23CSE111

OBJECT ORIENTED PROGRAMMING

LAB MANUAL



Department of Computer and Science Engineering

Amrita School of Engineering

Amrita Vishwa Vidyapeetham, Amaravati Campus

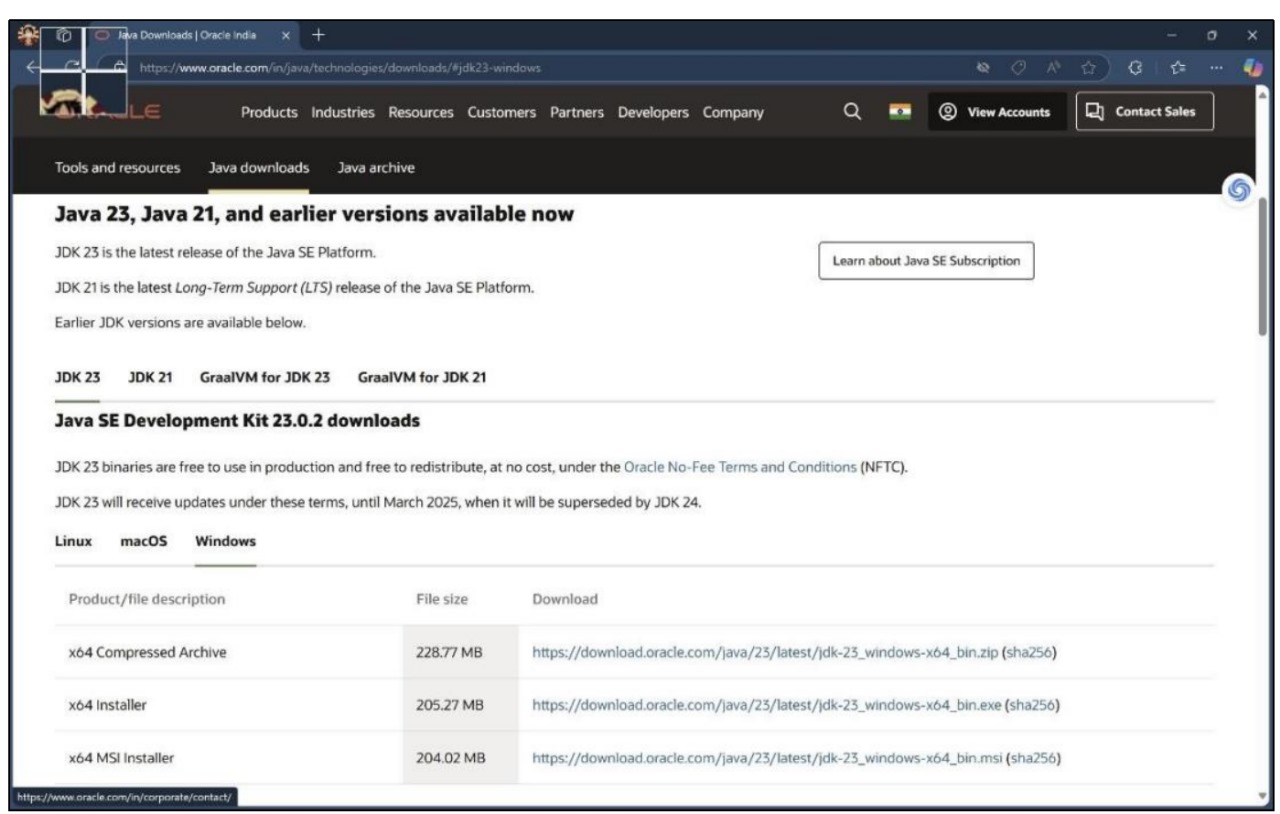
Name:-P SANTHAN

Sec:-CSE-A

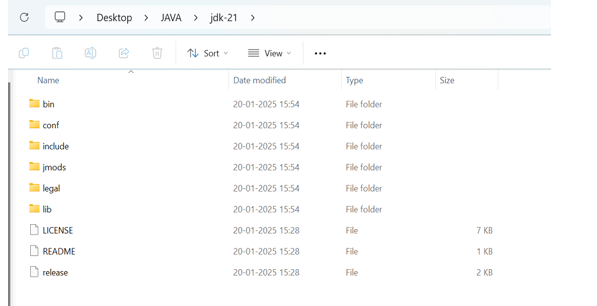
Roll.No:-AV.SC.U4CSE24305

Verified By:

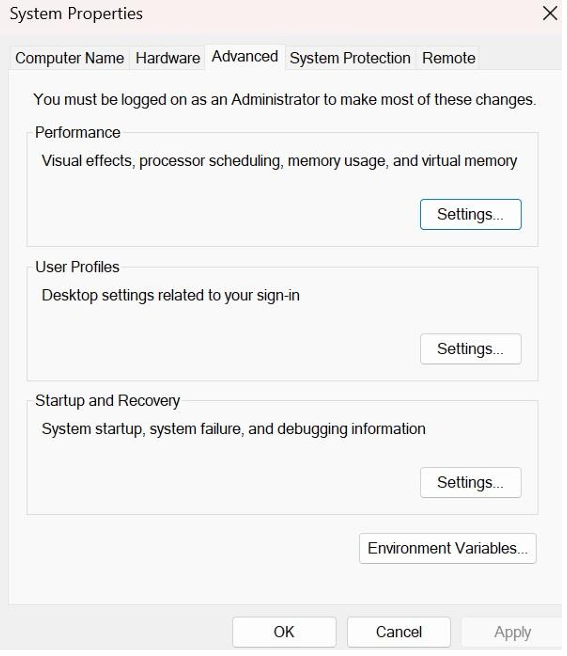
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| S.NO | Programs | Date | Page No | Signature |
| Week 1 |  | 27-01-2025 |  |  |
| 1 | Write the steps to download and install Java. |  |  |  |
| 2 | Write a java program to print the message “Welcome to java programming”. |  |  |  |
| 3 | Write a java program that prints student details(name, roll number and section of a student). |  |  |  |
| Week 2 |  | 3-02-2025 |  |  |
| 1 | Write a java program to calculate the area of a rectangle. |  |  |  |
| 2a) | Write a program to convert temperature from Celsius to Fahrenheit |  |  |  |
| 2b) | Write a program to convert temperature from Fahrenheit to Celsius. |  |  |  |
| 3 | Write a program to calculate the simple interest |  |  |  |
| 4 | Write a program to find the largest of three numbers using ternary operator |  |  |  |
| 5 | Write a program to find the factorial of a number |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

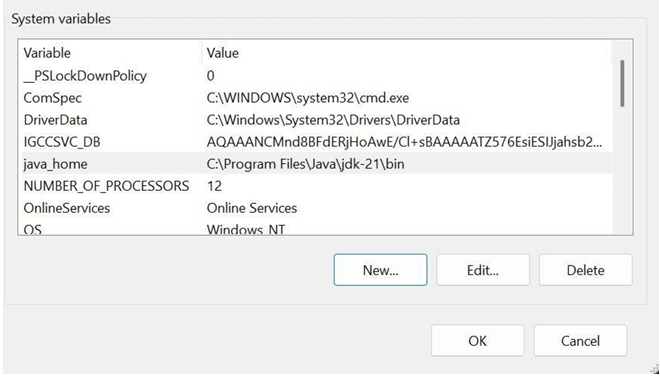
 ***WEEK –1*** Task:-1) Write the steps to download and install Java. Aim: To download and install java. Procedure: i. Visit oracle.com website to download Java.

ii. Download the version which supports LTS (JDK 21) x64 installer for windows. iii. Install and copy the path.

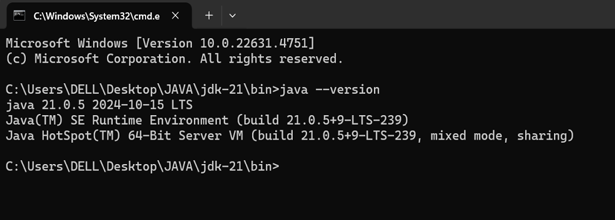


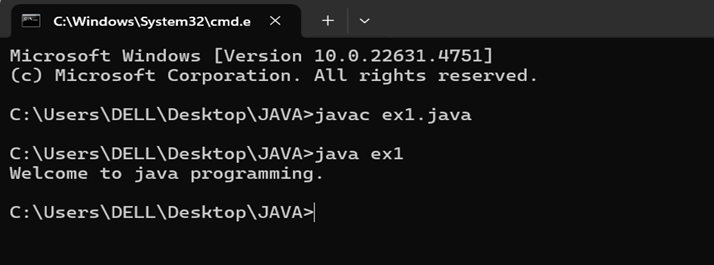
iv. Open environmental variables and add a new file with path.





v. Verify java version in command window.



Task:-2) Write a java program to print the message “Welcome to java programming”. *Code:-* public class Main { public static void main(String[] args){ System.out.println("Welcome to java programming."); } } Output:- 

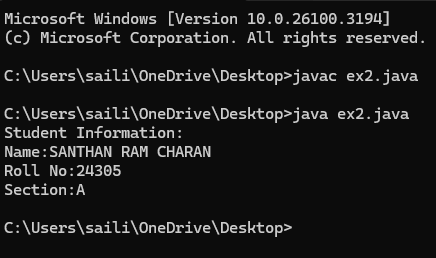
*Error:-*

|  |  |  |
| --- | --- | --- |
| S.No | Expected Error | Reason |
| 1 | ; | ; is expected at end |
| 2 | S | Capital S is expected for String and System. |

Task:-3)Write a java program that prints student details(name, roll number and section of a student).

Code: class ex2{ public static void main(String[] args){ String name = "SANTHAN RAM CHARAN "; int rollNo = 24305; String section = "A"; System.out.println("Student Information:"); System.out.println("Name:" + name); System.out.println("Roll No:" + rollNo); System.out.println("Section:" + section);

} } **Output:-**

****

|  |  |  |
| --- | --- | --- |
| S.No | Expected Error | Reason |
| 1 | S | Capital S is expected for String and System. |

**WEEK-2**

**1)JAVA PROGRAM TO CALCULATE THE AREA OF THE RECTANGLE:**

**CODE:**

import java.util.Scanner;

public class area {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

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

double length = scanner.nextDouble();

System.out.print("Enter the width of the rectangle: ");

double width = scanner.nextDouble();

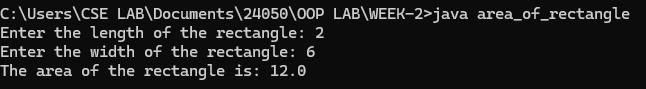
double area = length \* width;

