**23CSE111**

OOPS

**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: A. Gowtham Reddy**

**Roll No: AV.SC.U4CSE24014**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 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 | a) Write a java program to calculate the area of the rectangle.  b) Write a java program to convert temperature from Celsius to  Fahrenheit and viceversa.  c) Write a java program to calculate the simple intrest.  d) Write a java program to find largest of three numbers using  ternary operator.  e) Write a java program to find factorial of 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”. |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

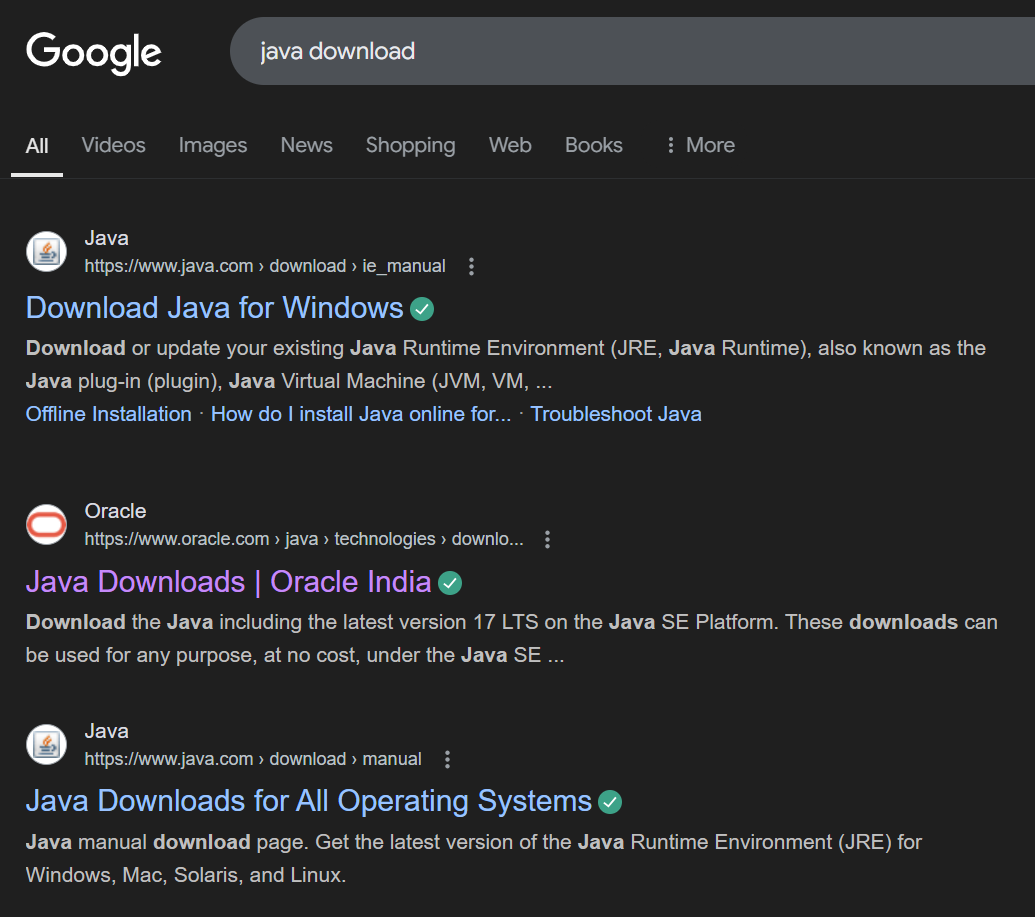
Week 1:-

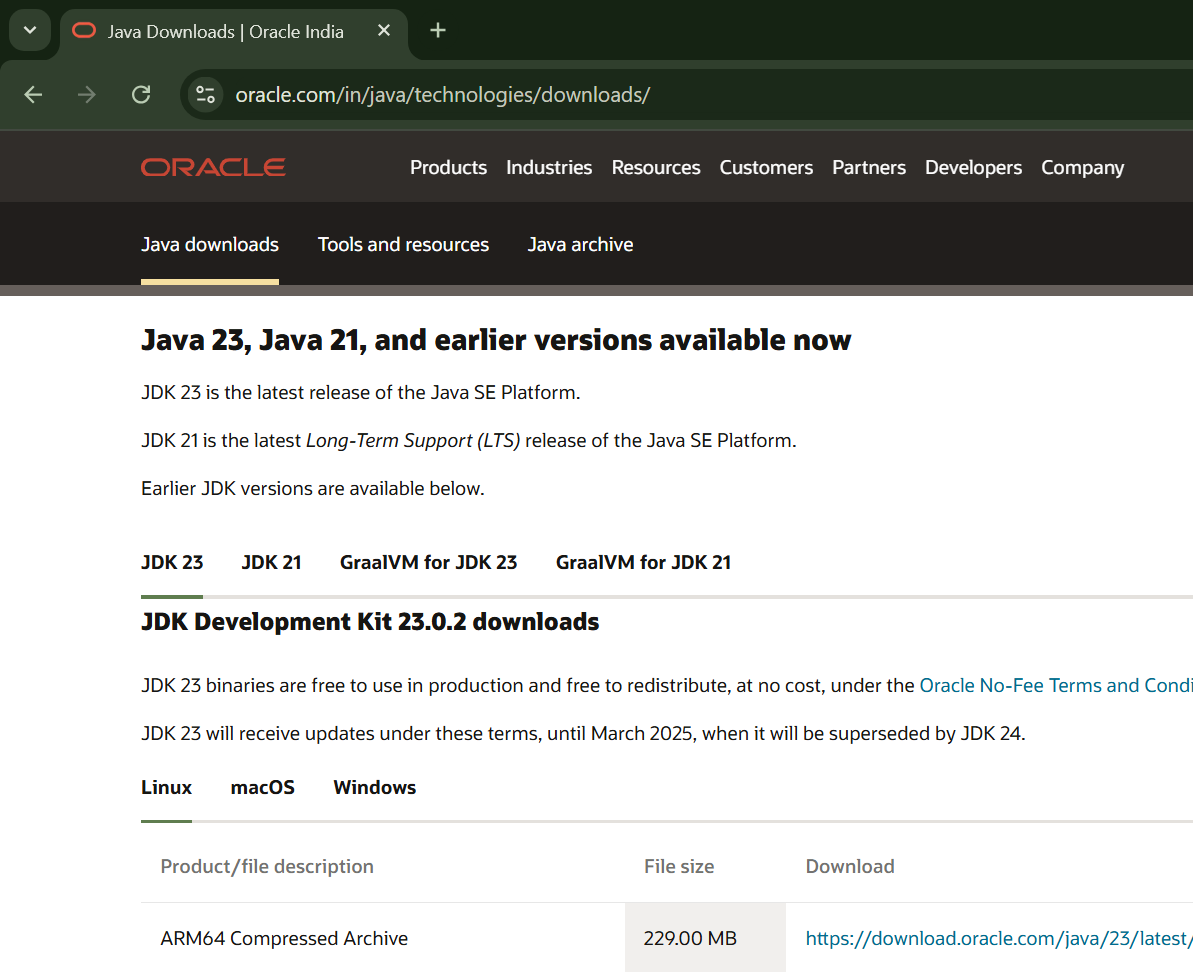
