**23CSE111**

**OBJECT ORIENTED PROGRAMMING**

**LAB REPORT**



**Department of Computer Science Engineering**   **Amrita School of Computing**

**Amrita Vishwa Vidyapeetham, Amaravati Campus**

**Name: M.JYOTHEESWAR**

**Roll No: 24230**

**Verified By :**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S.NO** | **Experiment** | **Page No** | **Remarks** | **Signature** |
| **1** | **Installation Process of JDK** | **3-4** |  |  |
| **2** | **Simple java program for printing basic details of student** | **4-5** |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**INDEX**

# WEEK-1

1. **AIM:**

**Process of Installing JDK (Java Development Kit)**

**Installing JDK (Java Development Kit):**

* 1. **Download JDK:**
* Go to the Oracle JDK download page in google and click on JDK-21 version which is Long term support (LTS) version.
* Click the download link as your operating system (Windows, macOS, or Linux).
  1. **Install JDK:**
* Once downloaded, run the installer.
* Follow the given instructions and keep clicking "Next" until it is done.
  1. **Set Environment Variables (Windows):**
* Open file explorer, then right click on This PC next select on properties then it will take you to the settings app then click on advanced system settings and then click on **Environment Variables**.
* Click on path and new under **System Variables**:

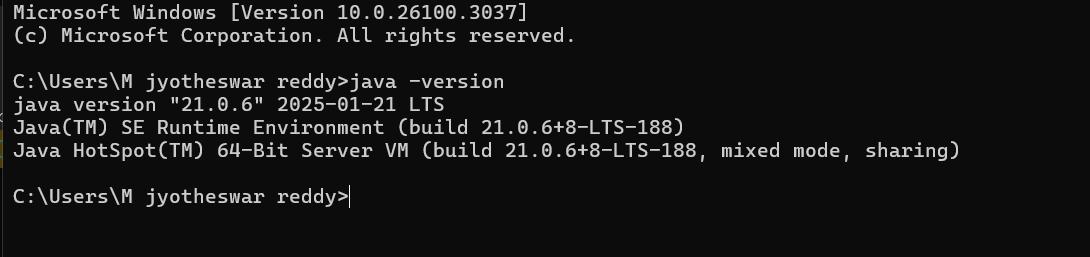
**Variable value:** The folder address where JDK is installed (like

C:\Program Files\Java\jdk-21\bin)

* Find Path under **System Variables**, click **New**, and add the path of the jdk-21(C:\Program Files\Java\jdk-21\bin)



**Checking JDK Version: -**

* 1. **Open Command Prompt:**
* Presswin+R, typecmd, and press Enter.
  1. **Check Version:**
* Type java -version and press Enter.
* Type javac --version and press Enter.
* 

**2)**

**AIM:**

**Simple Java Program for printing Name, Class, Roll No, of a Student**

Write your code in Notepad and execute it in cmd prompt

**CODE: -**

class Main

{

public static void main(String[] args)

{

System.out.println("Name:P.MANOJ");

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

System.out.println("Roll No:24241");

}

}

**Output: -**

A computer screen with white text

AI-generated content may be incorrect.

|  |  |  |
| --- | --- | --- |
| 1 | Syntax error | Semicolon added |
| 2 | Runtime error | Copied correct path |
| 3 | Name error | rectified |

Week-2

**1)**

**AIM ;**

**Simple Java Program for finding simple interest by taking input from**

**User**

**Code:**

**import java.util.Scanner;**

**class Main {**

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

**Scanner input = new Scanner(System.in);**

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

**double principal = input.nextDouble();**

**System.out.print("Enter the rate: ");**

**double rate = input.nextDouble();**

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

**double time = input.nextDouble();**

**double interest = (principal \* time \* rate) / 100;**

**System.out.println("Principal: " + principal);**

**System.out.println("Interest Rate: " + rate);**

**System.out.println("Time Duration: " + time);**

**System.out.println("Simple Interest: " + interest);**

**input.close();**

**}**

**}**

**Output**

****

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Error type** | **Reason for error** | **rectification** |
| **1** | **Runtime error** | **Incorrect path** | **Copied correct path** |
| **2** | **Syntax error** | **{ missing** | **{ added** |
| **3** | **Logical error** | **Wrong formula** | **Formula rectified** |