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

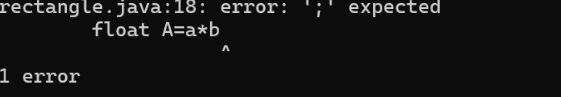
scanner.close();

}

}

**OUTPUT:**

Error:-



|  |  |  |
| --- | --- | --- |
| S.No | Expected Error | Reason |
| 1 | ; | That is  semicolon( ; )  is not present at the  end of the  expression. |

**2)**

**A)JAVA PROGRAM TO CONVERT TEMPERATURE FROM CELCIUS TO FAHRENHEIT:**

**CODE:**

import java.util.Scanner;

public class celcius\_to\_fahrenheit {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

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

double celsius = scanner.nextDouble();

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

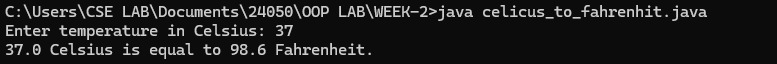
System.out.println(celsius + " Celsius is equal to " + fahrenheit + " Fahrenheit.");

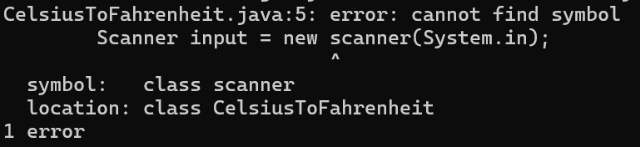
scanner.close();

}

}

**OUTPUT:**



Error:-

|  |  |  |
| --- | --- | --- |
| S.No | Expected Error | Reason |
| 1 | S | Capital S is expected for String and System. |

**B)JAVA PROGRAM TO CONVERT FAHRENHEIT TO CELCIUS :**

**CODE:**

import java.util.Scanner;

public class ftoc {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

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

double fahrenheit = scanner.nextDouble();

double celsius = (fahrenheit - 32) \* 5 / 9;

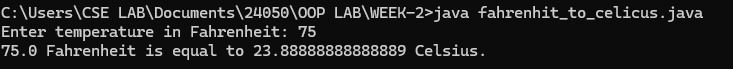
System.out.println(fahrenheit + " Fahrenheit is equal to " + celsius + " Celsius.");

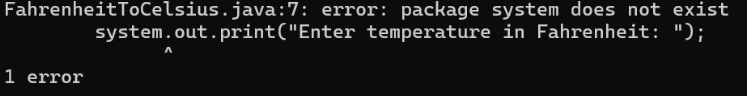
scanner.close();

}

}

**OUTPUT:**



Error:-

|  |  |  |
| --- | --- | --- |
| S.No | Expected Error | Reason |
| 1 | S | Capital S is expected for String and System. |

**3)JAVA PROGRAM TO CALCULATE THE SIMPLE INTEREST:**

**CODE:**

import java.util.Scanner;

public class SimpleInterest {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

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

double principal = scanner.nextDouble();

System.out.print("Enter the rate of interest (in %): ");

double rate = scanner.nextDouble();

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

double time = scanner.nextDouble();

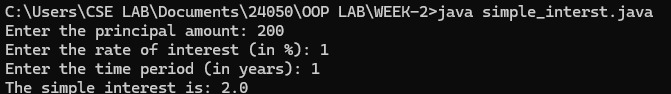
double simpleInterest = (principal \* rate \* time) / 100;

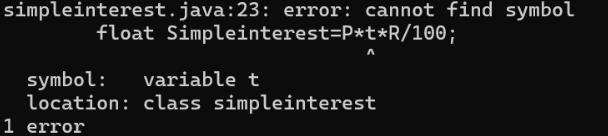
System.out.println("The simple interest is: " + simpleInterest);

scanner.close();

}

}

**OUTPUT:**

Error:-

|  |  |  |
| --- | --- | --- |
| S.No | Expected Error | Reason |
| 1 | T | Capital T is expected for String and System. |

**4)JAVA PROGRAM TO FIND THE LARGEST AMONG THREE NUMBER USING TERNARY OPERATOR:**

**CODE:**

import java.util.Scanner;

public class largest3 {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

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

int num1 = scanner.nextInt();

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

int num2 = scanner.nextInt();

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

int num3 = scanner.nextInt();

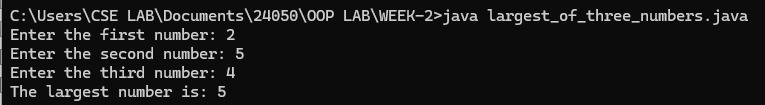
int largest = (num1 >= num2 && num1 >= num3) ? num1 : (num2 >= num1 && num2 >= num3) ? num2 : num3;

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

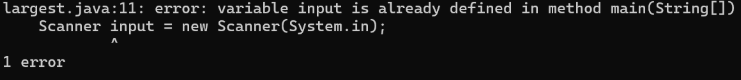
scanner.close();

}

}

**OUTPUT:**

**Error:-**



|  |  |  |
| --- | --- | --- |
| S.No | Expected Error | Reason |
| 1 | main(String []). | Variable input is already defines in method main(String []). |

**5)JAVA PROGRAM TO FIND FACTORIAL OF A NUMBER:**

**CODE:**

import java.util.Scanner;

public class factorial {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter a number to find its factorial: ");

int number = scanner.nextInt();

long factorial = 1;

if (number < 0) {

System.out.println("Factorial is not defined for negative numbers.");

