**23CSE111**

**OBJECT ORIENTED PROGRAMMING**

**LAB REPORT**



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

**Amrita Vishwa Vidyapeetham, Amaravati Campus**

**Name: M.SRIMANTH**

**Roll No: AV.SC.U4CSE24228**

**Verified By :**

**INDEX**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | WEEK-1 |  |  |  |
| 1 | Installation Process of JDK | **3-4** |  |  |
| 2 | Simple java program for  printing basic details of student | **4-5** |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | WEEK-2 |  |  |  |
| 1 | Java program for calculating simple interest | **6-7** |  |  |
| 2 | Java program for calculating area of rectangle | **7-8** |  |  |
| 3 | Java program for calculating area of Triangle |  |  |  |
| 4 | Java program for calculating Fibonacci series |  |  |  |
| 5 | Java program to convert temperature from Fahrenheit  to Celsius |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | WEEK-3 |  |  |  |
| **1** | Write a java program with following instructions | **19** |  |  |
| **2** | Write a java program with following instructions | **26** |  |  |
|  | WEEK-4 |  |  |  |
| **1** | Write a java program with following instructions | **34** |  |  |
| **2** | Write a java program with following instructions | **38** |  |  |
|  | WEEK-5 |  |  |  |
| **1** | Write a java program with following instructions |  |  |  |
| **2** | Write a java program with following instructions |  |  |  |
|  | WEEK-6 |  |  |  |
| **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-1

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

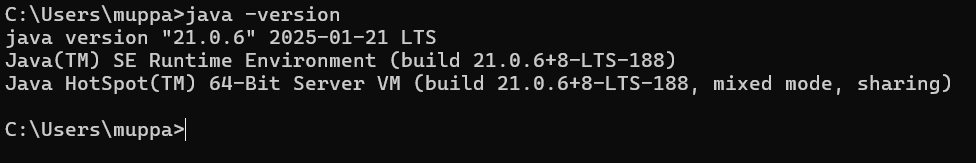
C:\Program Files\Java\jdk-21\bin)

* 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 :**

**CODE: -**

**public class details {**

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

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

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

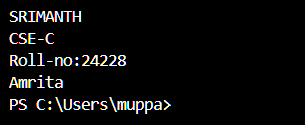
**System.out.println("Roll-no:24228");**

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

**}**

**}**

**Output: -**



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

**Week-2**

1) Java program for calculating simple interest:  
code:-  
import java.util.Scanner

class simpleinterest {

  public static void main(String[] args) {

    Scanner input = new Scanner(System.in);

    System.out.println("Enter the principal: ");

    double principal = input.nextDouble();

    System.out.println("Enter the rate: ");

    double rate = input.nextDouble();

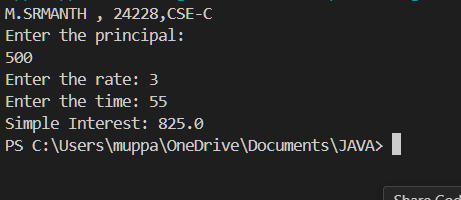
    System.out.println("Enter the time: ");

    double time = input.nextDouble();

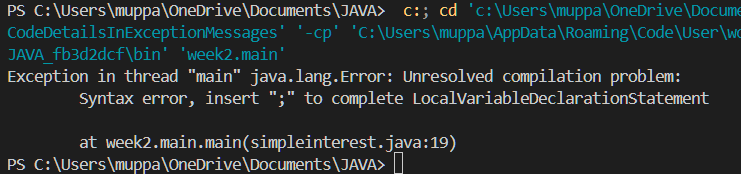
    double interest = (principal \* time \* rate) / 100;

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

    input.close();

  }Output:-  
  


Negative case:



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

2)Java program for calculating area of rectangle:  
Code:-  
import java.util.Scanner;

class area {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

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

double length = input.nextDouble();

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

double width = input.nextDouble();

double Area;

Area = length\*width;

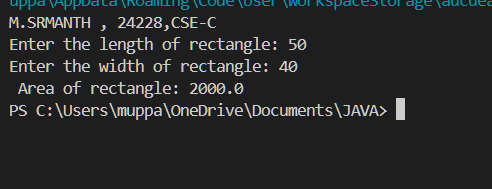
System.out.println(" Area of rectangle: " + Area);

input.close();

