**23CSE111**

**OBJECT ORIENTED PROGRAMMING**

**LAB REPORT**

****

**Department of Computer Science Engineering**

**Amrita School of Computing**

**Amrita Vishwa Vidyapeetham, Amaravati Campus**

**Name: NANDINI RAVULA**

Roll No: **AV.SC.U4CSE24310**

Branch: CSE

Section: C [D]

**Verified By**

**INDEX**

|  |  |  |
| --- | --- | --- |
|  | EXPERIMENTS | REMARKS |
| WEEK-1 |  |  |
| 1) | downloading and installation of java |  |
| 2) | write a java program that prints name, roll no, section of a student. |  |
| WEEK-2 |  |  |
| 1) | write a java program to find the simple interest where all the inputs are taken from the user |  |
| 2) | write a java program to find the fibonacci sequence of a given number |  |
| 3) | write a java program to find the area of rectangle |  |
| 4) | write a java program to find the area of triangle |  |
| 5) | write a java code to convert the temperature from celsius to fahrenheit and from fahrenheit to celsius. |  |
| 6) | write a java code to convert the temperature from celsius to fahrenheit and from fahrenheit to celsius. |  |
| WEEK-3 |  |  |
| 1) | 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 methods named car1, car2, car3. |  |
| 2) | to create a class bankaccount with methods deposit() and withdrawl |  |
| WEEK-4 |  |  |
| 1) | write a java program with class named “book”. the class should contain various attributes such as “title of the book”, “author”, year of publications”, it should also contain a constructor with parameter which initializes “title of the book”, “author”, “year of publication”. create a method which displays the details of the book. title of the book (), author (), year of publication ().display the details of two book, by creating 2 objects |  |
| 2) | to create a java program with class named “myclass” with a static variable count of “int” type, in sized to “zero” and a constant variable “pi” of type “double” initialized to 3.1415 as attributes of that class. define a contractor for “myclass” is created finally print the final values “count” and “pi” variables. |  |
| WEEK-5 |  |  |
| 1) | Create a calculator using the operations including addition, subtraction, multiplication, and division using multi-level inheritance and display the desired output.Hint: collect required variables using super class, Create each class for a parameter and each class must contain a method. |  |
| 2) | A vehicle rental company wants to develop a system that maintains information about different types of vechicles available for rent the company rents out cars and bikes, and they need a program to store details about each vehicle, such as brand and speed( should be in super class)  1. cars should have an additional property: no.of doors  2. Bikes should have a property indicating whether they have gears or not.  3. The system should also include a function to display details about each vehicle and indicate when a vehicle is starting.  4. Every class should have a constructor  Question:  1. Which oops concept is used in the above program  2. If the company decides to add a new type of vehicle, Truck, how would you modify the program?  a. Truck should include an additional property capacity (in tons)  b. Create a showTruckdetails() method to display the truck’s capacity.  c. Write a constructor for Truck that initializes all properties  3. Implement the truck class and update the main method to create a Truck object and also create an object for car and bike sub classes Finally, display the details |  |
| WEEK-6 |  |  |
| 1) | Write a java program to create a vehicle class with a method displayInfo().  Override this method in the car subclass to provide specific information a About car. |  |
| 2) | A college is developing an automated admissions systems 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 qualification.  1. UG admission require min of 60%  2. PG admission require min of 70% |  |
| 3) | To create a Java Program with class named “my class” with a Static Variable Count int type and initialize to 0 and A Constant Variable "pi" of type double initialized to 3.1415 has attributes of that class. Now defi a Constructor for my class that increments the Count Variable each time an object of my class is created. Finaly Print the final values of count. |  |
| 4) | Write a Java Program and create a Shape class with a method calcArea(). That is overloaded for different shapes like square and rectangle. Create a sub class circle that overrides the calcArea() for a circle. |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**WEEK-1**

**Program-1**

**AIM:** To download and install Java (JDK 21)

**PROCEDURE:**

This is the process for installation of JDK on windows.

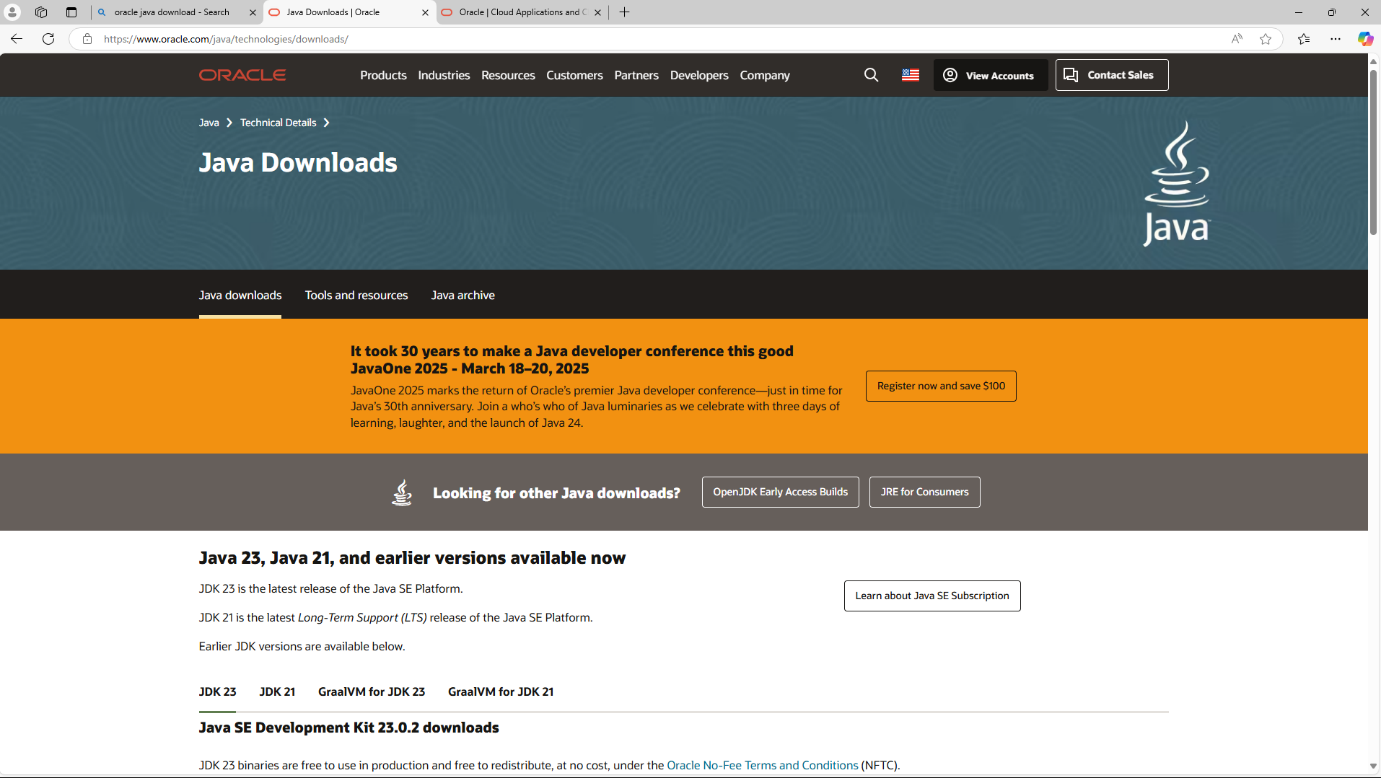
Follow the steps below to install Java on Windows:

* Download JDK (Java Development Kit)
* Run the Installer
* Configure Environment Variables
* Update the path variable
* Verify Installation in Command Prompt

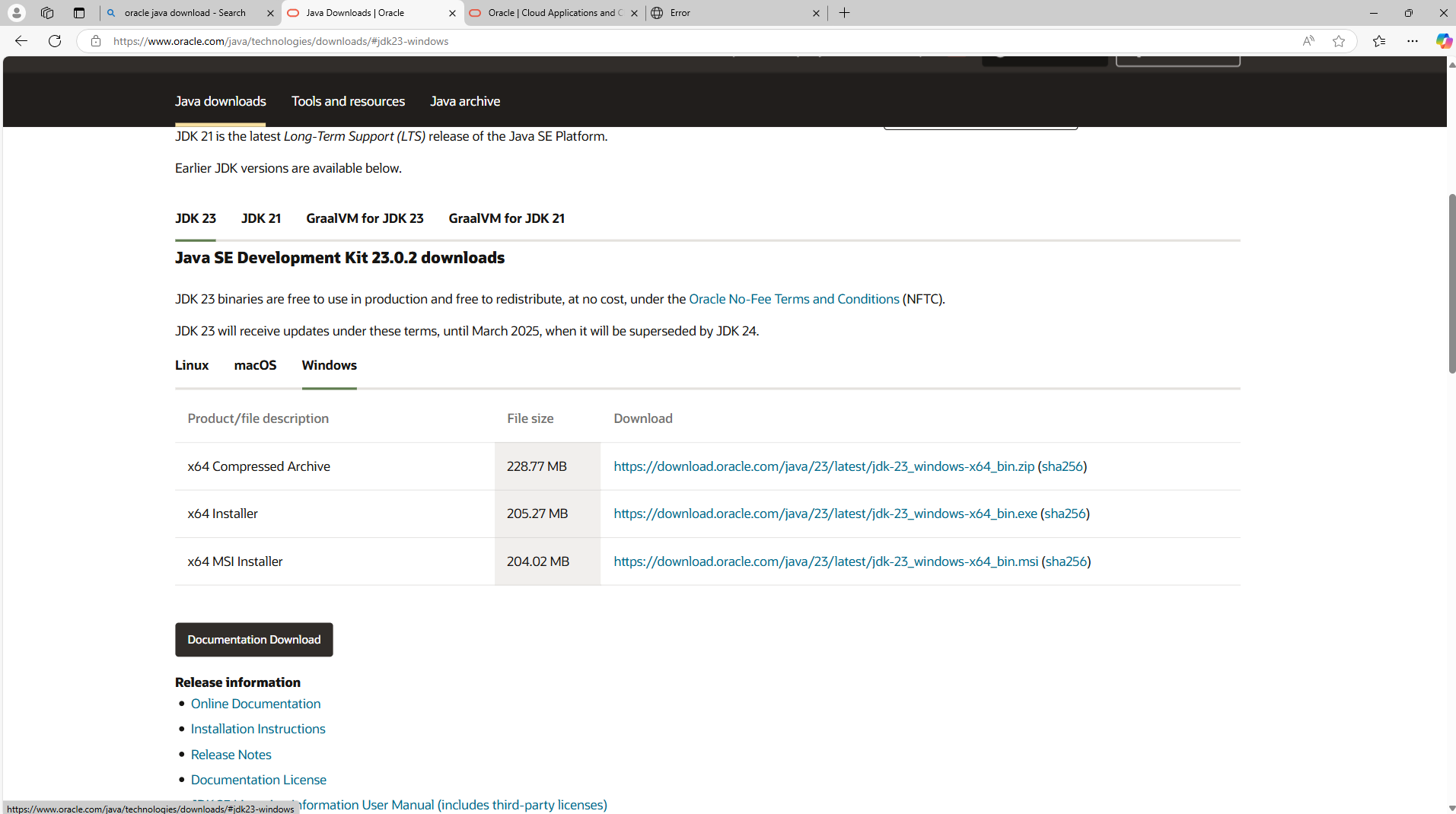
Here's a detailed explanation of each of the steps.

**Step 1**: Download JDK

1. Go to the official oracle website in the google search to download the JDK.
2. Locate the downloaded jdk-21\_windows-x64\_bin.exe file.
3. Double-click to launch the installer.
4. Click Next on the setup wizard.



