**23CSE111**

**OBJECT ORIENTED PROGRAMMING**

**LAB REPORT**



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

**Amrita Vishwa Vidyapeetham, Amaravati Campus**

**Name: K . Yagna Suvidh**

**Roll No: 24136**

**Verified By :**

INDEX

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| S.NO | **TITLE** | **PAGE NO** | **DATE** | **SIGNATURE** |
|  | **WEEK -1** | **5 - 8** |  |  |
| **1** | Explain the process of download & Installation of JDK |  |  |  |
| **2** | Write a program to print Student’s name, Roll no, Section |  |  |  |
|  | **WEEK-2** | **8 - 18** |  |  |
| **1** | Write a simple java program to calculate factorial of a number |  |  |  |
| **2** | Write a simple java program to find the simple interest by taking iinputs from the user |  |  |  |
| **3** | Write a program to calculate the Fibonacci sequence and take the input from the user |  |  |  |
| **4** | Write a program to find the area of triangle using hereon’s formula |  |  |  |
| **5** | Write a program to convert temperature from celssius to fahrenheit |  |  |  |
|  | **WEEK-3** | **18 - 29** |  |  |
| **1** | Write a java program with following instructions |  |  |  |
| **2** | Write a java program with following instructions |  |  |  |
|  | **WEEK-4** | **29 - 35** |  |  |
| **1** | Write a java program with following instructions |  |  |  |
| **2** | Write a java program with following instructions |  |  |  |
|  | **WEEK-5** | **35 - 43** |  |  |
| **1** | Write a java program with following instructions |  |  |  |
| **2** | Write a java program with following instructions |  |  |  |
|  | **WEEK -6** | **43 - 52** |  |  |
| **1** | Write a java program with following instructions |  |  |  |
| **2** | Write a java program with following instructions |  |  |  |
| **3** | Write a java program with following instructions |  |  |  |
| **4** | Write a java program with following instructions |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **WEEK -7** | **52 - 61** |  |  |
| 1 | Write the java program with an abstract class |  |  |  |
| 2 | Write the java program with an abstract class to find volume of the sphere |  |  |  |
| 3 | Write the java program with an abstract class to create pattern |  |  |  |
|  |  |  |  |  |

# WEEK-1

1. **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

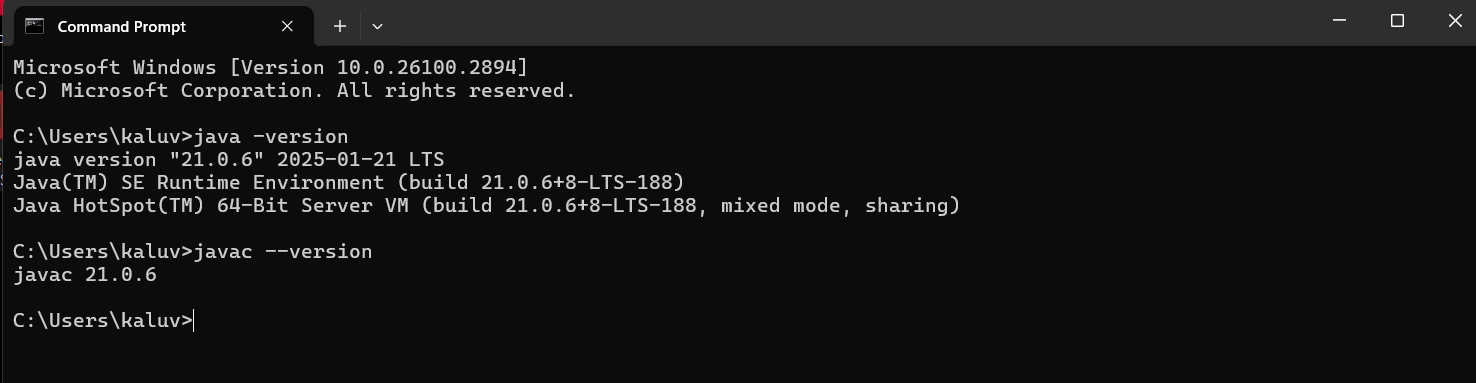
C:\Program Files\Java\jdk-21\bin)

* 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.



1. **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:K.Suvidh");

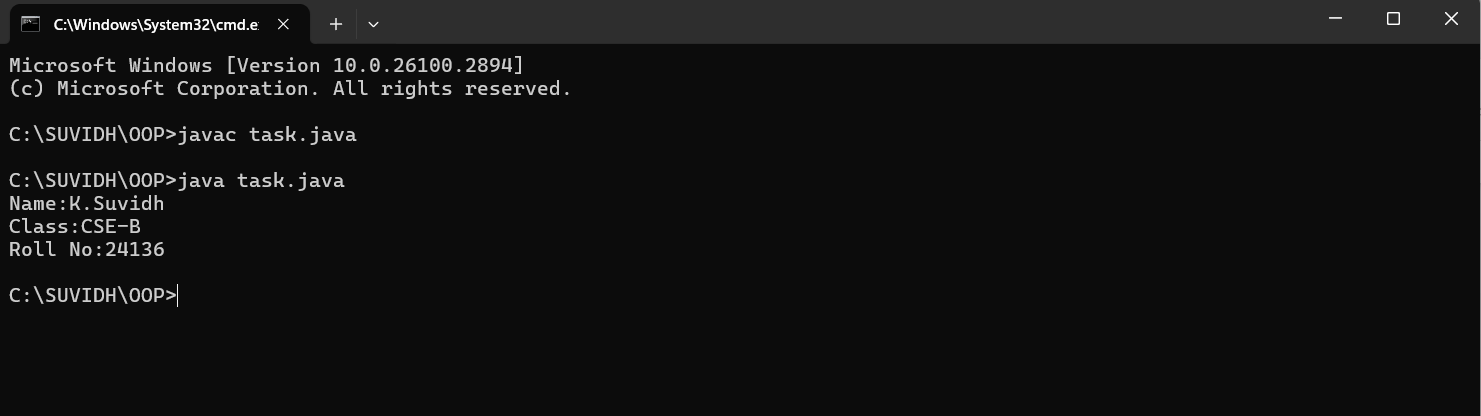
System.out.println("Class:CSE-B");

