**23CSE111**

**OBJECT ORIENTED PROGRAMMING**

**LAB REPORT**



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

**Amrita Vishwa Vidyapeetham, Amaravati Campus**

**Name: P. Komal Yashwanth**

**Roll No: 24243**

**Verified By :**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S.NO** | **Experiment** | **Page No** | **Remarks** | **Signature** |
| **1** | **Installation Process of JDK** | **3-4** |  |  |
| **2** | **Simple java program for printing basic details of student** | **5** |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**INDEX**

# WEEK-1

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

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

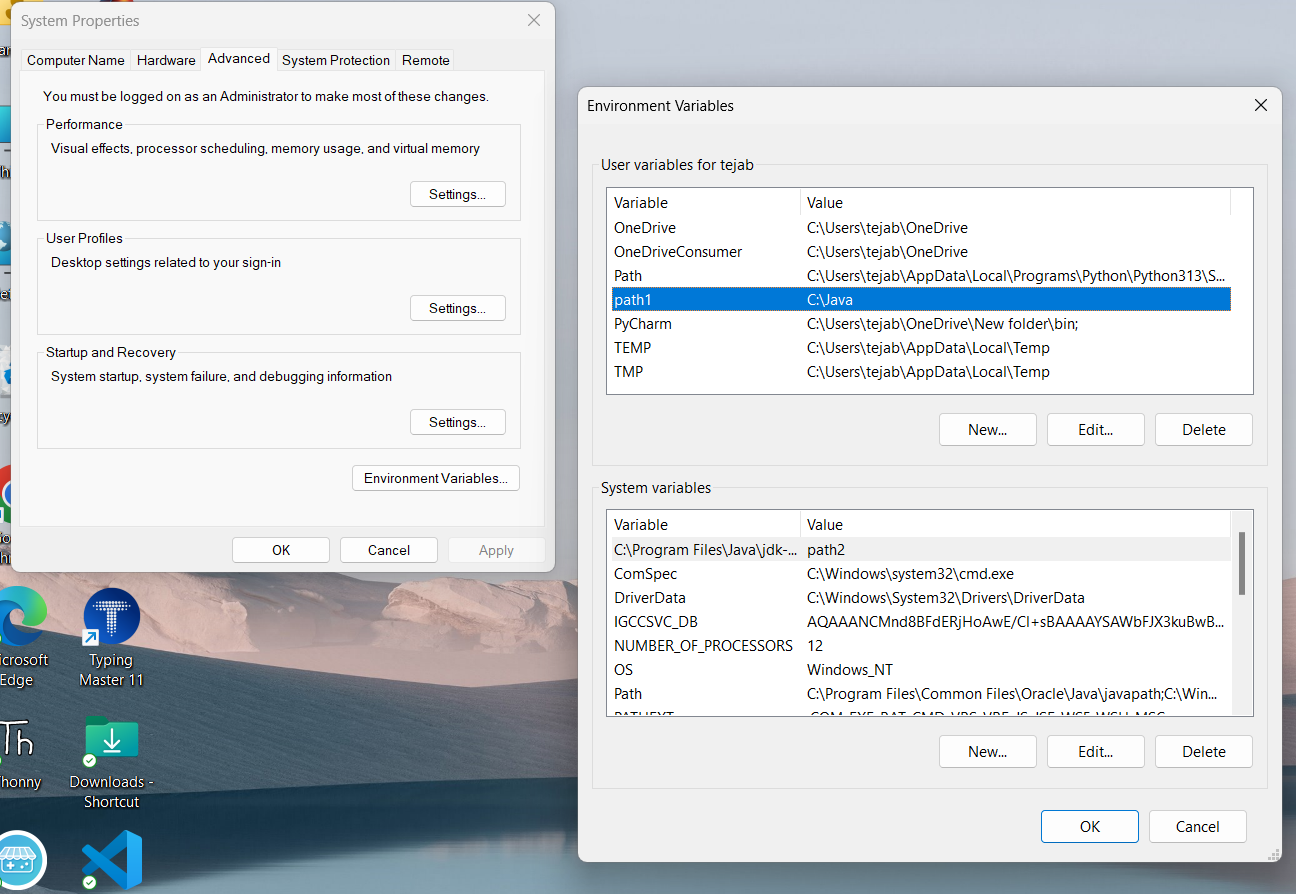
* 1. **Download JDK:**
* Go to the Oracle JDK download page in google and click on JDK-21 version which is Long term support (LTS) version.
* Click the download link as your operating system (Windows, macOS, or Linux).
  1. **Install JDK:**
* Once downloaded, run the installer.
* Follow the given instructions 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 4

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

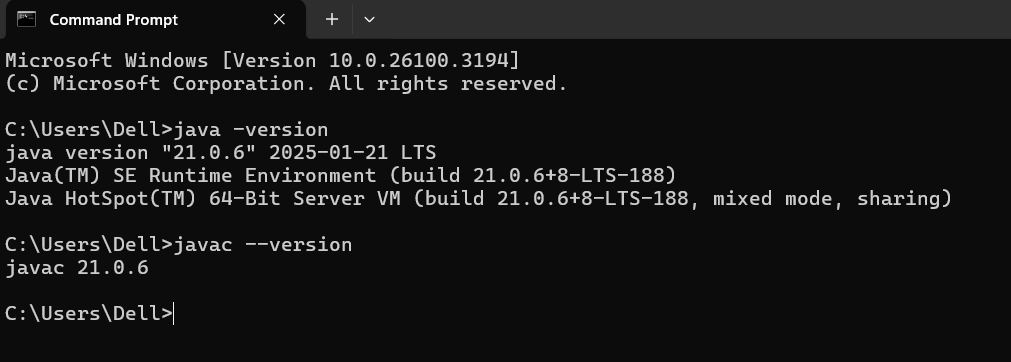
* Find Path under **System Variables**, click **New**, and add the path of the jdk-21(C:\Program Files\Java\jdk-21\bin)

001235456789



**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: Komal");

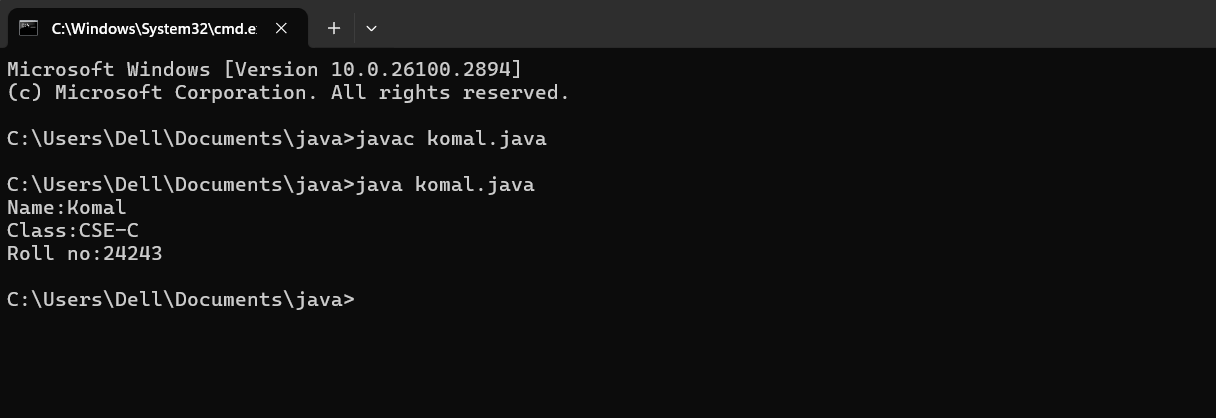
System.out.println("Class :CSE-C");