} else {

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

factorial \*= i;

}

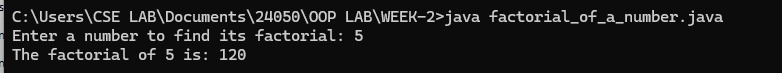
System.out.println("The factorial of " + number + " is: " + factorial);

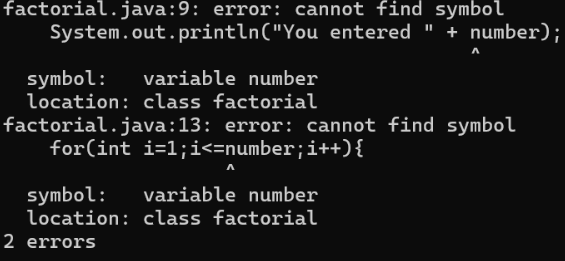
}

scanner.close();

}

}

**OUTPUT:**

Error:- 

WEEK – 3

1) Create the java program with the following instructions i) Create a class with name Car ii) Create 4 attributes named Car\_Color , Car\_brand, fuel\_type, mileage iii) Create 3 method named Start( ) , Stop( ), Service( ) iv) Create 3 objects Car1 , Car2 , Car3 v) Create a constructor which should print “Welcome to Car Garage” public class Car{

    public String carColor;

    private String carBrand;

    private String fuelType;

    public int mileage;

    Car(String carColor , String carBrand , String fuelType , int mileage){

    this.carColor =  carColor;

    this.carBrand = carBrand;

    this.fuelType = fuelType;

    this.mileage = mileage;

    System.out.println(carColor + " " + carBrand + " " + fuelType + " " +

mileage);

    }

    public void Start(){

    System.out.println("The car has just started");

    }

    public void Stop(){

    System.out.println("The car has just stopped");

    }

    public void Service(){

    System.out.println("The car is in good condition");

     }

    public static void main(String[] args){

   Car Car1 = new Car("Black","BMW","Petrol",20);

   Car Car2 = new Car("White","AUDI","Diesel",17);

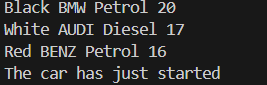
   Car Car3 = new Car("Red","BENZ","Petrol",16);

   Car1.Start();

   }

   }

OUTPUT:-



*Error:-*

|  |  |  |
| --- | --- | --- |
| S.No | Expected Error | Reason |
| 1 | } | } is expected at end of the class |
| 2 | Setting the parameters inside the constructor | We cannot pass the values inside constructor without setting them first |

Class Diagram:

|  |
| --- |
| Car |
| + carColor : String - carBrand : String - fuelType : String + mileage : int |
| + Car() : void + Start( ) : void + Stop( ) : void + Service( ) : void |

2)Write a java program to create a class BackAccount with two methods deposit( ) and withdraw( ) i) In deposit( ) whenever an amount is deposited it has to be updated with current amount ii) In withdraw( ) whenever an amount is withdrawn it has to be less than current amount else print “Insufficient funds”. Code: - public class BankAccount {

    private String Name;

    private int AccNo, CurrBal;

    public BankAccount(String Name, int AccNo, int CurrBal) {

        this.Name = Name;

        this.AccNo = AccNo;

        this.CurrBal = CurrBal;

        System.out.println("The customer is: " + this.Name);

    }

    public int deposit(int dAmt) {

        CurrBal += dAmt;

        return CurrBal;

    }

    public void withdraw(int wAmount) {

        if (wAmount <= CurrBal) {  // Allowing withdrawal if balance is equal

            CurrBal -= wAmount;

            System.out.println("Remaining Balance: " + CurrBal);

        } else {

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

        }

    }

    public static void main(String[] args) {

        BankAccount SaiKrishna = new BankAccount("SaiKrishna", 2000, 15000);

        SaiKrishna.withdraw(13000);  // Should print "Insufficient funds"

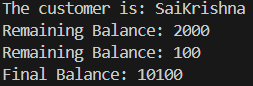
        SaiKrishna.withdraw(1900);   // Should print remaining balance

        int FinalAmount = SaiKrishna.deposit(10000);

        System.out.println("Final Balance: " + FinalAmount);

    }

}

OUTPUT:-

*Error:-*

|  |  |  |
| --- | --- | --- |
| S.No | Expected Error | Reason |
| 1 | } | } is expected at end of the class |
| 2 | Setting the parameters inside the constructor | We cannot pass the values inside constructor without setting them first |

Class Diagram:-

|  |
| --- |
| BankAccount |
| - Name : String - AccNo : String - CurrBal: String |
| + BankAccount( ) : void + deposit( ) : int + withdraw( ) : void |

WEEK4

1. CODE

public class Book{

// atributes

private String title;

private String author;

private int yearofpublication;

// constructor

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

this.title= title;

this.author= author;

this. yearofpublication= yearofpublication;

}

// method to display the details of the book

public void displaydetails(){

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

System.out.println("the autor of book is:"+ author);

System.out.println("year of publication:"+ yearofpublication );

}

// Main method to create and display details of two books

public static void main(String[] args){

// Creating two Book objects

Book book1= new Book("Solo levelling", "p.likhith", 2025);

Book book2= new Book("attack on titan", "p.likhith", 2026);

System.out.println("book 1 Details:");

book1.displaydetails();

