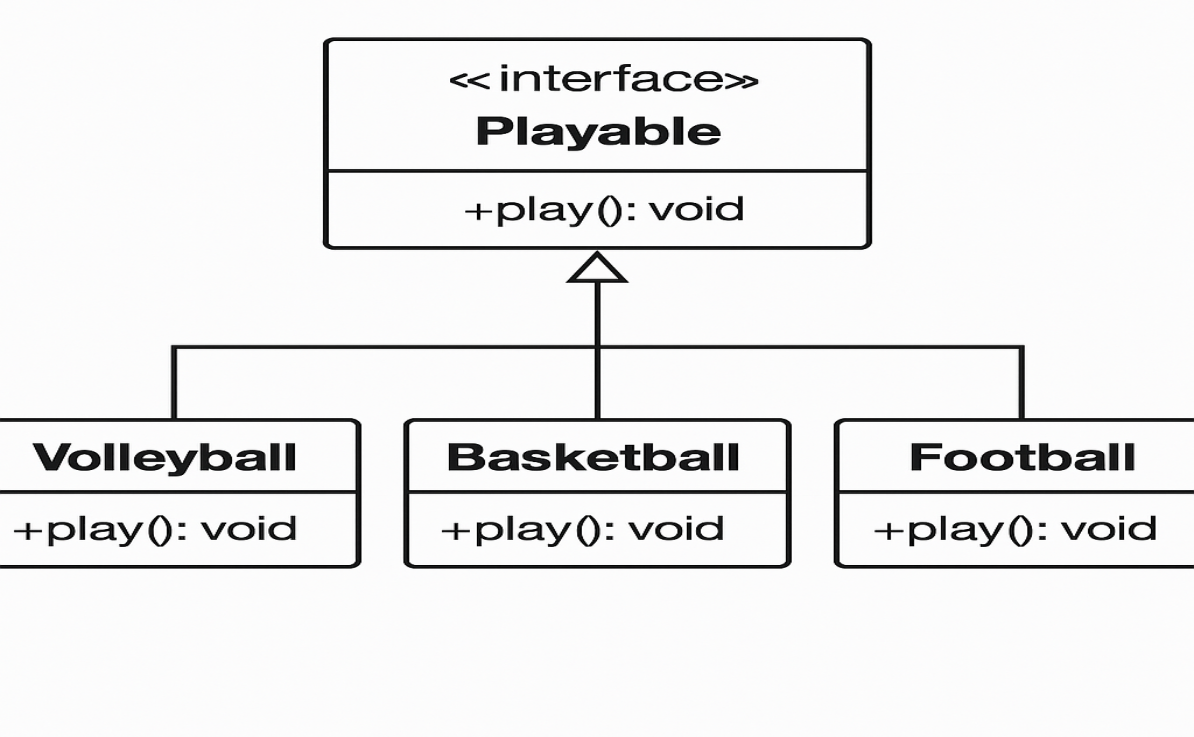
}

}

**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 Q3labweek8 {

public static void main(String[] args) {

UniversityPortal portal = new UniversityPortal();

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

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

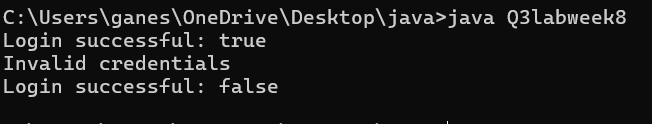
boolean loginSuccess2 = portal.login("student123", "wrong pass");

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

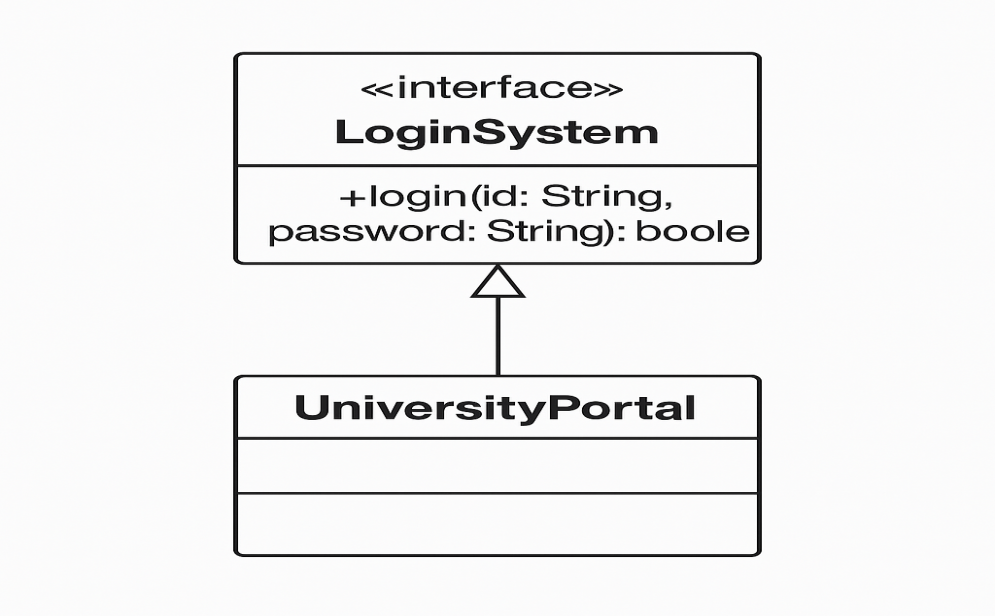
}

}

**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 |