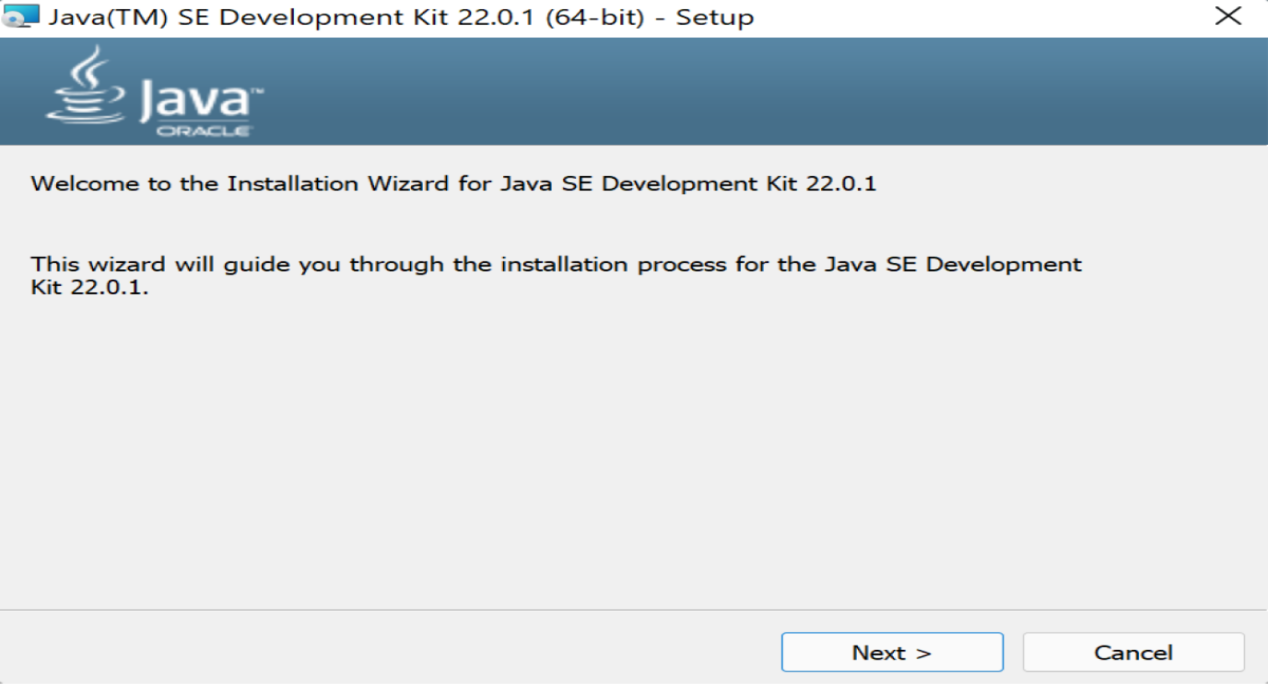
1. Choose the installation path (default is C:\Program Files\Java\jdk-21).
2. Click Next, then click Install.
3. Wait for the installation to complete.
4. Click Close once the installation is finished
5. Choose **x64 MSI Installer** on the windows tab and click on download link.



Step 2: Run the Installer

Now, go to your **downloads** folder and run the installer you just downloaded.

The screen below will be seen.



Simply click **Next** to proceed. Next you will be prompted another screen simply click next on that also.

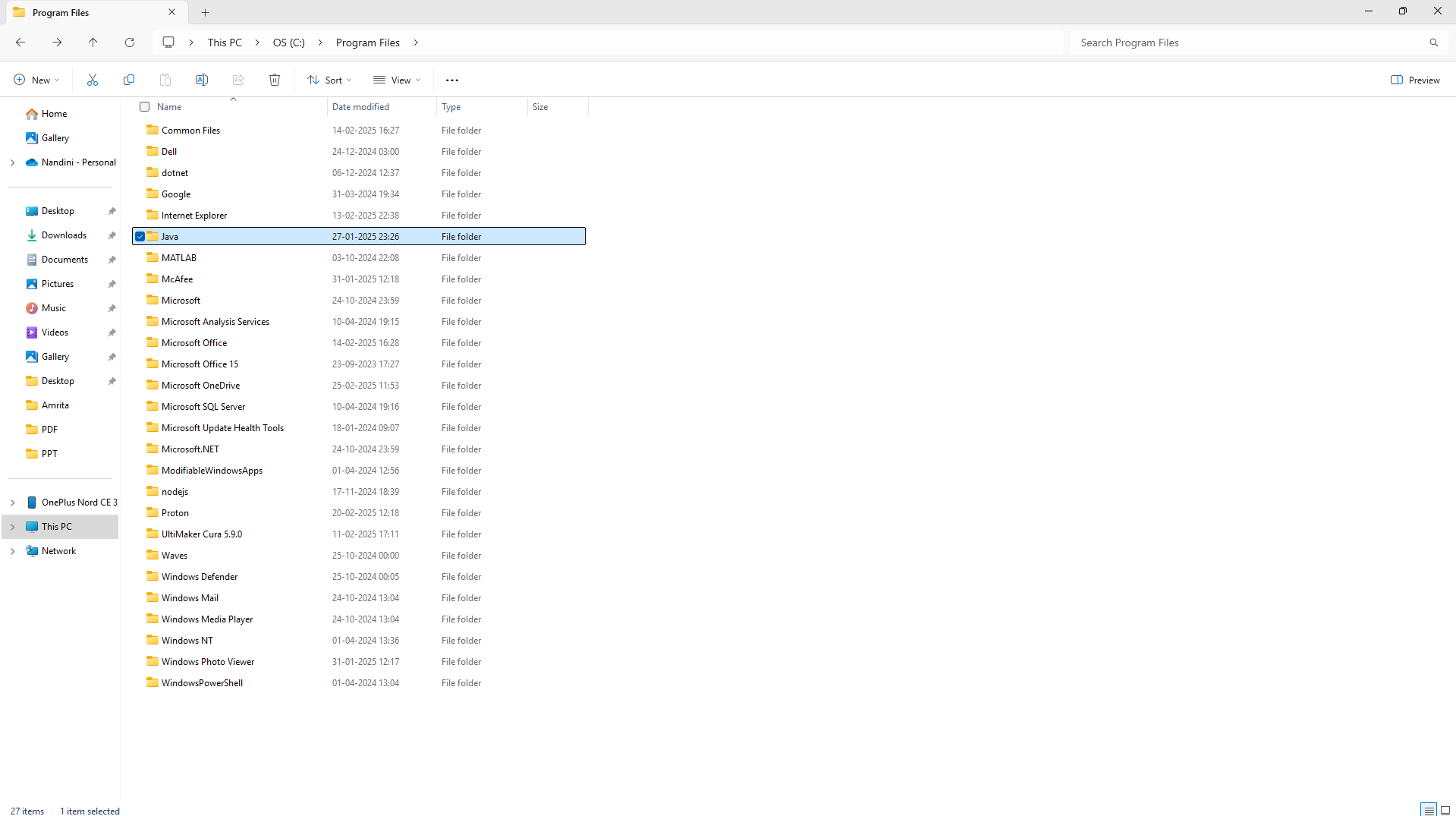
Step 3: Configure Environment Variables

After installation, you will need to tell your system where to find Java. This is done by setting environment variables.

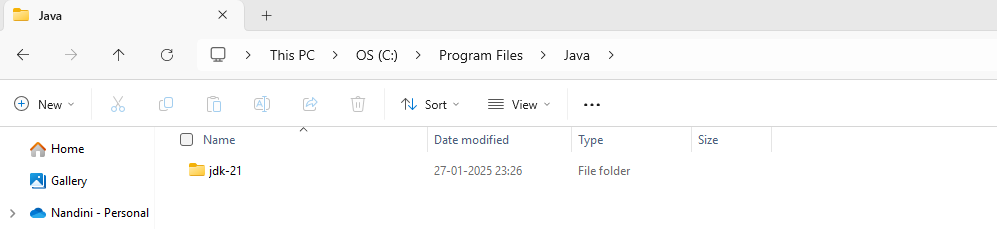
These are the ways to follow:

1)Go to file manager on your laptop or pc

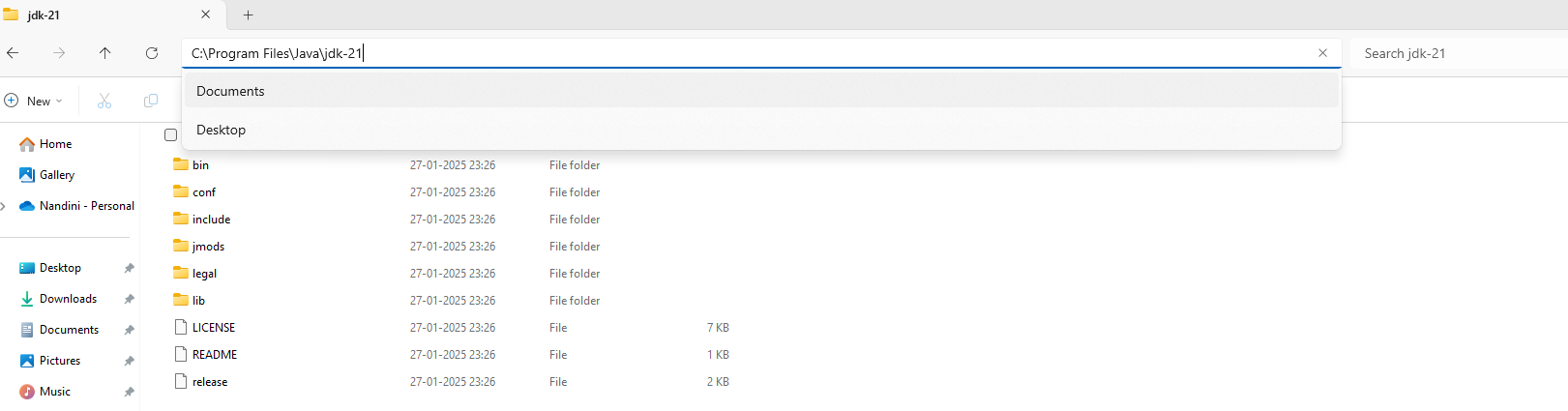
2) Go to “Windows C” Drive in File manager



3) Choose Program Files, select Java, then JDK 22, then select Bin.