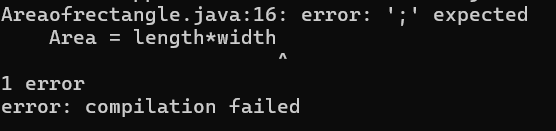
}

}

**Output:-**

****

**Negative case:**



**Error:-**

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

3)Java program for calculating area of Triangle:

Code:-

import java.util.Scanner;

public class TriangleArea {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

System.out.print("Enter the base of the triangle: ");

double base = input.nextDouble();

System.out.print("Enter the height of the triangle: ");

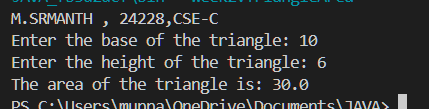
double height = input.nextDouble();

double area = (base \* height) / 2;

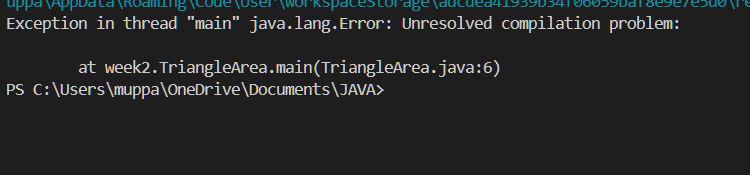
System.out.println("The area of the triangle is: " + area);

}

}  
**Output:-**

****

**Negative case:**

****

**Error:**

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

4)Java program for calculating Fibonacci series:-

Code:-

import java.util.Scanner;

public class fb {

 public static void main(String[] args) {

  int Length;

  Scanner input = new Scanner(System.in); //create object

  System.out.print("Please enter length: ");

  Length = input.nextInt();

  int[] num = new int[Length];

  num[0] = 0;

  num[1] = 1;

  for (int i = 2; i < Length; i++) {

   num[i] = num[i - 1] + num[i - 2];

  }

  System.out.println("Fibonacci Series: ");

  for (int i = 0; i < Length; i++) {

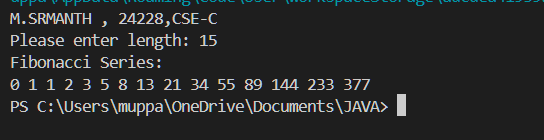
   System.out.print(num[i] + " ");

}

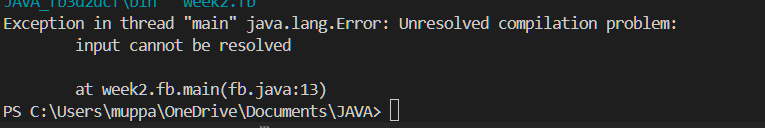
}

}

**Output:-**

****

**Negative case:**

****

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

5)Write a java program to convert temperature from Fahrenheit

to Celsius:-

Code:-

import java.util.\*;

class temp

{

public static void main(String args[])

{

Scanner sc = new Scanner(System.in); float c;

System.out.println("Enter celsius temperature:"); float f = sc.nextFloat();

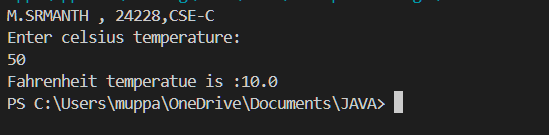
c = (f-32)\*5/9;

System.out.println("Fahrenheit temperatue is :"+c);

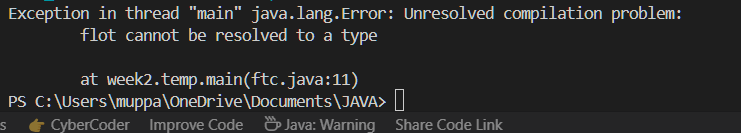
}

}

Output:-

****

**Negative case:**

****

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

**WEEK-3**

PROGRAM-1

Aim:-

To create java program with following instructions

1.Create a class with name car

2. Create four attributes named car\_color ,Car\_brand,fuel\_type,mileage

3. Create three methods named start(), stop(). Service()

4. Create three objects named car1,car2 and car3

Code:-

**i**mport java.util.\*;

class car

{

public String Car\_color;

public String Car\_brand;

public String fuel\_type;

public int mileage;

public void start()

{

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

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

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

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

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

}

public void service()