**2)**

**AIM:**

**Write a simple program to calculate factorial of a number and read the**

**input from user**

**code:**

**class Test {**

**static int factorial(int n)**

**{**

**int res = 1, i;**

**for (i = 2; i <= n; i++)**

**res \*= i;**

**return res;**

**}**

**public static void main(String[] args)**

**{**

**int num = 5;**

**System.out.println("Factorial of " + num + " is "**

**+ factorial(5));**

**}**

**}**

**Output**

****

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Error type** | **Reason for error** | **Rectification** |
| **1** | **Undeclared variable error** | **Missing variable** | **Variable declared** |
| **2** | **Missing import statement** | **Not importing packages** | **Packages imported** |
| **3** | **Logical error** | **Wrong formula** | **Formula rectified** |

**3)**

**AIM:**

**Write a program to to calculate the fibonacii sequence and take the input from user**

**Code:**

**import java.util.\*;**

**class fibo**

**{**

**public static void main(String args[])**

**{**

**Scanner sc = new Scanner(System.in);**

**int num;**

**int f3;**

**int f1 = 0;**

**int f2 = 1;**

**int i = 2;**

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

**num = sc.nextInt();**

**System.out.println(f1);**

**System.out.println(f2);**

**while(i<num)**

**{**

**f3 = f1+f2;**

**f1 = f2;**

**f2 = f3;**

**System.out.println(f3);**

**i = i+1;**

**}**

**}**

**}**

**Output:**

****

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Error type** | **Reason for error** | **Rectification** |
| **1** | **Logical error** | **Incorrect formula** | **Formula rectified** |
| **2** | **Run-time error** | **Incorrect path** | **Added correct path** |
|  |  |  |  |

**4)AIM;**

**Write a java program to convert temperature from Fahrenheit to celsius**

**Code:**

**class fah {**

**public static void main(String[] args)**

**{**

**double f = 50.0, c = 0.0;**

**c = (f - 32) / 1.8;**

**System.out.println(**

**"value of temperature in celsius:" + c);**

**}**

**}**

**Output:**

**A black screen with white text

AI-generated content may be incorrect.**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Error type** | **Reason for error** | **rectification** |
| **1** | **Syntax error** | **Missing ”** | **“ is added** |
| **2** | **Missing import error** | **Util package missing** | **Util package added** |
|  |  |  |  |

**5)AIM:**

**Write a java program to convert temperature from Celsius to Fahrenheit**

**Code:**

**class celsiustofahrenheit {**

**public static void main(String[] args)**

**{**

**double celsius = 10.0, fahrenheit = 0.0;**

**fahrenheit = (celsius \* 1.8) + 32;**

**System.out.println(**

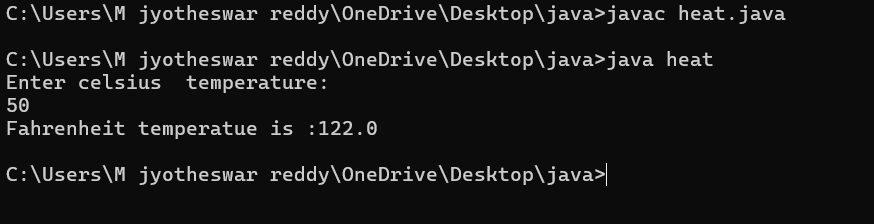
**" value of temperature in fahrenheit:"**

**+ fahrenheit);**

**}**

**}**

**Output:**

****

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Error type** | **Reason for error** | **Rectification** |
| **1** | **Runtime error** | **Incorrect path selection** | **Correct path added** |
| **2** | **Logical error** | **Incorrect logic** | **Correct logic** |
|  |  |  |  |

**6)AIM:**

**Write a simple program to find the area of rectangle:**

**Code:**

**import java.util.\*;**

**class Area{**

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

**int area;**

**Scanner sc = new Scanner(System.in);**

**System.out.println("Enter Length:");**

**int l = sc.nextInt();**

**System.out.println("Enter Breadth:");**

**int b = sc.nextInt();**

**area = l\*b;**

**System.out.println("Area of Rectangle :"+area);**

**}**

**}**

**Output:**

A computer screen shot of a black screen

AI-generated content may be incorrect.