System.out.println("Roll No:24243");

}

}

**Output: -**



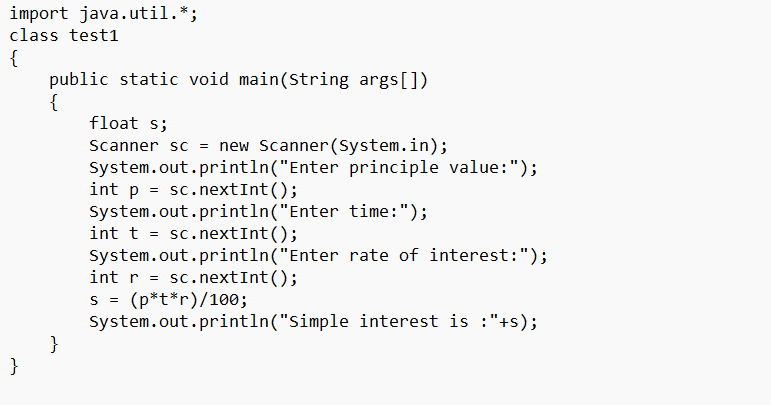
Errors

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

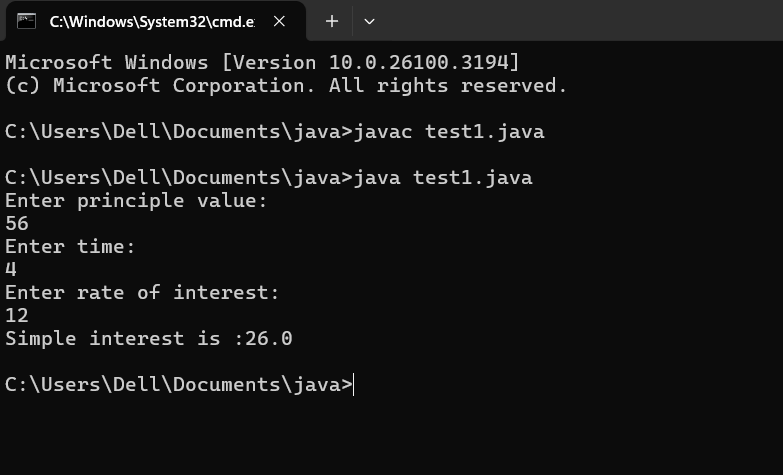
1. **Simple Java Program for finding simple interest by taking input from**

**User**

**Code:**

****

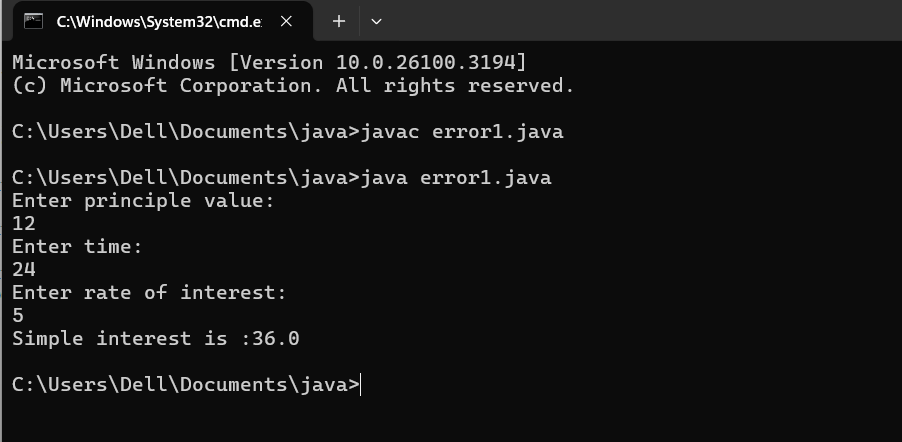
**Output:- Positive Case:**

****

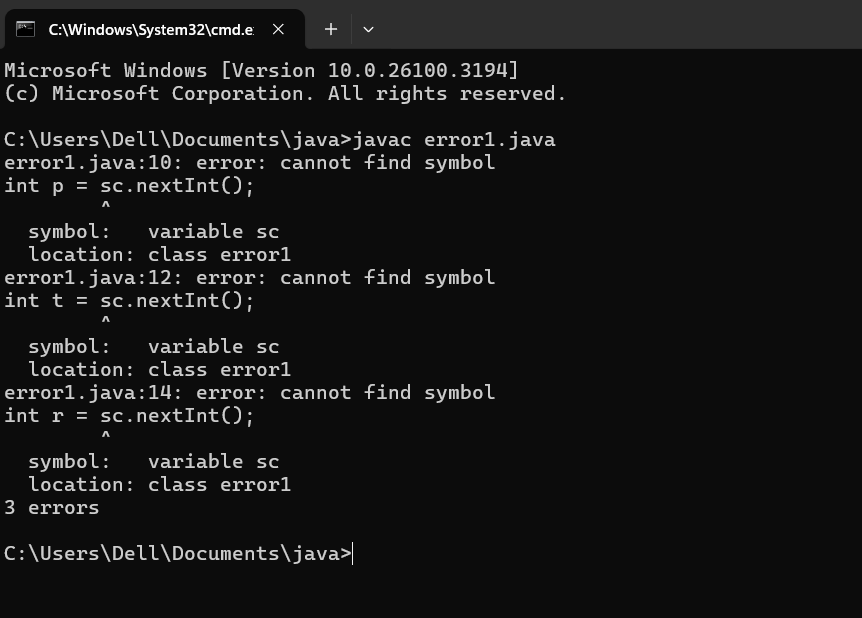
|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Error type** | **Reason for error** | **rectification** |
| **1** | **Logical Error** | **Due to incorrect formula the result may be invalid, In this case int s variable is assigned as (p+t+r)/100** | **We can achieve valid result by assigning proper formula s=(p\*t\*r)/100** |
| **2** | **Compilation Error** | **Scanner code is declared as SC, Java is case-sensitive.** | **Scanner Code is rectified by replacing sc instead of SC** |
| **3** | **Syntax Error** | **The line Scanner SC is missing the semicolon** | **The statement should end with semicolon** |