{

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

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

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

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

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

}

public void stop()

{

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

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

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

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

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

}

public static void main(String args[])

{ System.out.println("\n Srimanth\n\n");

car car1 = new car();

car1.Car\_color = "Black";

car1.Car\_brand = "BMW";

car1.fuel\_type = "Petrol";

car1.mileage = 100;

car1.start();

car car2 = new car();

car2.Car\_color = "Grey";

car2.Car\_brand = "Ferrari";

car2.fuel\_type = "EV";

car2.mileage = 500;

car2.stop();

car car3 = new car();

car3.Car\_color = "red";

car3.Car\_brand = "Jaguar";

car3.fuel\_type = "Diesel";

car3.mileage = 250;

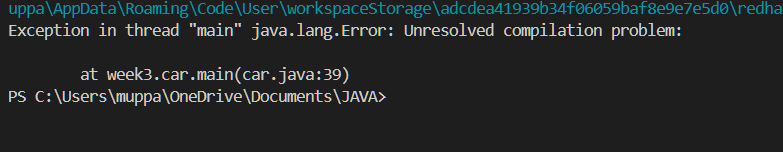
car3.service();

}

OUTPUT:-



**Negative case:**

****

PROGRAM-2

AIM:-

To create a Bank Account class with methods

DEPOSIT and WITHDRAWAL

Code:-

class BankAccount

{

private double balance;

public BankAccount(double initialBalance)

{

if(initialBalance > 0)

{

this.balance = initialBalance;

}

else

{

this.balance = 0;

}

}

public void deposit(double amount)

{

if(amount>0)

{

balance = balance+amount;

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

}

else

{

System.out.println("Amount deposited must be positive");

}

}

public double getBalance()

{

return balance;

}

}

public class Main1

{

public static void main(String args[])

{

BankAccount account = new BankAccount(1000);

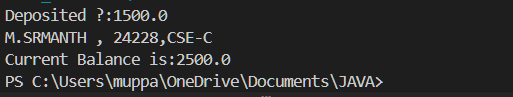
account.deposit(1500);

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

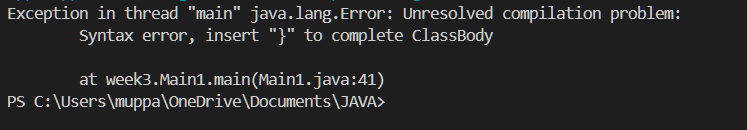
}

}

OUTPUT:-



Negative case:



**WEEK-4**

**PROGRAM:-1**

**Aim:-**

ToWrite a java program with class named book. the class should contain various attributes such as title , author ,year of obligation .it should also contain a constructer with parameters which initialised title , author , year of obligation.

CODE:-

class Book {

String title;

String author;

int yearOfPublication;

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

this.title = title;

this.author = author;

this.yearOfPublication = yearOfPublication;

}

public void displayDetails() {

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

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

System.out.println("Year of Publication: " + yearOfPublication);

}

public static void main(String[] args) {

System.out.println("M.SRMANTH , 24228,CSE-C");

Book book1 = new Book("Doomsday", "Schott", 1999);

Book1.displayDetails();

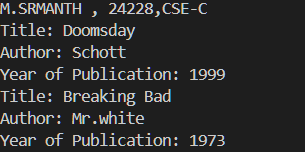
Book book2 = new Book("Breaking Bad", "Mr.white", 1973);

Book2.displayDetails();

}

}

**OUTPUT:-**

****

|  |  |
| --- | --- |
| Code Error | Code Rectififcation |
| 1.Two public classes should not be saved in same file | 1.Two public classes should be saved in different files |

**IMPORTANT POINTS:-**

1. While defining two classes for a code, we must be sure that we save both the classes in separate files.
2. While defining a method we should also define a function to call that method.

**CLASS DIAGRAM:-**

|  |
| --- |
| Book  - Title: String  - Author: String  - Year of publication: int  + Book(title: String,  Author: String;  Year of publication: int  + displayDetails( ): void |

**PROGRAM – 2:**

**AIM:-**

Create a java Program with class named myclass with static variable count of int type, initialized to zero and a constant variable “pi” of type double initialized to 3.14 as attributes of the class, ow define a constructor for “myclass” that increments the count variable each time an object of my class is created (count++), finally print the final values of count and pi variables create three objects.