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Error type** | **Reason for error** | **Rectification** |
| **1** | **Syntax error** | **Semi colon missing** | **Semi colon added** |
| **2** | **Missing import error** | **Import package missing** | **Import package added** |
|  |  |  |  |

**7)AIM:**

**Write a program to find the area of triangle by using heron’s formula**

**take the input from the user**

**Code:**

**port java.io.\*;**

**class GFG {**

**static double area(double h, double b)**

**{**

**return (h \* b) / 2;**

**}**

**public static void main(String[] args)**

**{**

**double h = 10;**

**double b = 5;**

**System.out.println("Area of the triangle: "**

**+ area(h, b));**

**}**

**}**

**Output:**

****

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Error type** | **Reason for error** | **Rectification** |
| **1** | **Logical error** | **Incorrect formula** | **Formula rectified** |
| **2** | **Name error** | **Undeclared variable** | **Variable declared** |
|  |  |  |  |

**WEEK 3**

**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 three methods named start(), stop(). Service()**

**4. Create three objects named car1,car2 and car3**

**Code:**

**import java.util.\*;**

**class car**

**{**

**public String Car\_color;**

**public String Car\_brand;**

**public String fuel\_type;**

**public int mileage;**

**public void start()**

**{**

**System.out.println("Car Started:");**

**System.out.println("Car color is :"+Car\_color);**

**System.out.println("Car Brand is:"+Car\_brand);**

**System.out.println("Car fuel type is:"+fuel\_type);**

**System.out.println("Car mileage is:"+mileage);**

**}**

**public void service()**

**{**

**System.out.println("Car Started:");**

**System.out.println("Car color is :"+Car\_color);**

**System.out.println("Car Brand is:"+Car\_brand);**

**System.out.println("Car fuel type is:"+fuel\_type);**

**System.out.println("Car mileage is:"+mileage);**

**}**

**public void stop()**

**{**

**System.out.println("Car Started:");**

**System.out.println("Car color is :"+Car\_color);**

**System.out.println("Car Brand is:"+Car\_brand);**

**System.out.println("Car fuel type is:"+fuel\_type);**

**System.out.println("Car mileage is:"+mileage);**

**}**

**public static void main(String args[])**

**{ System.out.println("\n manoj\n\n");**

**car car1 = new car();**

**car1.Car\_color = "Blue";**

**car1.Car\_brand = "BMW";**

**car1.fuel\_type = "Deisel";**

**car1.mileage = 10;**

**car1.start();**

**car car2 = new car();**

**car2.Car\_color = "Red";**

**car2.Car\_brand = "Tesla";**

**car2.fuel\_type = "EV";**

**car2.mileage = 300;**

**car2.stop();**

**car car3 = new car();**

**car3.Car\_color = "Yellow";**

**car3.Car\_brand = "MAHINDRA";**

**car3.fuel\_type = "Petrol";**

**car3.mileage = 20;**

**car3.service();**

**}**

**}**

**Output:**

**A computer screen shot of a black screen

AI-generated content may be incorrect.**

Class Diagram

|  |
| --- |
| Car |
| + car\_color: String  + car\_brand: String  + fuel\_type: String  + mileage: int |
| + Car(): void  + start(): void  + service(): void  + stop(): void |

**Concepts to be known:**

1. public String car\_color; - Used to declare a variable named car\_color, with data type as String with public accessibility.
2. Car(String car\_color,String car\_brand,String fuel\_type,int mileage){ } – It is a constructor (method with name same as class), which requires parameters such as car\_color (String data-type) and so on.
3. this.car\_color=car\_color; - “this” is a default method, which is used to point to the instance variables.
4. public void start(){} – used to declare a method, which will return nothing(void) in public accessibility.
5. Car car1=new Car("Red","Maruti","Diesel",20); - used to create a object in class Car, with object name as car1.

car1.start(); - Calling a method, under object car1.