System.out.println("Roll No:24136");

}

}

**Output: -**



**WEEK-2**

**1) AIM:**

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

**PROGRAM :**

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;

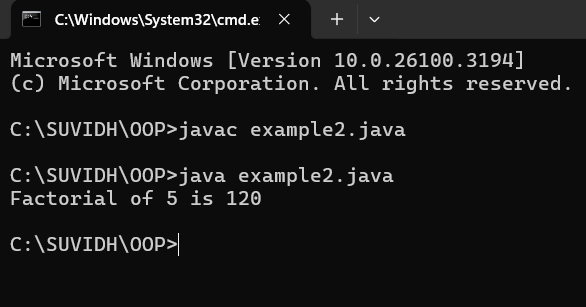
System.out.println("Factorial of " + num + " is "

+ 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 |

**2) AIM : Simple Java Program for finding simple interest by taking input from**

**PROGRAM :**

import java.util.Scanner;

class simple {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

System.out.print("Enter p :");

int p = input.nextInt();

System.out.print("Enter t :");

int t = input.nextInt();

System.out.print("Enter r :");

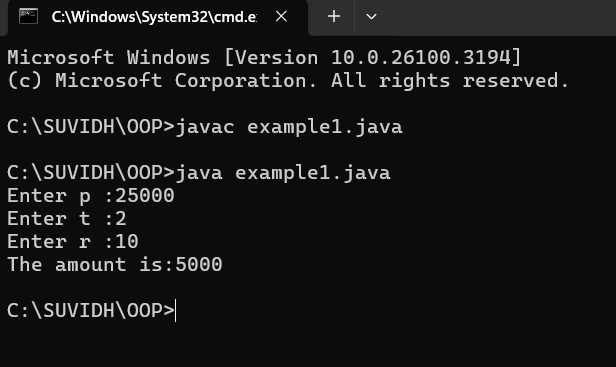
int r = input.nextInt();

System.out.println("The amount is:" + (p\*t\*r)/100);

}

}

**OUTPUT :**

****

**ERROR :**

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

**3) AIM : Write a program to to calculate the fibonacii sequence and take the input from user**

**PROGRAM :**

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);

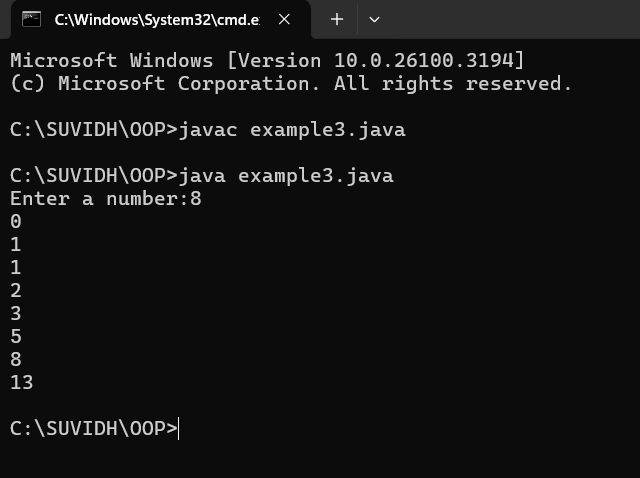
i = i+1;

}

}

}

**OUTPUT :**



**ERROR :**

|  |  |  |  |
| --- | --- | --- | --- |
| **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 find area of triangle using heron’s formula**

**and area of triangle**

**PROGRAM :**

import java.util.Scanner;

class Area {

public static void main(String args[]) {

Scanner input= new Scanner(System.in);

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

double a = input.nextDouble();

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

double b = input.nextDouble();

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

double c = input.nextDouble();

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

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

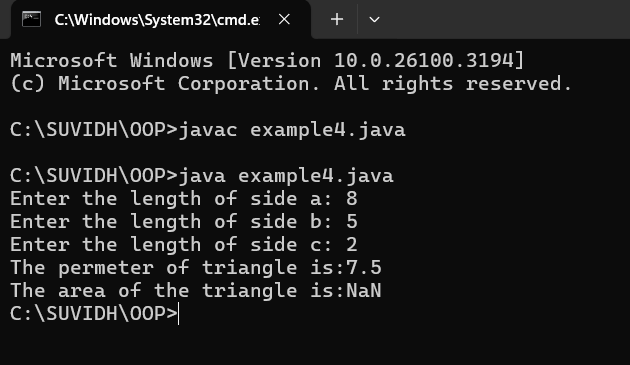
System.out.println("The permeter of triangle is:" + s);

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

}

}

**OUTPUT :**

****

**ERRORS :**

|  |  |  |  |
| --- | --- | --- | --- |
| **S no** | **Error type** | **Reason for error** | **Rectification** |
| 1 | Syntax error | Semicolon missing | Semi colon added |
| 2 | Missing Scanner | Creating scanner input | Scanner added |

**5) AIM : Write a java program to convert temperature from Celsius to**

**fahrenheit**

**PROGRAM :**

class celsiustofahrenheit {

    public static void main(String[] args)

    {

        double celsius = 10.0, fahrenheit = 0.0;

        fahrenheit = (celsius \* 1.8) + 32;

        System.out.println(

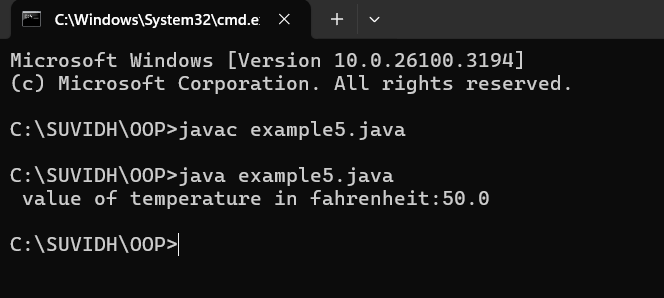
            " value of temperature in fahrenheit:"