System.out.println("book 2 Details:");

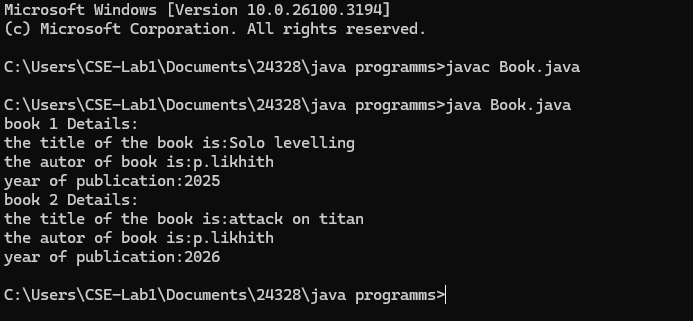
book2.displaydetails();

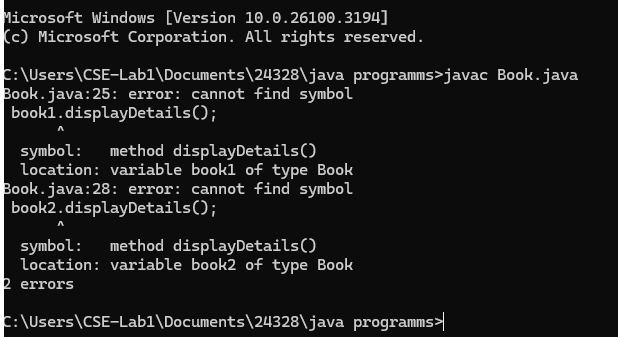
}

}

OUTPUT:

ERRORS:



2) public class MyClass {

// Static variable count of type int, initialized to zero

static int count = 0;

// Constant variable pi of type double, initialized to 314.15

final double pi = 314.15;

// Constructor for MyClass

public MyClass() {

// Increment the count variable each time an object is created

count++;

}

public static void main(String[] args) {

// Creating objects of MyClass

MyClass obj1 = new MyClass();

MyClass obj2 = new MyClass();

MyClass obj3 = new MyClass();

// Print the final value of count and pi

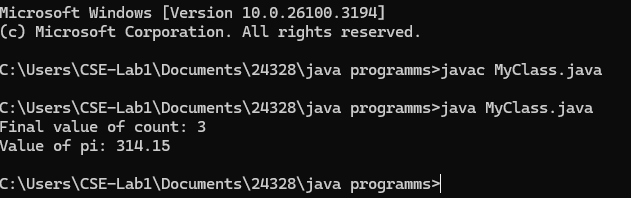
System.out.println("Final value of count: " + count);

System.out.println("Value of pi: " + obj1.pi); // Pi is the same for all objects

}

}

OUTPUT:



A black background with a black square

AI-generated content may be incorrect.

WEEK-5

1) Create a calc using the operations including add, sub, mul, div using multilevel inheritance and display the desired output. CODE:- public 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 operators {

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();

}

}

A screen shot of a computer

AI-generated content may be incorrect.Output:

ERRORS:-

|  |  |  |
| --- | --- | --- |
| S.No. | Expected Error | Reason |
| 1 | .variable | We must mention variable name to call the variable |
| 2 | static | Static variables contain only one value |

CLASS DIAGRAMS:-

A white rectangular object with black text

AI-generated content may be incorrect.

2) A vehicle rental company wants to develop a system that maintains information about different types of vehicles available for rent. The company rents out cars and bikes and they need a program to store details about each vehicle such as brand and speed cars should have an additional properties(attributes)- no.of doors , seating capacity 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 each class should have a constructor .

a) which oops concept is used in the above program ? Explain why it is useful in this scenario

b)If the company decides to add a new type of vehicle truck how would u modify the above program

1) truck should include an additional property called capacity(in tons) 2)create a show truck details method() to display the trucks capacity 3)write a constructor for truck that initializes all the properties c)Implement the truck class and update the main method to create the truck object and also create an object for car and bike subclass. Finally display its details

CODE:-

public class Vehicle {

    String brand;

    int speed;

    Vehicle(String brand, int speed) {

    this.brand = brand;

    this.speed = speed;

    }

    void displayDetails() {

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

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

    }

    void startVehicle() {

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

    }

    }

    // Car subclass

    class Car extends Vehicle {

    int noOfDoors;

    int seatingCapacity;

    Car(String brand, int speed, int noOfDoors, int seatingCapacity) {

    super(brand, speed);

    this.noOfDoors = noOfDoors;

    this.seatingCapacity = seatingCapacity;

    }

    @Override

    void displayDetails() {

    super.displayDetails();

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

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

    }

    }

    // Bike subclass

    class Bike extends Vehicle {

    boolean hasGears;

    Bike(String brand, int speed, boolean hasGears) {

    super(brand, speed);

    this.hasGears = hasGears;

    }

    @Override

    void displayDetails() {

    super.displayDetails();

    System.out.println("Has Gears: " + (hasGears ? "Yes" : "No"));

    }

    }

    // Truck subclass

    class Truck extends Vehicle {

    double capacity;

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

    super(brand, speed);

    this.capacity = capacity;

    }

    void showTruckDetails() {

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

    }

    @Override

    void displayDetails() {

    super.displayDetails();

    showTruckDetails();

    }

    }

    // Main class

 class VehicleRentalSystem {

    public static void main(String[] args) {

    Car car = new Car("Toyota", 150, 4, 5);

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

    Truck truck = new Truck("Volvo", 100, 15.5);

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

    car.displayDetails();

    car.startVehicle();

    System.out.println();

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

    bike.displayDetails();

    bike.startVehicle();

    System.out.println();

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

    truck.displayDetails();

    truck.startVehicle();

    }}

 OUTPUT:-

A screen shot of a computer

AI-generated content may be incorrect.