**2.)**

**AIM:**

**To create a class bankAccount with methods deposit() and withdrawl**

**Code:**

**class BankAccount**

**{**

**private double balance;**

**public BankAccount(double initialBalance)**

**{**

**if(initialBalance > 0)**

**{**

**this.balance = initialBalance;**

**}**

**else**

**{**

**this.balance = 0;**

**}**

**}**

**public void deposit(double amount)**

**{**

**if(amount>0)**

**{**

**balance = balance+amount;**

**System.out.println("Deposited $:"+amount);**

**}**

**else**

**{**

**System.out.println("Deposited amount must be positive");**

**}**

**}**

**public double getBalance()**

**{**

**return balance;**

**}**

**}**

**public class Main1**

**{**

**public static void main(String args[])**

**{**

**BankAccount account = new BankAccount(1000);**

**account.deposit(500);**

**System.out.println("Current Balance is:"+account.getBalance());**

**}**

**}**

**Output:**

A black screen with white text

AI-generated content may be incorrect.

**ERRORS:**

|  |  |  |
| --- | --- | --- |
| **Sno.** | **Error message** | **Error rectification** |
| 1. | error: ';' expected  cust1.withdraw(3050) | Add a “;”    cust1.withdraw(3050); |
| 2. | error: cannot find  symbol  thisCurrBal=CurrBal; | Add a “.”    this.CurrBal=CurrBal; |

**Concepts to be known:**

1. private String name; - Used to declare a variable named name, with data type as String with private accessibility.
2. BankAccount(String name,int Accno,int CurrBal){ } – It is a constructor (method with name same as class), which requires parameters such as name (String data-type) and so on.
3. this.CurrBal=CurrBal; - “this” is a default method, which is used to point to the instance variables.
4. public void withdraw(int WAmt){ } – used to declare a method, which will return nothing(void) in public accessibility, which requires a parameter WAmt(integer data type).
5. public int deposit(int DAmt){} - used to declare a method, which will return integer data type in public accessibility, which requires a parameter DAmt(integer data type).
6. BankAccount cust1=new BankAccount("Ram",5587,20000); - used to create a object in class BankAccount, with object name as cust1.
7. cust1.withdraw(50000); - Calling a method, under object cust1, by passing a parameter.

System.out.println("Your current balance after depositing money is:"+cust1.deposit(25000)); - Deposit method will return the value, which will be directly printed.

WEEK-4

**1.AIM:**

**WRITE A JAVA PROGRAM WITH CLASS NAMED “Book”. THE CLASS SHOUKD CONTAIN VARIOUS ATTRIBUTES SUCH AS TITLE, AUTHOR, YEAR OF**

**PUBLICATION. IT SHOULD ALSO CONTAIN A CONSTRUCTOR WITH**

**PARAMETERS WHICH INITIALIZES TITLE, AUTHOR, YEAR OF PUBLICATION**

**AND CREATE A METHOD WHICH DISPLAYS THE DETAILS OF 2 BOOKS.**

**PROGRAM:**

**public class Book {**

**public String title;**

**public String author;**

**public int year;**

**Book(String title, String author, int year) {**

**this.title = title;**

**this.author = author;**

**this.year = year;**

**}**

**public void displayDetails() {**

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

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

**System.out.println("Year of Publication" +year);**

**}**

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

**Book b1 = new Book("Math", "Ramanujan", 1950);**

**Book b2 = new Book("Physics", "CV Raman", 1960);**

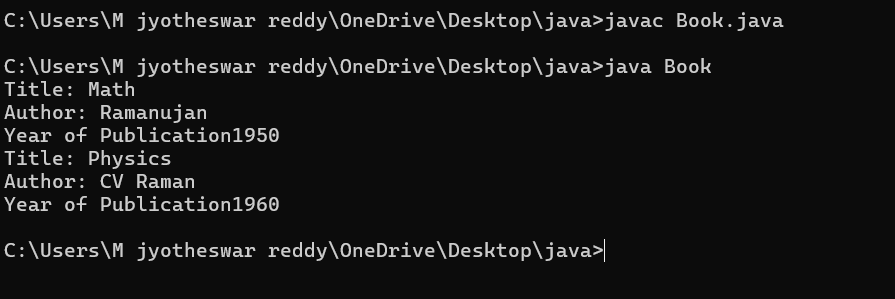
**b1.displayDetails();**

**b2.displayDetails();**

**}**

**}**

**Output:**

****

**ERROR:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **ERROR TYPE** | **Reason for error** | **Rectification** |
| **1.** | Syntax error | No semicolon | Semicolon added |
| **2.** | Runtime error | Incorrect path | Copied correct path |

**CLASS DIAGRAM:**

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

**IMPORTANT POINTS:**

1. **Constructor**:

* The constructor Book(String, String, int) is used to initialize the object when it is created.
* The keyword **this** is used to differentiate between class attributes and constructor parameters.