Program-1:-

Aim:-Download and Instal the Java Software

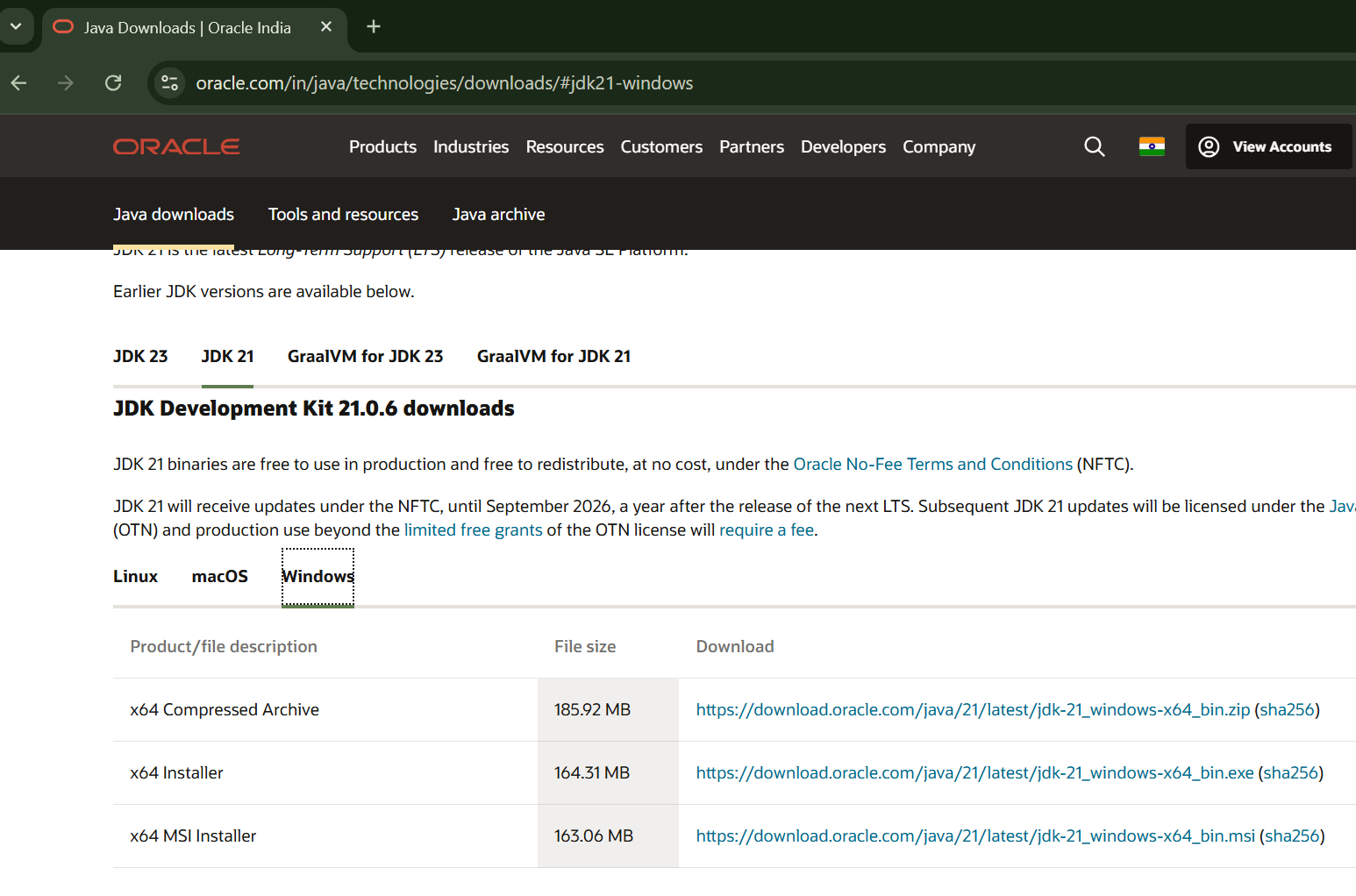
Procedure

Step-1:- Type Java download in search

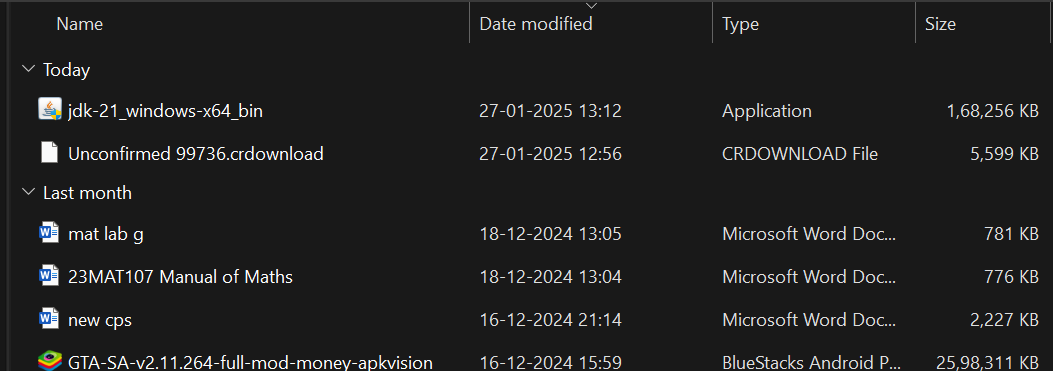


Step-2:-click on oracle java download and enter into oracle website 

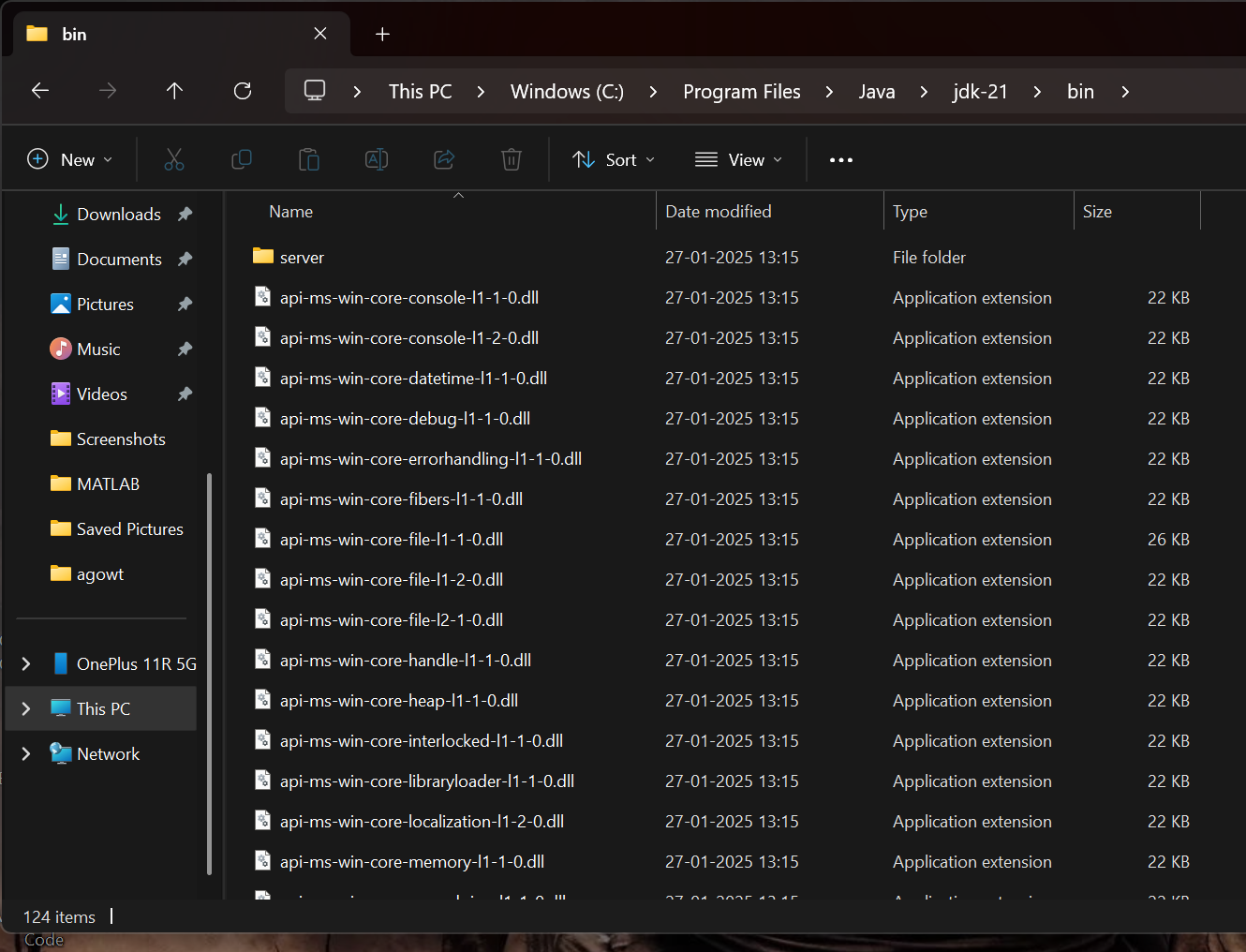
Step-3:-click on JDK21 and click on windows and later click on x64 instalier link to download



