**23CSE111**

**OBJECT ORIENTED PROGRAMMING**

**LAB REPORT**



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

**Amrita Vishwa Vidyapeetham, Amaravati Campus**

**Name: Mohan Sai**

**Roll No: 24219**

**Verified By :**

# INDEX

**INDEX**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S no** | **Title** | **Page no** | **Date** | **Signature** |
|  | **WEEK -1** |  |  |  |
| **1** | **Explain the process of download & Installation of JDK** | **4** |  |  |
| **2** | **Write a program to print Student’s name, Roll no, Section** | **7** |  |  |
|  | **WEEK-2** |  |  |  |
| **1** | **Write a simple java program to calculate factorial of a number** | **9** |  |  |
| **2** | **Write a simple java program to find the simple interest by taking iinputs from the user** | **10** |  |  |
| **3** | **Write a program to calculate the Fibonacci sequence and take the input from the user** | **13** |  |  |
| **4** | **Write a program to find the area of triangle using hereon’s formula**  **.** | **15** |  |  |
| **5** | **Write a program to convert temperature from celssius to fahrenheit** | **17** |  |  |
|  | **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. **AIM:**

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

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

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

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

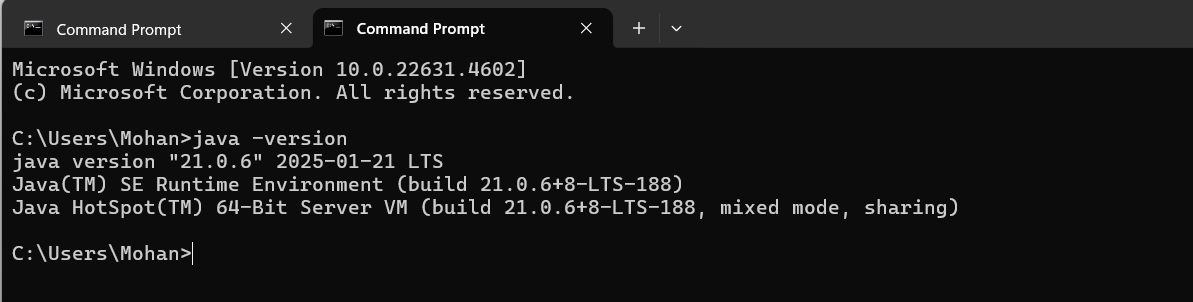
C:\Program Files\Java\jdk-21\bin)

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



**Checking JDK Version: -**

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

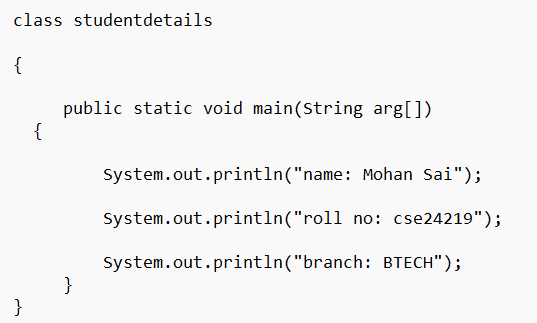


**2)AIM:**

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

Write your code in Notepad and execute it in cmd prompt

**CODE: -**



**Output: -**



**ERROR TABLE:**

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

**WEEK-2**

**1)**

**AIM ;**

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

**User**

**Code:**



**Output:**

****

**ERROR TABLE:**

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

**2)**

**AIM:**

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

**input from user**

**code:**



**Output**

****

**ERROR TABLE:**

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

**3)**

**AIM:**

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

**Code:**

import java.util.\*;

class fibo

{

public static void main(String args[])

{

Scanner sc = new Scanner(System.in);

int num;

int f3;

int f1 = 0;

int f2 = 1;

int i = 2;

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

num = sc.nextInt();

System.out.println(f1);

System.out.println(f2);

while(i<num)

{

f3 = f1+f2;

f1 = f2;

f2 = f3;

System.out.println(f3);

i = i+1;