4) Select and copy the path at the address bar 

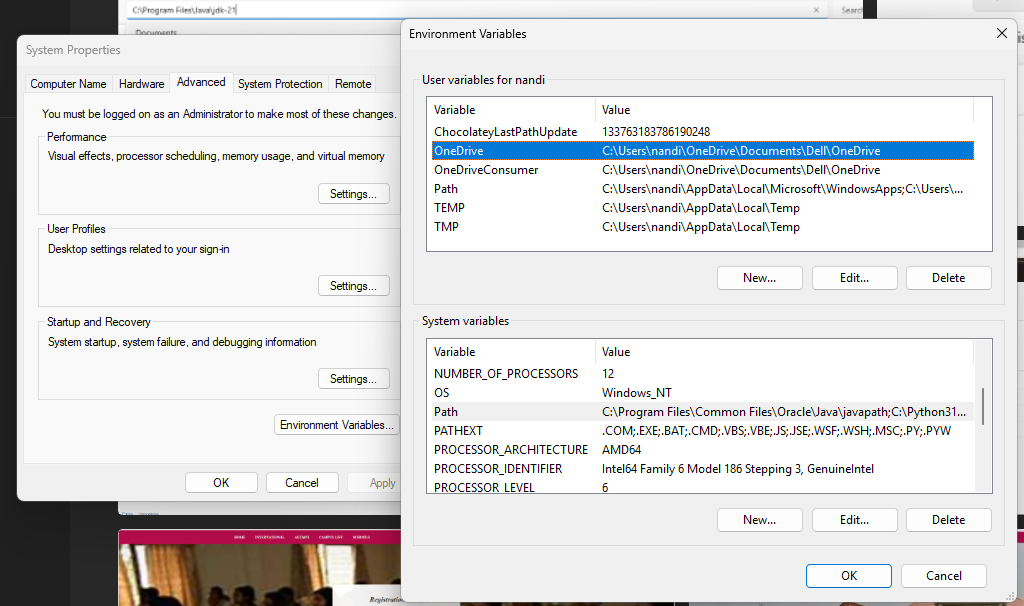
**Locate JDK Path**: Navigate through your file explorer to reach the JDK installation directory. Normally, it is located at

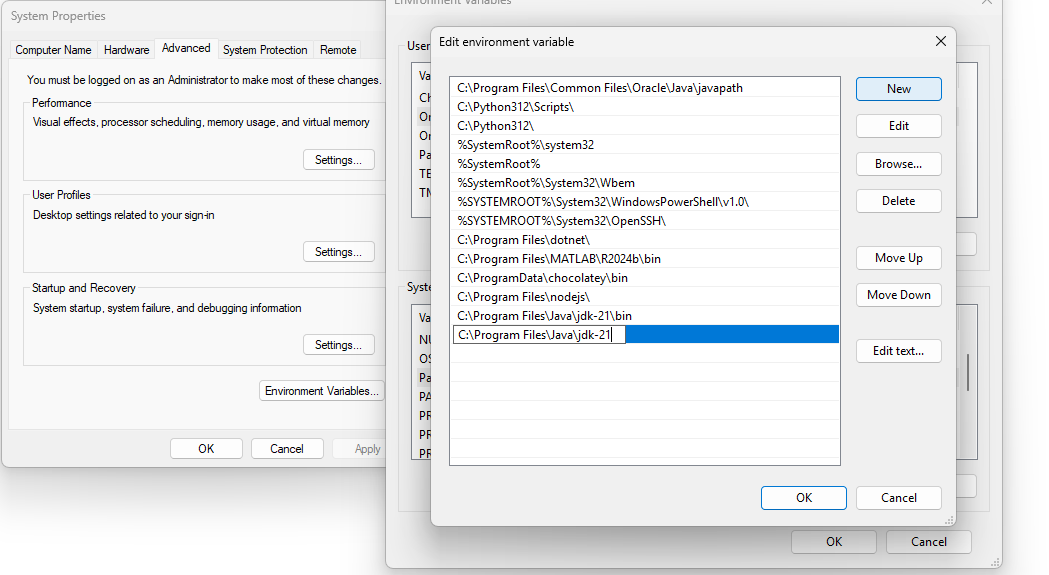


C:\Program Files\Java\jdk-22\bin

Copy this path

**Access Environment Variables**: Search **environment variable** on the terminal. In system properties, click on environment variables. You will be prompted to the screen below.





**Step 4: Update the Path Variable:**

Find the **Path** variable in the System variables section and click on **Edit**.

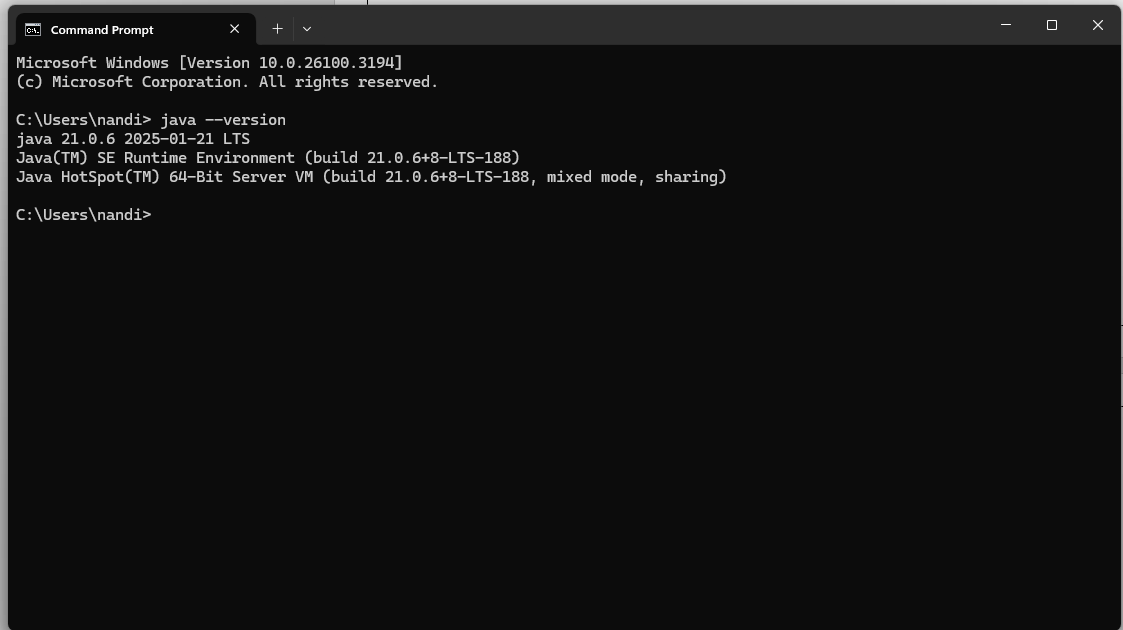
Then, click **New** and paste your JDK bin path (i.e. C:\Program Files\Java\jdk-22\bin).

Finally, click **Ok** to close each window.

**Step 5**: Verify your Installation

After the installation, you can verify whether Java is installed by using the following command in the command prompt.

java --version

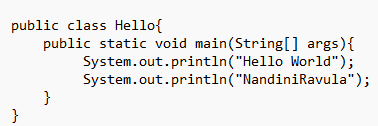


If Java is installed successfully, it will print the version information; otherwise, it will produce an error message indicating that the command is not recognized.

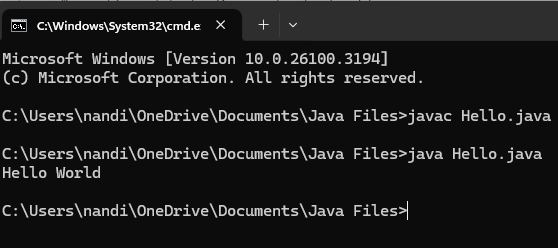
**PROGRAM-2:**

**AIM:** Write a Java program to print the message “Hello World”.

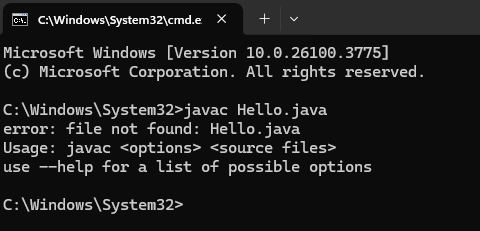
**Code**:



**POSITIVE OUTPUT**:



NEGATIVE OUTPUT;



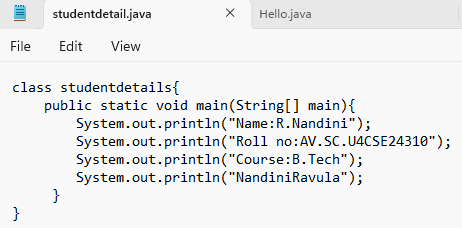
**ERROR TABLE:**

|  |  |
| --- | --- |
| ERRORS | RECTIFICATION |
| S in string is written in lowercase letter | The error is rectified by writing s in uppercase letter |

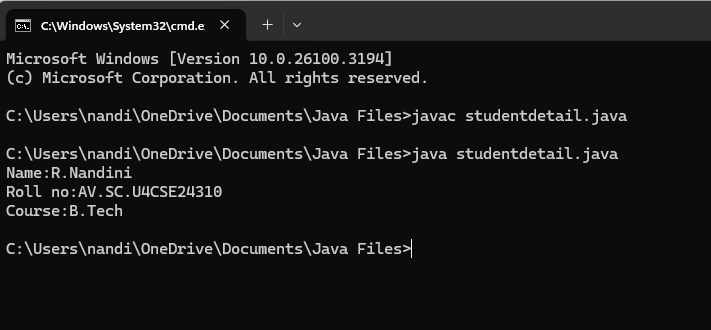
**PROGRAM-3:**

**AIM:** Write a Java Program that prints Name, Roll No,Course

**CODE:**

****

**OUTPUT:**



**ERROR TABLE:**

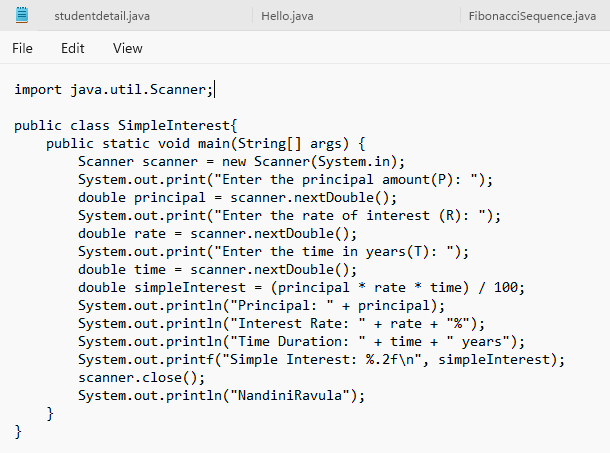
|  |  |
| --- | --- |
| ERROR | RECTIFICATION |
| In the statement at the end ; is not mentioned | Rectified by keeping ; at the end of the ststement |

**WEEK-2**

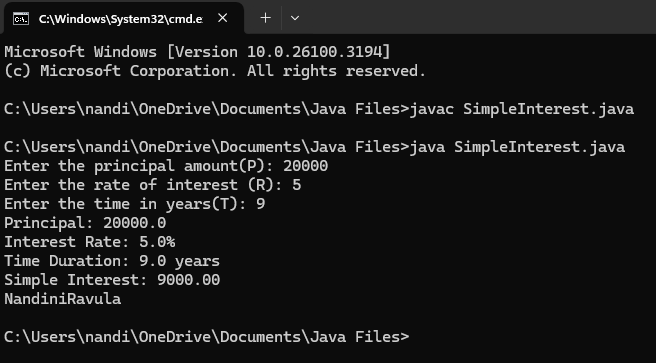
**Program-1**

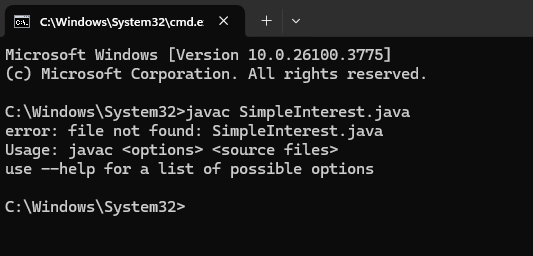
**AIM:** Write a java program to find the simple interest where all the inputs are taken from the user**.**