ERRORS:-

|  |  |  |
| --- | --- | --- |
| S.No. | Expected Error | Reason |
| 1 | .variable | We must mention variable name to call the variable |
| 2 | static | Static variables contain only one value |

CLASS DIAGRAM:-

A diagram of a vehicle

AI-generated content may be incorrect.

WEEK-6

1. Write a java program to create a Vehicle class with displayInfo() method overridden in Car subclass to provide info about car company , model , price, seating and petrol.

CODE:-

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();

    }}

OUTPUT:-

A screenshot of a computer

AI-generated content may be incorrect.

ERRORS:-

|  |  |  |
| --- | --- | --- |
| **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 |

CLASS DIAGRAM:-

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

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

2. 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%

CODE:-

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();``

    }

}

OUTPUT:-

A screen shot of a computer

AI-generated content may be incorrect.

Class Diagram:-

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

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

ERROR:-

|  |  |  |
| --- | --- | --- |
| **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 |

3. Create a calculator class with overloaded methods to perform additions

1.add two integers

2.add two double values

3.add three integers

CODE:-

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:-

A screen shot of a computer

AI-generated content may be incorrect.

CLASS DIAGRAM:-

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

ERRORS:-

|  |  |  |
| --- | --- | --- |
| **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

**AIM:1:**Write a java program to create an abstract class animal with an abstract method called sound create subclass lion and tiger that extend the animal class and implement the sound method to make a specific sound for each animal

Code:

abstract  class Animal {

    abstract void Sound();

}

class lion extends Animal {

    public void Sound(){

        System.out.println("lRoar");

    }}

class Tiger extends Animal {

    public void Sound(){

        System.out.println("tRoar");

          }}

class week7 {

    public static void main(String[] args){

        System.out.println("NAME: P. SANTHAN RAM CHARAN,ROLL NO :AV.SC.U4CSE24305,SEC:CSE-A");

lion l =new lion();

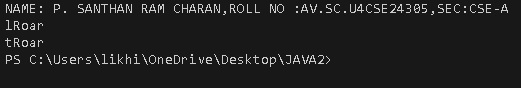
l.Sound();

Tiger t = new Tiger();

t.Sound();

    }}

**OUTPUT:**



IMPORTANT POINTS:

1. Abstract Class

• Animal is an abstract class.

• Abstract classes cannot be instantiated directly.

• It can have abstract methods (methods without a body) which must be overridden in child classes

2. Inheritance

• Lion and Tiger inherit from Animal using the extends keyword.

• They must override the Sound() method of the Animal class.

AIM:2: Write a java program to create an abstract class shape3D with abstract method calculatevolume () and calculatesurfacearea() create subclass sphere and cube that extends the shape3D class and implement the respective methods to calculate the volume and surface area of each shape.

Code:

abstract class Shape3D {

    abstract double calculateVolume();

    abstract double calculateSurfaceArea();

}

class Sphere extends Shape3D {

    private double radius;

    Sphere(double radius) {

        this.radius = radius;

    }

    @Override

    double calculateVolume() {

        return (4.0 / 3.0) \* Math.PI \* radius \* radius \* radius;

    }

    @Override

    double calculateSurfaceArea() {

        return 4 \* Math.PI \* radius \* radius;

    }

}

class  Cube extends Shape3D {

    private double side;

    Cube(double side) {

        this.side = side;

    }

    @Override

    double calculateVolume() {

        return side \* side \* side;

    }

    @Override

    double calculateSurfaceArea() {

        return 6 \* side \* side;

    }

}

class week7q2 {

    public static void main(String[] args) {

        System.out.println("NAME: P. SANTHAN RAM CHARAN ,ROLL NO :AV.SC.U4CSE24305,SEC:CSE-A");

        Sphere s = new Sphere(2.3);

        System.out.println("Volume of Sphere: " + s.calculateVolume());

        System.out.println("Surface Area of Sphere: " + s.calculateSurfaceArea());

        Cube c = new Cube(2.3);

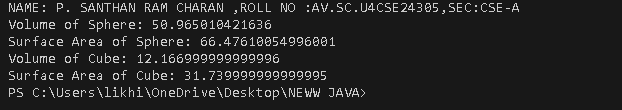
        System.out.println("Volume of Cube: " + c.calculateVolume());

        System.out.println("Surface Area of Cube: " + c.calculateSurfaceArea());

    }

}

OUTPUT:



IMPORTANT POINTS:

1. Abstract Class

• Shape3D is an abstract class.

• It defines two abstract methods: calculateVolume() and calculateSurfaceArea().

• Child classes must implement these methods.

2. Inheritance

• Sphere and Cube classes inherit (extends) from the Shape3D abstract class.

• They inherit the structure but provide their own specific calculations.  
  
AIM:3:write a java program using an abstract class to define a method for pattern printing.