**CODE:-**

public class myclass {

static int count = 0;

final double pi= 3.14;

public myclass() {

count++;

}

public static void main(String[] args) {

myclass object1 = new myclass();

myclass object2 = new myclass();

myclass object3 = new myclass();

System.out.println("M.SRMANTH , 24228,CSE-C");

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

System.out.println("Value of pi: " + object1.pi);

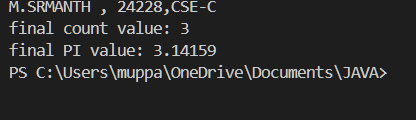
System.out.println("Value of pi: "+ object2.pi);

System.out.println("Value of pi: "+ object3.pi);

}

}

**OUTPUT:-**



**ERROR TABLE:**

|  |  |
| --- | --- |
| **Code Error** | **Code rectification** |
| 1. Not Putting the semi-colon after calling a function, | 1. Put the semi-colon after calling a function. |

**IMPORTANT POINTS:-**

1. We must declare the initial value of the variable before declaring the final one.
2. Here the main objective is to increase the count according to the number of objects we make, i.e the count increases when the no.of objects are increasing.

**CLASS DIAGRAM:-**

|  |
| --- |
| myclass   * Count: int * Pi: double   + myclass( )  + main(args: String[]): void |

WEEK-5

**AIM: Create a calculator using the operations including addition, subtraction**

**Multiplication and division using multilevel inheritance and display the desired**

**Output**

**Class Diagram:**

****

**Code:-**

**class addition**

**{**

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

**{**

**int addition = a+b;**

**return addition;**

**}**

**}**

**class subtraction extends addition**

**{**

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

**{**

**int subtraction = a-b;**

**return subtraction;**

**}**

**}**

**class multiplication extends subtraction**

**{**

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

**{**

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

**return multiplication;**

**}**

**}**

**class division extends multiplication**

**{**

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

**{**

**int division = a/b;**

**return division;**

**}**

**}**

**class calculator**

**{**

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

**{**

**division cal = new division();**

**System.out.println("M.SRMANTH , 24228,CSE-C");**

**System.out.println("Addition is:"+ cal.add(8,7));**

**System.out.println ("Subtraction is:"+cal.sub(5,2));**

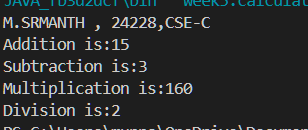
**System.out.println("Multiplication is:"+cal.mult(20,8));**

**System.out.println("Division is:"+cal.div(16,8));**

**}**

**}**

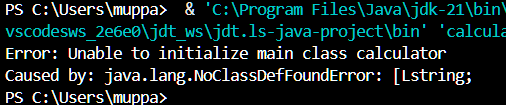
**Output:-**

****

**Error Table:-**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Error Type** | **Cause** | **Rectification** |
| **1** | **Constructor error** | **Invalid name to method** | **Defined class name** |
| **2** | **Syntax error** | **Expected ‘S’ instead of s** | **Added S to string** |
| **3** | **Logical error** | **Incorrect arithmetic**  **operation** | **Correct operation**  **rectified** |