2.**Method**:

* The method displayDetails() is used to display the book details.
* The **System.out.println()** method prints the details to the console.

3. **Object Creation**:

* Two objects b1 and b2 are created using the constructor.

**2.AIM:**

**WRITE A JAVA PROGRAM WITH CLASS NAMED “MyClass” WITH A STATIC VARIABLE COUNT OF INT TYPE. INTIALIZE IT TO ZERO AND A CONSTANT VARIABLE “Pi” OF TYPE DOUBLE INITIALIZED TO “3.14” AS ATTRIBUTES OF THAT CLASS. NOW DEFINE A CONSTRUCTOR FOR “MyClass”, THAT INCREMENTS THE COUNT VARIABLE EACH TIME AN OBJECT OF “MyClass” IS CREATED. FINALLY, PRINT THE FINAL VALUES OF ‘COUNT’ AND ‘PI’ VARIABLES AND CREATE 3 OBJECTS.**

**PROGRAM:**

public class MyClass {

static int count = 0;

static final double pi = 3.14;

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("Pi: " +pi);

}

}

**OUTPUT:**

**A screen shot of a computer

AI-generated content may be incorrect.**

**ERROR:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Error Type** | **Reason for error** | **Rectification** |
| **1.** | No class | No class name declared | Created class named ‘MyClass’ |
| **2.** | Syntax error | Not added keyword | Added keyword named ‘new’ |

**CLASS DIAGRAM:**

|  |
| --- |
| MyClass |
| -count: int (static)  -pi: double (static, final) |
| +MyClass()  +main(args: String[]):void |

**Week -5**

**1: Aim:**

**create a calculator using the operation including adding, subtraction,multiplication,and division using multilevel inheritance and display desired output**

**every class one method (int a ,int b)**

**pass values as parameter during method call**

**important points :**

1. **used contructor and ssigned values as int a and int b.**
2. **used multilevel unheritance .**
3. **used this keyword to assign values .**

**program:**

**class addition{**

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

**System.out.println("addition is done in calculator below:");**

**System.out.println("");**

**int addition = a+b;**

**return addition;**

**}**

**}**

**class subtraction extends addition {**

**public int sub(int a, int b){**

**System.out.println("subtraction is done in calculator below:");**

**System.out.println("");**

**int subtraction = a-b;**

**return subtraction;**

**}**

**}**

**class multiplication extends subtraction**

**{**

**public int mult(int a, int b) {**

**System.out.println("multiplication is done in calculator below:");**

**System.out.println("");**

**int multiplication = a\*b;**

**return multiplication;**

**}**

**}**

**class division extends multiplication{**

**public int div(int a,int b){**

**System.out.println("divission is done in calculator below:");**

**System.out.println("");**

**int division = a/b;**

**return division;**

**}**

**}**

**class calculator**

**{**

**public static void main(String args[])**

**{**

**division obj = new division();**

**addition img = new addition();**

**subtraction del = new subtraction();**

**multiplication zym = new multiplication();**

**System.out.println("Addition is:"+ img.add(99,101));**

**System.out.println("-----------------------------------");**

**System.out.println ("Subtraction is:"+del.sub(200,100));**

**System.out.println("-----------------------------------");**

**System.out.println("Multiplication is:"+zym.mult(50,4));**

**System.out.println("-----------------------------------");**

**System.out.println("Division is:"+obj.div(1000,100));**

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

**}**

**}**

**Output:**

**positive case:**

A screenshot of a computer program

AI-generated content may be incorrect.