            + fahrenheit);

    }

}

**OUTPUT :**

****

**ERRORS :**

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

**WEEK-3**

1. **AIM :**

**Write a java program with following instructions**

1. **Create a class with name car**
2. **Create four attributes named, car colour, car brand , fuel type , mileage.**
3. **Create 3 methods named start , stop, service .**
4. **Create 3 objects named c1,c2,c3**
5. **Create a constructor with parameters , car colour , car brand , fuel**

**Type and mileage**

**PROGRAM :**

class car {

//Attributes

String car\_brand;

String car\_colour;

String fuel\_type;

int mileage;

//Constructors

public car(String car\_brand,String car\_colour,String fuel\_type,int mileage){

this.car\_brand = car\_brand;

this.car\_colour = car\_colour;

this.fuel\_type = fuel\_type;

this.mileage = mileage;

}

//METHODS

public void start() {

System.out.println("CAR IS STARTED");

}

public void stop() {

System.out.println("CAR IS STOPPED");

}

public void service() {

System.out.println("CAR IS I SERVICE");

}

public void car\_details() {

System.out.println("CAR BRAND IS :" + car\_brand);

System.out.println("CAR COLOUR IS :" + car\_colour);

System.out.println("CAR FUEL TYPE IS :" + fuel\_type);

System.out.println("CAR MILEAGE IS :" + mileage);

}

//MAIN PROGRAM

public static void main(String[] args) {

// CREATING OBJECTS FOR CLASS CAR

car c1 = new car("MCLAREN","GREEN","PETROL",12);

car c2 = new car("FERRARI","RED","PETROL",18);

car c3 = new car("LAMBORGHINI","ORANGE","PETROL",20);

//CALLING METHODS

c1.car\_details();

System.out.println(" ");

c2.car\_details();

System.out.println(" ");

c3.car\_details();

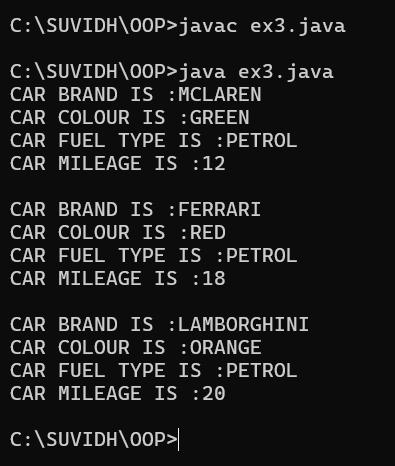
}

}

**CLASS DIAGRAM :**

|  |
| --- |
| **CAR** |
| car\_brand : String  car\_colour : String  fuel\_type : String  mileage : int |
| + car (String, String, String,int)  +start() : void  +stop() : void  +service : void  + car\_details : void |

**OUTPUT :**



**ERRORS:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Error type** | **Reason for error** | **Rectification** |
| 1 | Runtime error | Incorrect symbol in main program | { symbol is added |
| 2 | Logical error | Incorrect logic | Correct logic |

**2) AIM :**

**create a class named bankaccount with method deposit and with draw where the deposit method should accepts a parameter and when this method is called the deposited amount should be current balance .In addition to that when a withdraw method is called it has to verify whether withdraw amount is less than the current balance .If not display a message saying insufficient funds.Use the constructer to display the details of the customer (Customer name,account number , IFSC,branch) .Also create two customer objects c1,c2**

**PROGRAM :**

class bank {

    String name;

    String number;

    String IFSC;

    String branch;

    int balance;

//Constructors

public bank(String name, String number, String IFSC, String branch, int balance) {

    this.name = name;

    this.number = number;

    this.IFSC = IFSC;

    this.branch = branch;

    this.balance = balance;

}

//Method

public void bank\_details() {

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

        System.out.println("Account Number: " + number);

        System.out.println("IFSC Code: " + IFSC);

        System.out.println("Branch: " + branch);

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

    }

//Method for deposit

public void deposit(int amount) {

    if (amount >0) {

    balance += amount;

    System.out.println("Total balance is :" + balance);

    } else {

    System.out.println("Error");

    }

}

//Method for withdrawal

public void withdrawal(int amount) {

    if (amount < balance) {

        balance -= amount;

        System.out.println("Withdrawal amount is:" + amount);

        System.out.println("Updated balance is: " + balance);

        } else {

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

        }

    }

//Main function

public static void main(String[] args) {

//Creating objects for class bank

    bank c1 = new bank("Rahul", "1234567890", "SBI1997","GUNTUR",0);

    bank c2 = new bank("Iyer", "1234567890", "SBI1996","GUNTUR",0);

    //Calling Methods

    //Customer 1 details

    System.out.println("Customer 1 details");

    System.out.println("    ");

    c1.bank\_details();

    System.out.println("    ");

    System.out.println(“Customer 1 deposit");

    c1.deposit(1000);

    System.out.println("    ");

    System.out.println("Customer 1 withdrawal");

    c1.withdrawal(500);

    System.out.println("    ");

    System.out.println("    ");

    //CUSTOMER 2 DETAILS

    System.out.println("Customer 2 details");

    System.out.println("    ");

    c2.bank\_details();

    System.out.println("    ");

    System.out.println(“Customer 2 deposit");

    c2.deposit(5000);

    System.out.println("    ");

    System.out.println("Customer 2 withdrawal");

    c2.withdrawal(600);

}

}

**CLASS DIAGRAM :**

|  |
| --- |
| bank |
| name:String  number:String  IFSC:String  branch:String  balance:int |
| bank(name:String , number:String, IFSC:String, +branch:String, balance:int)  +bank\_details():void  +deposit(amount:int):void  +withdrawl(amount:int):void |