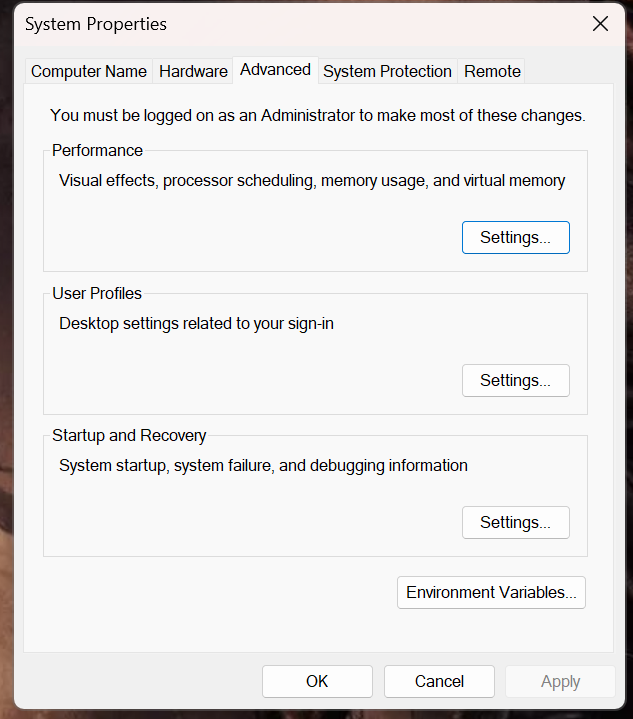
Step-4:-After completing download click on it’s file and then give permission to instal



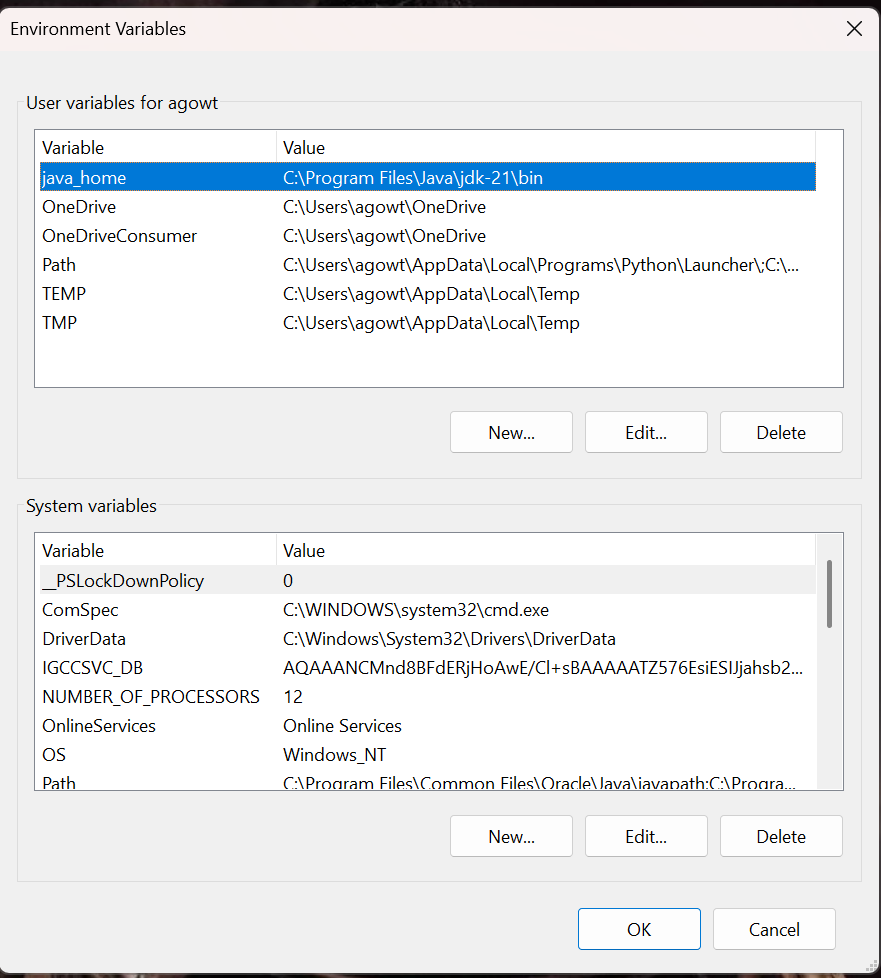
Step-5:-Then go to (This pc) in that click (windows{c}) in that click (Program files) in that click (Java) in that click (jdk-21) in that click (bin)