**CODE:**



**POSITIVE OUTPUT:**

****

**NEGATIVE OUTPUT;**

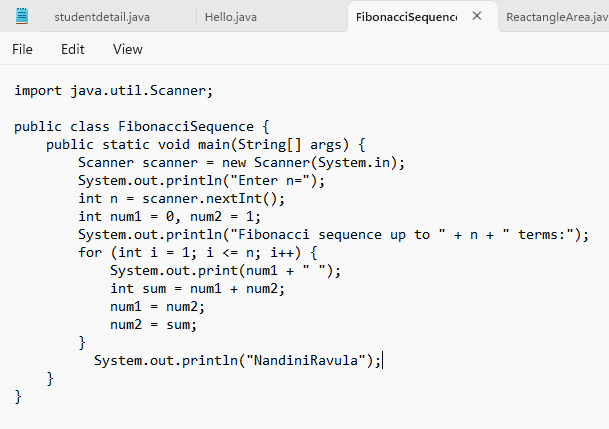
**ERROR TABLE:**

|  |  |  |
| --- | --- | --- |
| SI.NO | ERROR MESSAGE | ERROR RECTIFICATION |
| 1. | error: ';' expected          System.out.print("Enter the rate of interest (R) in percentage: ") | Insert: ‘;’          System.out.print("Enter the rate of interest (R) in percentage: "); |

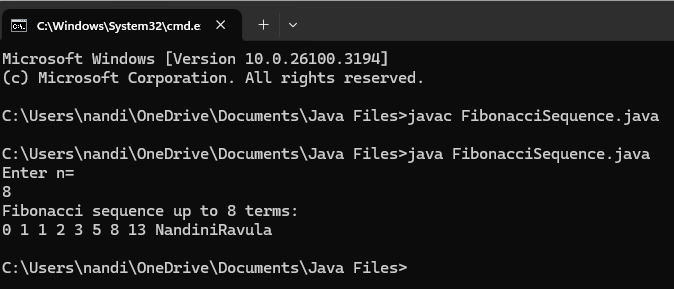
**PROGRAM-2:**

**AIM:** Write a java program to find the Fibonacci sequence of a given number

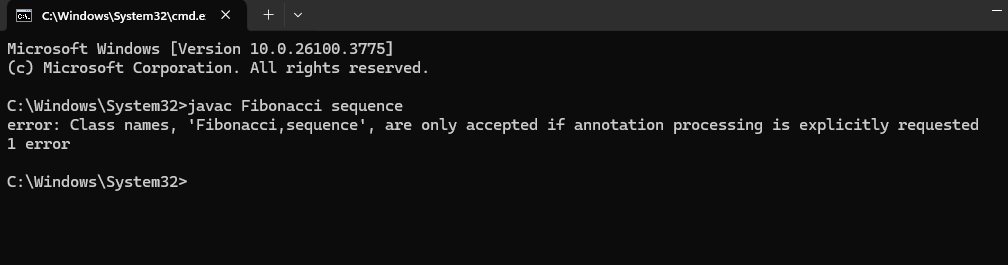
**CODE:**

****

**POSITIVE OUTPUT:**

****

**NEGATIVE OUTPUT:**

****

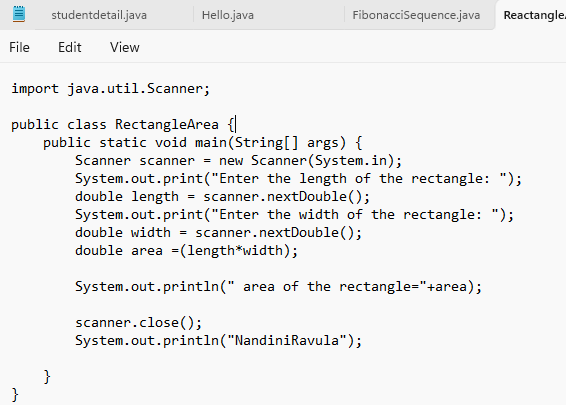
**ERROR TABLE:**

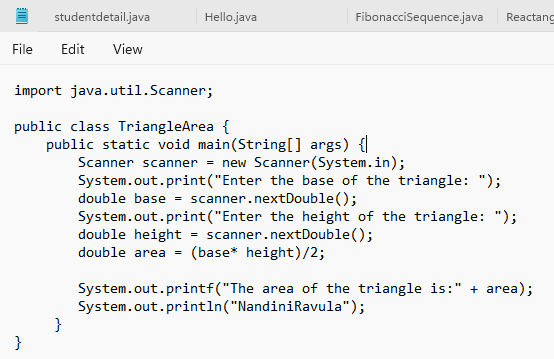
|  |  |  |
| --- | --- | --- |
| SI.NO | ERROR MESSAAGE | ERROR RECTIFICATION |
| 1. | Enter the number: 5  The factorial of 5 is 1  Error: factorial \*=1; | Replace i in 1 place   Replace:factorial \*=i; |

**Program-3:**

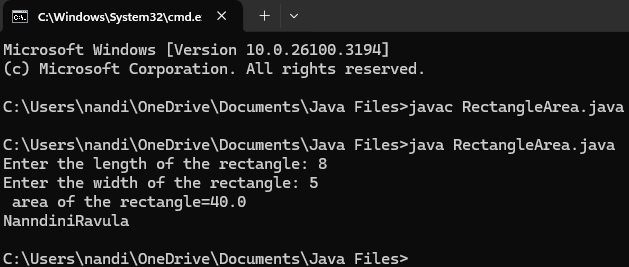
**AIM:** Write a java program to find the area of rectangle and triangle.

**CODE:**

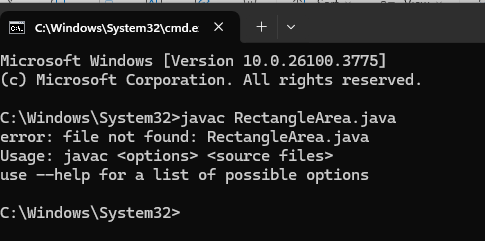
****

****

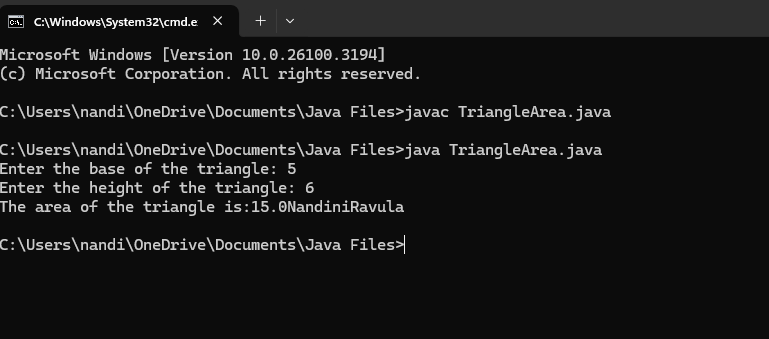
**POSITIVE OUTPUT:**

****

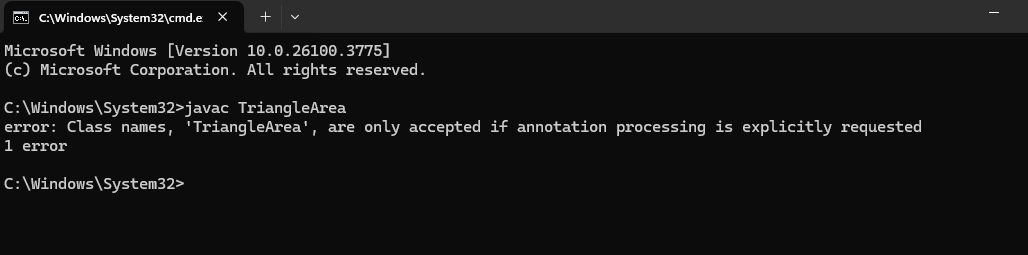
**NEGATIVE OUTPUT:**

****

**POSITIVE OUTPUT:**

****

**NEGATIVE OUTPUT:**

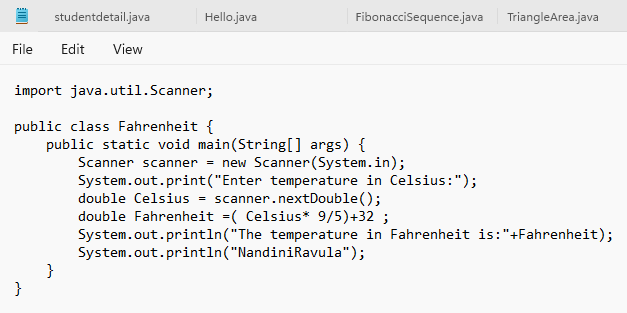
****

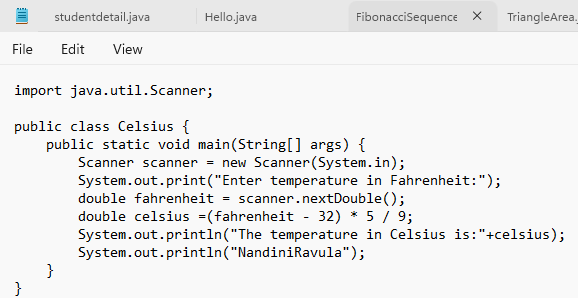
**ERROR TABLE:**

|  |  |  |
| --- | --- | --- |
| SI.NO | ERROR MESSAGE | ERROR RECTIFICATION |
| 1. | Error: float fahrenheit = celsius \* 9 / 5 + 32;  Reason:Formula mistake. | It should be ((celsius \* 9) / 5) + 32 |

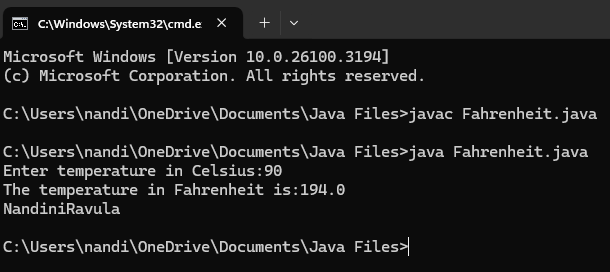
**PROGRAM-4:  
AIM:** Write a java code to convert the temperature from Celsius to Fahrenheit and from Fahrenheit to Celsius.

**CODE:**

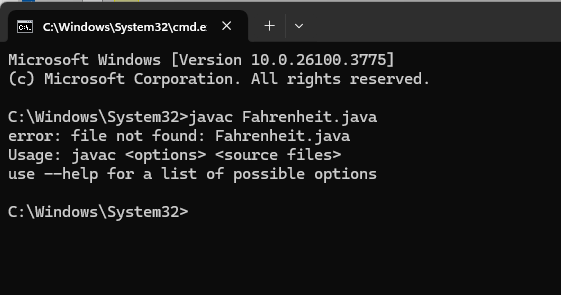
****

****

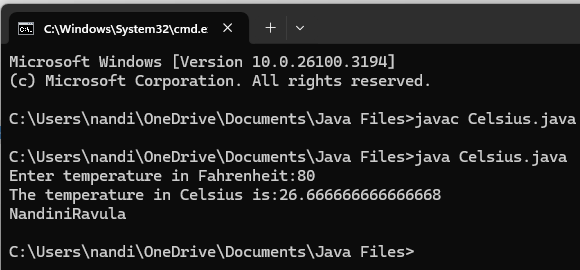
**POSITIVE OUTPUT:**

****

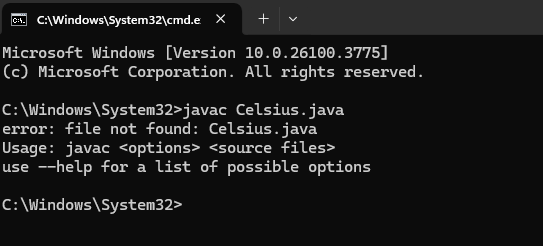
**NEGATIVE OUTPUT:**

****

**POSITIVE OUTPUT:**

****

**NEGATIVE OUTPUT:**

****

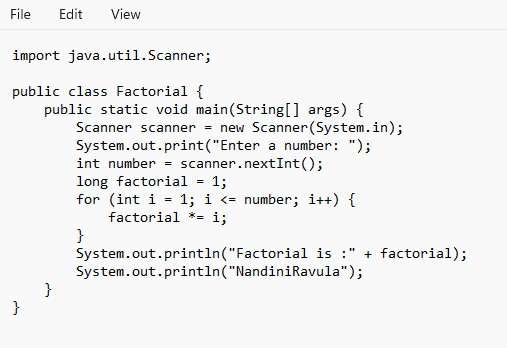
**ERROR TABLE:**

|  |  |  |
| --- | --- | --- |
| SI.NO | ERROR MESSAGE | ERROR RECTIFICATION |
| 1. | Error: Print statement with incorrect variable name System.out.println(fahrenheit + "°F is equal to " + Celsius + "°C") | 'Celsius' should be lowercase  It should be ‘celsius’. |