}

}

}

**Output:**

**\***

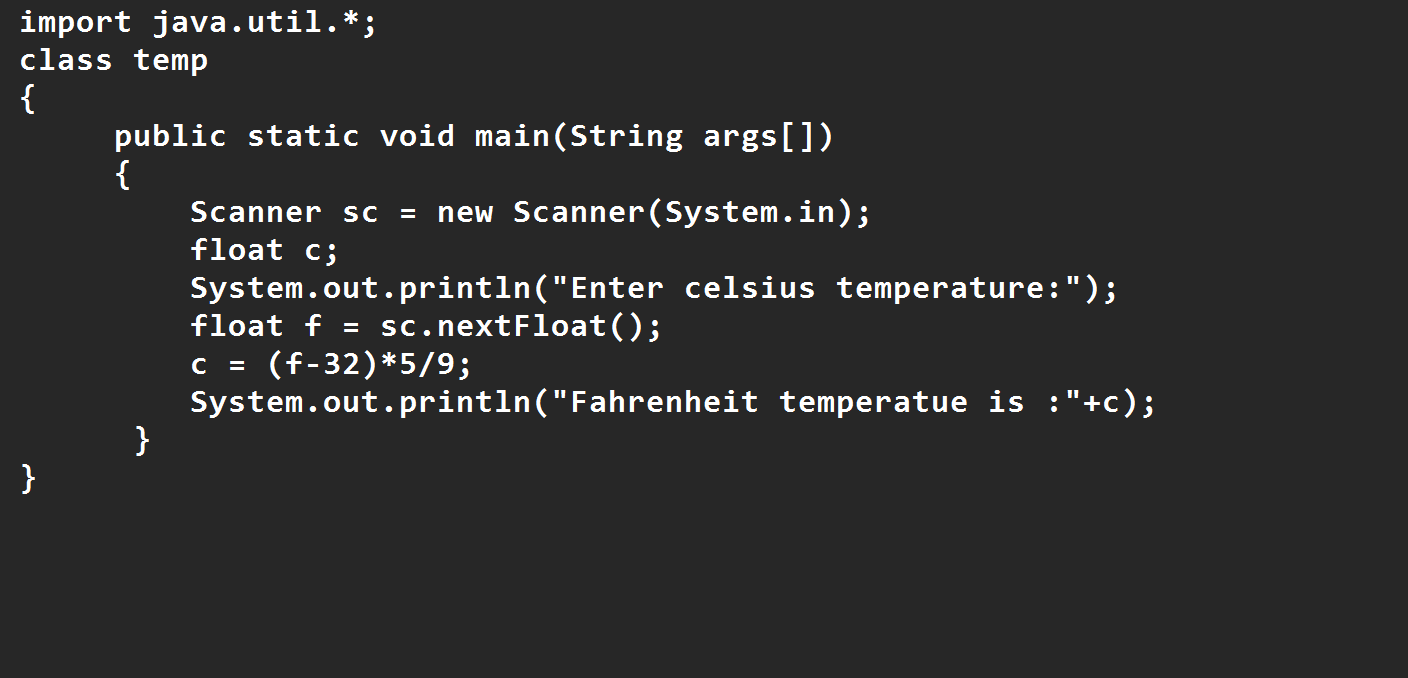
**ERROR TABLE:**

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

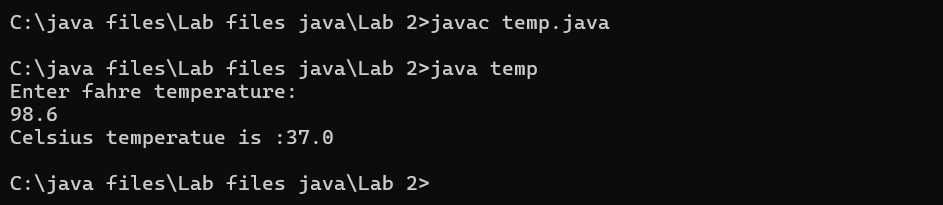
**4)AIM;**

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

**Code:**



**Output:**

****

**ERROR TABLE:**

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

**5)AIM:**

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

**Code:**



**Output:**

****

**ERROR TABLE:**

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

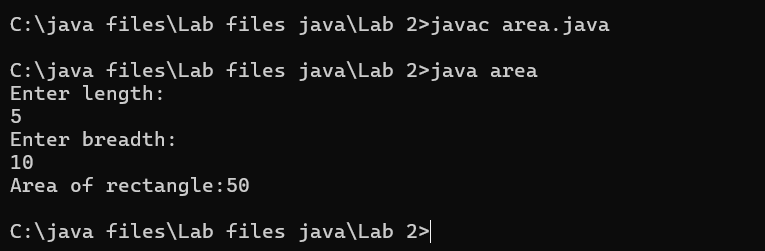
**6)AIM:**

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

**Code:**



**Output:**



**ERROR TABLE:**

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

**7)AIM:**

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

**take the input from the user**

**Code:**



**Output:**

****

**ERROR TABLE:**

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

**WEEK 3**

**Aim:**

**To create java program with following instructions**

**1.Create a class with name car**

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

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

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

**Code:**

import java.util.\*;

class car

{

public String Car\_color;

public String Car\_brand;

public String fuel\_type;

public int mileage;

public void start()