**Error table:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Error type** | **Reason for error** | **rectification** |
| **1.** | **Logical error** | **Incorrect formula** | **Correcting formula** |
| **2.** | **Runtime errror** | **Incorrect path** | **Using correct path** |
| **3.** | **Syntax error** | **Incorrect symbols** | **Using correct symbols** |

**Classs diagram:**

|  |
| --- |
| **Class : addition** |
| **+ add( int a ,int b)** |
|  |

|  |
| --- |
| **Class : subtraction** |
| **+ sub( int a ,int b)** |

|  |
| --- |
| **Class : multiplication** |
| **+ multi( int a ,int b)** |

|  |
| --- |
| **Class : divission** |
| **+ div( int a ,int b)** |

**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, bikes and truck and they need a program to Store details about each vehicle, such as brand and speed Cars should have an additional property: number of doors Bikes should have a property indicating whether they have gears or not The system should also include a function to display details about each vehicle And indicate when a vehicle is starting**

Program:

class vehicle{

String brand;

int speed;

public vehicle(String brand,int speed){

this.brand=brand;

this.speed=speed;

}

public static void main(String[] args) {

car obj1=new car("maruti",140,70);

bike obj2=new bike("unicorn",80,true);

truck obj3=new truck("zimban",120,20);

}

}

class car extends vehicle{

int noofdoors;

public car(String brand, int speed,int noofdoors) {

super(brand, speed);

this.noofdoors=noofdoors;

System.out.println(brand);

System.out.println(speed);

System.out.println(noofdoors);

}

}

class bike extends vehicle{

boolean gears;

public bike(String brand,int speed,boolean gears){

super(brand, speed);

this.gears=gears;

System.out.println(brand);

System.out.println(speed);

System.out.println(gears);

}

}

class truck extends vehicle{

int weight;

public truck(String brand,int speed,int weight){

super(brand,speed);

this.weight=weight;

System.out.println(brand);

System.out.println(speed);

System.out.println(weight);

}

public static void main(String[] args) {

car obj1=new car("maruti",143,70);

bike obj2=new bike("unicorn",80,true);

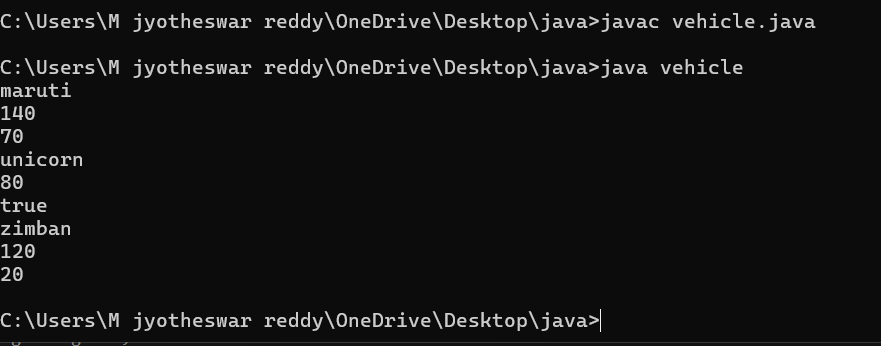
truck obj3=new truck("zimban",120,20);

}

}

**Out put:**

1. **positive case :**

****

Error table:

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Error type** | **Reason for error** | **rectification** |
| **1.** | **Logical error** | **Incorrect formula** | **Correcting formula** |
| **2.** | **Runtime errror** | **Incorrect path** | **Using correct path** |
| **3.** | **Syntax error** | **Incorrect symbols** | **Using correct symbols** |

Class diagram:

|  |
| --- |
| Class vehicle |
| + brand:string  + speed :int |
| Vehicle(String brand ,int speed) |

|  |
| --- |
| Class car |
| +noofdoors:int |
| car(String brand ,int speed ,noofdoors) |

|  |
| --- |
| Class bike |
| + gear : bool |
| bike(String brand ,int speed,Boolean grars) |

|  |
| --- |
| Class truck |
| + weight:int |
| truck(String brand ,int speed) |

**Important points:**

1.Inheritance:

Obtaining the qualities from parent class.

2.hirechial inheritance: different child class obtaing qualities from one parent.