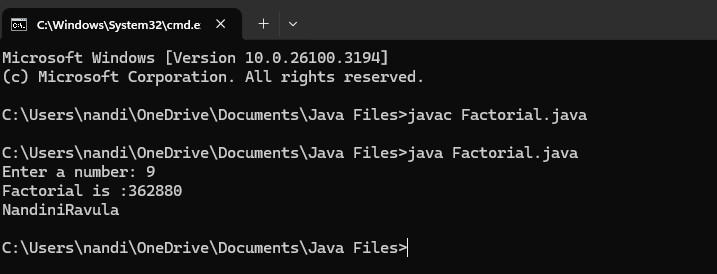
**PROGRAM-5**

**AIM:** Write a java code to find factorial of a number by taking input.

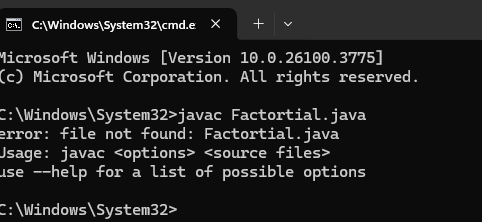
**CODE:**

****

**POSITIVE OUTPUT:**

****

**NEGATIVE OUTPUT:**

****

**ERROR TABLE:**

|  |  |  |
| --- | --- | --- |
| SI.NO | ERROR MESSAGE | ERROR RECTIFICATION |
| 1. | int firstTerm;  Error: variables not initialized properly | should be initialized          int n = 10; |

**WEEK-3**

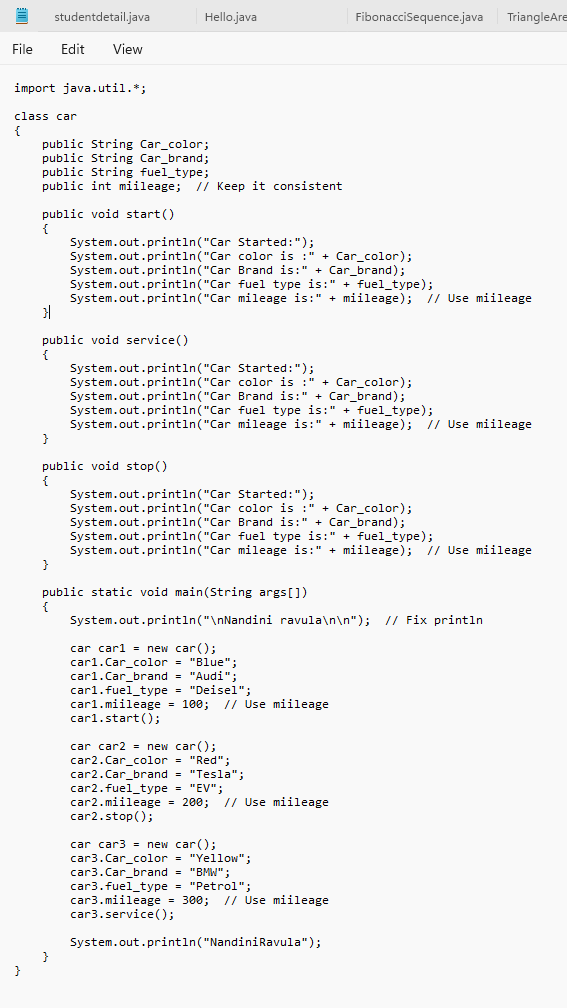
**Program-1**

**AIM:** To create java program with following instructions

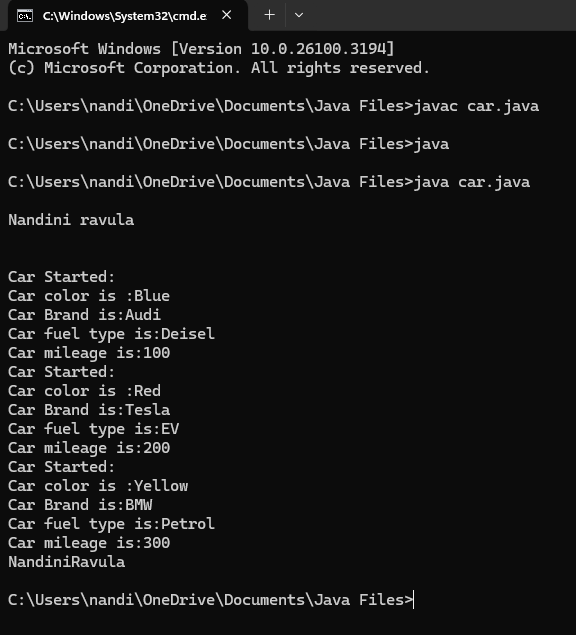
1. Create a class with name Car.
2. Create four attributes named Car color, Car brand, fuel type, mileage.
3. Create three methods named start (), stop (), service ().
4. Create three methods named Car1, Car2, Car3.
5. **CLASS DIAGRAM:**

|  |
| --- |
| CLASS CAR |
| +Car\_color: string  +Car\_brand: string  + fuel\_type:string  +mileage:int |
| +void start()  +void stop()  +void service() |

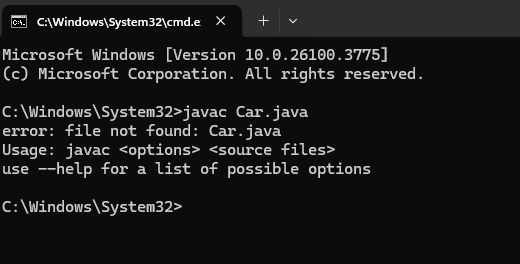
**CODE:**

****

**POSITIVE OUTPUT:**

****

**NEGATIVE OUTPUT:**

****

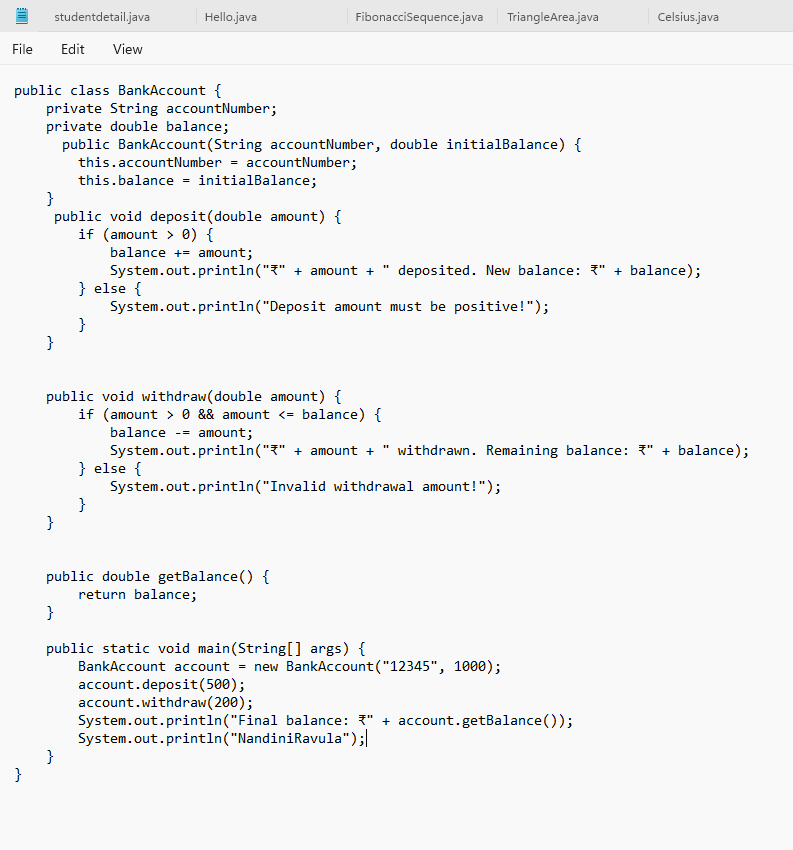
**ERROR TABLE:**

|  |  |  |
| --- | --- | --- |
| **SI.NO** | **ERROR MESSAGE** | **ERROR RECTIFICATION** |
| **1.** | **Error: car\_Color is undefined, should be car\_color** | **Replace: car\_Color with car\_color** |

**Program-2**

**Aim:** To create a class bank account with method deposit () and withdrawal ().

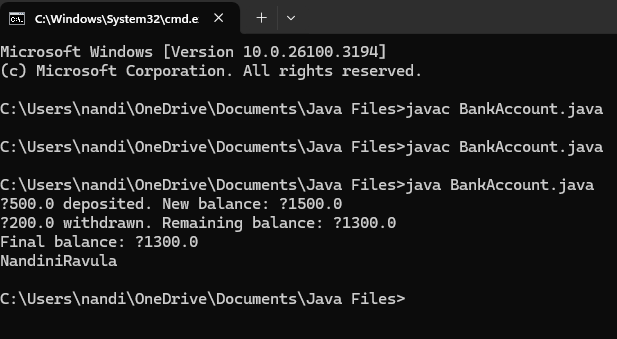
**CODE:**

****

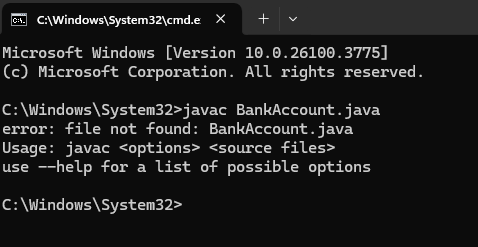
**CLASS DIAGRAM:**

|  |
| --- |
| CLASS BANKACCOUNT |
| -balance:double |
| +void deposit()  +void withdrawal() |

**POSITIVE OUTPUT:**

****

**NEGATIVE OUTPUT:**

****

**ERROR TABLE:**

|  |  |  |
| --- | --- | --- |
| SI.NO | ERROR MESSAGE | ERROR RECTIFICATION |
| 1. | this.existing = int.nextFloat(); | this.existing = input.nextFloat(); |

**WEEK-4**

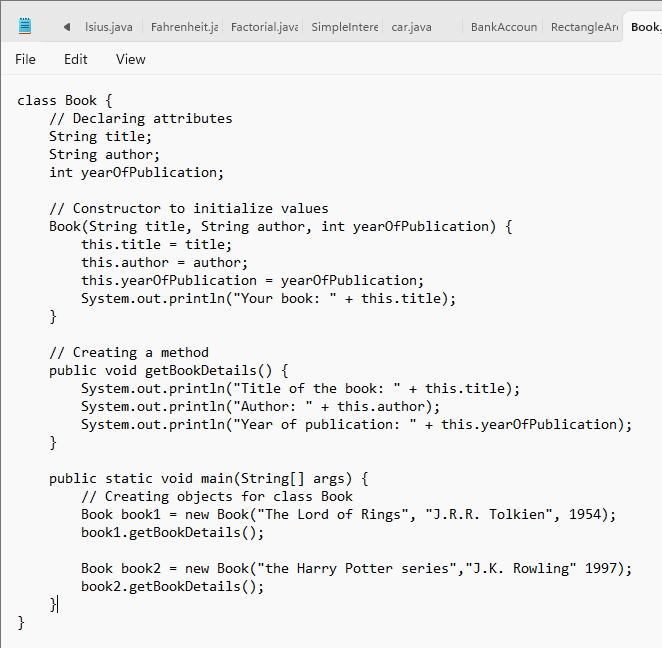
**Program-1**

**AIM:** Write a java program with class named “book”. The class should contain various attributes such as “Title of the book”, “Author”, Year of publications”, it should also contain a constructor with parameter which initializes “Title of the book”, “Author”, “Year of publication”. Create a method which displays the details of the book. Title of the book (), Author (), Year of publication ().Display the details of two book, by creating 2 objects.