**Negative Case:-**



**2.**

**Aim: A vehicle rental company wants to develop a system that maintains**

**Information about different types of vehicles available for rent**

**The Company rents out cars, bikes and truck and they need a program to**

**Store details about each vehicle, such as brand and speed**

**Cars should have an additional property: number of doors**

**Bikes should have a property indicating whether they have gears or not**

**The system should also include a function to display details about each vehicle**

**And indicate when a vehicle is starting**

**Class diagram**

****

**Code:-**

**class vehicle {**

**String brand;**

**int speed;**

**public vehicle(String brand, int speed) {**

**this.brand = brand;**

**this.speed = speed;**

**}**

**void displayDetails() {**

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

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

**}**

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

**System.out.println("M.SRMANTH , 24228,CSE-C");**

**car obj1 = new car("BMW", 45, 4);**

**obj1.displayDetails();**

**bike obj2 = new bike("Royal Enfield", 120, true);**

**obj2.displayDetails();**

**truck obj3 = new truck("Mahindra", 70, 100);**

**obj3.displayDetails();**

**}**

**}**

**class car extends vehicle {**

**int noofdoors;**

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

**super(brand, speed);**

**this.noofdoors = noofdoors;**

**}**

**void displayDetails() {**

**super.displayDetails();**

**System.out.println("No of Doors: " + noofdoors);**

**}**

**}**

**class bike extends vehicle {**

**boolean gears;**

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

**super(brand, speed);**

**this.gears = gears;**

**}**

**void displayDetails() {**

**super.displayDetails();**

**System.out.println("Has Gears: " + gears);**

**}**

**}**

**class truck extends vehicle {**

**int weight;**

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

**super(brand, speed);**

**this.weight = weight;**

**}**

**void displayDetails() {**

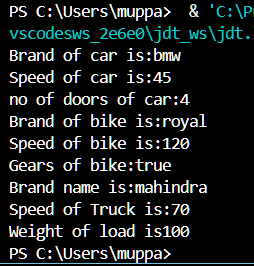
**super.displayDetails();**

**System.out.println("Load Capacity: " + weight + "kg");**

**}**

**}**

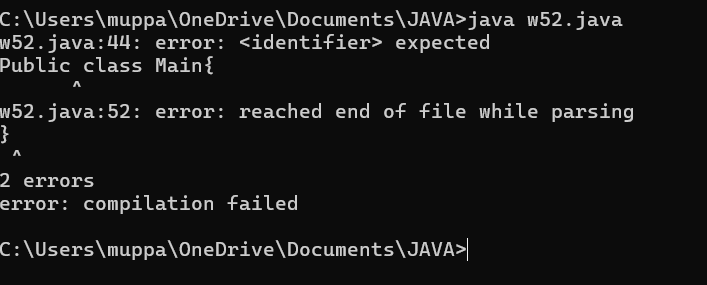
**Output:-**



**Error Table:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S No** | **Error Type** | **Cause** | **Rectification** |
| **1** | **Syntax Error** | **Semicolon missing** | **Added ;** |
| **2** |  |  |  |
| **3** |  |  |  |

**Negative Case:-**



**Important Points:-**

**Hierarchical Inheritence:**

**This is a type of inheritance occurs when multiple subclasses inherit from a**

**Single parent class**

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 machine {

    public static void main(String[] args){

 System.out.println("M.SRMANTH , 24228,CSE-C");

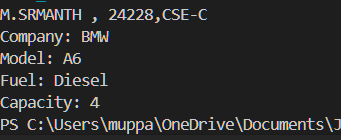
        car car1=new car();

         car1.displayInfo("BMW","A6","Diesel",4);

    }

}

**OUTPUT :**

****

**ERRORS :**

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

**CLASS DIAGRAM:**

|  |
| --- |
| Vehicle |
| +speed:int  +brand:string  +capacity:int |
| +DISPLAYINFO():void  + VEHICLE (INT )  ,SPEED INT CAPACITY ,STRING BRAND) |

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

**2)**

**Aim :** A college is developing an automated admission system that verifies students'

eligibility for undergraduate (UG) and postgraduate (PG) programs. Each program has

different 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%

**CLASS DIAGRAM :**

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

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

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

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

**Program :**

class student {

    public String name;

    public int percentage;

    student(String name, int percentage){

*this*.name = name;

*this*.percentage = percentage;

    }

}

class ugstudent extends student{

  ugstudent(String name, int percentage){

*super*(name,percentage);

  }

  void geteligibilty(){

*if* (percentage>=60){

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

    }*else*{

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

    }

  }

}

class pgstudent extends student{

    pgstudent(String name, int percentage){

*super*(name,percentage);

    }

    void geteligibilty(){

*if* (percentage>=70){

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

      }*else*{

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

      }

    }

}

public class admission{

    public static void main(String[] args){

 System.out.println("M.SRMANTH , 24228,CSE-C");

        ugstudent student1 = *new* ugstudent("ram",40);

        student1.geteligibilty();

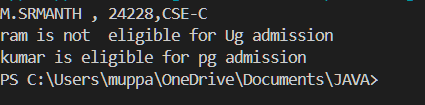
        pgstudent student2 =*new* pgstudent("kumar",90);

        student2.geteligibilty();

    }

}

**OUPUT :**

****

**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 |

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

(i) Add two integers.

(ii) Add two doubles.

(iii) Add three integers.