{

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

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

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

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

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

}

public void service()

{

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

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

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

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

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

}

public void stop()

{

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

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

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

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

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

}

public static void main(String args[])

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

car car1 = new car();

car1.Car\_color = "Blue";

car1.Car\_brand = "BMW";

car1.fuel\_type = "Deisel";

car1.mileage = 10;

car1.start();

car car2 = new car();

car2.Car\_color = "Red";

car2.Car\_brand = "Tesla";

car2.fuel\_type = "EV";

car2.mileage = 300;

car2.stop();

car car3 = new car();

car3.Car\_color = "Yellow";

car3.Car\_brand = "MAHINDRA";

car3.fuel\_type = "Petrol";

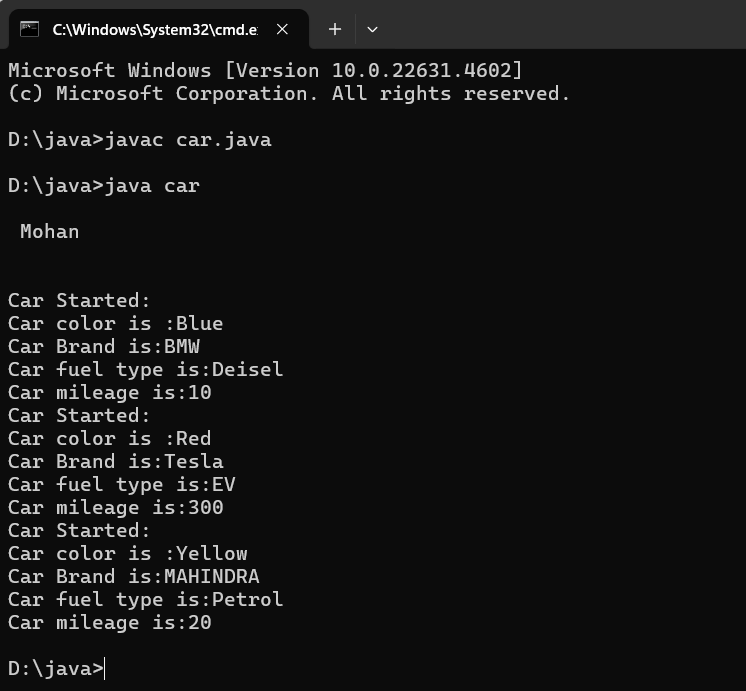
car3.mileage = 20;

car3.service();

}

}

**Output:**



Class Diagram

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

**Concepts to be known:**

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

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

**2.)**

**AIM:**

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

**Code:**

class BankAccount

{

private double balance;

public BankAccount(double initialBalance)

{

if(initialBalance > 0)

{

this.balance = initialBalance;

}

else

{

this.balance = 0;

}

}

public void deposit(double amount)

{

if(amount>0)

{

balance = balance+amount;

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

}

else

{

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

}

}

public double getBalance()

{

return balance;

}

}

public class Main1

{

public static void main(String args[])

{

BankAccount account = new BankAccount(1000);

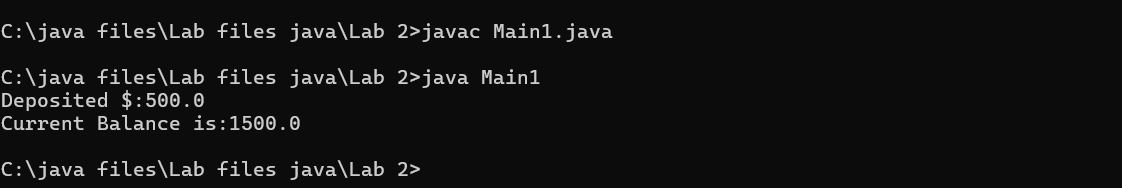
account.deposit(500);

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

}

}

**Output:**

****

**ERROR TABLE:**

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

**Concepts to be known:**

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

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

**WEEK-4**

**1.AIM:**

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

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

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

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

**PROGRAM:**

public class Book {

public String title;

public String author;

public int year;

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

this.title = title;

this.author = author;

this.year = year;

}

public void displayDetails() {

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

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

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

}

public static void main(String[] args) {

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

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

b1.displayDetails();

b2.displayDetails();

}

}

**Output:**



**NEGATIVE CASE:**

A black screen with white text

AI-generated content may be incorrect.