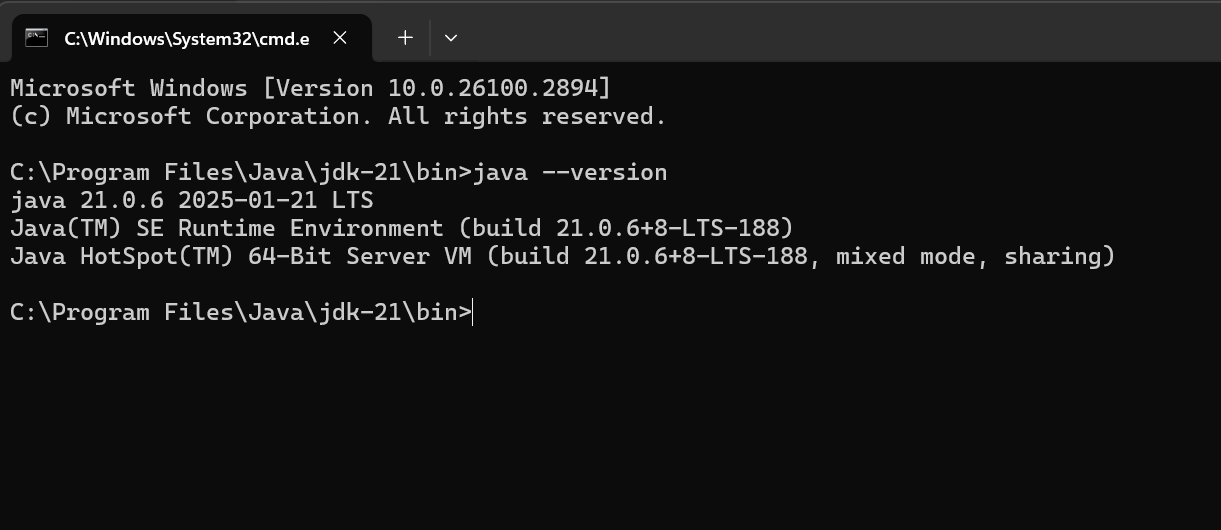
Step-6:-Select and copy path of opening the file and then press windows and search System Environmental



Step-7:-After opening Environment variables then past path of opening file in user variable and click on ok



Step-8:-To verify version open CMD and type java --version



Program : 2

Aim:-write a java program to print[welcome to java programming]