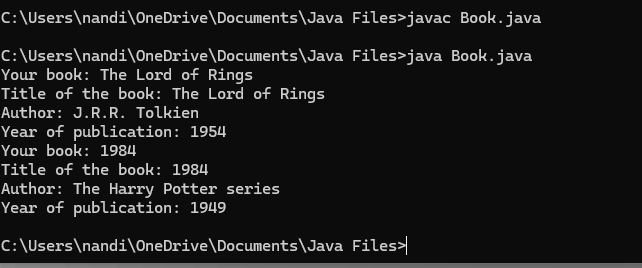
**CLASS DIAGRAM:**

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

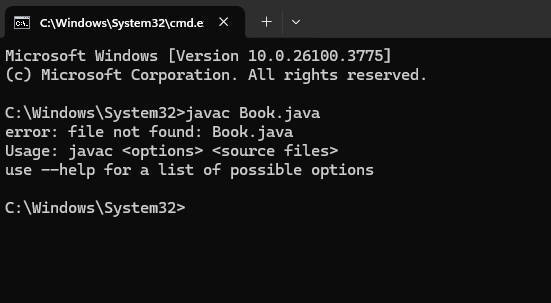
**CODE:**



**POSITIVE OUTPUT:**



NEGATIVE OUTPUT:



**Error:**

|  |  |  |
| --- | --- | --- |
| SI.NO | ERROR MESSAGE | ERROR RECTIFICATION |
| 1. | Not defining the function in a file. | To call the method we must define a function in a file. |

**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.

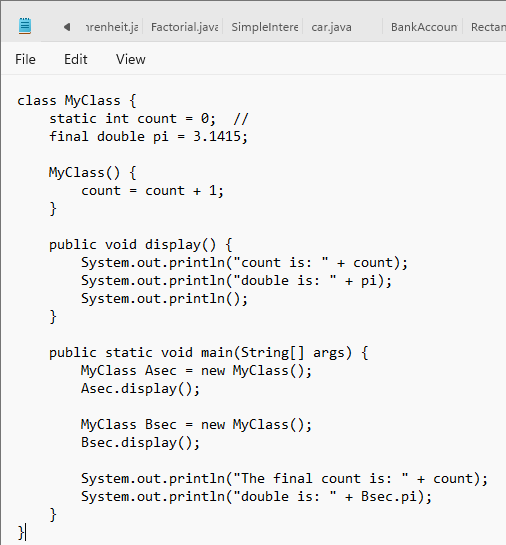
**Program-2**

**AIM:** To create a java program with class named “Myclass” with a static variable count of “int” type, in sized to “zero” and a constant variable “pi” of type “double” initialized to 3.1415 as attributes of that class. define a contractor for “Myclass” is created finally print the final values “count” and “pi” variables.

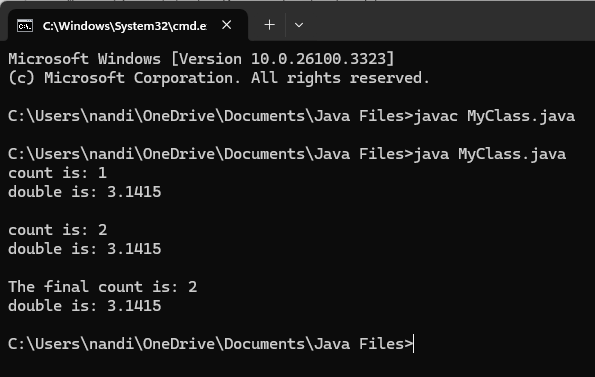
**CLASS DIAGRAM:**

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

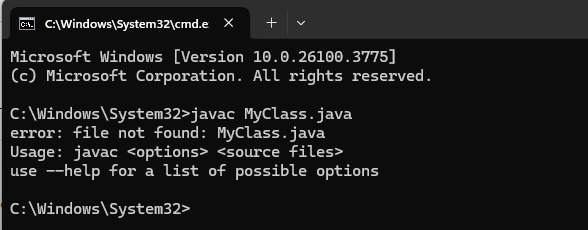
**CODE:**

****

**POSITIVE OUTPUT:**



NEGATIVE OUTPUT:



**Error:**

|  |  |  |
| --- | --- | --- |
| **SI.NO** | **ERROR MESSAGE** | **ERROR RECTIFICATION** |
| **1.** | Not Putting the semi-colon after calling a function, | 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.

**WEEK-5**

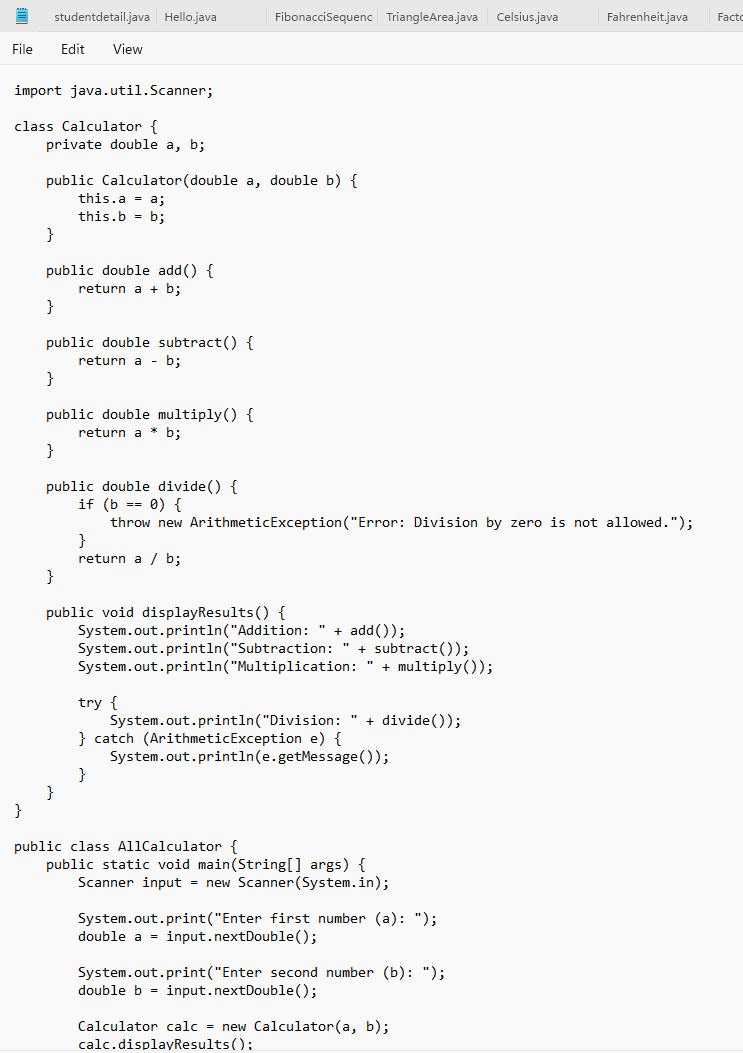
**Program-1**

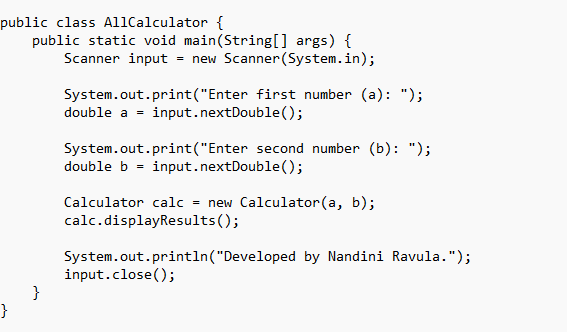
**AIM:: Create a calculator using the operations including addition, subtraction, multiplication and division using Multilevel Inheritance and display the desired output.**

**CLASS DIAGRAMS:**

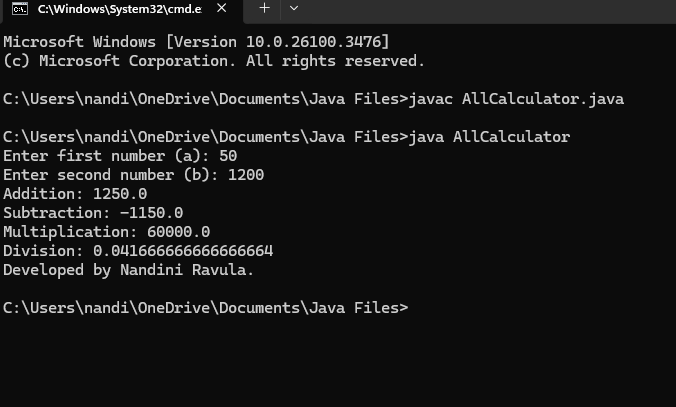
|  |
| --- |
| **Calculator** |
| **- a: double**  **- b: double** |
| **+ add(): double**  **+ subtract(): double**  **+ multiply(): double**  **+ divide(): double**  **+ displayResults(): void** |

**CODE:**

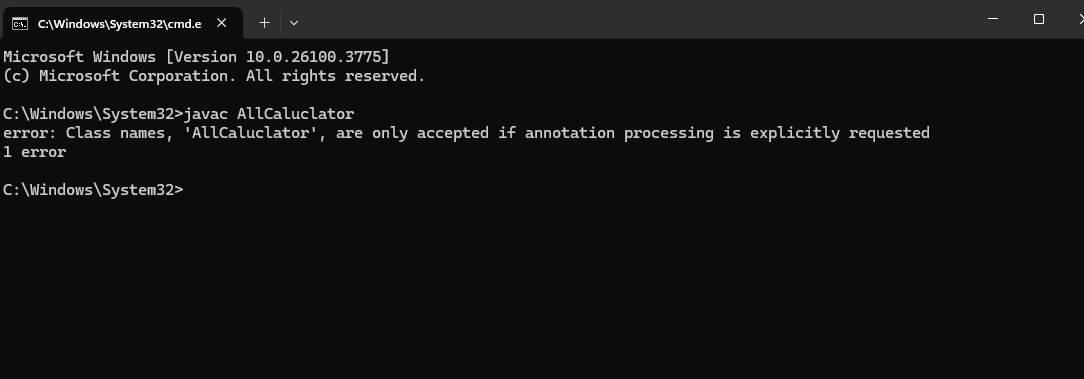




**POSITIVE OUTPUT:**



**NEGATIVE OUTPUT:**

****

Important Points:

Demonstrates **inheritance** with a single Calculator class

### **Error Table:**

| **Error** | **Possible Cause** | **Solution** |
| --- | --- | --- |
| **Main method not found** | Class name does not match file name (Rent instead of rent) | Ensure class name and file name match (Rent.java) |