Class diagram :

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

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

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

**Program :**

class Calcu {

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) {

        Calcu C1 = *new* Calcu();

        System.out.println("M.SRMANTH , 24228,CSE-C");

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

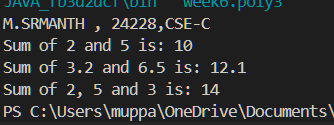
        System.out.println("Sum of 3.2 and 6.5 is: " + C1.add(2.6, 9.5));

        System.out.println("Sum of 2, 5 and 3 is: " + C1.add(1, 6, 7));

    }

}

**OUTPUT :**

****

**ERROR :**

|  |  |  |  |
| --- | --- | --- | --- |
| **1.** | **Logical error** | **Incorrect formula** | **Correcting formula** |
| **2.** | **Runtime errror** | **Incorrect path** | **Using correct path** |
| **3.** | **Syntax error** | **Incorrect symbols** | **Using correct symbols** |

**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 {

    void calculateArea( int a) {

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

    }

    void calculateArea(int a , int b) {

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

    }

}

class circle extends Shape {

    void calculateArea(double a){

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

  } }

class main {

    public static void main(String[] args) {

 System.out.println("M.SRMANTH , 24228,CSE-C");

        Shape s = new Shape();

        circle c = new circle();

        s.calculateArea(6);

        System.out.println("    ");

        s.calculateArea(4, 6);

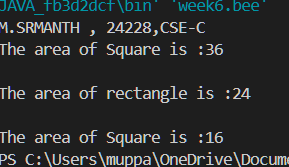
        System.out.println("    ");

        c.calculateArea(4);

    }

}

**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.

**CLASS DIAGRAM:**

|  |
| --- |
| < ABSTACT CLASS >  Animal |
| + sound() : void |

|  |
| --- |
| Lion |
| +sound() : void |

|  |
| --- |
| Tiger |
| +sound() : void |

**Code:**

abstract class Animal {

abstract void sound();

}

class Lion extends Animal {

void sound() {

System.out.println("Lion roars");

}

}

class Tiger extends Animal {

void sound() {

System.out.println("Tiger growls");

}

}

public class a17 {

public static void main(String[] args) {

System.out.println("Name: M. Srimanth,Roll No: 24228,Section: CSE-C");

Animal lion = new Lion();

Animal tiger = new Tiger();

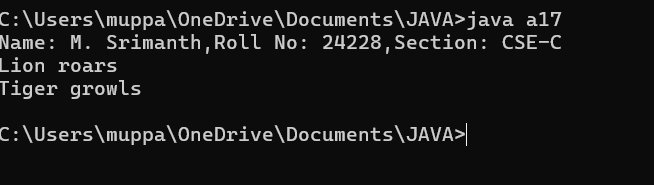
lion.sound();

tiger.sound();

}

}

**Output:**

****

**IMPORTANT POINTS:**

Abstract method sound() in base class.

Implemented in Lion and Tiger subclasses**.**

**2)**

**Aim:**

Write a Java program to create an abstract class Shape3D with abstract methods calculateVolume() and calculateSurfaceArea(). Create subclasses Sphere and Cube that extend the Shape3D class and implement the

respective methods to calculate the volume and surface area of each shape.

**CLASS DIAGRAM :**

|  |
| --- |
| <abstract>>  Shape3D |
| +calculateVolume(): double +calculateSurfaceArea(): double |

|  |
| --- |
| Sphere |
| - radius: int |
| +calculateVolume() +calculateSurfaceArea() |

|  |
| --- |
| CUBE |
| - a: int |
| +calculateVolume()  +calculateSurfaceArea() |

|  |
| --- |
| Sphere |
| - radius: int |
| +calculateVolume() +calculateSurfaceArea() |

**Code:**

abstract class Shape3D {

    abstract double calculateVolume();

    abstract double calculateSurfaceArea();

}

class Sphere extends Shape3D {

    double radius;

    Sphere(double radius) {

*this*.radius = radius;

    }

    double calculateVolume() {

*return* (4.0 / 3.0) \* Math.PI \* Math.pow(radius, 3);

    }

    double calculateSurfaceArea() {

*return* 4 \* Math.PI \* Math.pow(radius, 2);

    }

}

class Cube extends Shape3D {

    double side;

    Cube(double side) {

*this*.side = side;

    }

    double calculateVolume() {

*return* Math.pow(side, 3);