Week- 6:

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 a about car**

**Code:**

class vehicle {

String brand;

int speed;

int capacity;

Vehicle(String brand, int speed, int capacity) {

this.brand = brand;

this.speed = speed;

this.capacity = capacity;

}

public void displayInfo() {

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

System.out.println("--------------------------------");

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

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

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

}

}

class Car extends Vehicle {

Car(String brand, int speed, int capacity) {

super(brand, speed, capacity);

}

public void displayInfo() {

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

System.out.println("--------------------------------");

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

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

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

}

}

public class world {

public static void main(String[] args) {

Vehicle vehicle = new Vehicle("bmw", 150, 5);

vehicle.displayInfo();

System.out.println();

Car car = new Car("honda", 200, 7);

System.out.println("car details are given below :");

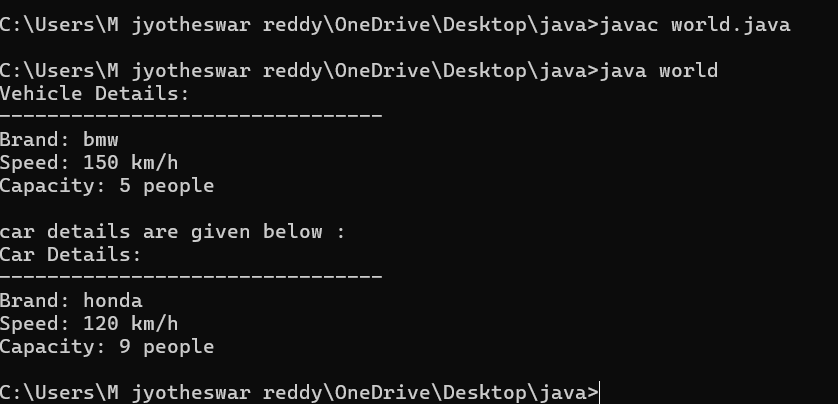
car.displayInfo();

}

}

OUT PUT :

1.POSITIVE CASE :



CLASS DIAGRAM:

|  |
| --- |
| CLASS VEHICLE |
| ATTRIBUTES :  + SPEED  + BRAND  + CAPACITY |
| +DISPLAYINFO():void  + VEHICLE (INT ,SPEED INT CAPACITY ,STRING BRAND) |

|  |
| --- |
| class car |
| car(int speed ,int capacity ,string brand)  displayinfo() : void |

Important points:

**1.Inheritance** allows the Car class to reuse the Vehicle class

functionality without repeating code.

2. **Method Overriding** enables the Car class to provide its own implementation of the displayInfo() method.

3. **Polymorphism** makes the code flexible, allowing different classes

(e.g., Car, Bike) to provide customized behavior for displayInfo().

2.AIM:

A college is developing an automated admission system that verifies student eligibility for UG and PG programs. Each program has different eligibility

criteria based on the student’s percentage in their previous Qualification:

UG admissions require a minimum of 60%

PG admissions require a minimum of 70%

Code:

class University {

String name;

int percentile;

University(String name, int percentile) {

this.name = name;

this.percentile = percentile;

}

public void office() {

System.out.println("Admission criteria are mentioned below.");

}

}

class PG extends University {

PG(String name, int percentile) {

super(name, percentile);

}

public void office() {

if (percentile >= 70) {

System.out.println(name + " is eligible for PG admission.");

} else {

System.out.println(name + " is not eligible for PG admission.");

}

}

}

class UG extends University {

UG(String name, int percentile) {

super(name, percentile);

}

public void office() {

if (percentile >= 60) {

System.out.println(name + " is eligible for UG admission.");

} else {

System.out.println(name + " is not eligible for UG admission.");

}

}

}

public class admissions {

public static void main(String[] args) {

UG ugStudent = new UG("uday", 95);

PG pgStudent = new PG("jyotheeswar", 96);

ugStudent.office();