**Program-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 and bikes 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**

**• Bike 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**

**• Every class should have a constructor**

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

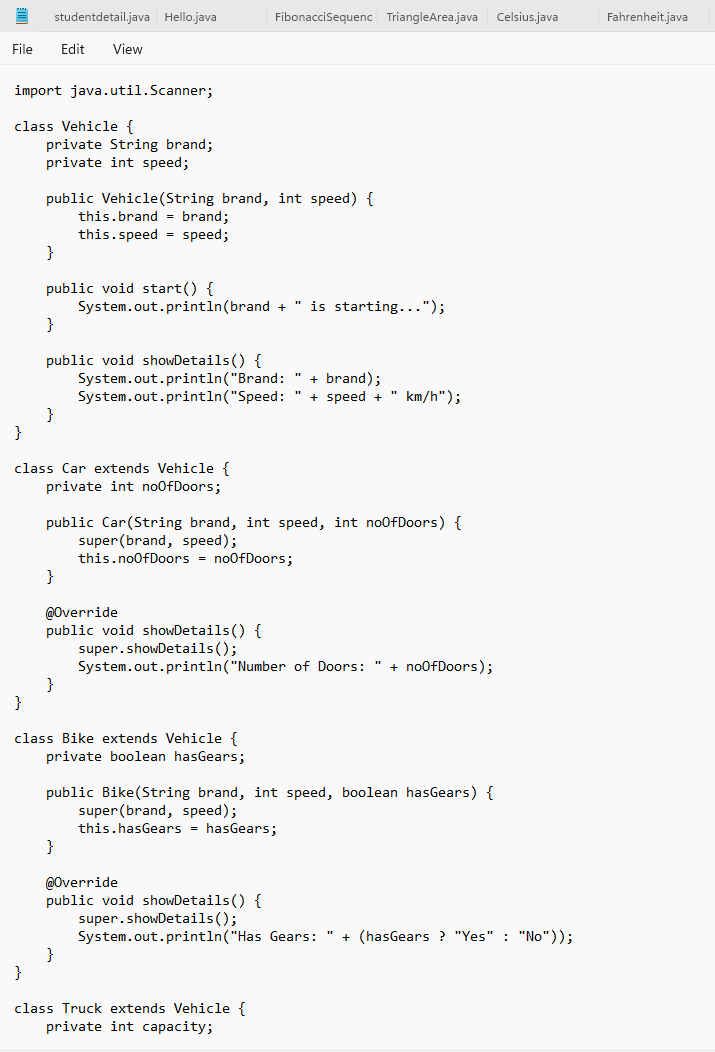
**2. If the company decides to add a new type of vehicle truck, how would you modify the program? Truck should include an additional property capacity(in tons). Create a showTruckDetails() method to display the truck’s capacity. Write a constructor for truck that initializes all properties.**

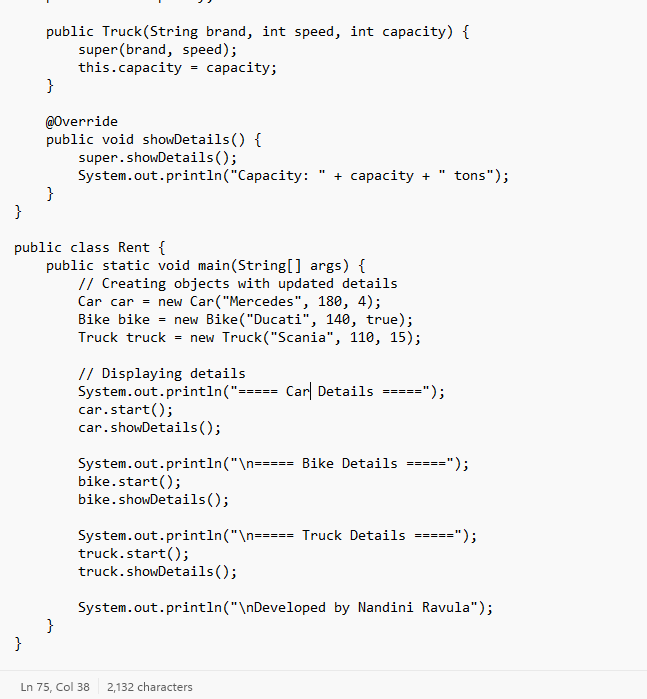
**3. Implement the truck class and update the main method to create a truck object, also create an object for car and bike subclassed. Finally display its details.**

**CLASS DIAGRAMS:**

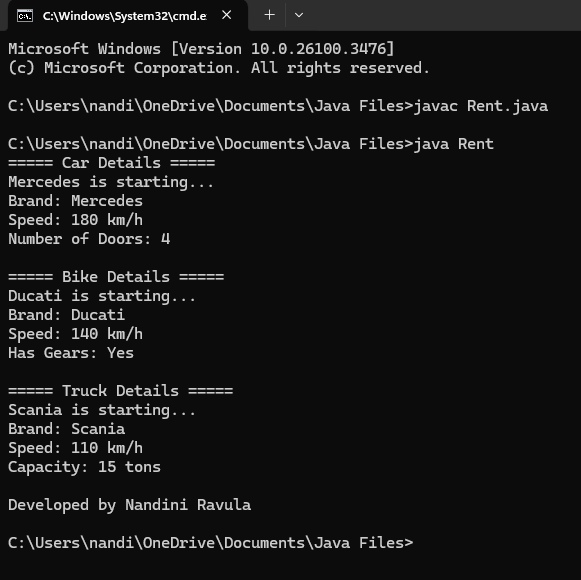
|  |  |  |
| --- | --- | --- |
| **Vehicle** | | |
| **- brand: String**  **- speed: int** | | |
| **+ start(): void**  **+ showDetails(): void** | | |
| **Car** | **Bike** | **Truck** |
| **- noOfDoors: int** | **- hasGears: boolean** | **- capacity: int** |
| **+ showDetails(): void** | **+ showDetails(): void** | **+ showDetails(): void** |

**CODE:**

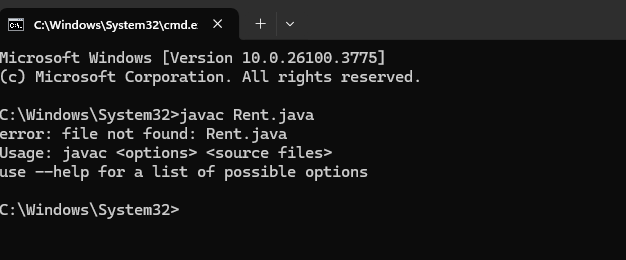




**POSITIVE OUTPUT:**



**NEGATIVE OUTPUT:**

****

Important Points:

Demonstrates **inheritance** (Car, Bike, Truck inherit from Vehicle).

Uses **method overriding** (showDetails() method in each subclass).

### **Error Table:**

| **Error** | **Possible Cause** | **Solution** |
| --- | --- | --- |
| **Method not found error** | showTruck() used in Truck, but other classes use showDetails() | Rename showTruck() to showDetails() in Truck |

**WEEK-6**

**Program-1-** **Write java program to create a vechical class with a method dispaly info().override this method in the car subclass to provide specific information about the car.**

**Class Diagram:**

|  |
| --- |
| **Vehicle** |
| + brand : String  + petrolType : String  + color : String |
| + display() : void |

|  |
| --- |
| **Car** |
| Car(String brand, String petrolType, String color) |

**CODE**- // Base class

class Vehicle {

String name;

int wheels;

// Constructor

Vehicle(String name, int wheels) {

this.name = name;

this.wheels = wheels;

}

// Method to display general vehicle information

void displayInfo() {

System.out.println("This is a vehicle named: " + name + " with " + wheels + " wheels.");

}

}

// Subclass

class Car extends Vehicle {

// Constructor for Car

Car(String name, int wheels) {

super(name, wheels); // call superclass constructor

}

// Overriding the displayInfo() method

@Override

void displayInfo() {

System.out.println("This is a car named: " + name + " with " + wheels + " wheels. It runs smoothly!");

}

}

// Main class (no public keyword to avoid filename conflict)

class Main {

public static void main(String[] args) {

Vehicle myVehicle = new Vehicle("Generic Vehicle", 2);

myVehicle.displayInfo(); // Calls Vehicle's displayInfo()

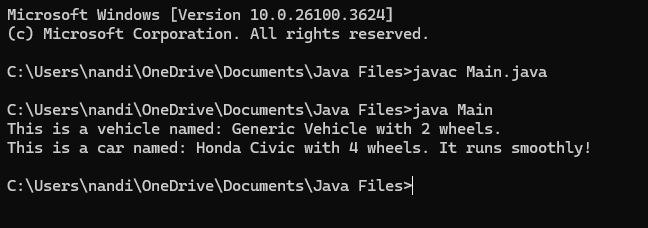
Car myCar = new Car("Honda Civic", 4);

myCar.displayInfo(); // Calls overridden displayInfo() in Car

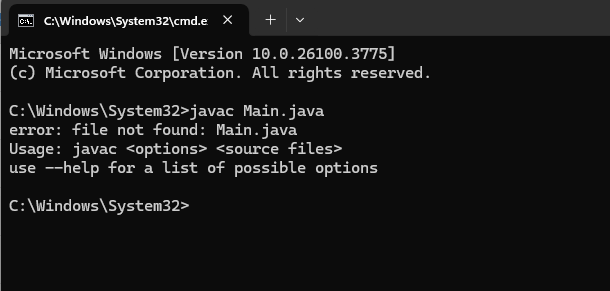
}

}

**POSITIVE OUTPUT-**



**NEGATIVE OUTPUT:**

****

**ERRORS:**

|  |  |  |
| --- | --- | --- |
| **Sl. no** | **Error** | **Error rectification** |
| 1. | error: class Main is public, should be declared in a file named Main.java  public class Main { | Save the file name as the name of the main class |

**IMPORTANT POINTS**-

1. We use the concept of method overriding where the names of the methods in the different classes. The method of the parent class is overridden by the method of the child class

**PROGRAM-2:**

A college is developing an automated admissions systems 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 qualification.

1. UG admission require min of 60%

2. PG admission require min of 70%

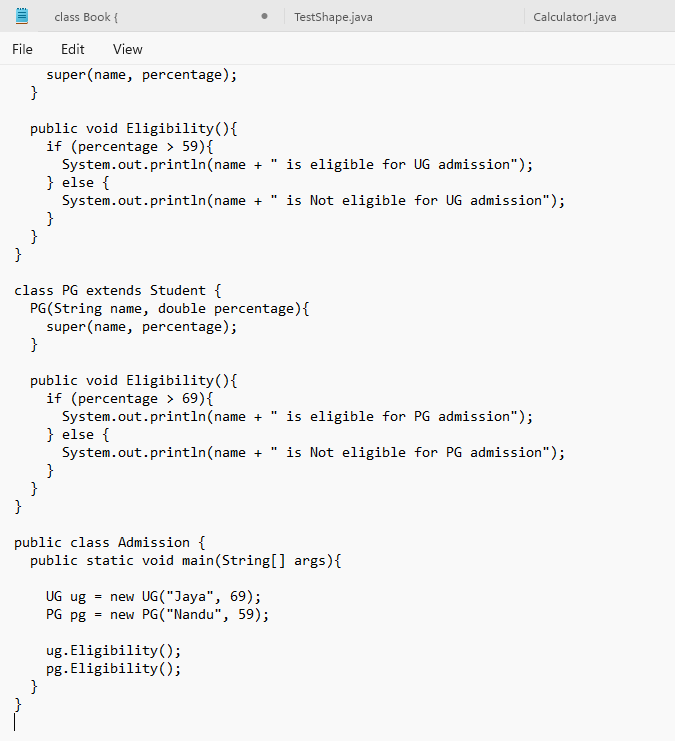
**CLASS DIAGRAM:**

|  |
| --- |
| Student |
| + name : String  + percentage : double |
| + Student(String name, double percentage): void  + Eligibility(): void |

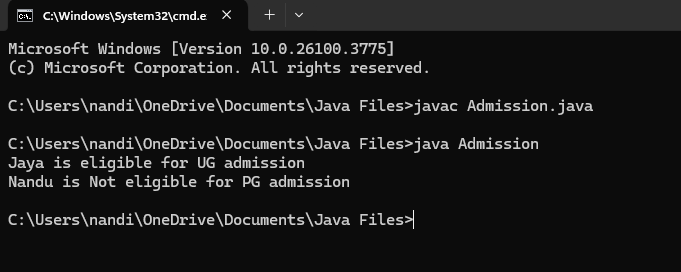
|  |
| --- |
| UG |
| UG(String name, double percentage) |

|  |
| --- |
| PG |
| PG(String name, double percentage) |

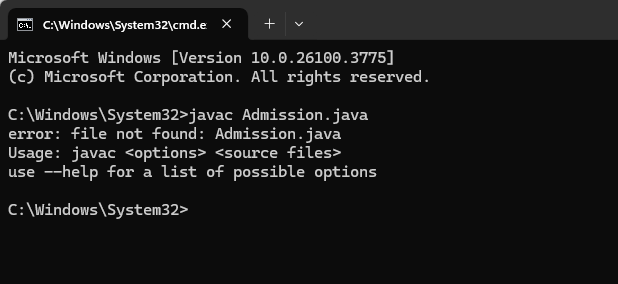
**CODE**-



**POSITIVE OUTPUT-**



**NEGATIVE OUTPUT:**

****

IMPORTANT POINTS-

1. The variables once declared in the super class need not be declared twice in any of the sub classes.

2. super keyword is used in sub classes to access the methods of super classes, they are basically the reverse of overriding.

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

A. Add two integers

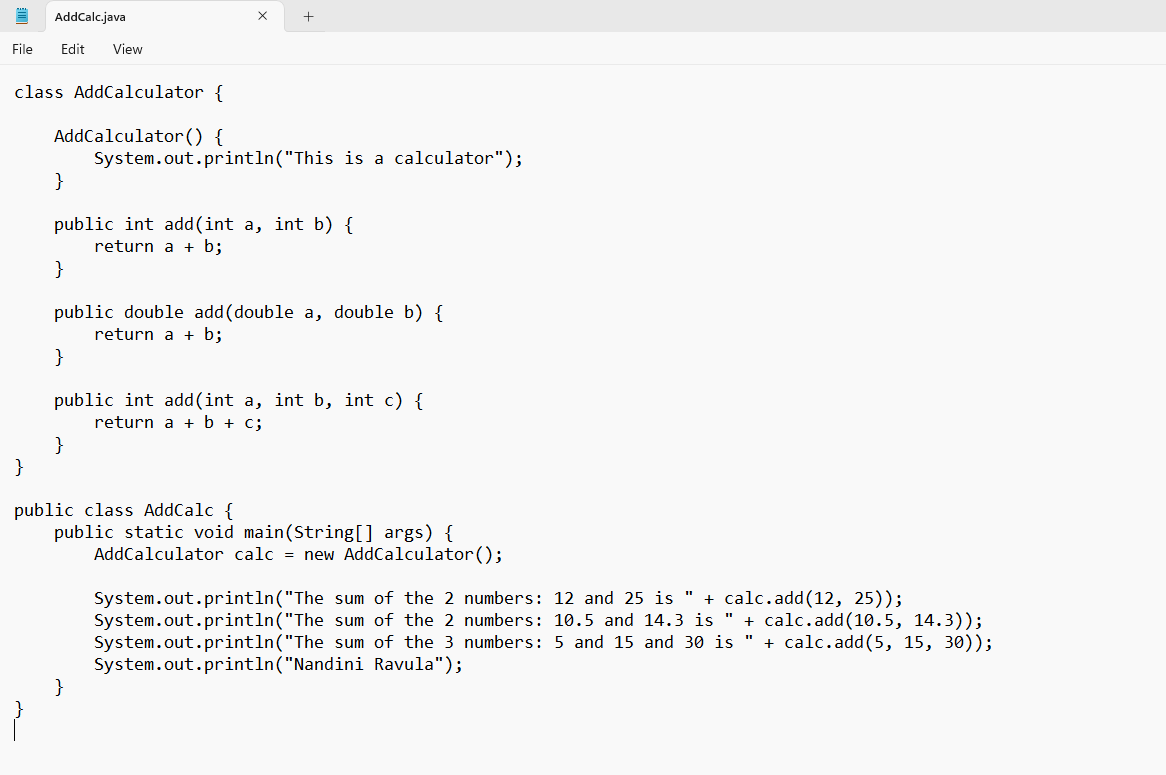
B. Add two double

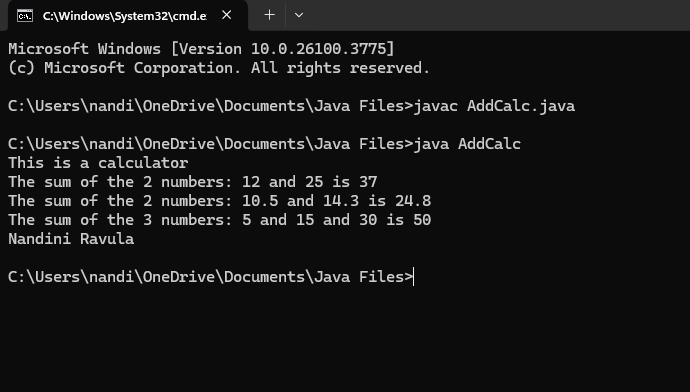
C. Add three integer

**CLASS DIAGRAM:**

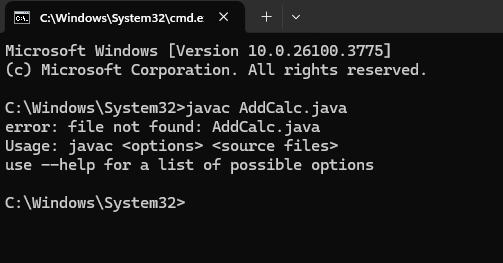
|  |
| --- |
| AddCalculator |
| + AddCalculator ()  + add(int a, int b) : int  + add(double a, double b) : double  + add(int a, int b, int c) : int |

**Code-**

****

**POSITIVE OUTPUT- **

**NEGATIVE OUTPUT-**

****

**ERRORS:**

|  |  |  |
| --- | --- | --- |
| **Sl. no** | **Error** | **Error rectification** |
| 1. | error: ';' expected System.out.println(name + " is elligible for UG admission") | Add a ‘;’ after the print statement. |

**IMPORTANT POINTS-**

1. The variables once declared in the super class need not be declared twice in any of the sub classes.

2. super keyword is used in sub classes to access the methods of super classes, they are basically the reverse of overriding.

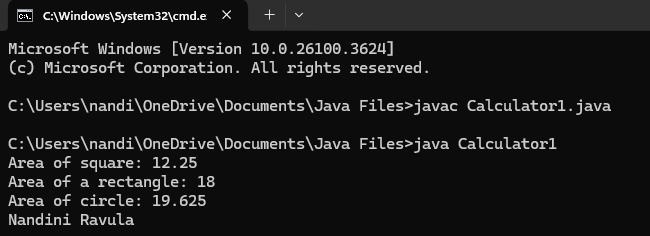
1. Create a shape class with a method CalculateArea() that is overloaded for different shpaes (e.g square, rectangle) then, create a subclass circle that overrides the calculatearea() method for a circle.
2. **Class Diagram:**

|  |
| --- |
| **Circle** |
| + r : double  + pi : double |
| + calcArea(int r) : double |
| **Shape** |
| + calcArea(int a) : int  + calcArea(int b, int h) : int |

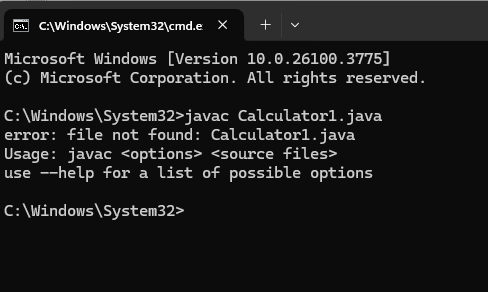
**CODE-**

****

**POSITIVE OUTPUT-**

****

**NEGATIVE OUTPUT:**

****

**ERRORS:**

|  |  |  |
| --- | --- | --- |
| **Sl. no** | **Error** | **Error rectification** |
| 1. | error: invalid method declaration; return type required  calcArea(int a){ | Enter the return type as per required. Here it is int |
| 2. | error: incompatible types: possible lossy conversion from double to int  return pi\*r\*r; | For calculating area of circle, we need to give return type double. |

**IMPORTANT POINTS**-

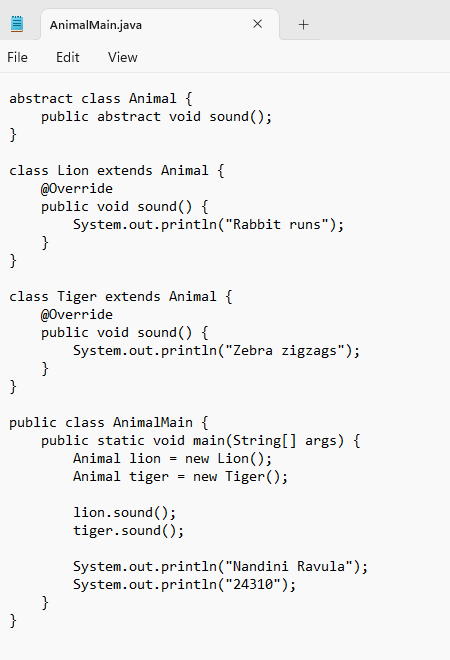
1. We use the concept of method overloading to calculate the area of square and rectangle in the parent class Shape.

2 we use method overriding in the child class Circle to calculate it’s area.

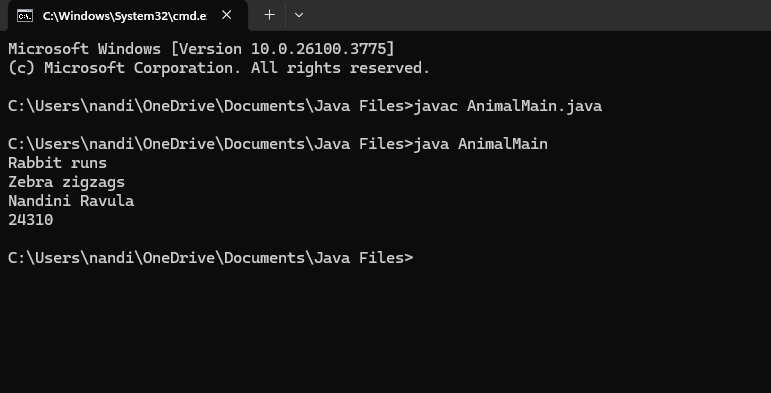
**WEEK-7**

1.Write a java program to create an abstract class animal with an abstract method called sound() .create sub class lion and tiger that extend the animal class and implement the sound method method to make a specifi sound for each animal.

**CODE-**

****

**OUTPUT-**

****