**ERROR TABLE:**

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

**CLASS DIAGRAM:**

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

**IMPORTANT POINTS:**

1. **Constructor**:

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

2.**Method**:

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

3. **Object Creation**:

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

**2.AIM:**

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

**PROGRAM:**

public class MyClass {

static int count = 0;

static final double pi = 3.14;

MyClass() {

count++;

}

public static void main(String[] args) {

MyClass obj1 = new MyClass();

MyClass obj2 = new MyClass();

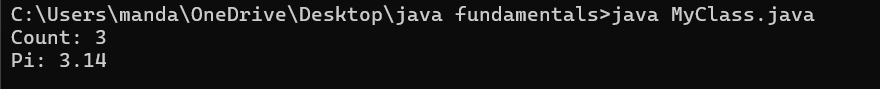
MyClass obj3 = new MyClass();

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

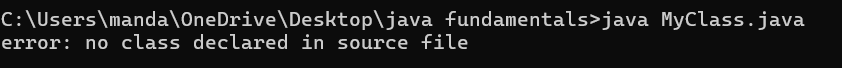
System.out.println("Pi: " +pi);

} }

**OUTPUT:**

****

**NEGATIVE CASE:**

****

**ERROR TABLE:**

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

**CLASS DIAGRAM:**

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

**IMPORTANT POINTS:**

**1.Static Keyword**

* Static members belong to the **class, not to individual objects**.
* Only one copy of the static variable is maintained for all objects.

**2.Static Variable**

* **static int count**:
  + Shared among all objects of the class.
  + It is initialized only once and not for every object.
  + It increments every time the constructor is called.

**3.Final Variable**

* **static final double pi**:
  + The **final** keyword makes the variable constant.
  + Its value **cannot be changed** once assigned.
  + It must be initialized at the time of declaration.

**WEEK-5**

1.AIM:

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

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

**Output.**

**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 obj = new division();

System.out.println("Addition is:"+ obj.add(10,2));

System.out.println ("Subtraction is:"+obj.sub(8,4));

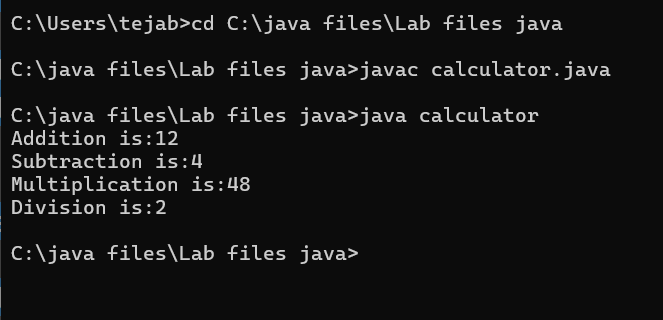
System.out.println("Multiplication is:"+obj.mult(12,4));

System.out.println("Division is:"+obj.div(8,4));

  }

}

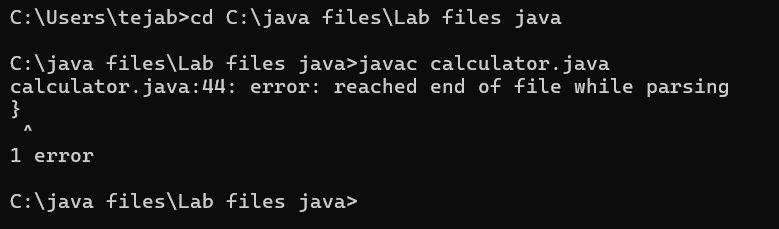
**Output:**



Error Table

|  |  |  |  |
| --- | --- | --- | --- |
| S.no | Error Type | Cause | Rectification |
| 1 | Constructor error | Invalid name to method | Defined class name |
| 2 | Syntax error | Expected ‘(‘ | Added parenthesis |
| 3 | Logical error | Incorrect arithmetic  operation | Correct operation  rectified |