    }

    double calculateSurfaceArea() {

*return* 6 \* Math.pow(side, 2);

    }

}

public class a18 {

    public static void main(String[] args) {

        System.out.println("Name: M. Srimanth\nRoll No: 24228\nSection: CSE-C");

        Shape3D sphere = *new* Sphere(6);

        Shape3D cube = *new* Cube(4);

        System.out.println("Volume: " + sphere.calculateVolume());

        System.out.println("Surface Area: " + sphere.calculateSurfaceArea());

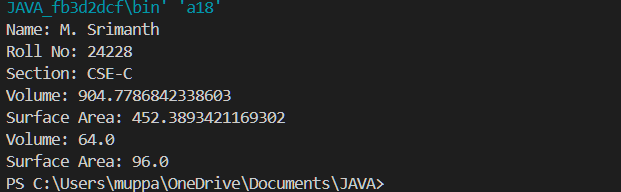
        System.out.println("Volume: " + cube.calculateVolume());

        System.out.println("Surface Area: " + cube.calculateSurfaceArea());

    }

}

**Output:**

****

**IMPORTANT POINTS:**

Abstract class Shape3D with volume & surface area methods.

3.

**AIM:-**

Write a java using an abstract class to define a method for Pattern Printing

**CLASS DIAGRAM :**

|  |
| --- |
| <<abstract>>  PatternPrinter |
| +printpattern(int): void  +display(String): void |

|  |
| --- |
| starpattern |
| +printpattern(int) |

|  |
| --- |
| numberpattern |
| +printpattern(int) |

**Code:-**

abstract class PatternPrinter

{

   abstract void printpattern(int n);

   public void display(String title)

   {

       System.out.println("\n=== "+ title+"===");

   }

}

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 num=1;

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

       {

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

          {

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

              num++;

          }

          System.out.println();

      }

  }

}

class Printer{

  public static void main(String[] args){

   System.out.println("M.SRIMANTH");

   System.out.println("CSE-C");

   System.out.println("24228");

   int rows = 5;

   PatternPrinter star = *new* starpattern();

   star.display("Star pattern");

   star.printpattern(rows);

   PatternPrinter number = *new* numberpattern();

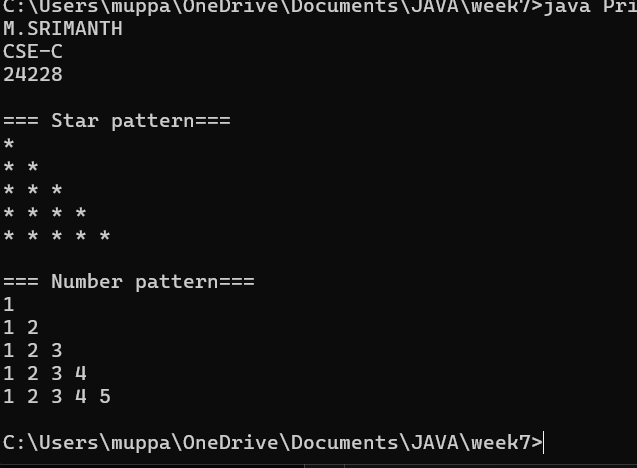
   number.display("Number pattern");

   number.printpattern(rows);

}

}

**OUTPUT:-**

****

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.