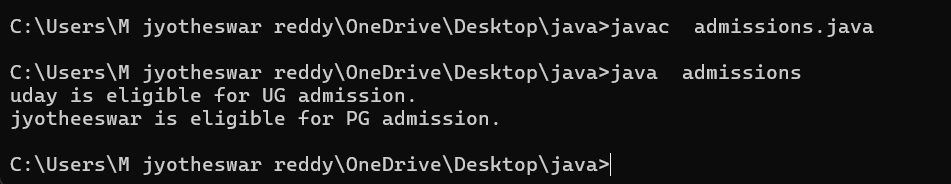
pgStudent.office();

}

}

Output :

positive case :



Error table:

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Error type** | **Reason for error** | **rectification** |
| **1.** | **Logical error** | **Incorrect formula** | **Correcting formula** |
| **2.** | **Runtime errror** | **Incorrect path** | **Using correct path** |
| **3.** | **Syntax error** | **Incorrect symbols** | **Using correct symbols** |

Class diagram :

|  |
| --- |
| University |
| - name: String  - percentile: int |
| + University(String, int)  + office(): void |

|  |
| --- |
| PG |
| + PG(String, int)  + office(): void |

|  |
| --- |
| UG |
| + UG(String, int)  + office(): void |

|  |
| --- |
| admissions |
| + main(String[]): void |

IMPORTANT POINTS:

1.**Inheritance** allows PG and UG classes to reuse the University class’s attributes and constructor.

2. **Method Overriding** customizes the office() method in PG and UG classes to implement specific admission criteria.

3. **Polymorphism** enables the same office() method to behave

differently for PG and UG objects based on their typ

3.AIM :

Create a Shape class with a method calculateArea() that is overloaded forDifferent shapes. Then, create a subclass circle that overrides the calculateArea() method for a circle

CODE:

class Shape {

public void calculateArea(int length, int width) {

int area = length \* width;

System.out.println("Area of Rectangle: " + area);

}

public void calculateArea(double base, double height) {

double area = 0.5 \* base \* height;

System.out.println("Area of Triangle: " + area);

}

public void calculateArea(int side) {

int area = side \* side;

System.out.println("Area of Square: " + area);

}

}

class Circle extends Shape {

public void calculateArea(double radius) {

double area = 3.14 \* radius \* radius;

System.out.println("Area of Circle: " + area);

}

}

public class area {

public static void main(String[] args) {

Shape shape = new Shape();

shape.calculateArea(5, 60);

shape.calculateArea(7.0, 34.0);

shape.calculateArea(9);

Circle circle = new Circle();

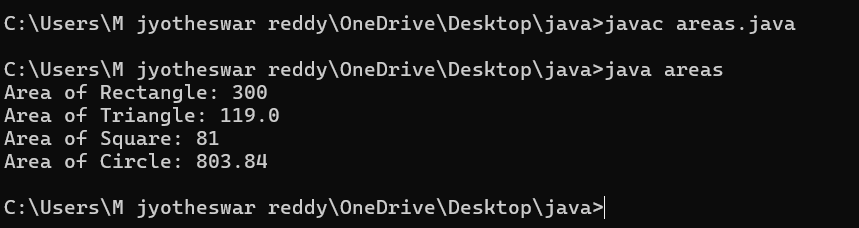
circle.calculateArea(16.0);

}

}

Out put:

positive case :



Error table:

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Error type** | **Reason for error** | **rectification** |
| **1.** | **Logical error** | **Incorrect formula** | **Correcting formula** |
| **2.** | **Runtime errror** | **Incorrect path** | **Using correct path** |
|  |  |  |  |
| **3.** | **Syntax error** | **Incorrect symbols** | **Using correct symbols** |

**Claas diagram :**

|  |
| --- |
| **Class shapes** |
| **+ calculateArea(int, int): void**  **+ calculateArea(double, double): void**  **+ calculateArea(int): void** |

|  |
| --- |
| Class circle |
| + calculateArea(double): void |

|  |
| --- |
| Area |
| + main(String[]): void |

IMPORTANT POINTS :

**1.Method Overloading** allows the Shape class to calculate areas for different shapes (rectangle, triangle, square) using multiple calculateArea() methods with different parameters.

**2.Method Overriding** in the Circle class provides a custom implementation of calculateArea() for calculating the area of a circle.

**3.Polymorphism** enables objects of different types (e.g., Shape, Circle)

to call the appropriate version of calculateArea() based on the object type and input parameters.