**OUTPUT :**

****

**ERRORS:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Error type** | **Reason for error** | **Rectification** |
| 1 | syntax error | Forgot to keep main word in main program | Main is added |
| 2 | Logical error | Incorrect logic | Correct logic |

**WEEK-4**

1. **AIM : Write a java program with class named book the class should contain various attributes such as title ,author, year of publication .It should also containa constructor with parameters with initializes title ,author and year ofpublication . create a method which displays the details of the book (displaythedetails of two book i.e, create two books and objects with details)**

**PROGRAM:**

class book {

    String title;

    String author;

    int year;

//CONSTRUCTORS

public book(String title, String author, int year) {

    this.title = title;

    this.author = author;

    this.year = year;

}

//METHODS

public void details() {

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

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

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

}

//MAIN FUNCTION

public static void main(String[] args) {

    //CREATING OBJECTS

    book b1 = new book("Spider Man", "A.K", 2002);

    book b2 = new book("Horry Potter", "J.K", 1986);

    //CALLING METHODS

    System.out.println("BOOK 1 DETAILS");

    System.out.println("    ");

    b1.details();

 System.out.println("    ");

    System.out.println("BOOK 2 DETAILS");

 System.out.println("    ");

    System.out.println("    ");

    b2.details();

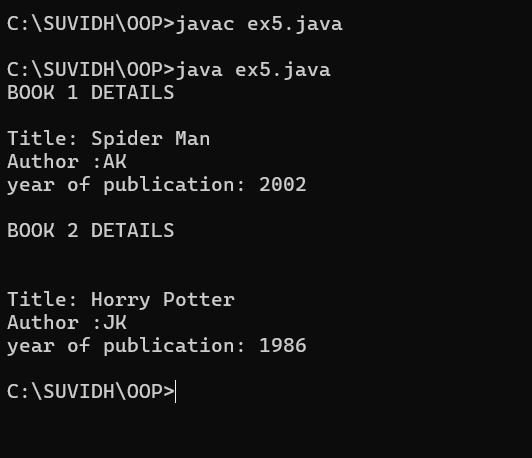
}

}

**CLASS DIAGRAM :**

|  |
| --- |
| book |
| title : String  author : String  year : int |
| + book(title: String, author: String, year: int)  + details(): void |

**OUTPUT :**

****

**ERRORS :**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Error type** | **Reason for error** | **Rectification** |
| 1 | syntax error | Forgot to keep }  At last | } is added |
| 2 | Logical error | Incorrect logic | Correct logic |

1. **AIM : write a java program to create a class named myclass with a static**

**Variable count of int type and initialize to zero and constant variable pie of double data type ,initialize to 3.1415 as attributes of that class now define a constructor for my class that increments the count variables each time an objectof my class is created variable each time an object of myclass is created. Finally print the final values of count and pie variables.**

**PROGRAM :**

class MyClass {

    static int count = 0;

    final double PIE = 3.1415;

//CONSTRUCTORS

public MyClass() {

        count++;

    }

//MAIN FUNCTION

    public static void main(String[] args) {

        //CREATING OBJECTS

        MyClass obj1 = new MyClass();

        MyClass obj2 = new MyClass();

        MyClass obj3 = new MyClass();

        //PRINTING COUNT

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

        System.out.println("PIE constant value: " + obj1.PIE);

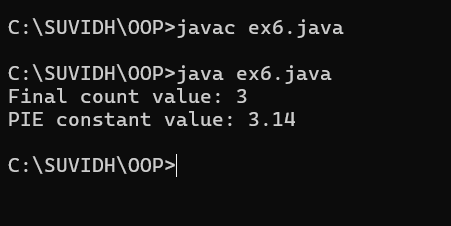
    }

}

**CLASS DIAGRAM :**

|  |
| --- |
| Myclass |
| Count : int  PIE : double |
| +Myclass() |

**OUTPUT :**

****

**ERRORS:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Error type** | **Reason for error** | **Rectification** |
| 1 | syntax error | String forgot in main function | String is added |
| 2 | Logical error | Incorrect logic | Correct logic |

**WEEK-5**

1. **AIM :** create a calculator using the operations including addition, subtraction, multiplication and division using multilevel inheritance and display the desired

**Output.**

**PROGRAM :**

public class calculator {

    int a = 18;

    int b = 1;

void addition(){

    System.out.println("Addition is :" + (a+b));

}

}

class multi1 extends calculator{

    void subtraction(){

        System.out.println("Subtraction is :" + (a-b));

    }

}

class multi2 extends calculator{

    void multiplication(){

        System.out.println("Multiplication is :" + (a\*b));

    }

}

class multi3 extends calculator{

    void division(){

        System.out.println("Division is :" + (a/b));

    }

}

class multipleinheritance {

    public static void main(String[] args) {

        multi1 m1 = new multi1();

        multi2 m2 = new multi2();

        multi3 m3 = new multi3();

        m1.addition();

        System.out.println("    ");

        m1.subtraction();

        System.out.println("    ");

        m2.multiplication();

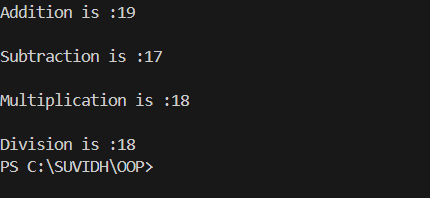
        System.out.println("    ");

        m3.division();

}

}

**OUTPUT :**



**ERRORS :**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Error type** | **Reason for error** | **Rectification** |
| 1 | syntax error | String forgot in main function | String is added |
| 2 | Logical error | Incorrect logic | Correct logic |

**2)** AIM**:** A vehicle rental company wants to develop a system that maintains informationabout different types of vehicles available for rent. The company rents outcars and bikes and they need a program to store details about each variable such asbrand and speed.

• Cars should have an additional property: numbers of doors, seating capac

• 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 & indicates when a vehicle is starting.