**CLASS DIAGRAM :**

|  |
| --- |
| <<interface>>  Shape |
| +getPerimeter() |

|  |
| --- |
| Rectangle |
| -l : int  - b: int |
| +getPerimeter() |

|  |
| --- |
| Triangle |
| - s1, s2, s3 |
| +getPerimeter() |

|  |
| --- |
| Circle |
| - r: int |
| +getPerimeter() |

**Code:**

**interface Shape {**

**public double getPerimeter();**

**}**

**class Rectangle implements Shape {**

**double length, breadth;**

**Rectangle(double length, double breadth) {**

***this*.length = length;**

***this*.breadth = breadth;**

**}**

**public double getPerimeter() {**

***return* 2 \* (length + breadth);**

**}**

**}**

**class Circle implements Shape {**

**double radius;**

**Circle(double radius) {**

***this*.radius = radius;**

**}**

**public double getPerimeter() {**

***return* 2 \* Math.PI \* radius;**

**}**

**}**

**class Triangle implements Shape {**

**double side1, side2, side3;**

**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 a20 {**

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

**System.out.println("Name: M. Srimanth\nRoll No: 24228\nSection: CSE-C");**

**Shape rectangle = *new* Rectangle(5, 6);**

**Shape circle = *new* Circle(5);**

**Shape triangle = *new* Triangle(5, 5, 5);**

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

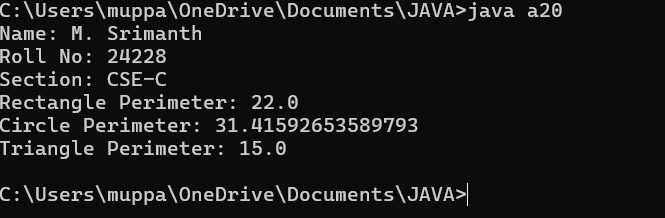
**System.out.println("Circle Perimeter: " + circle.getPerimeter());**

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

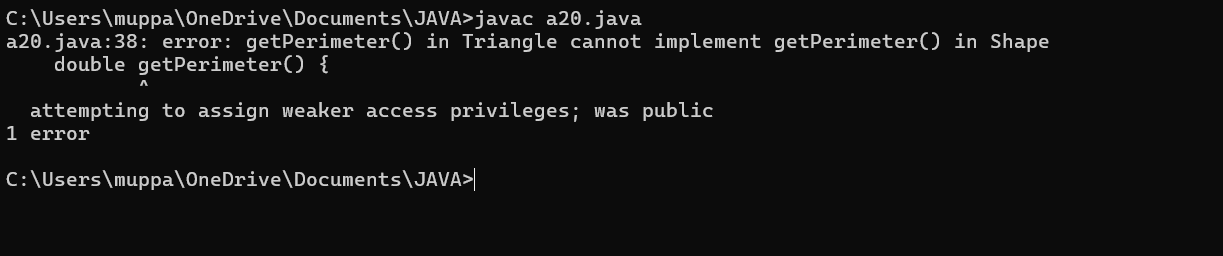
**}**

**}**

**Output:**

****

**Negative case:**

****

**Error:**

|  |  |  |
| --- | --- | --- |
| **S.no** | **Error** | **rectification** |
| **1** | **Not declared method as public** | **Changed method to public** |

**IMPORTANT POINTS:**

Interface implementation for geometric shapes.

Demonstrates polymorphism using interface.

**2.**

**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.

**CLASS DIAGRAM :**

|  |
| --- |
| <<interface>>  playable |
| +play(): void |

|  |
| --- |
| football |
| +play() |

|  |
| --- |
| Volleyball |
| +play() |

|  |
| --- |
| basketball |
| +play() |

**Code:**

**interface Playable {**

**void play();**

**}**

**class Football implements Playable {**

**@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");**

**}**

**}**

**public class PlayableDemo {**

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

**System.out.println("Name   : M. SRIMANTH");**

**System.out.println("Roll No: 24228");**

**System.out.println("Section: CSE-C\n");**

**Playable football = *new* Football();**

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

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

**football.play();**

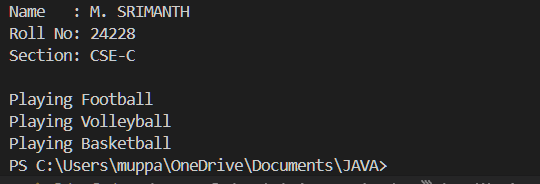
**volleyball.play();**

**basketball.play();**

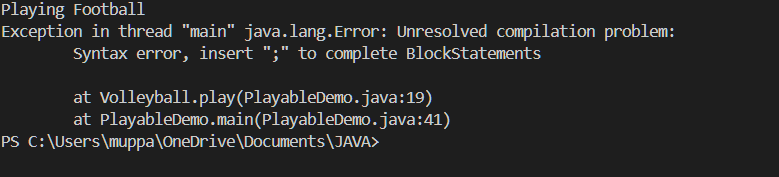
**}**

**}**

**Output:**

****

**Negative case:-**

****

|  |  |  |
| --- | --- | --- |
| **S.no** | **Error** | **rectification** |
| **1** | **Not inserted ; while printing the output** | **Inserted ;** |

**IMPORTANT POINTS:**

Interface method play() in Football, Volleyball, Basketball.