**Negative case:**

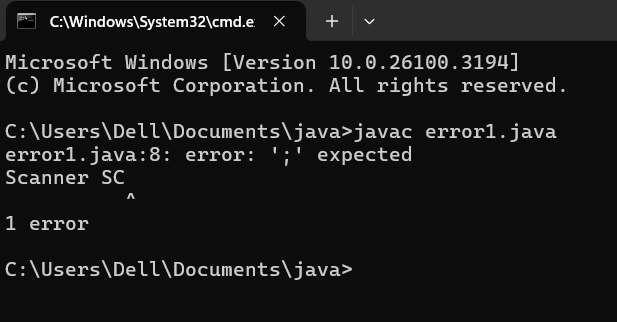
**I)**



**II)**

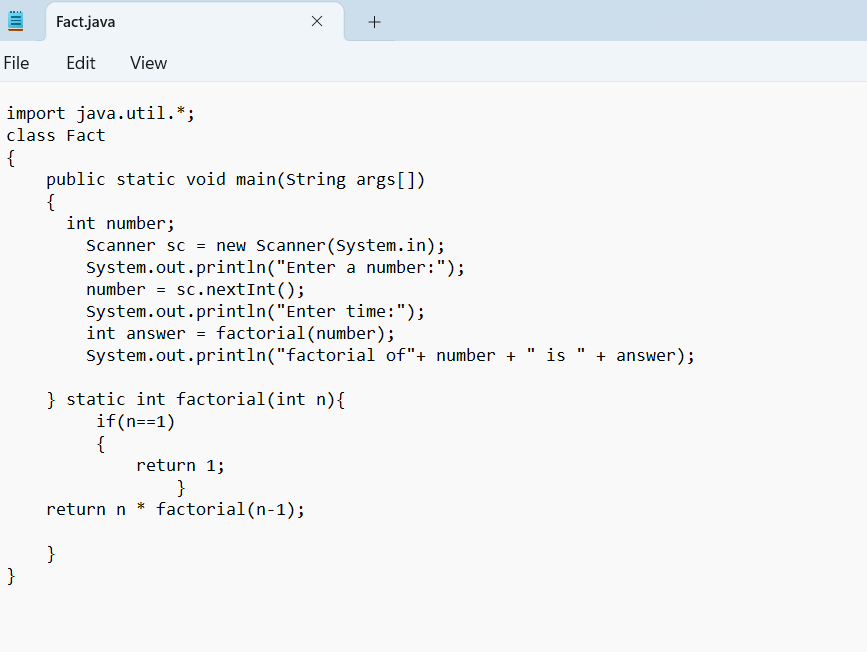
****

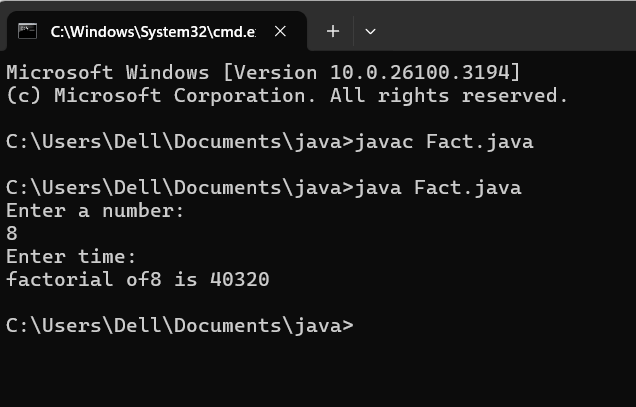
**III)**

****

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

**input from user Code:**

****

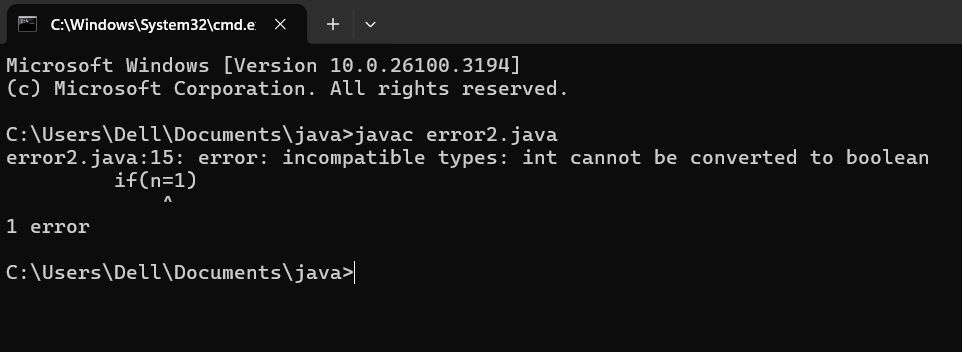
****

**ERRORS:**

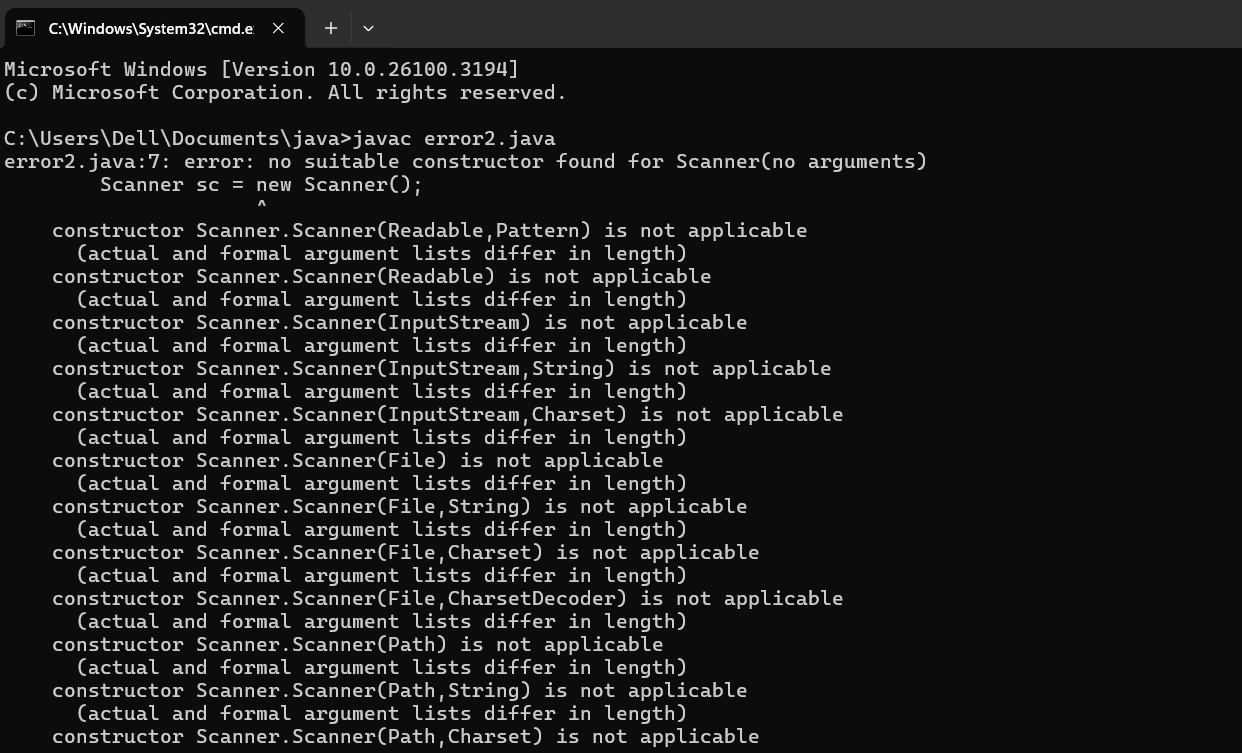
|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Error type** | **Reason for error** | **Rectification** |
| **1** | **Syntax Error** | **Incorrect Conditional Statement .n=1** | **The “=” is assignment operator, So in order to compare , the “==” should be used** |
| **2** | **Scanner Constructor Error** | **The Scanner constructor requires input like System.in inside the parenthesis , but It is missing** | **This error could be rectified by typing System.in inside parenthesis.** |
| **3** | **Logical error** | **if(n == 1)**  **{**  **return 0;**  **}** | **The factorial of 1 is 1 , so the if statement should be**  **if(n==1)**  **{**  **return 1;**  **}** |