Input:-

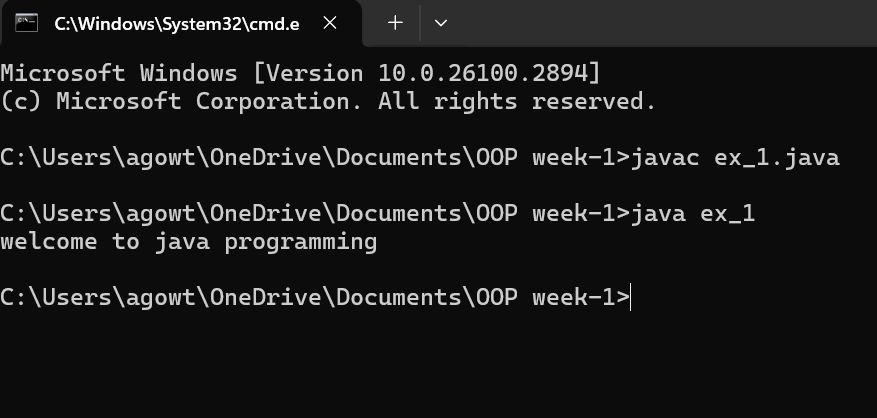
class ex\_1{

public static void main(String[] args){

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

}

}

Output:- 

Program : 3

Aim:-write a java program that prints name, roll no, section of the student

Input:-

public class my\_profile{

public static void main(String args[]) {

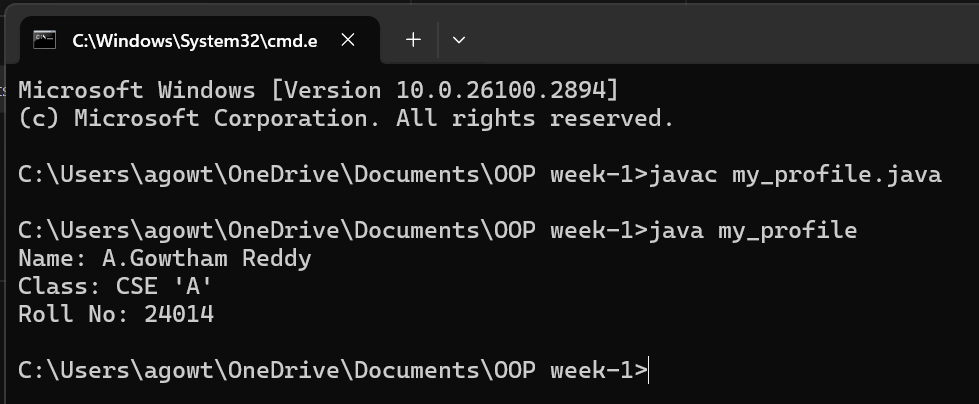
System.out.println("Name: A.Gowtham Reddy");

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

System.out.println("Roll No: 24014");

}

}

Output:-

Week 2:-

Program-1:-

Aim:-

Procedure

* Program : 1

**Q) Write a java program to calculate the area of the rectangle.**

import java.util.Scanner;

class rectangle{

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");

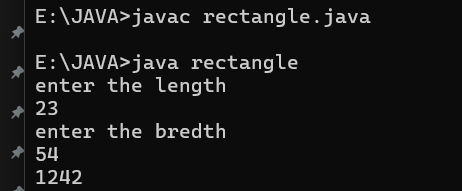
int bred=input.nextInt();

int area=len\*bred;

System.out.println(area);

}}

OUTPUT



|  |  |  |
| --- | --- | --- |
| **S.NO** | **Errors** | **Rectification** |
| 1 | ; | ; is expected at end |
| 2 | area | Declaration of int type variable |

* Program : 2

**Q) Write a java program to convert temperature from Celsius to Fahrenheit and viceversa.**

import java.util.Scanner;

class temperature{

public static void main(String[]args){

Scanner input =new Scanner(System.in);

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

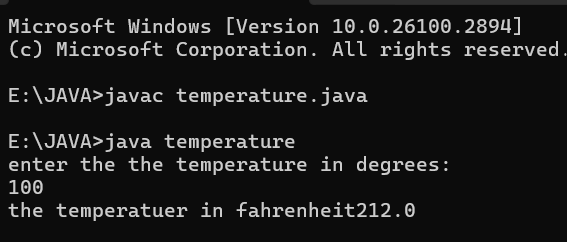
double deg=input.nextDouble();

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

}

}

OUTPUT



|  |  |  |
| --- | --- | --- |
| **S.NO** | **Errors** | **Rectification** |
| 1 | ; | ; is expected at end |
| 2 | Input.close(); | The input is expected to be closed. |

* Program : 3

**Q) Write a java program to calculate the simple intrest**.

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();

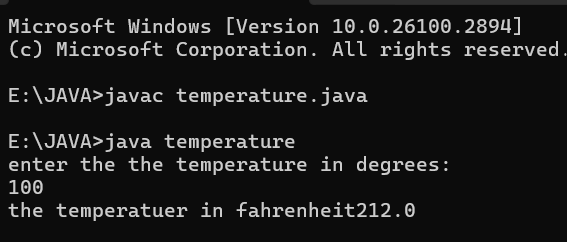
float si=(p\*t\*r)/100;

System.out.println(si);

}

}

OUTPUT



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

* Program : 4

**Q) Write a java program to find largest of three numbers using ternary operator.**

import java.util.Scanner;

class largest{

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();

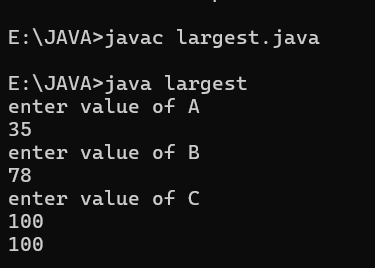
int large=(a>b)?((a>c)?a:c):((b>c)?b:c);

System.out.println(large);

}

}

OUTPUT



|  |  |  |
| --- | --- | --- |
| **S.NO** | **Errors** | **Rectification** |
| 1 | ? | Checks the condition |
| 2 | : | Comparing between two variables |

* Program : 5

**Q) Write a java program to find factorial of number.**

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++){

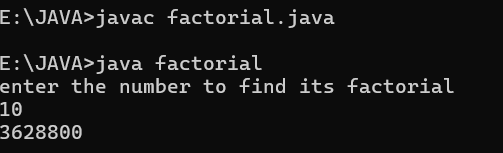
sum=sum\*i;}

System.out.println(sum);

}

}

OUTPUT



|  |  |  |
| --- | --- | --- |
| **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. |

WEEK-3

* Program : 1

Q) **Write a java program with the following instructions**.