• Every class should have constructor.

1. Which OOP concept is used in the above program? Explain why it is useful this scenario.

2. If the company decides to add a new type of vehicle: Truck, how would modify the program?

• Truck should include an additional property capacity(in tons)

• Create a ShowTruckDetails() method to display the truck’s capacity.

• Write a constructor for Truck that initializes all properties.

3. Implement the truck class and update the main method create a truck object & also create an object for car & bike sub classed. Finally display its details

**PROGRAM :**

class vehicle{

    String brand;

    int speed;

    vehicle(String brand,int speed){

        this.brand=brand;

        this.speed=speed;

    }

    void Details(){

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

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

    }

}

class CARS extends vehicle{

    int doors;

    int capacity;

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

        super(brand, speed);

        this.doors=doors;

        this.capacity=capacity;

    }

    void cardetails (){

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

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

    }

}

class Bikes extends vehicle{

    Boolean gears;

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

        super(brand, speed);

        this.gears=gears;

    }

    void bikedetails(){

        if (gears==true)

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

        else

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

    }

}

class Trucks extends vehicle{

    int tons;

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

        super(brand, speed);

        this.tons=tons;

    }

    void truckdetails(){

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

    }

}

class Rent{

    public static void main(String[] args){

        CARS c=new CARS("BMW",157,6,80);

        System.out.println("    ");

        c.cardetails();

        System.out.println("    ");

        c.Details();

        System.out.println("    ");

        Bikes b=new Bikes("Mahindra",200,true);

        b.bikedetails();

        System.out.println("    ");

        b.Details();

        System.out.println("    ");

        Trucks t=new Trucks("MG",40,5);

        t.truckdetails();

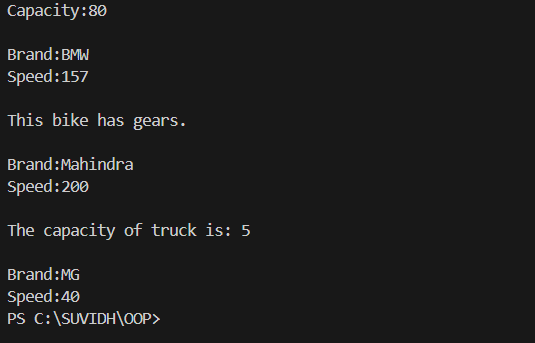
        System.out.println("    ");

        t.Details();

    }

}

**OUTPUT :**



**ERROR :**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Error type** | **Reason for error** | **Rectification** |
| 1 | syntax error | String forgot in main function | String is added |
| 2 | Logical error | Incorrect logic | Correct logic |

**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 about a car.**

**PROGRAM :**

class vehicle{

    String company;

    String model;

    String fuel;

    int capacity;

    void displayInfo(String company,String model,String fuel,int capacity){

        System.out.println("The details of vehicle: ");

        this.company=company;

        this.model=model;

        this.fuel=fuel;

        this.capacity=capacity;

    }

}

class car extends vehicle{

    void displayInfo(String company,String model,String fuel,int capacity){

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

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

        System.out.println("Fuel: "+fuel);

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

    }

}

class poly1{

    public static void main(String[] args){

        car car1=new car();

        car1.displayInfo("TATA","Model","Diesel",6);

    }

}

**OUTPUT :**

**ERRORS :**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Error type** | **Reason for error** | **Rectification** |
| 1 | Syntax error | [] is missed | [] is added |
| 2 | Logical error | Incorrect logic | Correct logic |

**2) AIM:** A college is developing an automated admission system that verifies student eligibility for undergraduate (UG) and postgraduate (PG) programs. Each program hasdifferent eligibility criteria based on the students' percentage in their previous qualifications.

(i)UG admissions require a minimum of 60%.

(ii)PG admissions require a minimum of 70%

**PROGRAM :**

class College{

    String name;

    int percentage;

    void geteligibility(String name,int percentage){

        this.name=name;

        this.percentage=percentage;

    }

}

class UG extends College{

    void geteligibility(String name,int percentage){

        if (percentage>=60){

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

        }

        else{

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

        }

    }

}

class PG extends College{

    void geteligibility(String name,int percentage){

        if (percentage>=70){

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

        }

        else{

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

        }

    }

}

class poly2{

    public static void main(String[] args){

        UG ug=new UG();

        ug.geteligibility("Person-1",40);

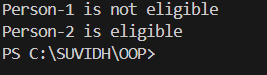
        PG pg=new PG();

        pg.geteligibility("Person-2",80);

    }

}

**OUTPUT:**



**ERRORS:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Error type** | **Reason for error** | **Rectification** |
| 1 | syntax error | String forgot in main function | String is added |
| 2 | Logical error | Incorrect logic | Correct logic |

**3) AIM :** Create a Calculator class with overloaded methods to perform addition:

(i) Add two integers.

(ii) Add two doubles.

(iii) Add three integers.

**PROGRAM :**

class Calcee{

    public int add(int a,int b){

        return a+b;

    }

    public double add(double a,double b){

        return a+b;

    }

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

        return a+b+c;

    }

}

class poly3{

    public static void main(String[] args){

        Calcee C1=new Calcee();

        System.out.println("Sum of 2 and 5 is: "+C1.add(2,5));

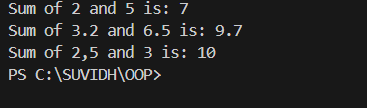
        System.out.println("Sum of 3.2 and 6.5 is: "+C1.add(3.2,6.5));

        System.out.println("Sum of 2,5 and 3 is: "+C1.add(2,5,3));

    }

}

**OUTPUT :**

****

**ERRORS :**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Error type** | **Reason for error** | **Rectification** |
| 1 | syntax error | String forgot in main function | String is added |
| 2 | Logical error | Incorrect logic | Correct logic |

**4) AIM :** Create a Shape class with a method calculateArea() that is overloaded for different shapes (e.g., square, rectangle). Then, create a subclass Circle that overrides the calculateArea() method for a circle.