**Negative Case:**

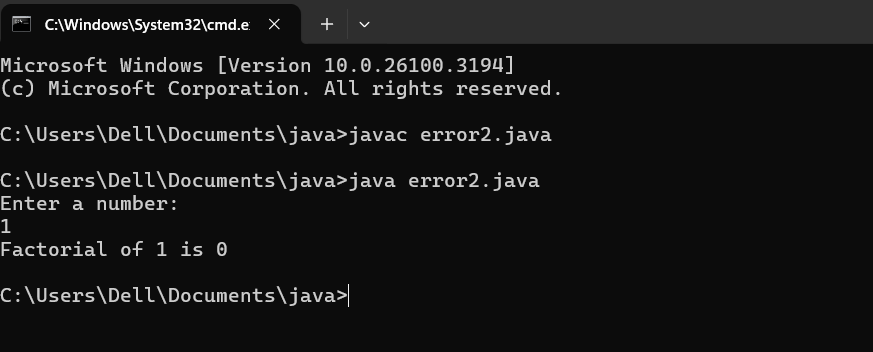
**I)**

****

**II)**

****

**III)**

****

**3.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;**

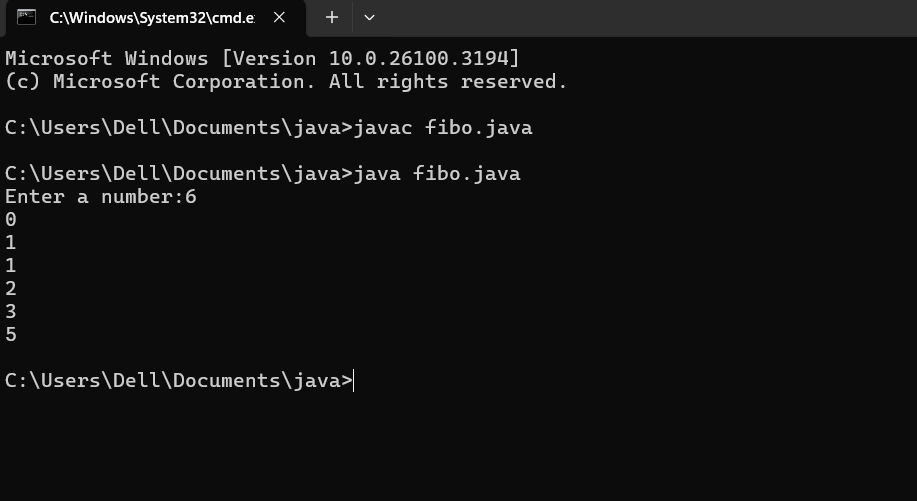
**}**

**}**

**}**

**Output:**

**POSITIVE CASE:**

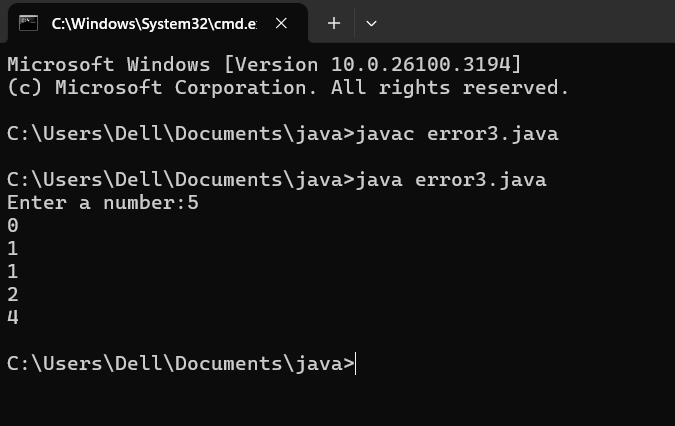
****

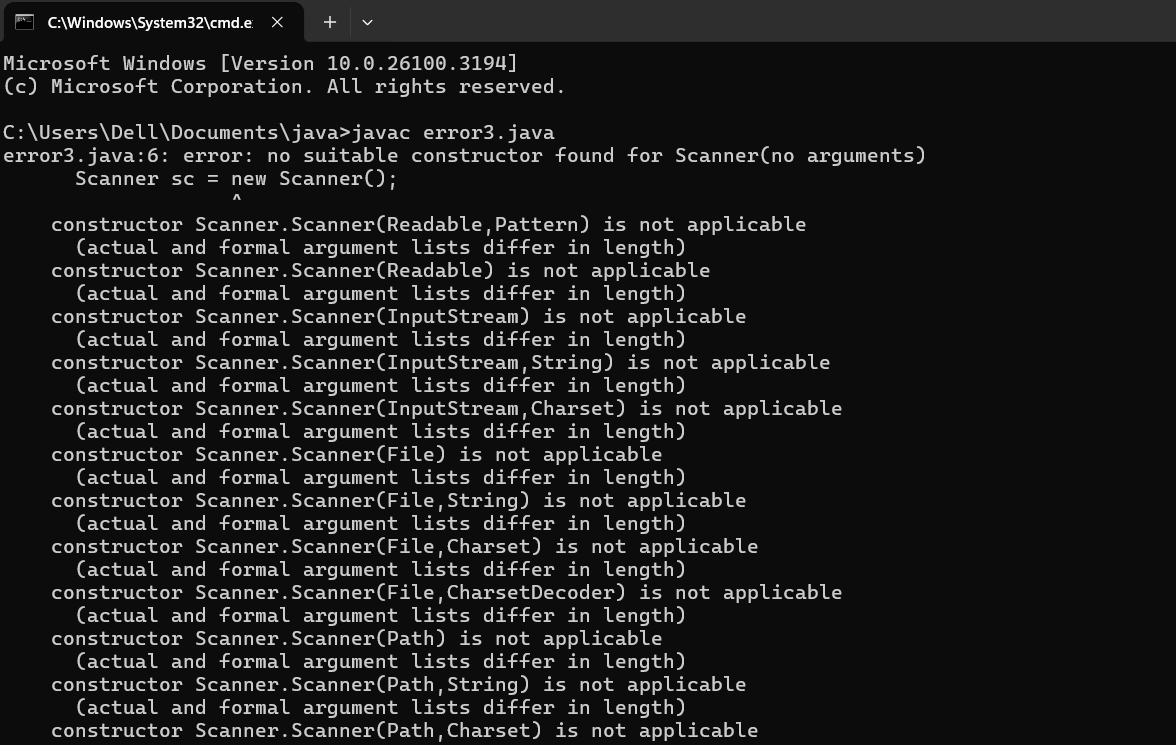
**ERRORS:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Error type** | **Reason for error** | **Rectification** |
| **1** | **Logical error** | **The variables are assigned as f1=f3**  **f2=f1** | **But the variables should be assigned as f1=f2 and f2=f3** |
| **2** | **Run-time error** | **The Scanner constructor requires input like System.in inside the parenthesis , but It is missing** |  |