1. Create a class with name car.
2. Create four attributes named car\_colour,car\_brand,fuel\_type,top\_speed.
3. Create three method named “Start\_Racing”,”End\_Race”.{ }
4. Create three objects named Car1,Car2,Car3.
5. 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() |

// 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();

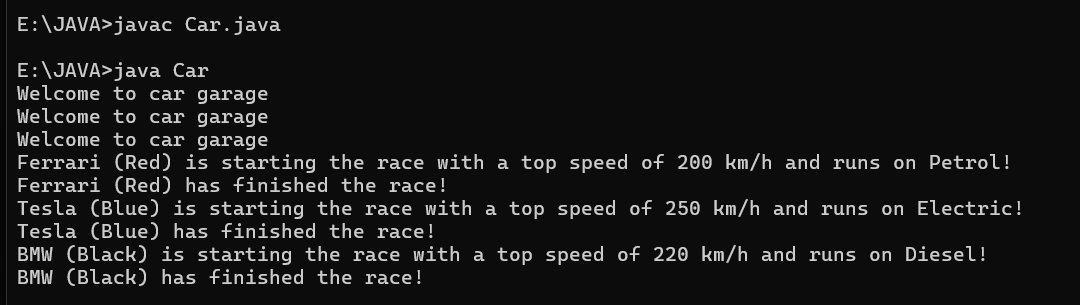
car3.startRacing();

car3.endRace();

}

}

Output:



|  |  |  |
| --- | --- | --- |
| **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. |

* Program : 2

Q ) 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 |

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);

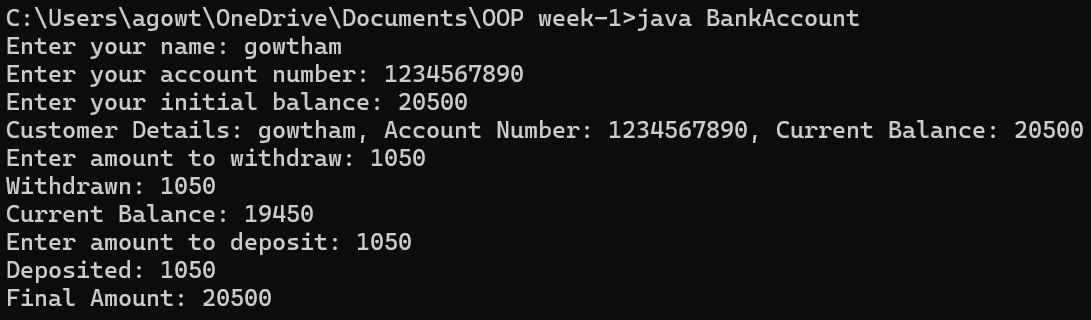
// Close the scanner

scanner.close();

}

}

Output:



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

WEEK-4

* 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 |

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();

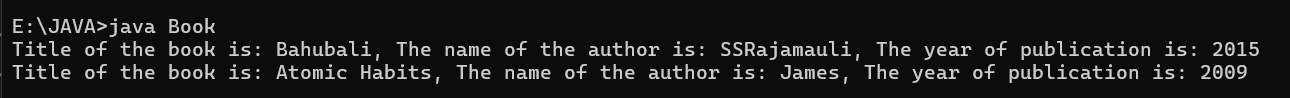
book2.display();

}

}

// class ends here

Output:



|  |  |  |
| --- | --- | --- |
| **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. |

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 |

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);

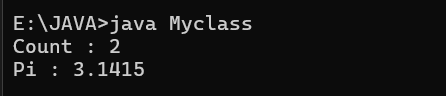
System.out.println("Pi : " + c1.pi);

}

}

// class ends

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-5

* Program : 1

Q)  **Create a calculator using the operations including addition using subtraction**

**multiplication and division using multilateral inheritance and display thr 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) |

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); 29

}

}

class aacalc extends acalc {

f loat 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();

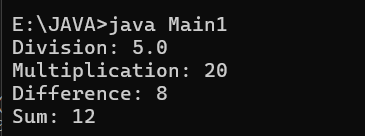
c.mult();

c.add();

}

}

Output:



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

* 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);

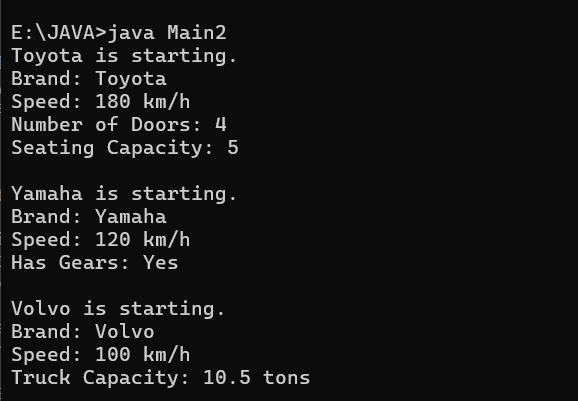
truck.start();

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

* Program : 1

Q)**Write a java program to create a vehicle class with a method override this method in the**

**class subclass to provide. Specific information about the cars that is “car company , model ,**

**price ,seating capacity, petrol or diesel, true or false”**

Class Diagram:

|  |
| --- |
| Vehicle |
|  |
| * displayInfo() |

|  |
| --- |
| Car |
| - carCompany: String  - model: String  - price: double  - seatingCapacity: int  - fuelType: String  - isAvailable: boolean |
| + displayInfo()   + Car(company: String, model: String, price: double, seatingCapacity: int, fuelType: String, isAvailable: boolean) |