**PROGRAM :**

class Shape { // class shape

    void calculateArea( int a) { // method 1

        System.out.println("The area of Square is :" + (a\*a) );

    }

    void calculateArea(int a , int b) { // method 2

        System.out.println("The area of rectangle is :" + (a\*b));

    }

}

class circle extends Shape { // inheritance class

    void calculateArea(double a){ // method overloading

        System.out.println("The area of circle is :" + (3.14\*a\*a));

  } }

class main { // main program

    public static void main(String[] args) {

        // creating objects for class

        Shape s = new Shape();

        circle c = new circle();

        // calling methods

        s.calculateArea(4);

        System.out.println("    ");

        s.calculateArea(4, 5);

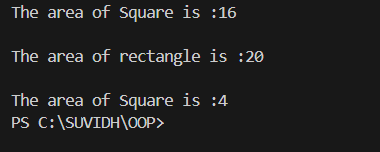
        System.out.println("    ");

        c.calculateArea(2);

    }

}

**OUTPUT :**

****

**ERRORS :**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Error type** | **Reason for error** | **Rectification** |
| 1 | Syntax error | ; is missed | ; is added |
| 2 | Logical error | Incorrect logic | Correct logic |

**WEEK-7**

1. **AIM:** Write a Java program to create an abstract class Animal with an abstract method called sound(). Create subclasses Lion and Tiger that extend the Animal class and implement the sound() method to make a specific sound for each animal.

**PROGRAM:**

abstract class Animal { // abstract class

    abstract void sound(); // abstract method

}

class Lion extends Animal { // Sub class

    void sound() {

        System.out.println("Lion Roar...!"); // sub class method to override the abstract method

    }

}

class Tiger extends Animal {

    void sound() {

        System.out.println("Tiger Roar...!"); // sub class method to override the abstract method

    }

}

class Sound { // main program

    public static void main(String[] args) {

        //Creating objects for the sub classes

        Lion l = new Lion();

        Tiger t = new Tiger();

        System.out.println("Name : SUVIDH"  + "Roll No : AV.SC.U4CSE24136" + "Section : CSE-B");

        System.out.println("    ");

        l.sound(); // calling sub class method

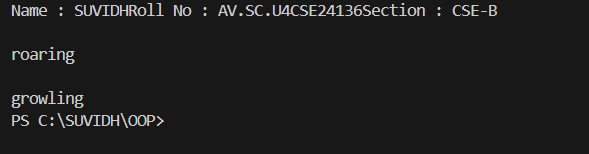
        System.out.println("    ");

        t.sound();// calling sub class method

    }

}

**OUTPUT:**

****

**ERRORS:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Error type** | **Reason for error** | **Rectification** |
| 1 | Syntax error | Abstract key word is missed before method | Abstract keyword is added |
| 2 | Logical error | Incorrect logic in subclass method | Corrected logic in subclass method |

**2)Aim :** Write a Java program to create an abstract class Shape3D with abstract methods calculateVolume() and calculateSurfaceArea(). Create subclasses Sphere andCube that extend the Shape3D class and implement the respective methods to calculate the volume and surface area of each shape.

**PROGRAM :**

import java.math.\*; //pacakage importing

abstract class Shapes3D { //abstract class

    abstract void CalculateVolume(); // abstract method

    abstract void CalculateSurfaceArea(); // abstract method

}

class Sphere extends Shapes3D { //Sub class extends abstract class

    int radius;

    Sphere(int radius) { // sub class Constructor

        this.radius = radius;

    }

    void CalculateVolume() {

        System.out.println("Volume of sphere is: " + (4.0 / 3.0) \* Math.PI \* radius \* radius \* radius);

    }

    void CalculateSurfaceArea() {

        System.out.println("Surface area of sphere is: " + 4 \* Math.PI \* radius \* radius);

    }

}

class Cube extends Shapes3D {//Sub class extends abstract class

    int side;

    Cube(int side) { // sub class Constructor

        this.side = side;

    }

    void CalculateVolume() {

        System.out.println("Volume of cube is: " + side \* side \* side);

    }

    void CalculateSurfaceArea() {

        System.out.println("Surface area of cube is: " + 6 \* side \* side);

    }

}

public class Shapes18 {

    public static void main(String[] args) {

        Sphere sp = new Sphere(5);

        Cube c = new Cube(4);

        System.out.println("Name : K.SUVIDH  Section : CSE-B  Roll no : AV.SC.U4CSE24136");

        System.out.println("    ");

        sp.CalculateSurfaceArea();

        System.out.println("    ");

        sp.CalculateVolume();

        System.out.println("    ");

        c.CalculateSurfaceArea();

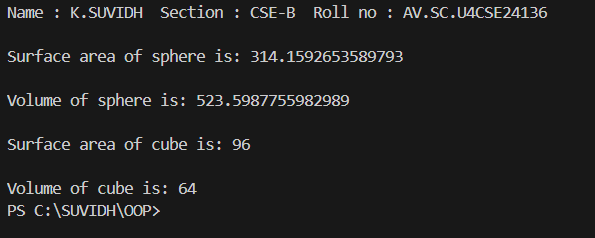
        System.out.println("    ");

        c.CalculateVolume();

    }

}

**OUTPUT:**

****

**ERRORS:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Error type** | **Reason for error** | **Rectification** |
| 1 | Syntax error | package is missed before abstract class | Package is imported |
| 2 | Logical error | Incorrect logic in subclass method | Corrected logic in subclass method |

**3)Aim :** 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 printpattern(int n) and a concrete method to display the pattern title.

Implement two subclasses:

1) Star pattern - Prints a right-angled triangle of stars(\*).

2) Number pattern - Prints a right- angled triangles of increasing numbers.