**NEGATIVE ERRORS:**

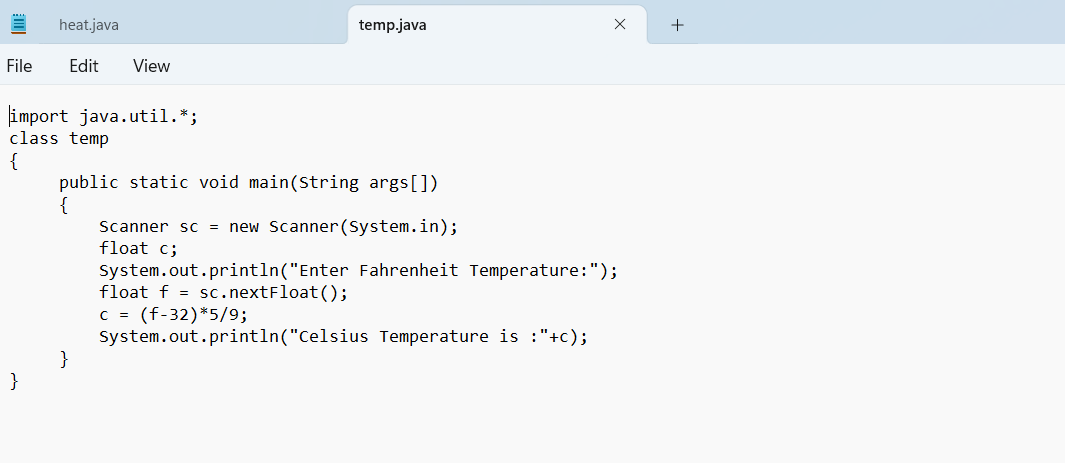
**I)**

****

**II)**

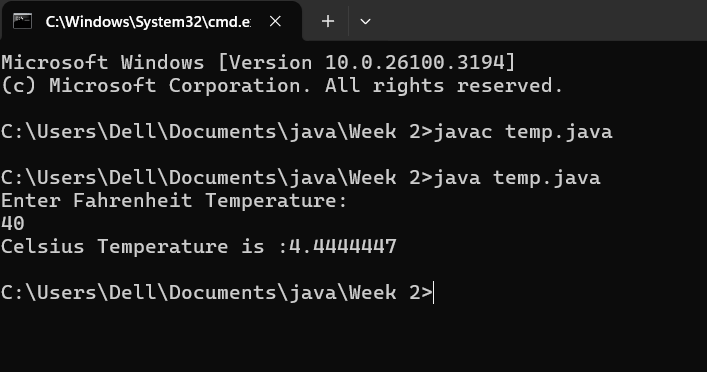
**4.Write a java program to convert temperature from Fahrenheit to Celsius**

**CODE:**

****

**Output:**

**POSTIVE CASE:**

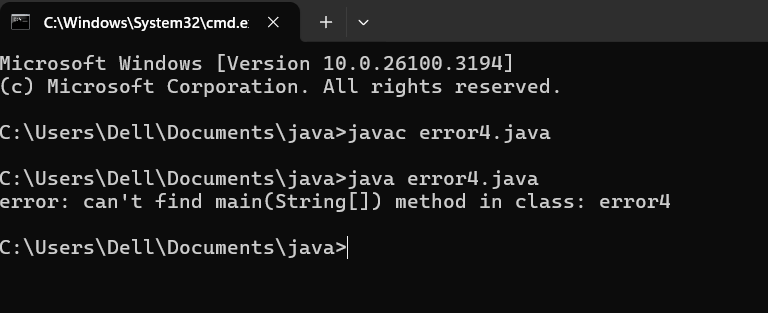
****

**ERRORS:**

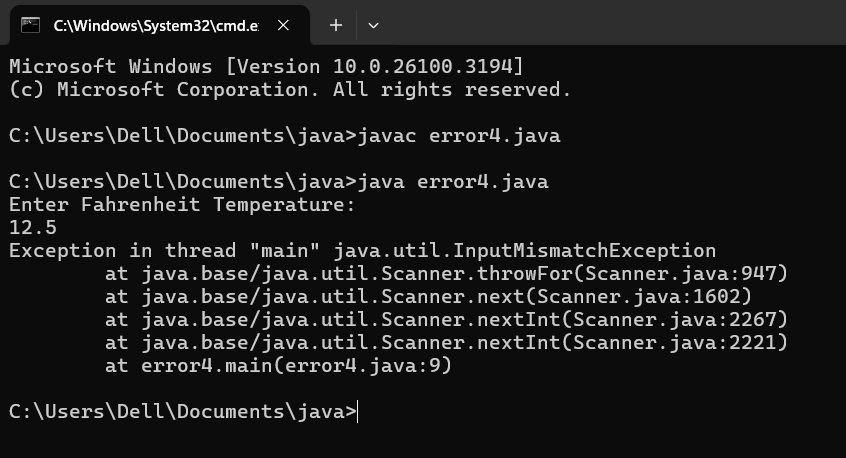
|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Error type** | **Reason for error** | **rectification** |
| **1** | **Syntax error** | **The argument should be String args[] instead of String arg.** | **The code should be**  **public static void main(String args[])** |
| **2** | **Incorrect Input Method for float** | **nextInt() is used instead of nextFloat()** | **float f = sc.nextFloat(); should be typed to rectify the error** |
| **3** | **Logical error** | **The formula for conversion of Fahrenheit to Celsius is written as c=(f-32)\*5//9** | **This error could be rectified by using the right formula: c=(f-32)\*5/9** |