Create an abstract class named pattern printer with an abstract method print pattern (int n) and a concrete method to display the pattern title

Implement two subclass:

* + - * 1. Star pattern: Print a right-angle triangle of stars(\*) .
        2. Number pattern : Print a right-angled triangle of increasing numbers.

In the main ()method, create objects of both subclass and print the pattern of given number of rows.

Code:

abstract class PatternPrinter {

    abstract void printPattern(int n);

    void displayTitle(String title) {

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

        System.out.println("");

    }

  }

  class StarPattern extends PatternPrinter {

    void printPattern(int n) {

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

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

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

            }

            System.out.println();

        }

    }

  }

  class NumberPattern extends PatternPrinter {

    void printPattern(int n) {

        int number = 1;

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

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

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

                number++;

            }

            System.out.println();

        }

    }

  }

  class week7q3 {

    public static void main(String[] args) {

      System.out.println("NAME:P.SANTHAN RAM CHARAN, ROLL NO:24305,SEC:CSE-A ");

        int rows = 5;

        StarPattern star = new StarPattern();

        star.displayTitle("Star Pattern");

        star.printPattern(rows);

        NumberPattern number = new NumberPattern();

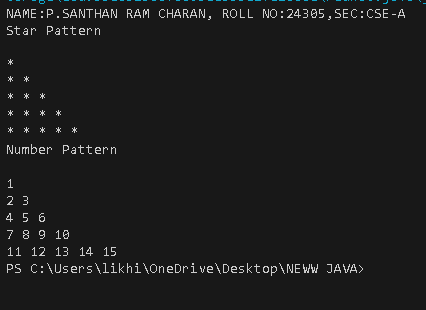
        number.displayTitle("Number Pattern");

        number.printPattern(rows);

    }

  }

  OUTPUT:



IMPORTANT POINTS:

1. Abstract Class

• PatternPrinter is an abstract class.

• It has one abstract method printPattern(int n) — must be overridden in child classes.

• It also has a normal method displayTitle(String title) which prints the title before the pattern.

2. Inheritance

• StarPattern and NumberPattern inherit from PatternPrinter.

• They override printPattern() with their own way of printing the pattern.

WEEK-8

AIM1: )Write a java program to create an interface shape with the getperimeter() method.Create 3classes rectangle,circle and triangle that implements shape interface.Implement getperimeter() method for each of 3classes.

CODE:

interface Shape {

    double getPerimeter();

}

class Rectangle implements Shape {

    private double length;

    private double breadth;

    Rectangle(double length, double breadth) {

        this.length = length;

        this.breadth = breadth;

    }

    public double getPerimeter() {

        return 2 \* (length + breadth);

    }}

class Circle implements Shape {

    private double radius;

    Circle(double radius) {

        this.radius = radius;

    }

    public double getPerimeter() {

        return 2 \* Math.PI \* radius;

    }}

class Triangle implements Shape {

    private double a, b, c;

    Triangle(double a, double b, double c) {

        this.a = a;

        this.b = b;

        this.c = c;

    }

    public double getPerimeter() {

        return a + b + c;

    }}

public class week8q1 {

    public static void main(String[] args) {

        System.out.println("NAME: P SANTHAN RAM CHARAN ,ROLL NO :AV.SC.U4CSE24306,SEC:CSE-A");

        Rectangle r = new Rectangle(5, 3);

        System.out.println("Perimeter of Rectangle is: " + r.getPerimeter());

        Circle c = new Circle(7);

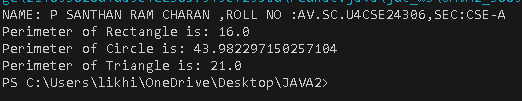
        System.out.println("Perimeter of Circle is: " + c.getPerimeter());

        Triangle t = new Triangle(9, 8, 4);

        System.out.println("Perimeter of Triangle is: " + t.getPerimeter());

    }}

OUTPUT:



ERRORS

|  |  |  |
| --- | --- | --- |
| **S.no** | **Error** | **Rectification** |
| 1 | Compilation error | Add correct return type |
| 2 | Access modifier issues(OOP error) | Change the modifier or add getter/setter |

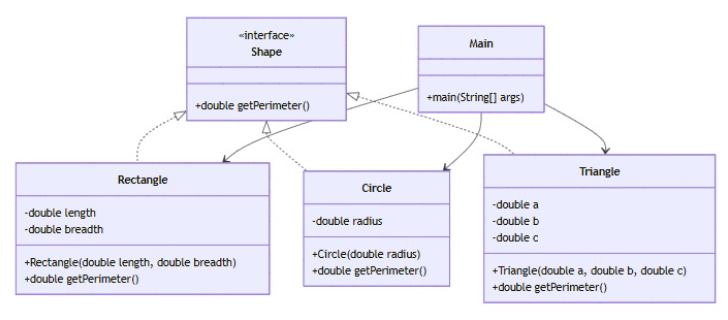
IMPORTANT POINTS

Getter and setter :

A getter method is used to retrieve the value of a private variable

A setter method is used to set or update the value of a private variable.

**Class Diagram:**

****

**AIM2:** **Write a java program to create an interface label with method play that takes no arguments and return void create 3 classes football,volley ball and basketball that implements interface and override play() method to play respective sports.**

**Code:**

**interface Playable { // interface named Playable with a method play()**

**void play();**

**}**

**class Football implements Playable { // Football class implements Playable interface**

**@Override**

**public void play() {**

**System.out.println("Playing Football!");**

**}**

**}**

**class Volleyball implements Playable {**

**@Override**

**public void play() {**

**System.out.println("Playing Volleyball!");**

**}**

**}**

**class Basketball implements Playable {**

**@Override**

**public void play() {**

**System.out.println("Playing Basketball!");**

**}**

**}**

**class main { // Main class**

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

**System.out.println("NAME: P.SANTHAN RAM CHARAN ,ROLL NO :AV.SC.U4CSE24305,SEC:CSE-A");**

**Playable football = new Football(); // Create objects**

**Playable volleyball = new Volleyball();**

**Playable basketball = new Basketball();**

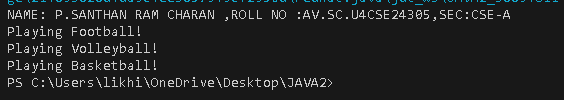
**football.play();**

**volleyball.play();**

**basketball.play();**

**}**

**}**

**OUTPUT:** ****

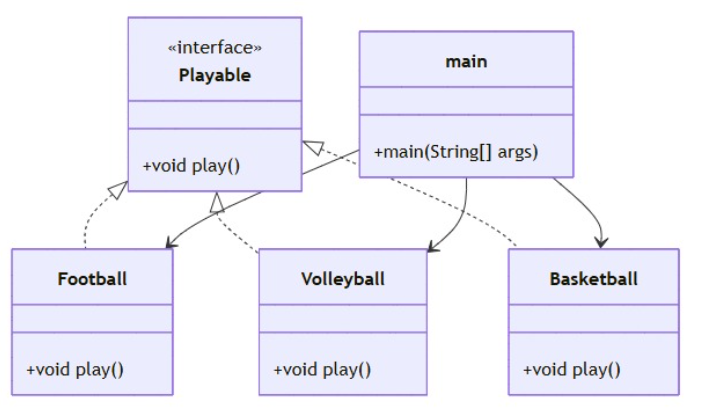
**ERRORS:**

|  |  |  |
| --- | --- | --- |
| **S.no** | **Error** | **Rectification** |
| **1.** | **Football is not abstract and does not override abstract method play()** | **Implement the play method from interface** |
| **2.** | **Playables.java but public class play test** | **Class play test is public,should be declared in a file named playtest.java** |

**IMPORTANT POINTS:-**

* **The Playable interface defines the contract (play() method).**
* **Each class (Football, Volleyball, Basketball) implements the interface and provides a specific behavior for play().**
* **This demonstrates polymorphism using interfaces in Java.**

**CLASS DIAGRAM:**

****

**AIM3:Write a java program to implement a login system using interfaces.**

**CODE:**

**import java.util.Scanner;**

**interface Login {**

**boolean authenticate(String username, String password);**

**}**

**class UserLogin implements Login {**

**private final String validUsername = "admin";**

**private final String validPassword = "password123";**

**@Override**

**public boolean authenticate(String username, String password) {**

**return validUsername.equals(username) && validPassword.equals(password);**

**}**

**}**

**public class LoginSystem {**

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

**System.out.println("NAME: P SANTHAN RAM CHARAN ,ROLL NO: 24305, SEC: CSE-A");**

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

**UserLogin userLogin = new UserLogin();**

**System.out.print("Enter username: ");**

**String username = scanner.nextLine();**

**System.out.print("Enter password: ");**

**String password = scanner.nextLine();**

**if (userLogin.authenticate(username, password)) {**

**System.out.println("Login successful!");**

**} else {**

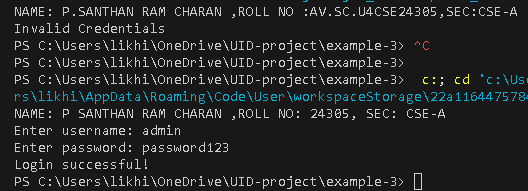
**System.out.println("Login failed! Invalid username or password.");**

**}**

**scanner.close();**

**}**

**}**

**Output:** ****

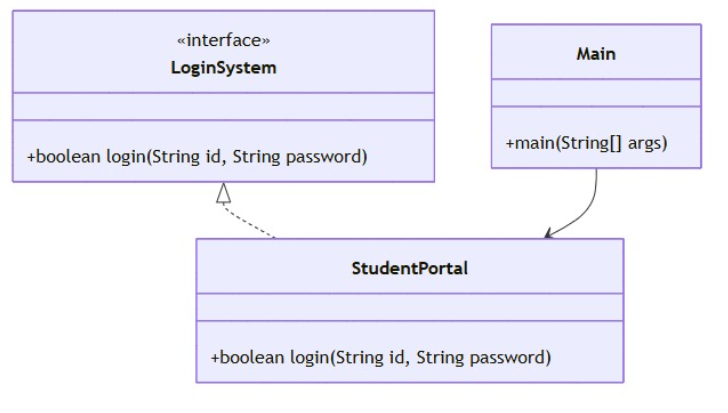
**Errors:**

|  |  |  |
| --- | --- | --- |
| **S.No** | **Error** | **Error rectification** |
| **1** | **Login\_System() not**  **overridden in University classes.** | **Implement Login\_System() in**  **university class using @override annotation** |
| **2** | **Error: ”;” expected** | **Giving the ‘;’ in wanted places** |

**Important points:**

**Boolean type is used in many applications when we need to check the input is true or false.**

**CLASS DIAGRAM:**

****