**Negative Case:**



Class diagram:



**Important Points:**

Inheritence:

The concept of OOP where a class inherits the properties and behaviours from

Another class (parent class) which promotes code reusability and hieratchical relationships

Multilevel Inheritence:

This is a type of inheritance in which a class inherited from another class, and

That superclass, in turn, inherits from yet another class, creating a chain of

inheritence

extends:

The extends keyword defines the relation of child class with the parent class

**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 speedCars 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**

**Code:**

class vehicle{

String brand;

int speed;

public vehicle(String brand,int speed){

this.brand=brand;

this.speed=speed;

}

public static void main(String[] args) {

car obj1=new car("ford",34,4);

bike obj2=new bike("hero",100,true);

truck obj3=new truck("tata",60,40);

}

}

class car extends vehicle{

int noofdoors;

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

super(brand, speed);

this.noofdoors=noofdoors;

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

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

System.out.println("no of doors of car:"+noofdoors);

}

}

class bike extends vehicle{

boolean gears;

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

super(brand, speed);

this.gears=gears;

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

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

System.out.println("Gears of bike:"+gears);

}

}

class truck extends vehicle{

int weight;

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

super(brand,speed);

this.weight=weight;

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

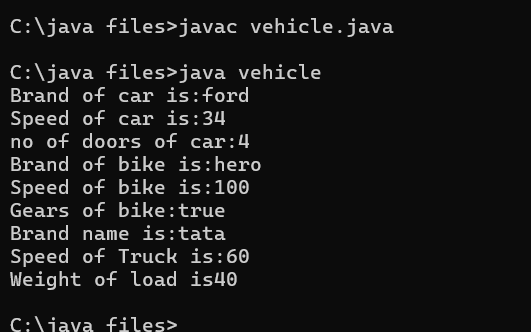
System.out.println("Speed of Truck is:"+speed);

System.out.println("Weight of load is"+weight);

}

}

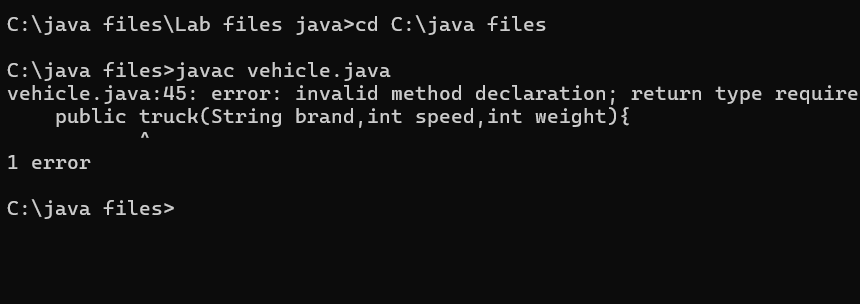
Output:



Error Table:

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

Negative Case:



**Class diagram:**



**Important Points:**

Hierarchical Inheritence:

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

Single

**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.**

**CODE:**

**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){**

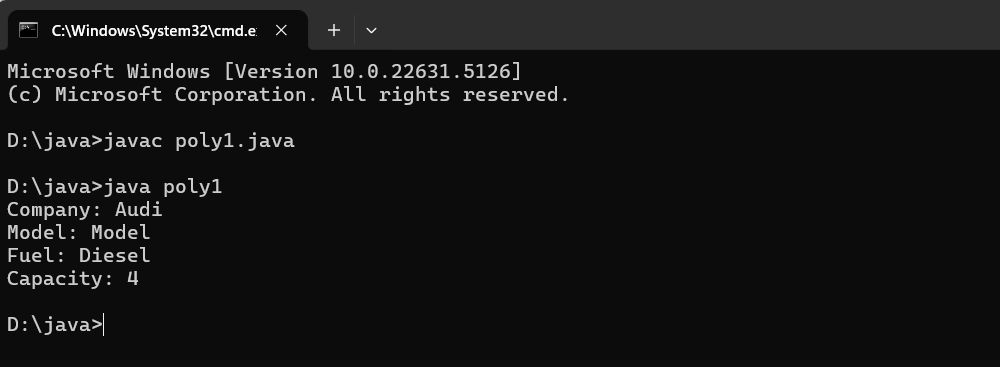
**car car1=new car();**

**car1.displayInfo("Audi","Model","Diesel",4);**

**}**

**}**

**OUTPUT:**



**ERROR TABLE:**

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

1. **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%**

**CODE:**

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);

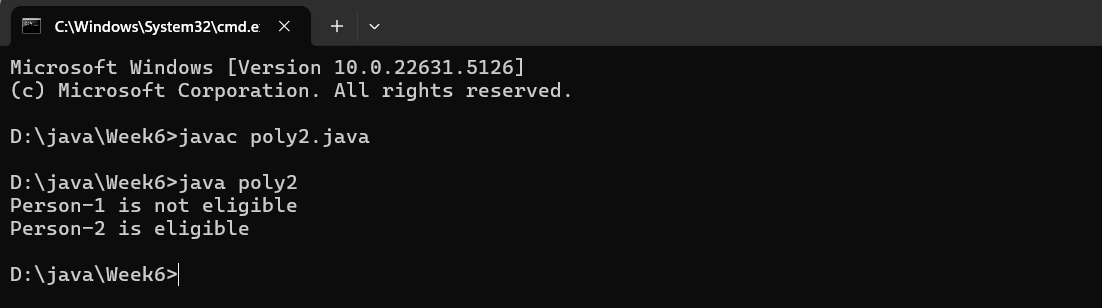
        PG pg=new PG();

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

    }

}

**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 |

1. **Aim :**

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

**(i) Add two integers.**

**(ii) Add two doubles.**

**(iii) Add three integers**.

**Program :**

**CODE:**

class Calc{

    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){

        CalcC1=new Calc();

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

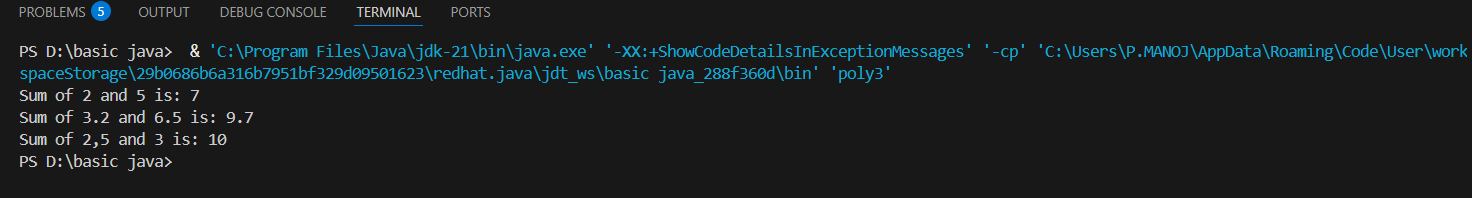
        System.out.println("Sum of 3.2 and 6.5 is: "+C1.add(3.2,6.5));

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

    }

}

**OUTPUT :**

****

**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 |

**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) {

        Shape s = new Shape();

        circle c = new circle();

        s.calculateArea(4);

        System.out.println("    ");

        s.calculateArea(4, 5);

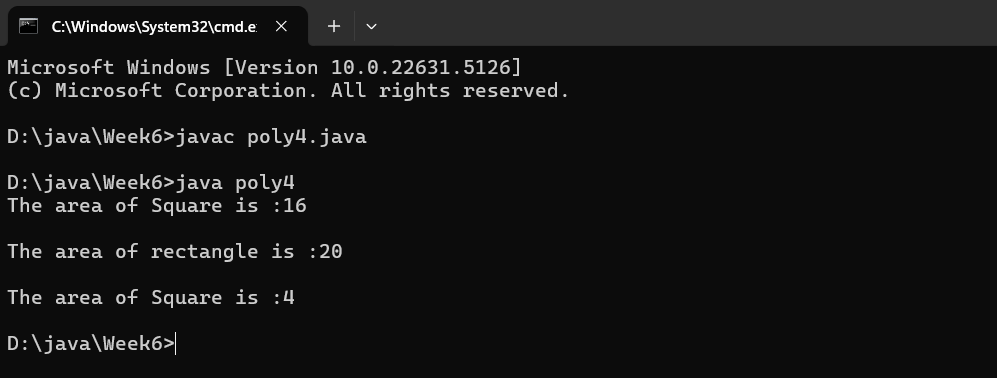
        System.out.println("    ");

        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 |