**NEGATIVE CASE:**

**I)**

****

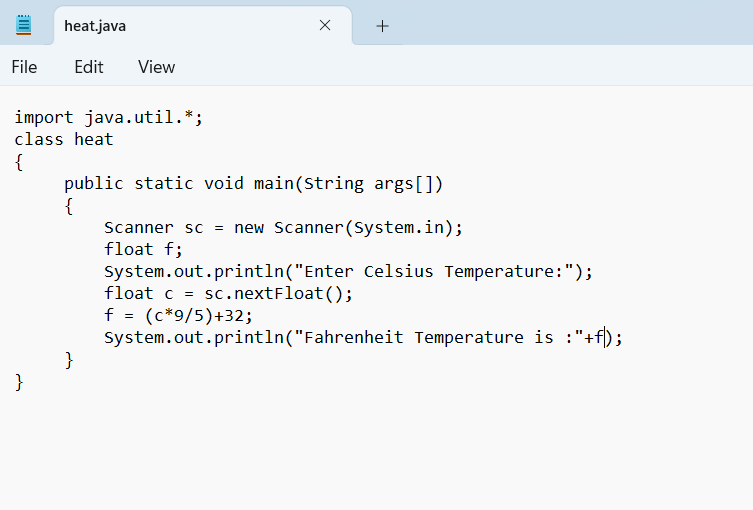
**II)**

****

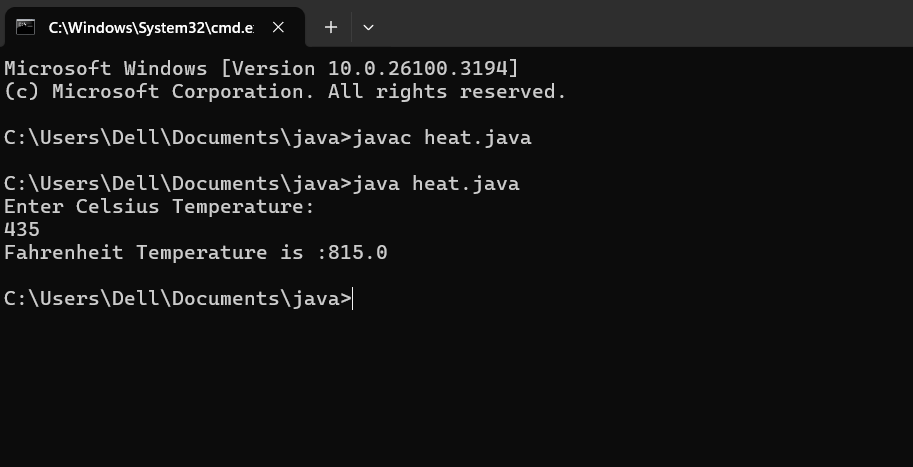
**III)**

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

**Code**

****

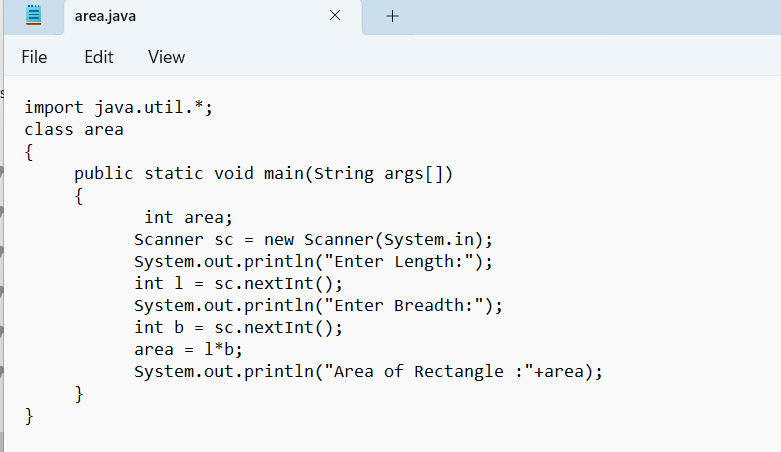
**Output:**

****

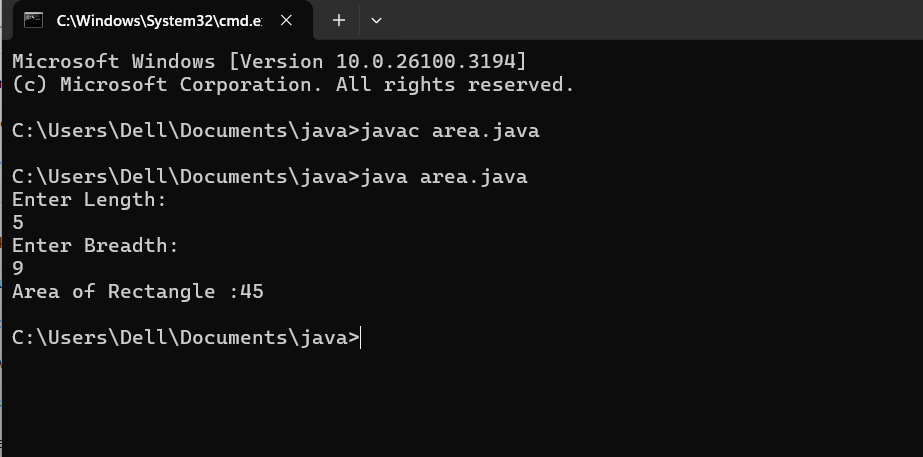
|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Error type** | **Reason for error** | **rectification** |
| **1** | **Syntax error** | **Missing ”** | **“ is added** |
| **2** | **Missing import error** | **Util package missing** | **Util package added** |
| **3** |  |  |  |

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

**Code:**

****

**Output:**

****

|  |  |  |  |
| --- | --- | --- | --- |
| **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 komal\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";**