// Base class

class Vehicle {

// Method to be overridden

public void displayInfo() {

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

}

}

// Subclass

class Car extends Vehicle {

private String company;

private String model;

private double price;

private int seatingCapacity;

private String fuelType; // "Petrol" or "Diesel"

private boolean isAvailable; // true or false

// Constructor

public Car(String company, String model, double price, int seatingCapacity, String fuelType, boolean isAvailable) {

this.company = company;

this.model = model;

this.price = price;

this.seatingCapacity = seatingCapacity;

this.fuelType = fuelType;

this.isAvailable = isAvailable;

}

// Overriding the displayInfo method

@Override

public void displayInfo() {

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

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

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

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

System.out.println("Fuel Type: " + fuelType);

System.out.println("Available: " + (isAvailable ? "Yes" : "No"));

}

}

// Main class to test the functionality

public class VehicleMain {

public static void main(String[] args) {

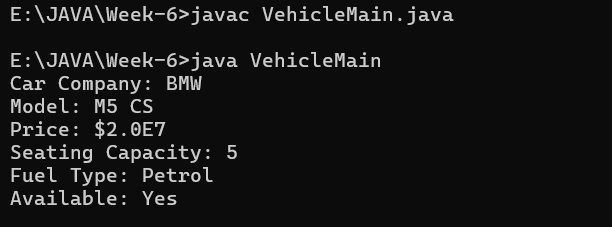
Vehicle myCar = new Car("BMW", "M5 CS", 20000000, 5, "Petrol", true);

myCar.displayInfo();

}

}

Output:



|  |  |  |
| --- | --- | --- |
| **S.NO** | **Errors** | **Rectification** |
| 1 | 2CR | We must give the price in the form of int type not in the string type |
| 2 | boolean | We have to give Boolean value in the valid form. |

* Program : 2

Q)**Write a java program that a Collage is developing a automated admission system that**

**verifies student eligibility for UG and PG programs each program has a different eligibility**

**criteria based on the students percentage in the previous qualification.**

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

Class Diagram:

|  |
| --- |
| Admission |
| +eligibility():void |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| |  | | --- | | UG | | + eligible(): void | | |  | | --- | | PG | | + eligible(): void | |

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 CollageMain {

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);

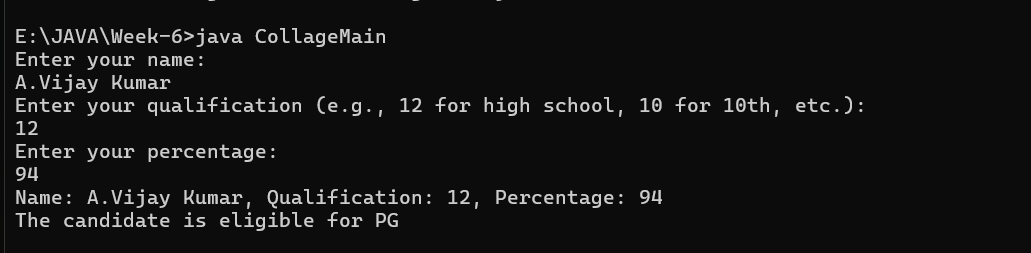
}

candidate.Eligibility();``

}

}

Output:



|  |  |  |
| --- | --- | --- |
| **S.NO** | **Errors** | **Rectification** |
| 1 | Class Naming | Rename the class to CollegeMain to reflect the correct context and adhere to naming conventions. |
| 2 | Output Formatting | Improve output messages for clarity and professionalism. Format the output for better readability. |

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

* **Add two doubles**
* **Add two doubles**
* **Add three Integers**

Class Diagram:

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

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

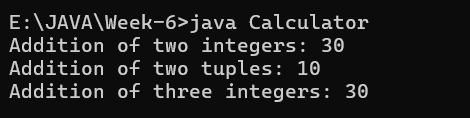
int result3 = calc.add(5, 10, 15);

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

}

}

Output:



|  |  |  |
| --- | --- | --- |
| **S.NO** | **Errors** | **Rectification** |
| 1 | } | Ending the class and main method is required. |
| 2 | Output Formatting | Improve output messages for clarity and professionalism. Format the output for better readability. |

Q)**Write a java program 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.**

Class Diagram:

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

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

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);

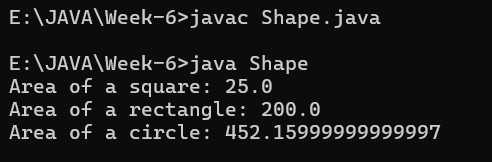
double circle = shape.areaOfCircle(12);

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

}

}

Output:



|  |  |  |
| --- | --- | --- |
| **S.NO** | **Errors** | **Rectification** |
| 1 | } | Ending the class and main method is required. |
| 2 | Output Formatting | Improve output messages for clarity and professionalism. Format the output for better readability. |