In the main() method, create Objects

Star Pattern Number pattern

\* 1

\*\* 1 2

\*\*\* 1 2 3

\*\*\*\* 1 2 3 4

\*\*\*\*\* 1 2 3 4 5

**PROGRAM :**

abstract class PatternPrinter { // abstract class

    abstract void printPattern(int n); // abstract method

    void displayTitle(String title) {

        System.out.println(title);

    }

}

class StarPattern extends PatternPrinter { // Star pattern subclass

    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 {// Number pattern subclass

    void printPattern(int n) {

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

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

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

            }

            System.out.println();

        }

    }

}

 class PatternProgram { //Main program

    public static void main(String[] args) {

        //  creating objects for Star & Number pattern classes

        StarPattern sp = new StarPattern();

        NumberPattern np = new NumberPattern();

        System.out.println("Name:K.SUVIDH Roll no:AV.SC.U4CSE24136

 Section:CSE-B");

        System.out.println("    ");

        sp.displayTitle("Star Pattern");

        sp.printPattern(5);

        System.out.println("    ");

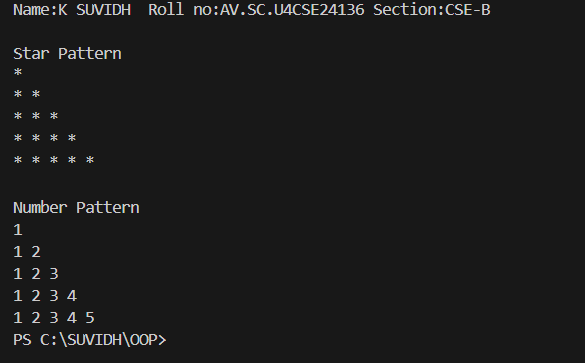
        np.displayTitle("Number Pattern");

        np.printPattern(5);

    }

}

**OUTPUT:**

****

**ERRORS:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Error type** | **Reason for error** | **Rectification** |
| 1 | Syntax error | For loop increment condition is missed in subclass method | Increment condition is added in subclass method |
| 2 | Logical error | Incorrect logic in subclass method | Corrected logic in subclass method |

**WEEK-8**

1. **Aim :** Write a Java program to create an interface Shape with the getPerimeter() method. Create three classes Rectangle, Circle, and Triangle that implement the Shape interface. Implement the getPerimeter() method for each of the three classes.

**PROGRAM :**

interface Shape {

    double getPerimeter();

}

class Rectangle implements Shape {

    private double length;

    private double width;

    public Rectangle(double length, double width) {

        this.length = length;

        this.width = width;

    }

    public double getPerimeter() {

        return 2 \* (length + width);

    }

}

class Circle implements Shape {

    private double radius;

    public Circle(double radius) {

        this.radius = radius;

    }

    public double getPerimeter() {

        return 2 \* Math.PI \* radius;

    }

}

class Triangle implements Shape {

    private double side1;

    private double side2;

    private double side3;

    public Triangle(double side1, double side2, double side3) {

        this.side1 = side1;

        this.side2 = side2;

        this.side3 = side3;

    }

    public double getPerimeter() {

        return side1 + side2 + side3;

    }

}

public class perimeter {

    public static void main(String[] args) {

        Shape rectangle = new Rectangle(5, 3);

        Shape circle = new Circle(4);

        Shape triangle = new Triangle(3, 4, 5);

        System.out.println("    ");

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

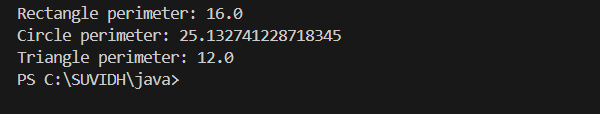
        System.out.println("Circle perimeter: " + circle.getPerimeter());

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

    }

}

**OUTPUT:**



**ERRORS:**

|  |  |  |  |
| --- | --- | --- | --- |
| S.No | Error Type | Cause | Rectification |
| 1 | Compilation error | Return type missing | Add correct return type |
| 2 | Access modifier issues(OOP error) | Trying to access private members | Change the modifier or add getter/setter |

1. **Aim:** Write a Java program to create an interface Playable with a method play() that takes no arguments and returns void. Create three classes Football, Volleyball, and Basketball that implement the Playable interface and override the play() method to play the respective sports.

**PROGRAM:-**

interface Playable {

    void play();

}

class Football implements Playable {

    public void play() {

        System.out.println("Playing Football: Kicking the ball towards the goal!");

    }

}

class Volleyball implements Playable {

    public void play() {

        System.out.println("Playing Volleyball: Serving and spiking the ball over the net!");

    }

}

class Basketball implements Playable {

    public void play() {

        System.out.println("Playing Basketball: Dribbling and shooting the ball into the basket!");

    }

}

public class PlayableTest {

    public static void main(String[] args) {

        Playable football = new Football();

        Playable volleyball = new Volleyball();

        Playable basketball = new Basketball();

        System.out.println(" Football ");

        football.play();

        System.out.println(" Volleyball");

        volleyball.play();

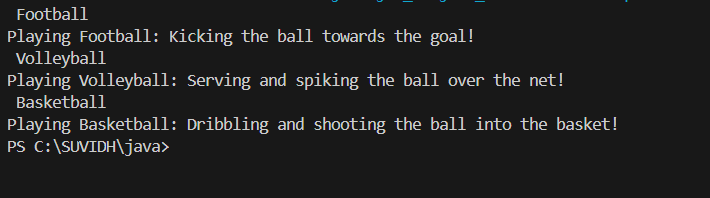
        System.out.println(" Basketball");

        basketball.play();

    }

}

**OUTPUT:**



**ERRORS:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Error Type** | **Cause** | **Rectification** |
| **1** | File not found exception | Wrong selection of path | Select path correctly |
| **2** | Instantiation of abstract class | Trying to create an object of abstract class without any subclass or interface | Use subclass or interface implementation |

1. **Aim:** write a java program to implement a login system using interfaces.

**PROGRAM:**

interface LoginSystem {

    boolean Login(String ID, int pass);

}

class CollegePortal implements LoginSystem {

    public boolean Login(String ID, int pass) {

        if ((ID=="suvidh") && (pass==101045)){

            System.out.println("Login Successful..!");

            return true;

        }else {

            System.out.println("Invalid Password");

            return false;

        }

    }

}

class LoginPortal {

    public static void main(String[] args) {

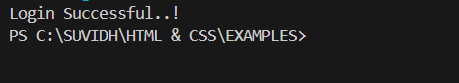
        CollegePortal CP = new CollegePortal();

        CP.Login("suvidh", 101045);

    }

}

**OUTPUT:**

****

**ERROR:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Error type** | **Reason for error** | **Rectification** |
| 1 | Syntax error | Error in If statement condition | If statement condition is corrected |

**WEEK-9**

1. **Aim :** Write a Java Program to create a method that takes an integer as a parameter and throws an exception if the number is even

**PROGRAM:**

public class Evenexceptionexample{

    public static void checkNumber(int number) throws Exception{

        if (number%2==0) {

            throw new Exception("Even number not allowed: "+number);

    }

    else{

        System.out.println("Odd number accepted: " + number);

    }

}

public static void main(String[] args){

    try {

        checkNumber(4);

    }

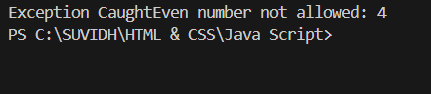
catch(Exception e){

        System.out.println("Exception Caught"+e.getMessage());

        }}

}}

**OUTPUT:**



**ERROR:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Error Name** | **Error Rectification** |
| 1 | Closing Brackets | Need to Close the brackets |
| 2 | Class Name Error | Give the class name correctly |

1. **Aim :** Write a Java program to create a method that reads a file and throws an exception if the file is not found

**PROGRAM :**

import java.io.\*;

public class FileReadExample{

    public static void main(String[] args) throws IOException{

        try{

            BufferedReader br = new BufferedReader(new FileReader("C:\\Users\\suvidh\\OneDrive\\Desktop\\.vscode\\java\\week-9\text.txt"));

            String line;

            while ((line = br.readLine()) != null)

            {

                System.out.println(line);

            }

            br.close();

        }

        catch(FileNotFoundException e){

            System.err.println("File not found: " +e.getMessage());

        }

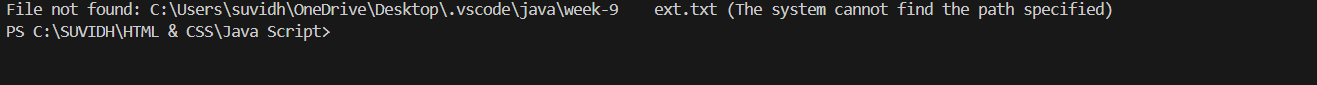
        catch(IOException e) {

            System.err.println("Error reading file: " + e.getMessage());

    }}

}

**OUTPUT:**

****

**ERROR:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Error Name** | **Error Rectification** |
| 1 | Closing Brackets | Need to Close the brackets |
| 2 | Class Name Error | Give the class name correctly |

1. **Aim:** Write a Java program to handle arithmetic exception using try,catch and finally.

**PROGRAM:**

import java.util.Scanner;

public class Finally{

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter two integers:");

int num1 = scanner.nextInt();

int num2 = scanner.nextInt();

try {

int result = num1 / num2;

System.out.println("Result: "+result);

} catch (ArithmeticException e){

System.out.println("Error: Cannot divide by zero.");

}

finally{

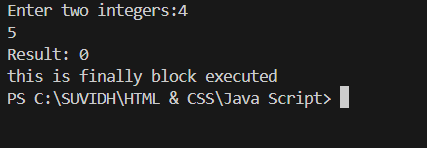
System.out.println("this is finally block executed");

}

}

}

**OUTPUT :**

****

**ERROR :**

|  |  |  |
| --- | --- | --- |
| SI.NO | Error name | Error Rectification |
| 1 | Method Access modifiers | Adding appropriate access modifiers |
| 2 | No encapsulation | Make fields private and provide getter/setters |

1. **Aim:** Write a Java program to simulate a University system using inner classes. Create an outerclass named university with a variable universityName.

Inside it,define two non-static inner classes

1.Department-with variables like deptName,deptcode and a method to displaydetails

2. Student-with variables like studentName,RollNo and a method to displaydetails

3.Create an object for each class and call their methods to display their details along with the university Name

**PROGRAM:**

class University{

    String universityname="Amrita Vishwa Vidyapeetam";

    class Department{

    String deptname="CSE";

    String deptcode="CSE111";

    void Deptdetails(){

    System.out.println("Department Name is:"+deptname);

    System.out.println("Department Code is:"+deptcode);

    }

    }

    class Student{

    }

    }

    public static void main(String[] args){

    University uni=new University();

    System.out.println("Name of the university is:"+uni.universityname);

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

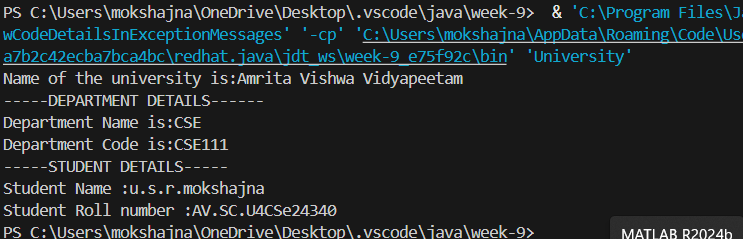
    University.Department dept=uni.new Department();

    dept.Deptdetails();

    }

    }

**OUTPUT :**



**ERROR:**

|  |  |  |
| --- | --- | --- |
| SI.NO | Error name | Error Rectification |
| 1 | Method Access modifiers | Adding appropriate access modifiers |
| 2 | No encapsulation | Make fields private and provide getter/setters |