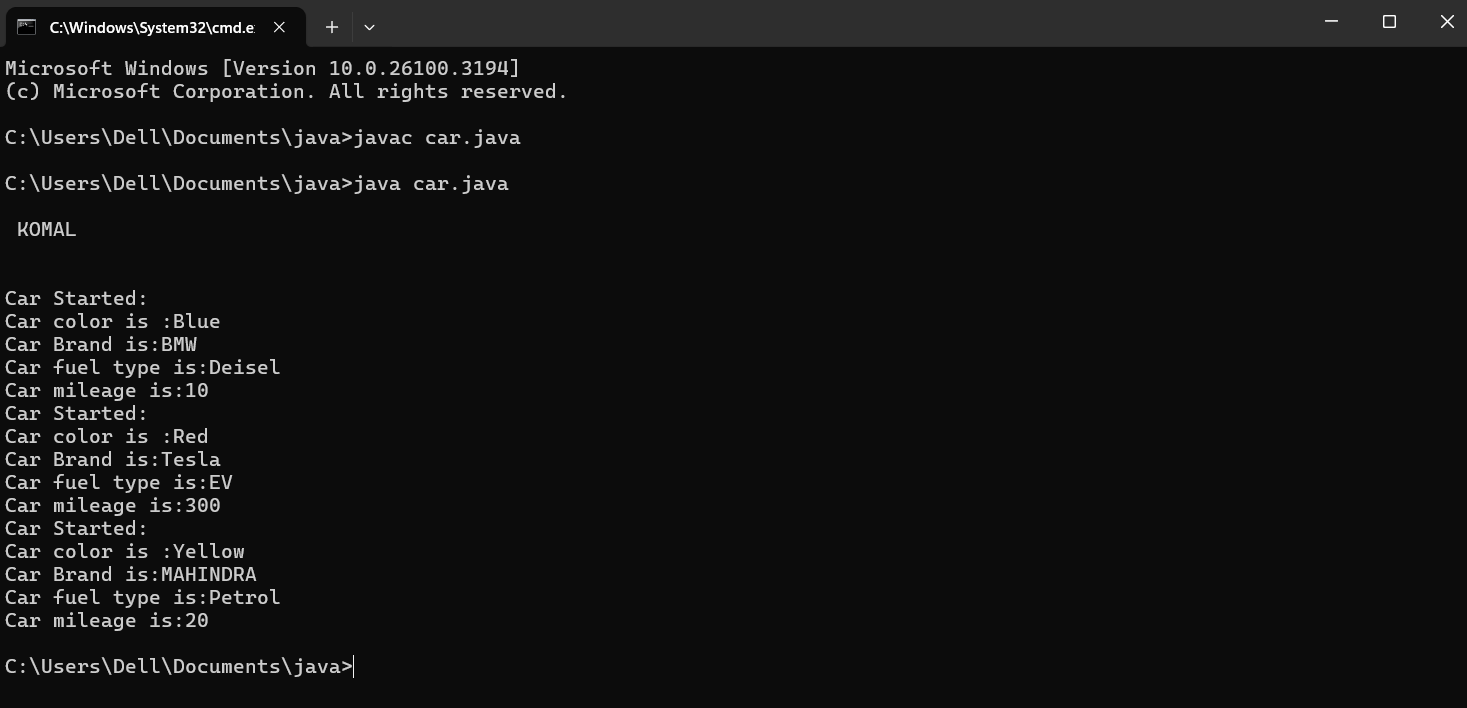
**car3.mileage = 20;**

**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. 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);**

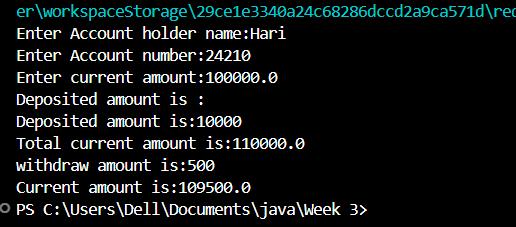
**account.deposit(500);**

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

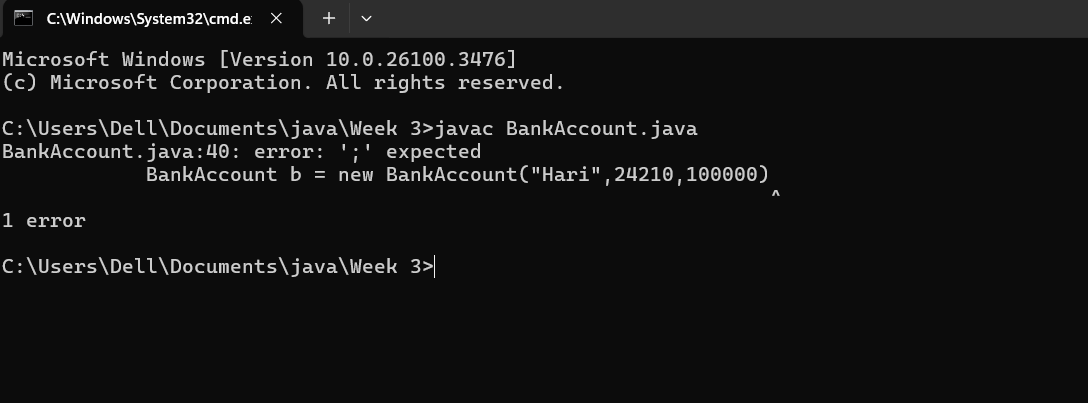
**}**

**}**

Output:



Negative Case:



**Important points:**

**Constructor: The Constructor creates and initializes objects of a class. They are called**

**when an object is created to a class.**

**This Keyword: The This keyword refers to the current instance of a class.It is used to**

**Access class variables and methods.**

**ERRORS:**

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

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);**

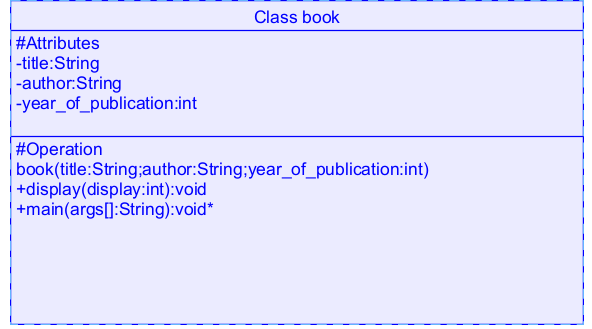
**b1.displayDetails();**

**b2.displayDetails();**

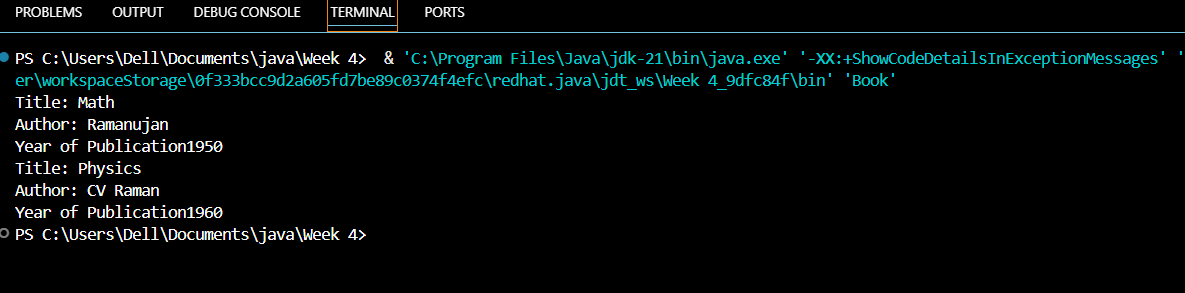
**}**

**}**

**Class Diagram:**

****

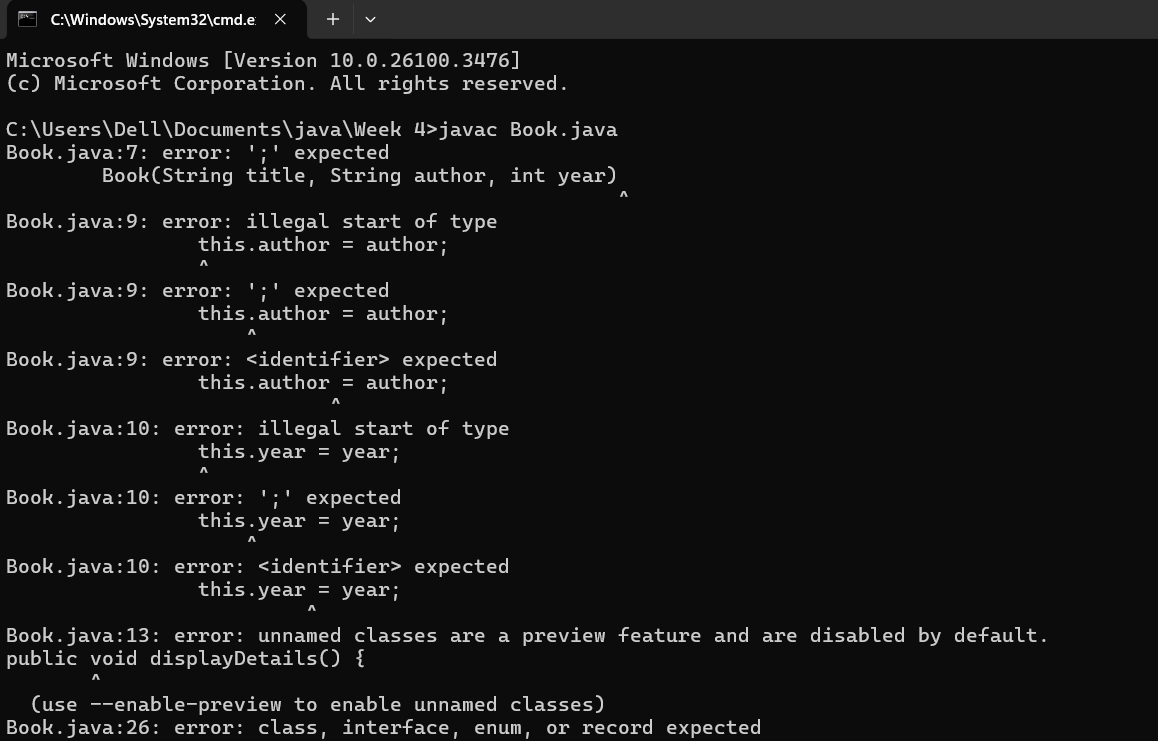
OUTPUT:



**Error Table:**

|  |  |  |  |
| --- | --- | --- | --- |
| **s.no** | **Error name** | **Cause of error** | **Rectification** |
| **1** | **Name Error** | **Undefined name** | **Correct variable**  **Name replaced** |
| **2** | **Syntax Error** | **Missing Parenthesis** | **Parenthesis Added** |
| **3** | **Logical Error** | **Incorrect Condition** | **Condition Rectified** |

**Negative Case:**

****

**Important points:**

**Constructor: The Constructor creates and initializes objects of a class. They are called**

**when an object is created to a class.**

**This Keyword: The This keyword refers to the current instance of a class.It is used to**

**Access class variables and methods.**

**2.Create a java program with class named “myclass” with a static variable**

**“count” of int type, initialized to zero and a constant variable “pi” of type**

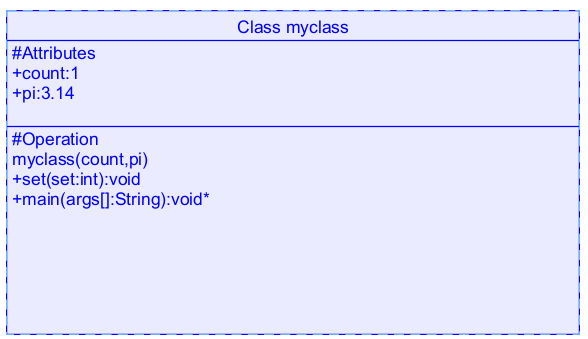
**Double initialized to 3.14 as attributes of the class. Now define a constructor**

**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. Create three objects**

**Class Diagram:**



**Code:**

**class myclass**

**{**

**static int count=0;**

**static double pi=3.14;**

**myclass()**

**{**

**count = count+1;**

**}**

**public void set()**

**{**

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

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

**}**

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

**{**

**myclass m = new myclass();**

**m.set();**

**myclass m1 = new myclass();**

**m1.set();**

**myclass m2 = new myclass();**

**m2.set();**

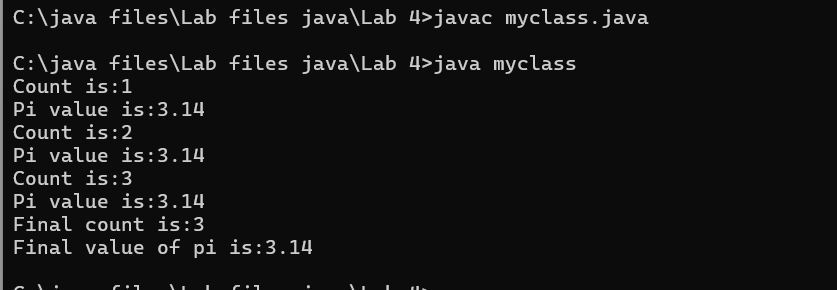
**System.out.println("Final count is:"+count);**

**System.out.println("Final value of pi is:"+pi);**

**}**

**}**

**Output:**

****

**ERROR:**

|  |  |  |  |
| --- | --- | --- | --- |
| **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’** |

Negative Case:



**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**

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

Multiplication and division using multilevel inheritance and display the desired

Output

**INPUT:-**

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));

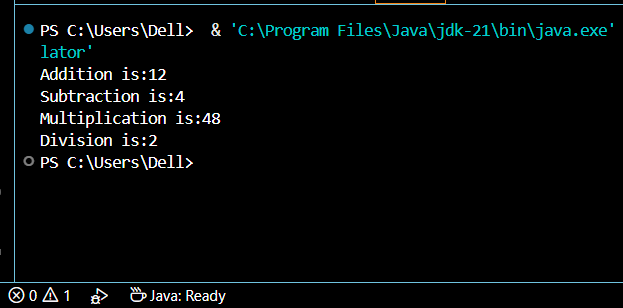
           System.out.println("Multiplication is:"+obj.mult(12,4));

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

    }

}

Output:



**CLASS DIAGRAM:-**

|  |
| --- |
| CLASS ADDITION |
| +add(int a, int b):int |

|  |
| --- |
| Class Subtraction |
| +sub(int a, int b):int |

|  |
| --- |
| Class Multiplication |
| +mult(int a, int b):int |

|  |
| --- |
| Class Division |
| +div(int a, int b):int |

**Error-table:-**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Error Type** | **Reason for error** | **Rectification** |
| **1.** | Constructive error | Invalid method name declared | Created class name |
| **2.** | Syntax error | Haven’t included ‘;’ | Added ‘;’ |

**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

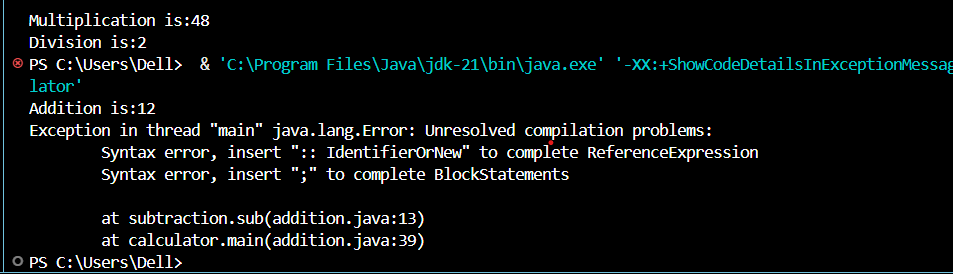
**Multilevel Inheritence:**

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

Inheritance

**ERROR:**

****

**PROGRAM-2**

A vehicle rental company wants to develop a system that maintains

Information about different types of vehicles available for rent

The Company rents out cars, 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.

**INPUT:-**

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);

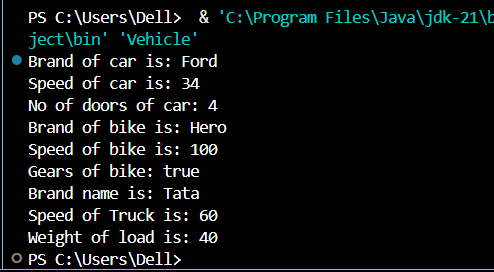
System.out.println("Speed of Truck is: " + speed);

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

}

}

OUTPUT:



**ERROR TABLE:-**

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

**Important Points**

**Hierarchical Inheritence:**

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

Single parent class

ERROR:

