**23CSE111**

**LAB MANUAL**

A logo with pink letters

Description automatically generated

**Department of CSE**

**Amrita School of Engineering**

**Amrita Vishwa Vidyapeetham, Amaravati Campus**

**Verified By :- Name: CH LOHITH**

**Roll No: 24039**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| S.NO | Programs | Date | Pg:No | Signature |
| 1 | 1. Download and Install Java Software. 2. Write a java program to print message “Welcome to java programming”. 3. Write a java program that prints name , roll number, section of a student. |  |  |  |
| 2 | i. To calculate the area of the rectangle  ii. Program to convert the temperature in celsius to Fahrenheit.  iii. Program to calculate the simple interest.  iv. Program to find the largest of three numbers using the ternary operators.  v. Program to find the factorial of the number. |  |  |  |
| 3 | i. Create the java program for the cars with constructor and methods.  ii. Create the java program to withdraw and deposit money in the bank account. |  |  |  |
| 4 | i. Create the java program for the books by using the constructor and display its details using methods.  ii. Program to explain the final and the static variables. |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

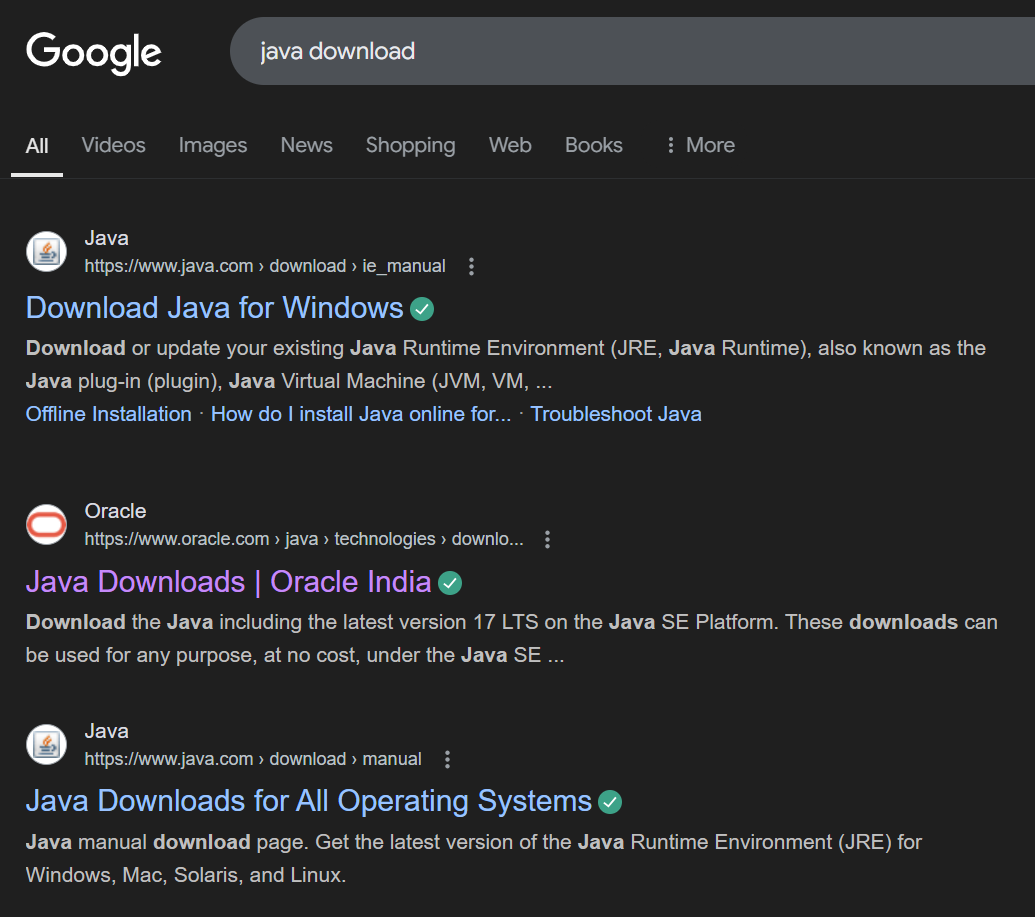
Week 1:-

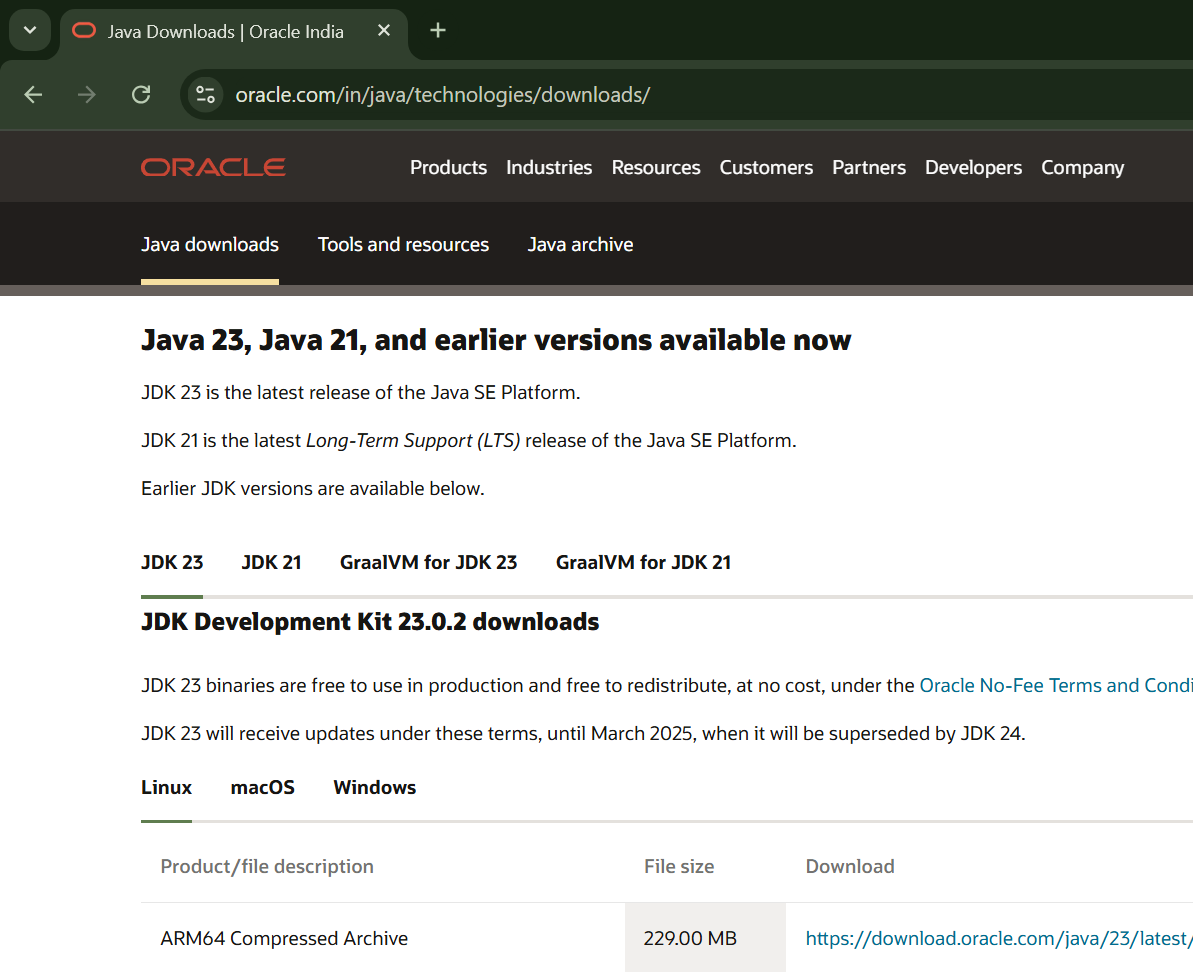
Program-1:-

Aim:-Download and Install the Java Software

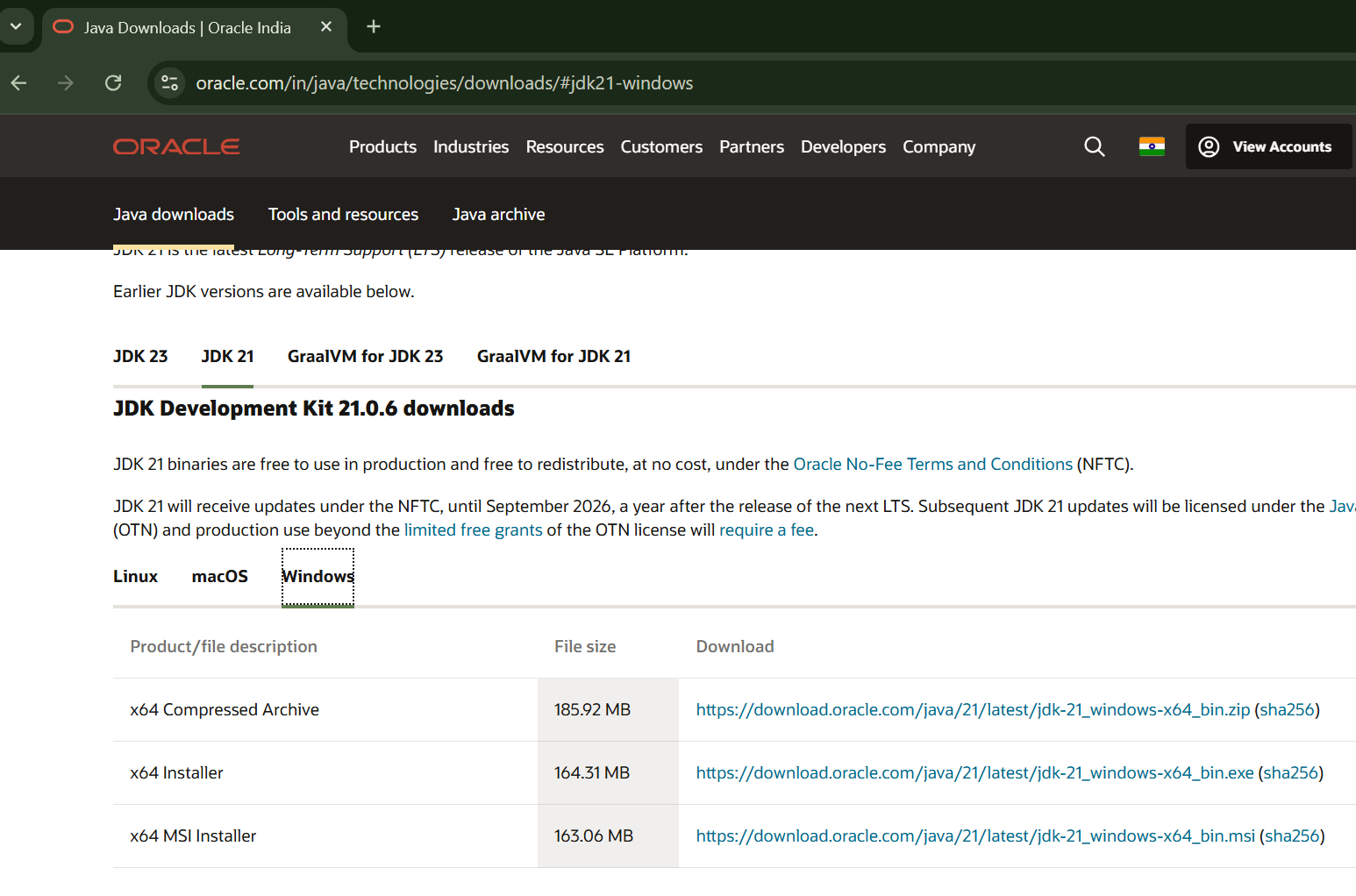
Procedure

Step-1:- Type Java download in search

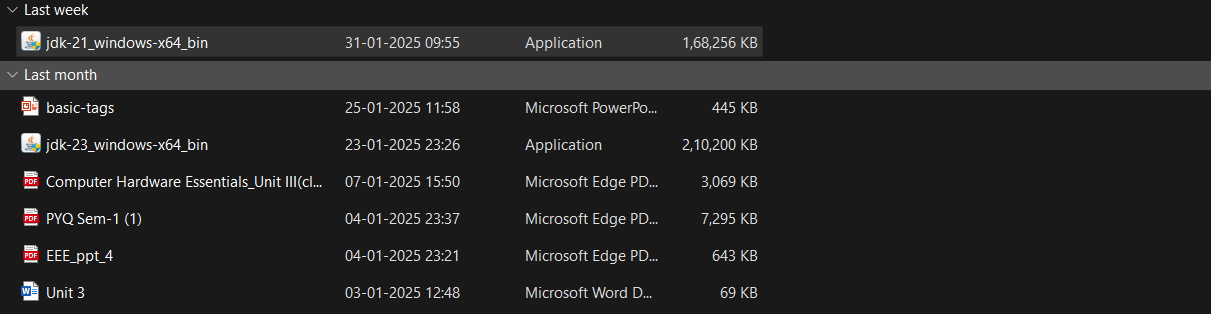


Step-2:-click on oracle java download and enter into oracle website 

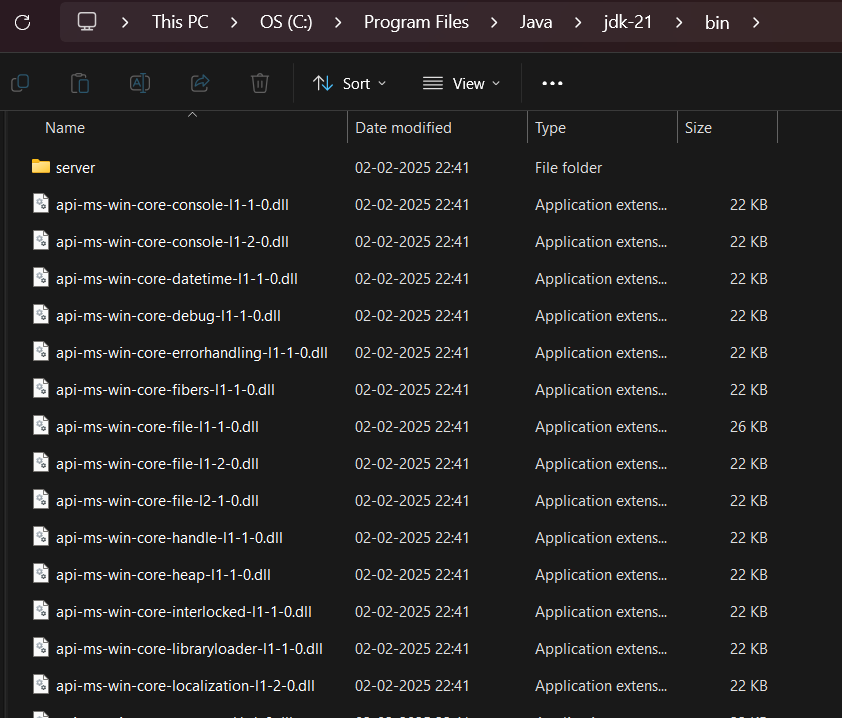
Step-3:-click on JDK21 and click on windows and later click on x64 instalier link to download



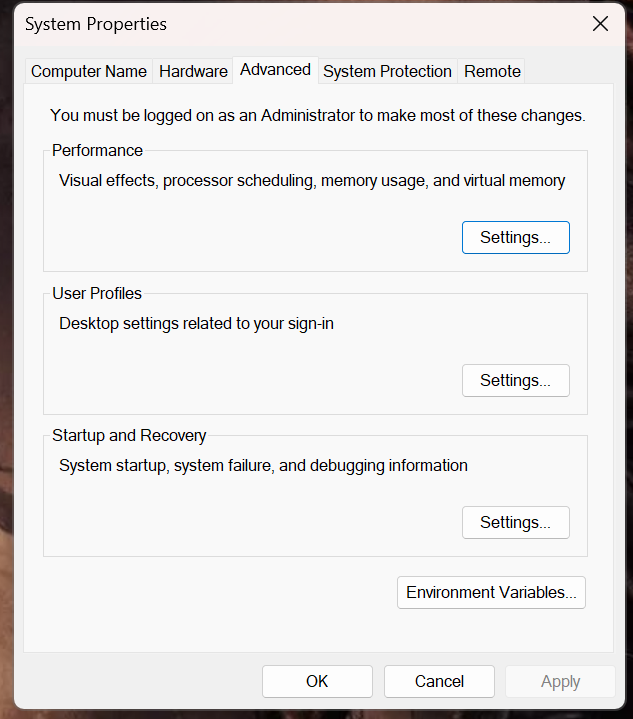
Step-4:-After completing download click on it’s file and then give permission to install



Step-5:-Then go to (This pc) in that click (windows{c}) in that click (Program files) in that click (Java) in that click (jdk-21) in that click (bin)

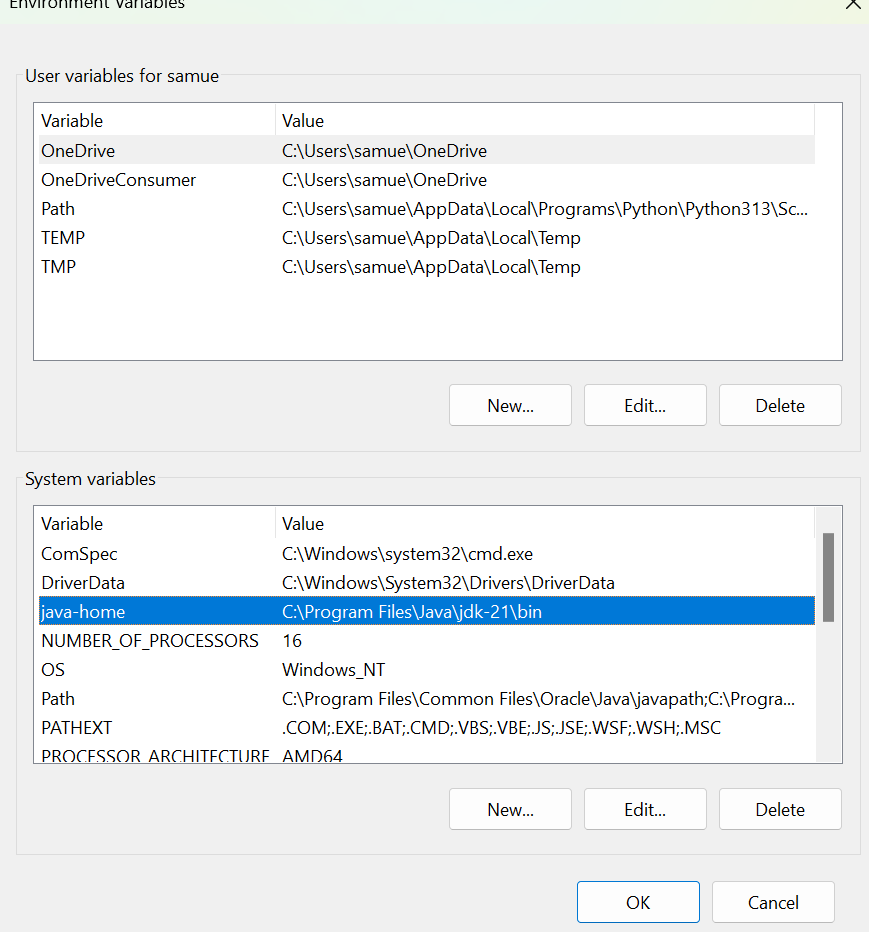


Step-6:-Select and copy path of opening the file and then press windows and search System Environmental



Step-7:-After opening Environment variables then past path of opening file in user variable and click on ok

.



Step-8:-To verify version open CMD and type java --version



Program : 2

Aim:-write a java program to print[welcome to java programming]

Input:-

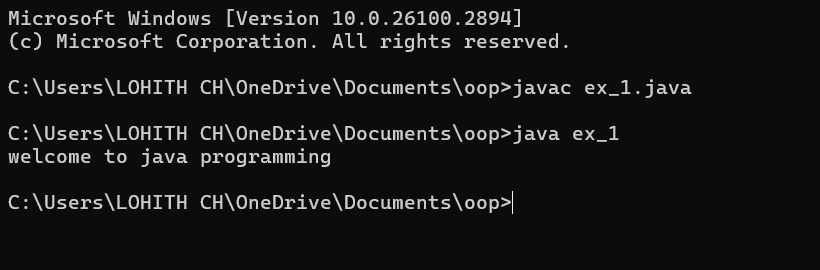
class ex\_1{

public static void main(String[] args){

System.out.println("welcome to java programming");

}

}

Output:- 

Program : 3

Aim:-write a java program that prints name, roll no, section of the student

Input:-

class ex\_2 {

public static void main(String args[]) {

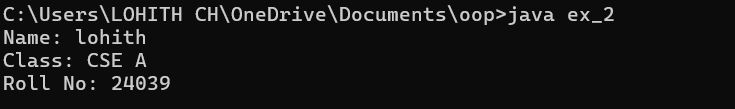
System.out.println("Name: lohith");

System.out.println("Class: CSE A");

System.out.println("Roll No: 24039");

}

}

Output:- 

***WEEK-2***

Program-1:

Aim: to write a java program to find area of rectangle

Input:

import java.util.Scanner;

public class RectangleArea {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

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

double length = scanner.nextDouble();

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

double width = scanner.nextDouble();

double area = length \* width;

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

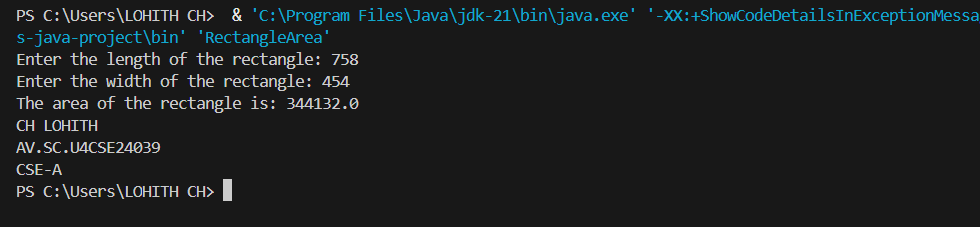
scanner.close();

System.out.println("CH LOHITH"+'\n'+"AV.SC.U4CSE24039"+'\n'+"CSE-A");

}

}

Output:



Error table:

|  |  |  |
| --- | --- | --- |
| Error | Error cause | Error rectification |
| error: ';' expected | ‘;’ is missed in the end of print statement | Placed ‘; ’ at the end of the statement |
| error:cannot find symbolScannerinput=new scanner(System.in); | Placed small s in place of capital S | Replaced capital s in place of small s to rectifiy the error |

**Program-2**

Aim: write a java program to convert temp from celsius to farenheit

Input:

import java.util.Scanner;

public class TemperatureConverter {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter temperature in Celsius: ");

double celsius = scanner.nextDouble();

double fahrenheit = (celsius \* 9/5) + 32;

System.out.println("Temperature in Fahrenheit: " + fahrenheit);

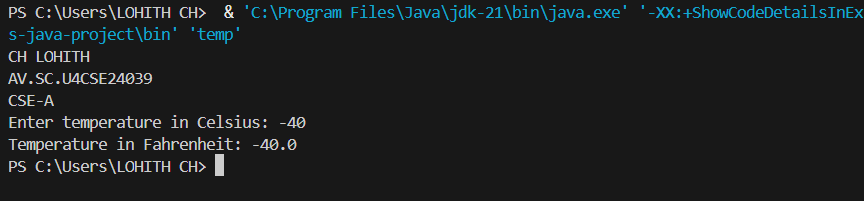
scanner.close();

System.out.println("CH LOHITH"+'\n'+"AV.SC.U4CSE24039"+'\n'+"CSE-A");

}

}

Output:



Error table:

|  |  |  |
| --- | --- | --- |
| Error | Error cause | Error rectification |
| error: ';' expected  scanner.close() | Forgot ‘;’ at the end the statement | Rectified by placing ’;’  it |
| error: incompatible types: possible lossy conversion from double to int  int celsius = scanner.nextDouble(); | Placed int in place of double | Rectified by replacing double |

**Program-3:**

Aim: write a java program to calculate simple interest

Input:

import java.util.Scanner;

public class SI{

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter the principal amount: ");

double principal = scanner.nextDouble();

System.out.print("Enter the rate of interest: ");

double rate = scanner.nextDouble();

System.out.print("Enter the time period in years: ");

double time = scanner.nextDouble();

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

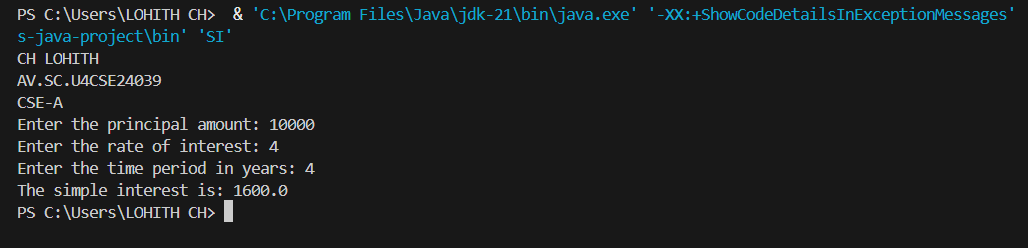
System.out.println("The simple interest is: " + interest);

scanner.close();

}

}

Output:



Error table:

|  |  |  |
| --- | --- | --- |
| Error | Error cause | Error rectification |
| error: class Sinterest is public, should be declared in a file named Sinterest.java  public class Sinterest {  ^ | Placed capital s I place of small ‘s’ | Replaced by placing small ‘s’ |

**Program-4**

Aim: write a java program to find the largest of 3 numbers using terenary operator

Input:

import java.util.Scanner;

public class largestnumber {

public static void main(String[] args) {

        System.out.println("CH LOHITH"+'\n'+"AV.SC.U4CSE24039"+'\n'+"CSE-A");

Scanner scanner = new Scanner(System.in);

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

int num1 = scanner.nextInt();

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

int num2 = scanner.nextInt();

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

int num3 = scanner.nextInt();

int largest = (num1 > num2) ? (num1 > num3 ? num1 : num3) : (num2 > num3 ? num2 : num3);

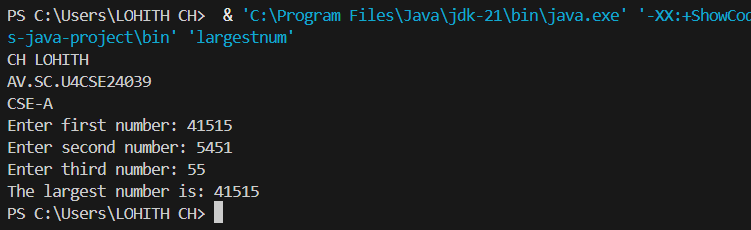
System.out.println("The largest number is: " + largest);

scanner.close();

}

}

Output:



Error table:

|  |  |  |
| --- | --- | --- |
| Error | Error cause | Error rectification |
| error: reached end of file while parsing  ((num1 > num3) ? num1 : num3) | Missed ‘} ’ in the end of the program | Rectified by replacing it |
| error: illegal start of expression  }  ^ | Missed’}’in the starting | Rectified by replacing it |

**Program-5**

Aim: write a java program to find the factorial of a number

Input:

import java.util.Scanner;

public class FactorialCalculator {

public static void main(String[] args) {

System.out.println("CH LOHITH"+'\n'+"AV.SC.U4CSE24039"+'\n'+"CSE-A");

Scanner scanner = new Scanner(System.in);

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

int number = scanner.nextInt();

long factorial = 1;

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

factorial \*= i;

}

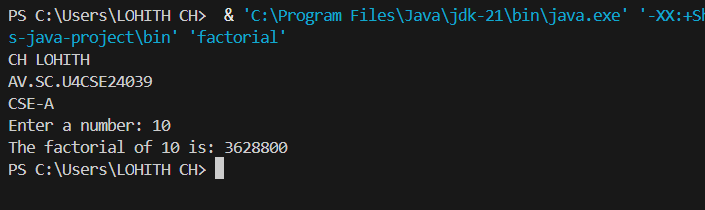
System.out.println("The factorial of " + number + " is: " + factorial);

scanner.close();

}

}

Output:



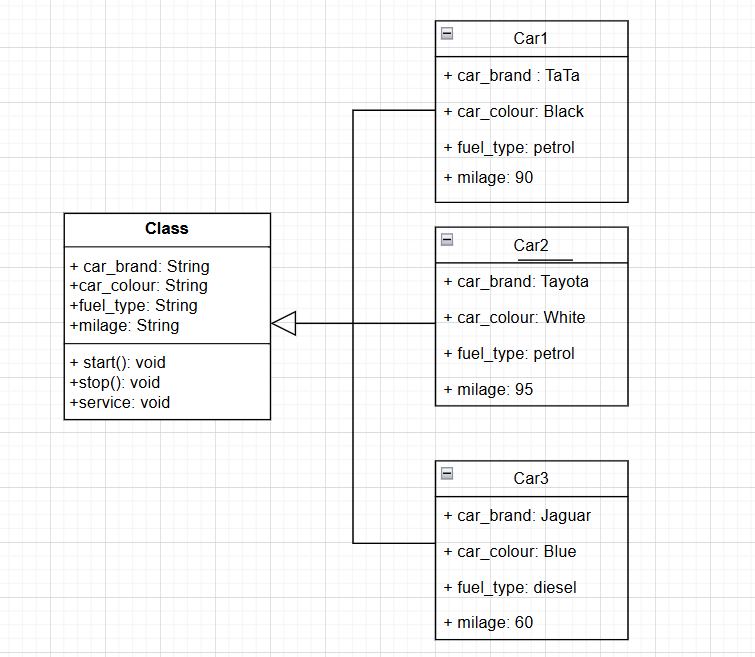
Error table:

|  |  |  |
| --- | --- | --- |
| Error | Error cause | Error rectification |
| error: unclosed string literal  System.out.print("Enter a number: );  ^ | Missed “ in the end | Rectified by replacing “ |

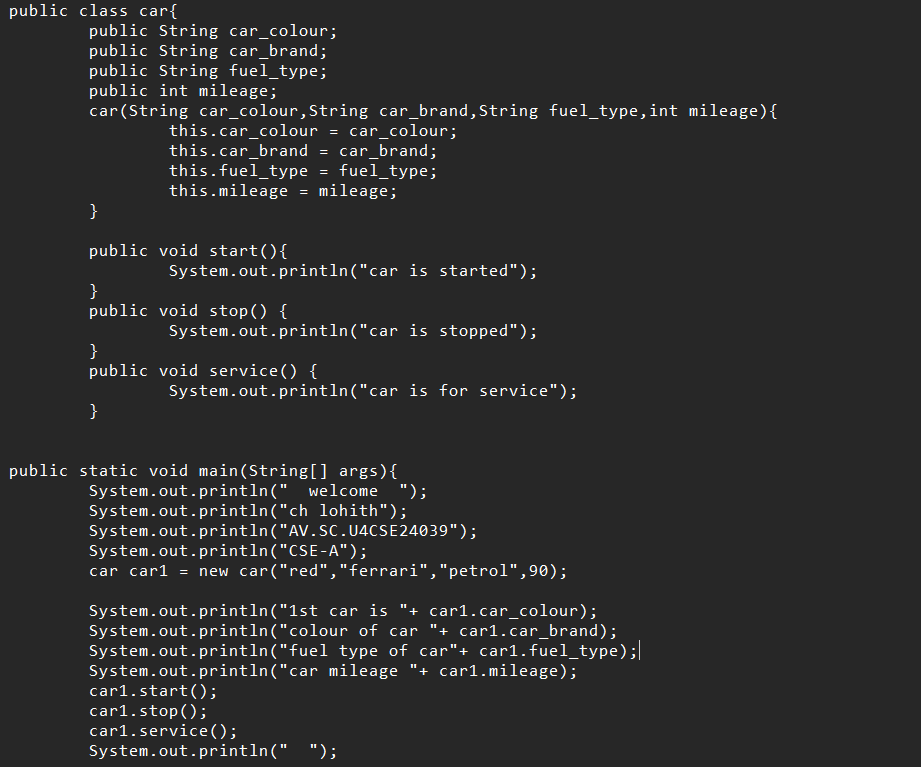
**Week-3:**

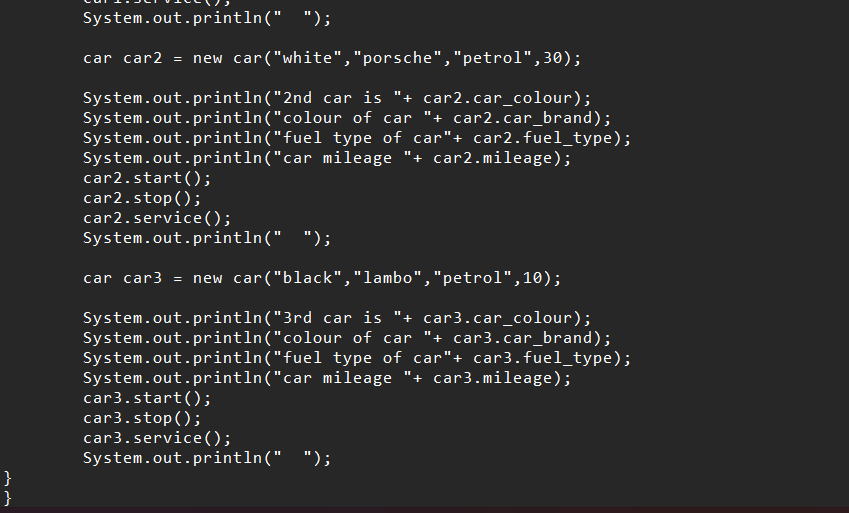
**Aim: (i)** Create the java program for the cars with constructor and methods.

**Class Diagram:**

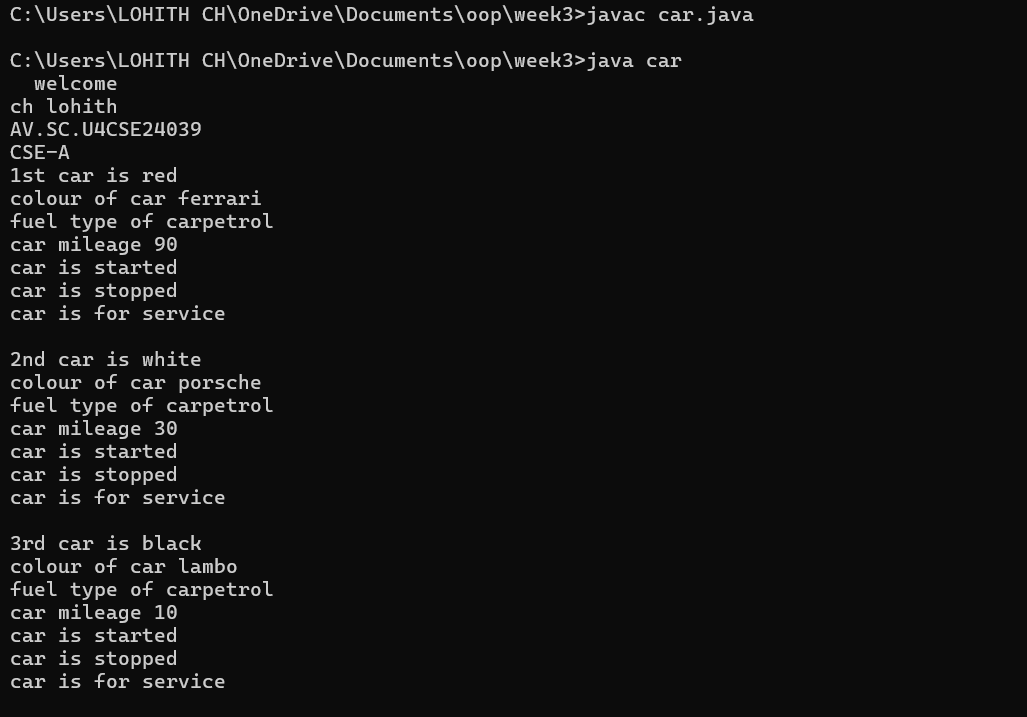
****

**Program:**

****

****

**Output:**

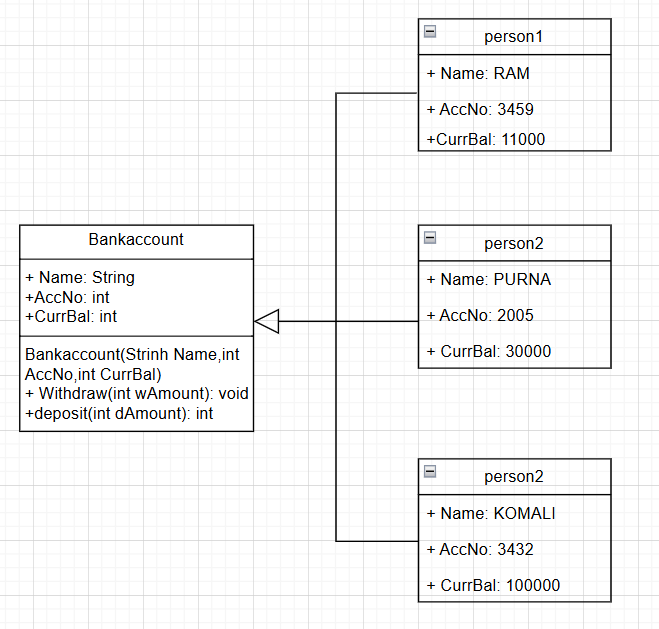


**Error:**

|  |  |  |
| --- | --- | --- |
| **Error Type** | **Incorrect Code** | **Corrected Code** |
| **Class Naming Issue** | class main{ | class Main{ |
| **Incorrect Object Description** | "1st car is "+car2.car\_brand; | "2nd car is "+car2.car\_brand; |

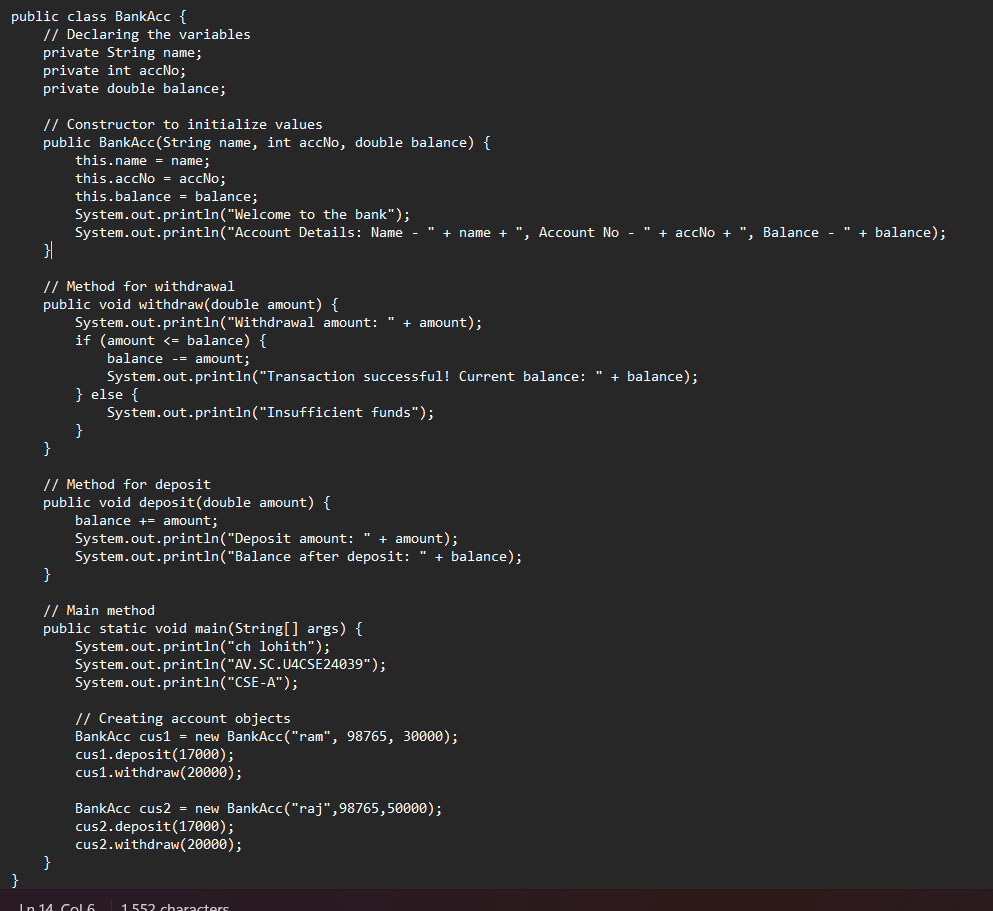
**Aim: (ii)** Create the java program to withdraw and deposit money in the bank account.

**Class Diagram:**

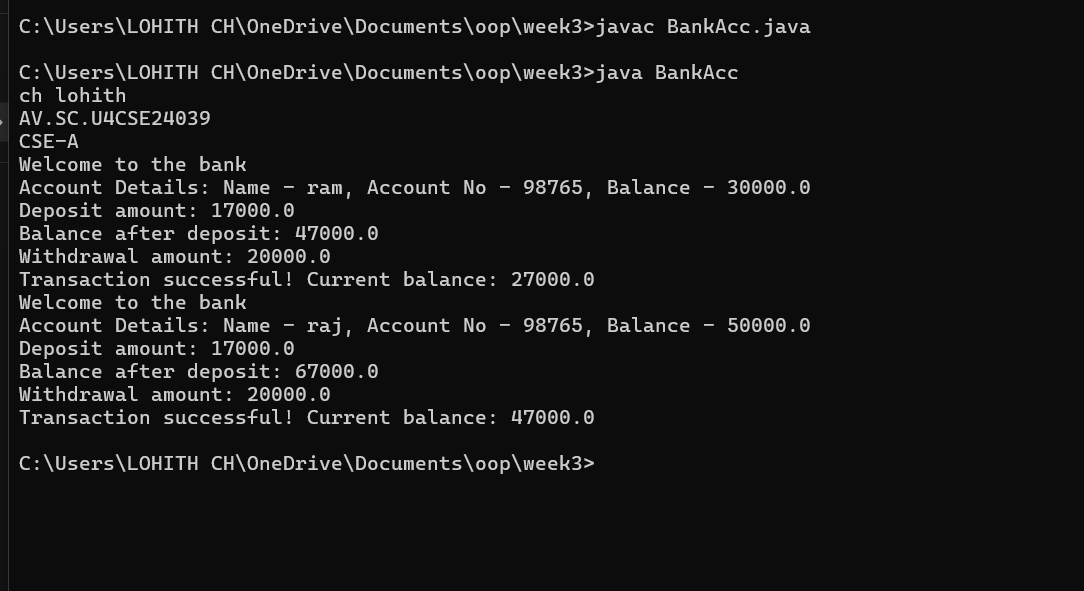
****

**Procedure:**

Code:

****

**Output:**

****

**Error:**

|  |  |  |
| --- | --- | --- |
| **Error Type** | **Incorrect Code** | **Corrected code** |
| **Class Name Capitalization** | class Bankaccount | class BankAccount (Java follows PascalCase for class names) |
| **Object Naming Issue** | BankAccount person-1 (hyphen is not allowed) | BankAccount person1 |
| **Missing Semicolon** | System.out.println ("Balance is "+ person-1.deposit (50,000)) | System.out.println ("Balance is "+ person1.deposit (50000)); (semicolon added) |

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

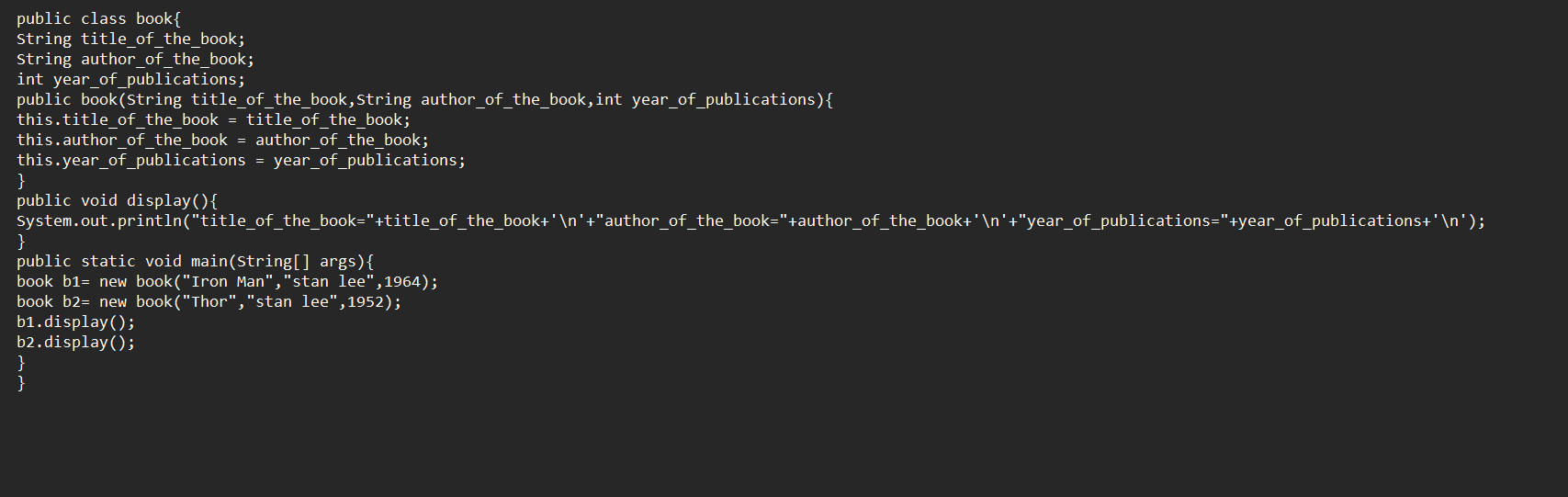
**Week-4:**

**Aim: (i)** Create the java program for the books by using the constructor and display its details using methods.

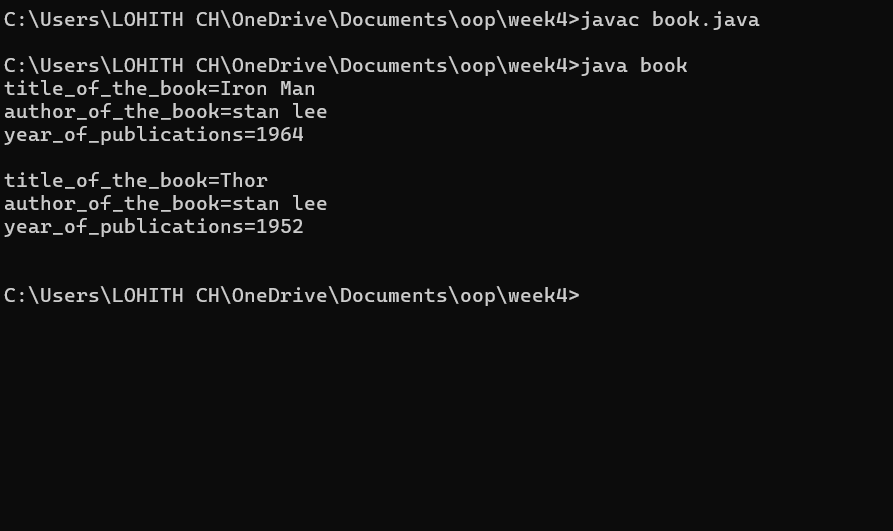
**Class Diagram:**

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

**Proceure:**

Code: 

**Output:**

****

**Errors:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Error Type** | |  | | --- | | **Incorrect Code** |  |  | | --- | |  | | **Corrected Code** |
| **Class Name Capitalization** | public class book | public class Book (Java follows PascalCase for class names) |
| **Constructor Name Mismatch** | new book(...) | new Book(...) (Constructor name must match class name) |

**Aim: (ii)** Program to explain the final and the static variables.

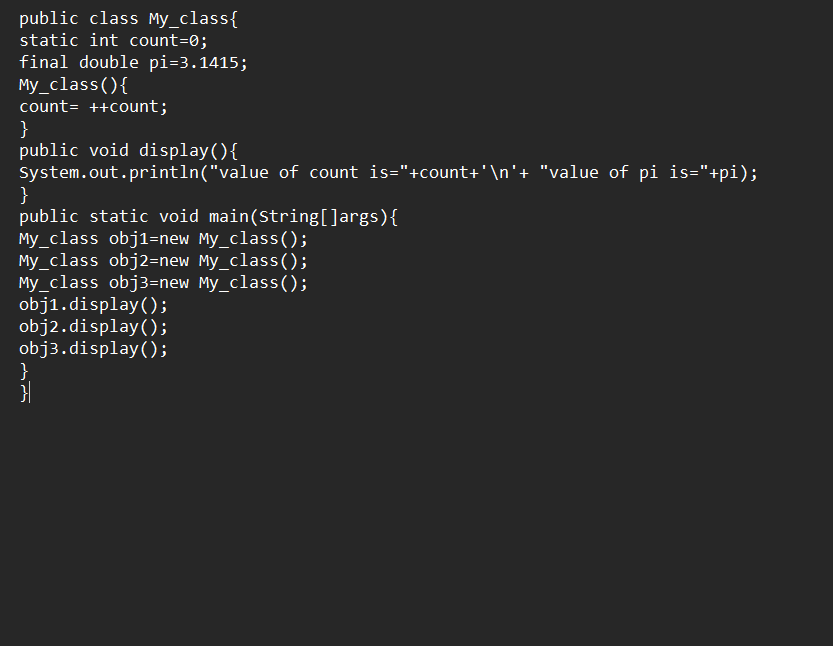
**Class Diagram:**

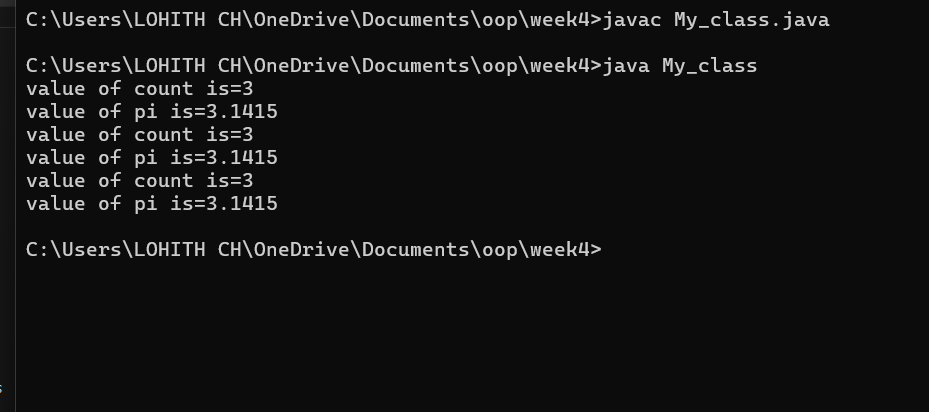
|  |
| --- |
| **MyClass** |
| **- Count: int**  **+ pi: double** |
| **+ MyClass()**  **+ getCount(): int** |

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |

**Procedure:**

Code:



**Output: **

**Error:**

|  |  |  |
| --- | --- | --- |
| **Error Type** | **Incorrect Code** | **Corrected Code** |
| **Attempt to Modify final Variable** | pi = 3.14; (if added inside the constructor or method) | Remove this line (final variables cannot be reassigned) |
| **Incorrect Class Name** | public class Myclass | public class MyClass (Java follows PascalCase for class names) |

**WEEK-5**

1. **Create a calc using the operations including add, sub, mul, div using multilevel inheritance and display the desired output.**

**Class Diagram:**

|  |
| --- |
| **Basic Operations** |
| **+ add (a,b)**  **+subtract (a,b)** |

|  |
| --- |
| **Multiplication** |
| **+Multiply (a,b)** |

|  |
| --- |
| **Division** |
| **+ Divide (a,b)** |

|  |  |
| --- | --- |
| **Subtraction** | |
| **+ subtraction(a,b)** | |
|  | |  | |

|  |
| --- |
| **Calculator** |
| **+calculate (op,a,b)** |

**Program:**

**public class cacluator {**

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

**int addition = a+b;**

**return addition;**

**}**

**}**

**class advanced\_cacluator extends cacluator{**

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

**int subraction=a-b;**

**return subraction;**

**}**

**}**

**class scientific\_cacluator extends advanced\_cacluator{**

**public int multi(int a , int b){**

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

**return multiplication;**

**}**

**}**

**class super\_cacluator extends scientific\_cacluator{**

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

**if(b !=0){**

**return a/b;**

**}else{**

**System.out.println("the ans is undefine");**

**return a/b;**

**}**

**}**

**}**

**class main{**

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

**super\_cacluator obj = new super\_cacluator();**

**System.out.println("additon is:"+obj.add(4,3));**

**System.out.println("substraction is:"+obj.sub(4,3));**

**System.out.println("multiplication is:"+obj.multi(4,3));**

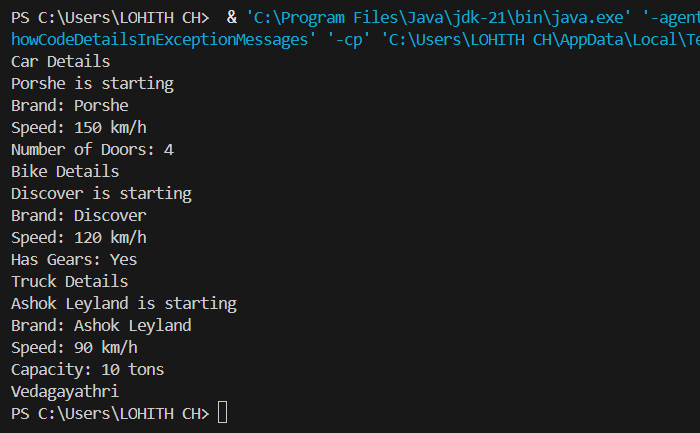
**System.out.println("division is:"+obj.div(4,3));**

**System.out.print ln("division is:"+obj.div(4,0));**

**}**

**}**

**Output:**

****

**Error table:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Errors** | **Rectification** |
| **1** | **.variable** | **We must mention variable name to call the variable.** |
| **2** | **static** | **Static variables contain only one value.** |

**B.Program : 2**

**Q) 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 a need a program to store details about each vehicle such as brand and speed .**

* **Cars should have an additional properties .**
* **“Number of doors “ seating capacity.**
* **Bikes should have a property indicating whether they have gears are not ?**
* **The system should also include a fuction to display details about each vehicle and indicate when a vechicle is starting .**
* **If the company describes to add a new type of vechile ‘truck’ how would you modify above program.**
* **Truck should include an addition property capacity ‘in tons’.**
* **Create a show truck details method to display the trucks capacity.**
* **Write a constructor for truck that initializes all properties.**
* **Implement the truck class and update the main method to create a truck object and also create an object and also create an object car and bike subclass find display it details.**

**Class Diagram:**

|  |
| --- |
| **Vechile** |
| * **Brand: String** * **speed: int** |
| **+ Vechile(String,b int)**  **+ Start()**  **+ DisplayDetails()** |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | **Car** | | * **numberofdoors: int** * **seatingCapacity: int** | | **+ car(String,int,int,int)**  **+ displaydetails()** | | |  | | --- | | **Bike** | | **-hasGears: boolean** | | **+ Bike(String,int,Boolean)**  **+displayetails()** | |

|  |
| --- |
| **Truck** |
| **-capacity: double** |
| **+ truck(String,int,double)**  **+showtruckdetails()**  **+displaydetails()** |

**// Base class for Vehicle**

**public class vehicle {**

**public String brand;**

**public int speed;**

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

**this.brand = brand;**

**this.speed = speed;**

**}**

**public void start() {**

**System.out.println(brand + " is starting");**

**}**

**public void showDetails() {**

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

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

**}**

**}**

**class Car extends vehicle {**

**private int noOfDoors;**

**public Car(String brand, int speed, int noOfDoors) {**

**super(brand, speed);**

**this.noOfDoors = noOfDoors;**

**}**

**public void showDetails() {**

**super.showDetails();**

**System.out.println("Number of Doors: " + noOfDoors);**

**}**

**}**

**class Bike extends vehicle {**

**private boolean hasGears;**

**public Bike(String brand, int speed, boolean hasGears) {**

**super(brand, speed);**

**this.hasGears = hasGears;**

**}**

**public void showDetails() {**

**super.showDetails();**

**System.out.println("Has Gears: " + (hasGears ? "Yes" : "No"));**

**}**

**}**

**class Truck extends vehicle {**

**private int capacity;**

**public Truck(String brand, int speed, int capacity) {**

**super(brand, speed);**

**this.capacity = capacity;**

**}**

**public void showTruck() {**

**super.showDetails();**

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

**}**

**}**

**class main {**

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

**Car car = new Car("Porshe", 150, 4);**

**Bike bike = new Bike("Discover", 120, true);**

**Truck truck = new Truck("Ashok Leyland", 90, 10);**

**System.out.println("Car Details");**

**car.start();**

**car.showDetails();**

**System.out.println("Bike Details");**

**bike.start();**

**bike.showDetails();**

**System.out.println("Truck Details");**

**truck.start();**

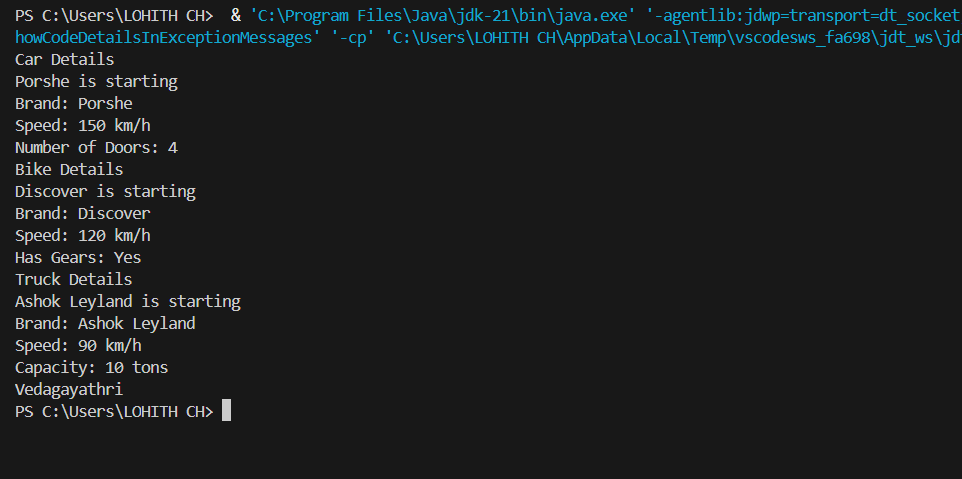
**truck.showTruck();**

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

**}**

**}**

**Output:**

****

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Errors** | **Rectification** |
| **1** | **.variable** | **We must mention variable name to call the variable.** |
| **2** | **static** | **Static variables contain only one value.** |

**WEEK-6**

1. **Write a java program to create a vehicles class with a method displayinfo() override this method in the car subclass to provide specific information about a car**

* **Company**
* **Model**
* **Price**
* **Seating capacity**
* **Petrol or not**

**Class Diagram:**

|  |
| --- |
| **Vehicle** |
| **+displayInfo(): void** |

|  |
| --- |
| **car** |
| **+displayInfo(): void** |

**Program:**

**class vehicle**

**{**

**public void displayinfo()**

**{**

**System.out.println("This is the vehicle information");**

**}**

**}**

**class car extends vehicle**

**{**

**public void displayinfo(String car\_company , String car\_model, int car\_price,int car\_seating\_capacity,boolean petrol)**

**{**

**System.out.println("the car company is " + car\_company+'\n'+"the car model is "+car\_model +'\n'+"the car price is "+car\_price+'\n'+"the car seating capacity"+ car\_seating\_capacity+'\n'+"car fuel tank type is "+petrol);**

**}**

**}**

**class main**

**{**

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

**{**

**car c = new car();**

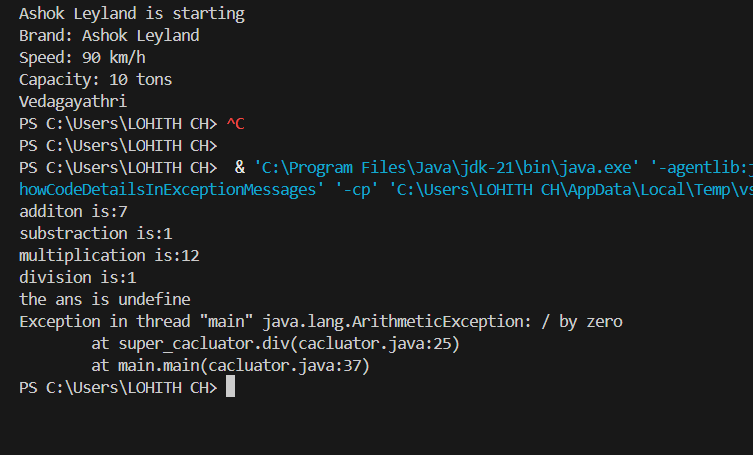
**c.displayinfo();**

**c.displayinfo("BMW","Mrcedies",19934,6,true);**

**}**

**}**

**Output:**

****

**Error table:**

|  |  |  |
| --- | --- | --- |
| **S.No** | **Expected Error** | **Reason** |
| **1** | **Setting the parameters inside the constructor** | **We cannot pass the values inside constructor without setting them first** |
| **2** | **}** | **Ending the class and main method is required** |

**2Q) A college is developing automated admission system that verifies students eligibility for UG and PG programs .Each program has different eligibility criteria based on the students percentage in their previous qualification.**

* **UG admission require minimum 60%**
* **PG admission require minimum 70%**

**Program:**

**class Admi**

**{**

**double UG=0;**

**double PG=0;**

**public void samp()**

**{**

**this.UG=0;**

**this.PG=0;**

**}**

**public void samp(double UG, double PG)**

**{**

**this.UG=UG;**

**this.PG=PG;**

**}**

**}**

**class Ssion extends Admi**

**{**

**public void UG()**

**{**

**if ( UG<=60 )**

**{**

**System.out.println("In valid for UG because of low score");**

**}**

**else**

**{**

**System.out.println("Student is eligible for UG");**

**}**

**}**

**public void PG()**

**{**

**if ( PG<=70 )**

**{**

**System.out.println("In valid for PG because of low score");**

**}**

**else**

**{**

**System.out.println("Student is eligible for PG");**

**}**

**}**

**}**

**class Admission**

**{**

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

**{**

**Ssion A = new Ssion();**

**A.samp(69,79);**

**A.UG();**

**A.PG();**

**}**

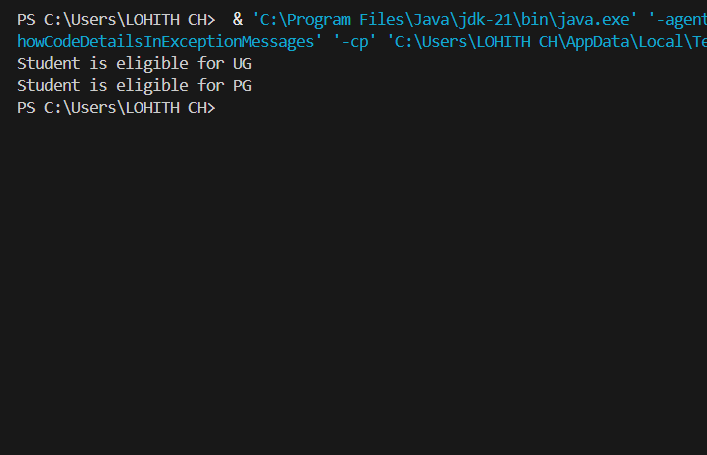
**}**

**Class Diagram:**

|  |
| --- |
| **adm** |
| **elg():void** |

|  |  |
| --- | --- |
| **ug** | **pg** |
| **+elg():void** | **+elg():void** |

**Output:**

****

**Error table:**

|  |  |  |
| --- | --- | --- |
| **S.No** | **Expected Error** | **Reason** |
| **1** | **Setting the parameters inside the constructor** | **We cannot pass the values inside constructor without setting them first** |
| **2** | **}** | **Ending the class and main method is required** |

**3Q)Create a calculator class with overloading methods to perform addition**

* **Add two doubles**
* **Add two integer**
* **Add three integer**

**Program:**

**class cacluato{**

**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 main{**

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

**cacluato d = new cacluato();**

**System.out.println( "sum of 5 and 6 is:"+ d.add(5, 6));**

**System.out.println("sum of 5.5 and 6.6 is:"+ d.add(5.5,6.6));**

**System.out.println("sum of 5,6 and 7 is:"+ d.add(5,6,7));**

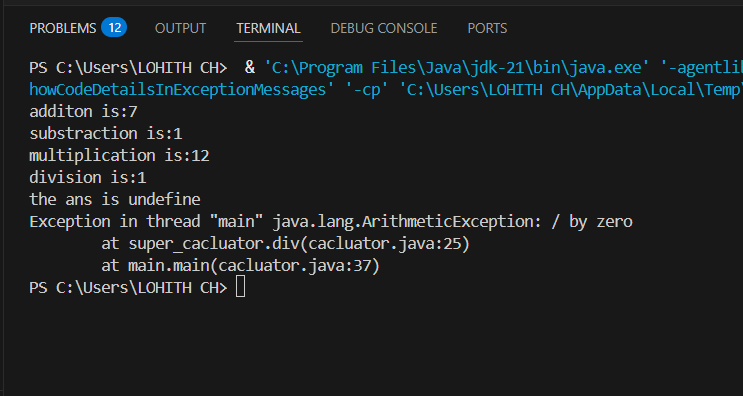
**}**

**}**

**Class Diagram:**

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

**Output:**

****

**Error table:**

|  |  |  |
| --- | --- | --- |
| **S.No** | **Expected Error** | **Reason** |
| **1** | **Setting the parameters inside the constructor** | **We cannot pass the values inside constructor without setting them first** |
| **2** | **}** | **Ending the class and main method is required** |

**4Q)Create a shape class with a method calculate area that is overloaded for different shapes Square,Rectangle then create a sub class circle that overerides the calculate area methods for a circle.**

**Program:**

**class shape{**

**public Double area( Double a ){**

**return a\*a;**

**}**

**public  Double area(Double length , Double breadth ){**

**return length\*breadth;**

**}**

**}**

**class circle extends shape{**

**public Double area(Double radius){**

**return 3.14\*radius\*radius;**

**}**

**}**

**class main{**

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

**shape s = new shape();**

**circle c = new circle();**

**System.out.println("area of square is "+s.area(5.0));**

**System.out.println("area of rectangle is"+s.area(5.0,6.0));**

**System.out.println("area of circle is "+c.area(3.0));**

**}**

**}**

**Class Diagram:**

|  |
| --- |
| **shape** |
| **+calarea(float side):float**  **+calarea(float l,float b):float**  **+calarea(float c):float** |

|  |
| --- |
| **Circle** |
| **+calarea(double r):double** |

**Output:**

**Error table:**

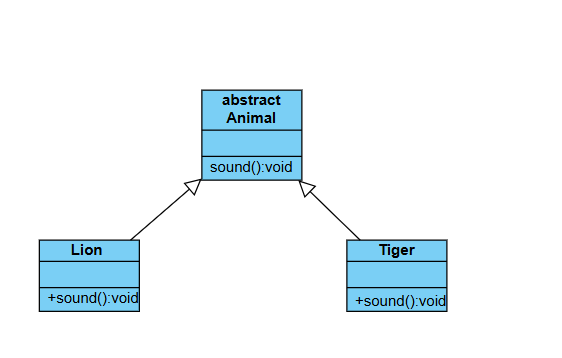
|  |  |  |
| --- | --- | --- |
| **S.No** | **Expected Error** | **Reason** |
| **1** | **Setting the parameters inside the constructor** | **We cannot pass the values inside constructor without setting them first** |
| **2** | **}** | **Ending the class and main method is required** |

**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:**

****

**CODE:**

**abstract class animals{**

**abstract void sound();**

**}**

**class lion extends animals{**

**public void sound(){**

**System.out.println("lion roar");**

**}**

**}**

**class tiger extends animals{**

**public void sound(){**

**System.out.println("tiger growl");**

**}**

**}**

**class main{**

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

**System.out.println("CH LOHITH"+'\n'+"AV.SC.U4CSE24039"+'\n'+"CSE-A");**

**tiger t = new tiger();**

**t.sound();**

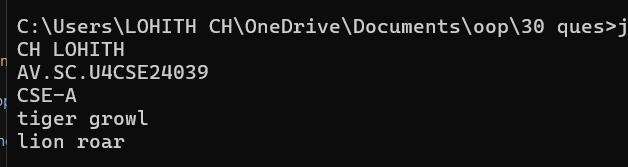
**lion l = new lion();**

**l.sound();**

**}**

**}**

**OUTPUT:**

****

**ERROR: Error table:**

|  |  |  |
| --- | --- | --- |
| **Type** | **Description** | **Fix** |
| **Method name mismatch** | **Wrong case used in method name** | **Change Eat() to eat()** |

**IMPORTANT POINTS:**

**1. Animal class is declared as an abstract.**

**2.the method sound has no parameters in super class and subclasses.**

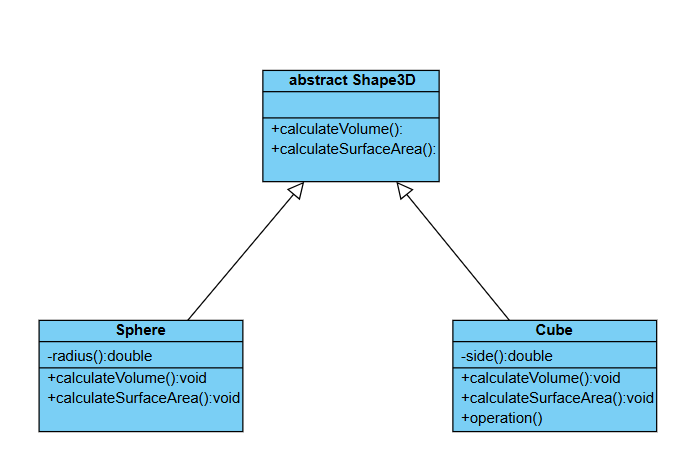
**3.each subclass extends from super class.**

**4.method overriding is happened here**

**5.the method sound declared as abstract in super class.**

**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:**

****

**CODE:**

**abstract class Shape3D{**

**double volume, surfaceArea;**

**abstract void calculateVolume();**

**abstract void calculateSurfaceArea();**

**}**

**class Sphere extends Shape3D{**

**double radius;**

**Sphere(double radius){**

**this.radius=radius;**

**}**

**public void calculateVolume(){**

**volume = 4.0/3.0 \* Math.PI \* radius \* radius \* radius;**

**System.out.println("Volume of Sphere is: " + volume);**

**}**

**public void calculateSurfaceArea(){**

**surfaceArea = 4.0 \* Math.PI \* radius \* radius;**

**System.out.println("Surface Area of Sphere is: " + surfaceArea);**

**}**

**}**

**class Cube extends Shape3D{**

**double side;**

**Cube(double side){**

**this.side=side;**

**}**

**public void calculateVolume(){**

**volume = side \* side \* side;**

**System.out.println("Volume of Cube is: " + volume);**

**}**

**public void calculateSurfaceArea(){**

**surfaceArea = 6.0 \* side \* side;**

**System.out.println("Surface Area of Cube is: " + surfaceArea);**

**}**

**}**

**class Main {**

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

**System.out.println("CH LOHITH"+'\n'+"AV.SC.U4CSE24039"+'\n'+"CSE-A");**

**Sphere sphere = new Sphere(4);**

**sphere.calculateSurfaceArea();**

**sphere.calculateVolume();**

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

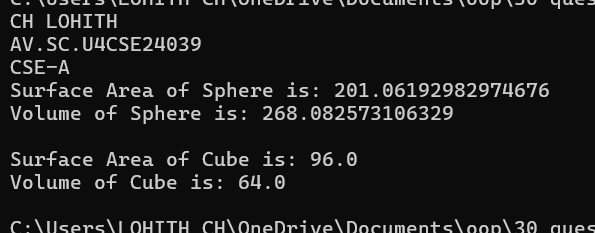
**Cube cube = new Cube(4);**

**cube.calculateSurfaceArea();**

**cube.calculateVolume();**

**}**

**}**

**Output:** ****

**Error table:**

|  |  |  |
| --- | --- | --- |
| **Type** | **Description** | **Fix** |
| **Calculation Error** | **Missing π (3.14) in volume formula** | **Add \* 3.14 in volume formula** |
| **Redundant Code** | **Unused variables in cube class** | **Remove unused breadth and height** |

**Important Points and Explanation:**

* **Animal is an abstract class with an abstract method eat().**
* **Lion and Tiger extend Animal and override eat().**
* **Both classes have their own sound() method.**
* **Objects are created in the main() method.**
* **Methods eat() and sound() are called to show output.**

**3.AIM: Write a java program using an abstract class to define a method for pattern printing.**

**create an abstract class named PatternPrinter with an abstract method PrintPattern(int n) and a concrete method to display the pattern title**

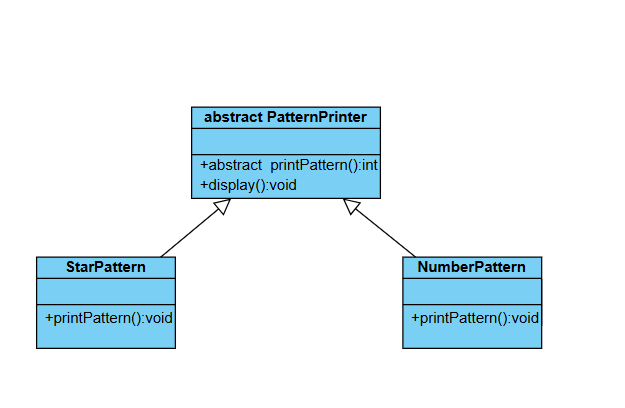
**Implement two subclasses:**

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

**2)Number pattern-prints a right angled triangle of increasing numbers**

**In the main() method ,create objects of both subclasses and print the patterns for a given number of rows**

**CLASS DIAGRAM:**

****

**CODE:**

**abstract class PatternPrinter{**

**public abstract void printPattern(int n);**

**public void display(String str){**

**System.out.println("/n " + str);**

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

**}**

**}**

**class StarPattern extends PatternPrinter{**

**public 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{**

**public void printPattern(int n){**

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

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

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

**}**

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

**}**

**}**

**}**

**class Main{**

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

**System.out.println("CH LOHITH"+'\n'+"AV.SC.U4CSE24039"+'\n'+"CSE-A");**

**StarPattern s =new StarPattern();**

**s.display("Star Pattern");**

**s.printPattern(5);**

**NumberPattern n = new NumberPattern();**

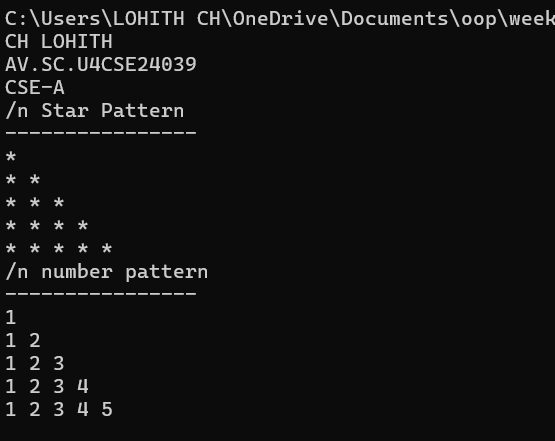
**n.display("number pattern");**

**n.printPattern(5);**

**}**

**}**

**OUTPUT:**

****

**ERROR TABLE:**

|  |  |  |
| --- | --- | --- |
| **S.No** | **ExpectedError** | **Reason** |
| **1** | **Setting the parameters inside the constructor** | **We cannot pass the values inside constructor without setting them first** |
| **2** | **}** | **Ending the class and main method is required** |

**IMPORTANT POINTS:**

**1. Abstract Class (PatternPrinter)**

* **Defines:**
  + **Abstract Method → printPattern(int n) (must be implemented by subclasses).**
  + **Concrete Method → displayTitle(String title) (prints a formatted pattern title).**
* **Cannot be instantiated directly (must be extended).**

**2. Subclasses (StarPattern and NumberPattern)**

* **Extend PatternPrinter and override printPattern(int n).**
* **StarPattern:**
  + **Prints a right-angled triangle of stars (\*)**

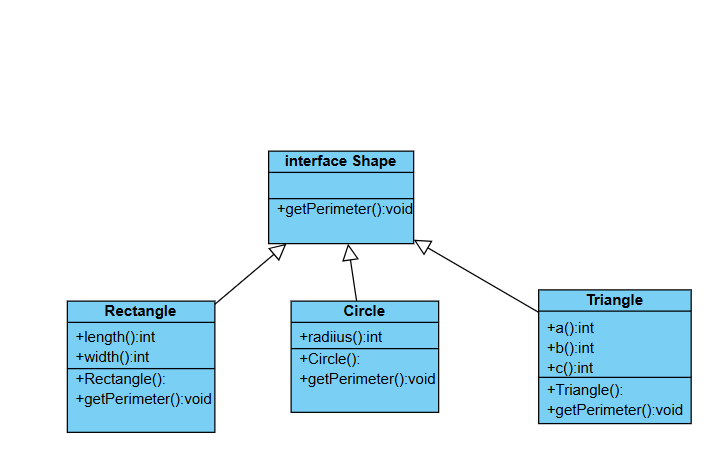
**NumberPattern:**

* **Prints a right-angled triangle of increasing numbers.**

**WEEK-8**

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

**CLASS DIAGRAM:**

****

**CODE:**

 interface Shape{

    void getPerimeter();

}

class Rectangle implements Shape{

    double l,b;

    public Rectangle(double l, double b){

        this.l=l;

        this.b=b;

    }

    public void getPerimeter(){

        System.out.println("Perimeter of Rectangle: " + (2\*(l+b)));

    }

}

class Circle implements Shape{

    double r;

    public Circle(double r){

        this.r=r;

    }

    public void getPerimeter(){

        System.out.println("Perimeter of Circle: " + (2\*Math.PI\*r));

    }

}

class Triangle implements Shape{

    double a,b,c;

    public Triangle(double a,double b,double c){

        this.a=a;

        this.b=b;

        this.c=c;

    }

    public void getPerimeter(){

        System.out.println("Perimeter of Triangle: " + (a+b+c));

    }

}

 class Main {

    public static void main(String[] args) {

        System.out.println("CH Lohith\nCSE-A\nAV.SC.U4CSE24039\n");

        Rectangle rect = new Rectangle(12,5);

        rect.getPerimeter();

        Circle circle = new Circle(4);

        circle.getPerimeter();

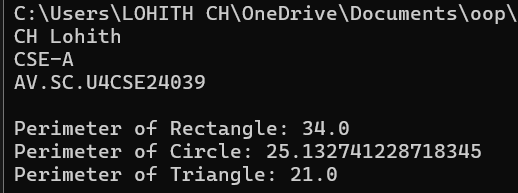
        Triangle triangle = new Triangle(5,7,9);

        triangle.getPerimeter();

    }

}

**Output:**



**Errors:**

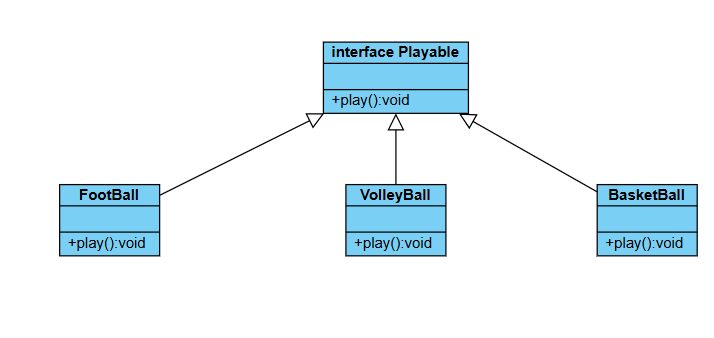
|  |  |  |
| --- | --- | --- |
| **Type** | **Description** | **Fix** |
| **Syntax Error** | **Typo in method name getPerimter() in Rectangle class** | **Rename method to getPerimeter()** |
| **Logical Error** | **Constructor parameter height doesn't match field name width** | **Change parameter name to width or assign this.width = height** |

**Important Points and Explanation:**

* **Interface is used to achieve abstraction and enforce a contract that any implementing class must follow.**
* **This shows runtime polymorphism where the same method (getPerimeter) has different behavior based on the shape.**
* **Method overriding requires the exact method name and parameters. Java uses method signatures to distinguish between methods.**

**2.AIM: Write a java program to create an interface playable with a method play() that takes no arguments and return 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:**

****

**CODE:**

**import static java.lang.System.\* ;**

**interface playable {**

**void play();**

**}**

**class Football implements playable {**

**public void play() {**

**System.out.println("Football is most popular sport");**

**}**

**}**

**class Volleyball implements playable {**

**public void play() {**

**System.out.println("volley ball is popular sport in china");**

**}**

**}**

**class Basketball implements playable{**

**public void play() {**

**System.out.println("basket ball is famous in USA");**

**}**

**}**

**class main{**

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

**System.out.println("CH LOHITH"+'\n'+"AV.SC.U4CSE24039"+'\n'+"CSE-A");**

**Football f = new Football();**

**Volleyball v = new Volleyball();**

**Basketball b = new Basketball();**

**f.play();**

**v.play();**

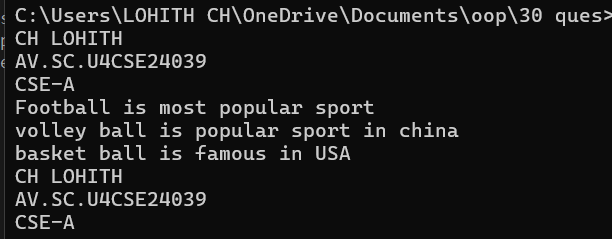
**b.play();**

**System.out.println("CH LOHITH"+'\n'+"AV.SC.U4CSE24039"+'\n'+"CSE-A");**

**}**

**}**

**Output:**

****

**Errors:**

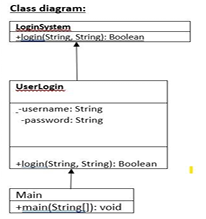
|  |  |  |
| --- | --- | --- |
| **Type** | **Description** | **Fix** |
| **Compile-time Error** | **Object instantiation uses new Football() instead of new football()** | **Change to new football() to match class definition** |

**Important Points and Explanation:**

* **The playable interface is defined with a method void play();. This sets a rule that any class implementing this interface must define its own version of the play() method.**
* **Football, volleyball, and basketball classes all implement the playable interface.**
* **Each class provides its specific implementation of the play() method, printing a message about the national sport of a country.**

**3Q) write a java program to implement a login system using interface.**

**Class Diagram:**

****

**Code:**

**interface Loginsystem {**

**boolean user(String username, String password);**

**}**

**class UserLogin implements Loginsystem {**

**private String validUsername;**

**private String validPassword;**

**public UserLogin(String validUsername, String validPassword) {**

**this.validUsername = validUsername;**

**this.validPassword = validPassword;**

**}**

**@Override**

**public boolean user(String username, String password) {**

**// Check if the entered username and password match the stored values**

**return username.equals(validUsername) && password.equals(validPassword);**

**}**

**}**

**public class login {**

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

**UserLogin login = new UserLogin("hemu", "pp@123");**

**String username = "hemu";**

**String password = "pp@123";**

**if (login.user(username, password)) {**

**System.out.println("Login successful!");**

**} else {**

**System.out.println("Invalid  credentials.");**

**}}}**

**Output:**

****

**Errors:**

|  |  |
| --- | --- |
| **Errors** | **Rectification** |
| **“{“** | **“{“** |
| **“Class”** | **“class”** |

**Important points:**

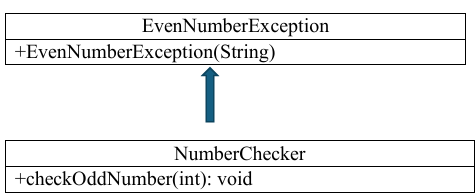
**Interface Login system: This interface defines a contract for any class that wants to implement a login system. It declares a single method user(String username, String password) which takes a username and password as input and returns a boolean indicating whether the login is successful. This promotes abstraction by defining what a login system should do, not how it should do it.**

**WEEK-09**

**Program-1**

**Aim: Write a java program to create a method that takes integers as parameter and throws n exception if the number is even**

**Class Diagram:**

****

**Code:**

**class EvenNumberException extends Exception {**

**public EvenNumberException(String message) {**

**super(message);}}**

**class NumberChecker {**

**public static void checkOddNumber(int number) throws EvenNumberException {**

**if (number % 2 == 0) {**

**throw new EvenNumberException("Even number not allowed: " + number); } else {**

**System.out.println("Valid odd number: " + number);**

**}}**

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

**int[] testNumbers = {1, 2, 3, 4, 5, 6};**

**for (int num : testNumbers) {**

**try {**

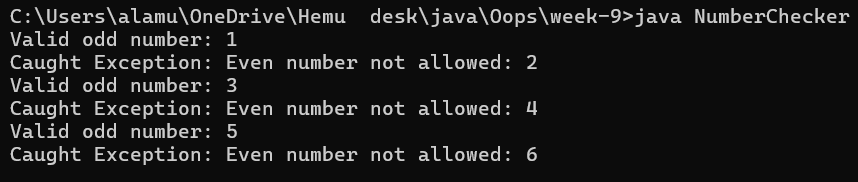
**checkOddNumber(num);**

**} catch (EvenNumberException e) {**

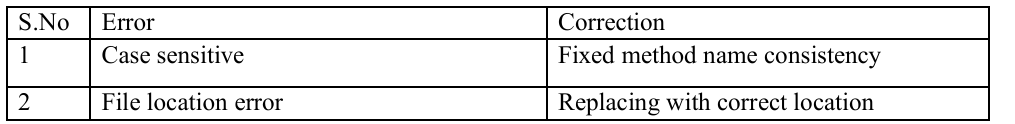
**System.out.println("Caught Exception: " + e.getMessage());**

**}}}}**

**OUTPUT:**

****

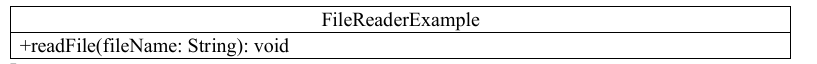
**ERRORS:**

****

**PROGRAM-02**

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

**Class Diagram:**

****

**Code:**

**import java.io.\*;**

**public class FileReaderExample {**

**public static void readFile(String fileName) throws FileNotFoundException {**

**File file = new File(fileName);**

**if (!file.exists()) {**

**throw new FileNotFoundException("File not found: " + fileName);**

**}**

**try (BufferedReader br = new BufferedReader(new FileReader(file))) {**

**String line;**

**System.out.println("Contents of the file:");**

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

**System.out.println(line);**

**}**

**} catch (IOException e) {**

**System.out.println("An error occurred while reading the file: " + e.getMessage());**

**}**

**}**

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

**String fileName = "example.txt"; // Change this to a valid file path**

**try {**

**readFile(fileName);**

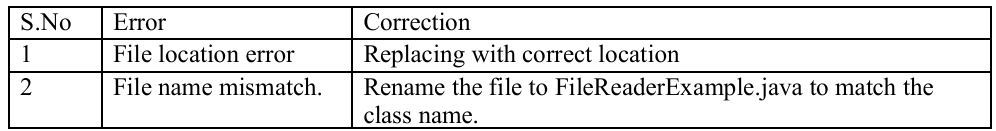
**} catch (FileNotFoundException e) {**

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

**}}}**

**OUTPUT:**

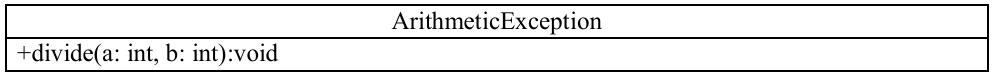
**ERRORS:**

****

**PROGRAM-03**

**AIM: Write a java program to handle arithmetic exception using try, catch, finally.**

**Class Diagram:**

****

**Code:**

**class ArithmeticExceptionExample {**

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

**int a = 10;**

**int b = 0;**

**try {**

**int result = a / b;**

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

**} catch (ArithmeticException e) {**

**System.out.println("Caught ArithmeticException: " + e.getMessage());**

**} finally {**

**System.out.println("Finally block executed.");**

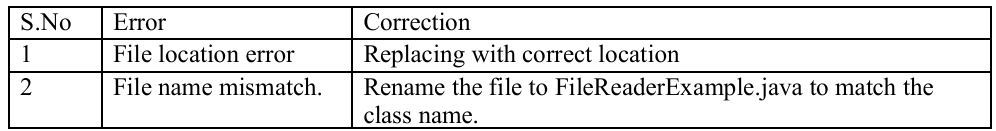
**}**

**System.out.println("Program continues after exception handling.");**

**}**

**}**

**OUTPUT:**

**ERRORS: **

**PROGRAM-04**

**Aim: Write a java program to stimulate a University System using inner classes.**

**• Create an outer class named University with a variable University Name**

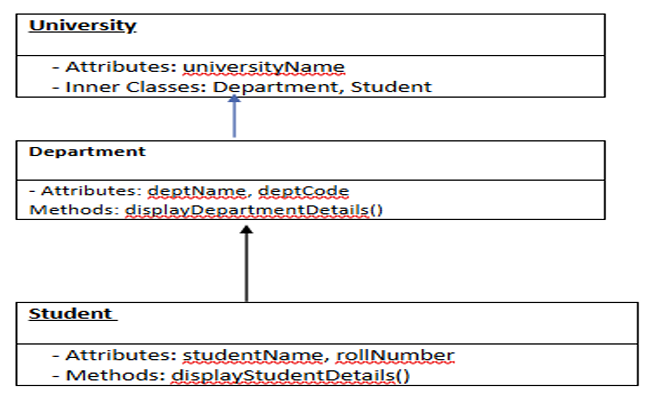
**• Inside it, define two non-static inner classes**

**1)Department – With variables like dept Name and dept Code, and a method to display department details**

**2) Student- with variables like Student name, student roll no and method to display student details**

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

**Class Diagram:**

****

**Code:**

**class University {**

**private String universityName;**

**public University(String universityName) {**

**this.universityName = universityName;**

**}**

**public class Department {**

**private String deptName;**

**private String deptCode;**

**public Department(String deptName, String deptCode) {**

**this.deptName = deptName;**

**this.deptCode = deptCode;**

**}**

**public void displayDepartmentDetails() {**

**System.out.println("University: " + universityName);**

**System.out.println("Department Name: " + deptName);**

**System.out.println("Department Code: " + deptCode);**

**}**

**}**

**public class Student {**

**private String studentName;**

**private String rollNo;**

**public Student(String studentName, String rollNo) {**

**this.studentName = studentName;**

**this.rollNo = rollNo;**

**}**

**public void displayStudentDetails() {**

**System.out.println("University: " + universityName);**

**System.out.println("Student Name: " + studentName);**

**System.out.println("Roll No: " + rollNo); } }**

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

**University university = new University("Amrita Viswa Vidhapeetham University");**

**University.Department department = university.new Department("Computer Science", "23CSE111");**

**University.Student student = university.new Student("Hemu", "AV.SC.U4CSE24004");**

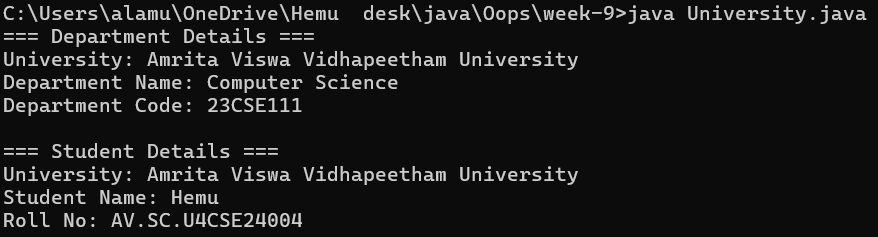
**System.out.println("=== Department Details ===");**

**department.displayDepartmentDetails();**

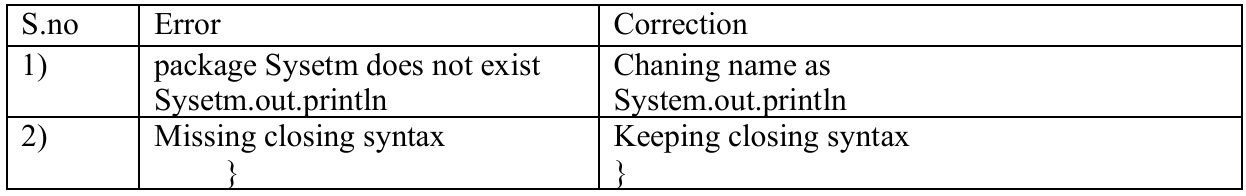
**System.out.println("\n=== Student Details ===");**

**student.displayStudentDetails();}}**

**OUTPUT:**

****

**ERRORS:**

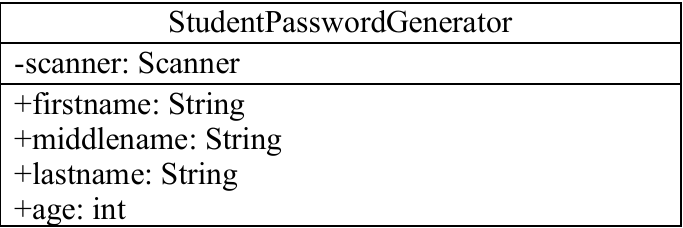
****

**WEEK-10**

**PROGRAM-01**

**AIM: write a java program to generate a password for a student using his/her initials and age. The password displayed should the string consists of first character, of first name, middle name, last name, with age.**

**CLASS DIAGRAM:**

****

**CODE:**

**import java.util.Scanner;**

**public class StudentPasswordGenerator {**

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

**Scanner scanner = new Scanner(System.in);**

**System.out.print("Enter First Name: ");**

**String firstName = scanner.nextLine();**

**System.out.print("Enter Middle Name: ");**

**String middleName = scanner.nextLine();**

**System.out.print("Enter Last Name: ");**

**String lastName = scanner.nextLine();**

**System.out.print("Enter Age: ");**

**int age = scanner.nextInt();**

**char firstInitial = firstName.charAt(0);**

**char middleInitial = middleName.charAt(0);**

**char lastInitial = lastName.charAt(0);**

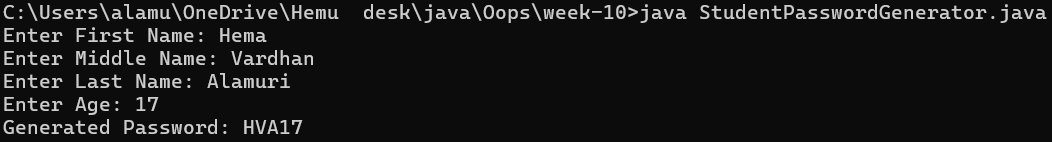
**String password = "" + firstInitial + middleInitial + lastInitial + age;**

**System.out.println("Generated Password: " + password);**

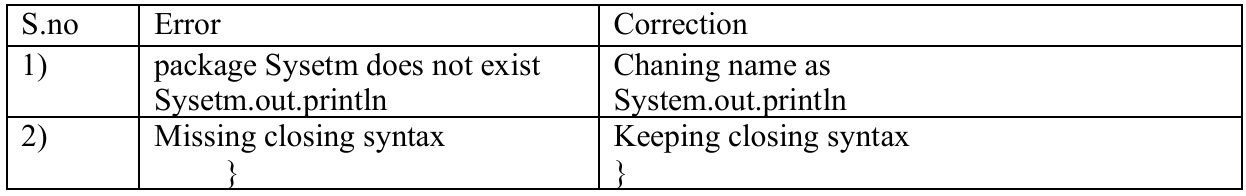
**}**

**}**

**OUTPUT:**

****

**ERRORS:**

****

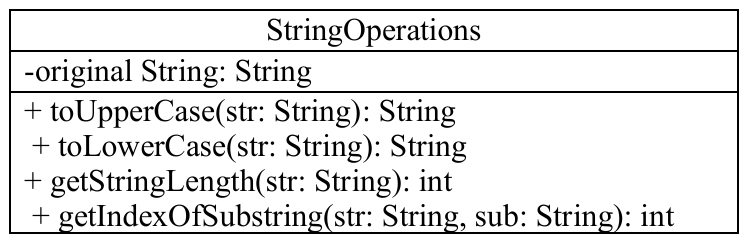
**PROGRAM-02**

**AIM: design and implement a java program that will do the following operations to this string “Welcome! You are practicing strings concepts”. a)covert all alphabets to capital letters and print out the result.**

**b)convert all alphabets to lower-case letters and print out the result.**

**c)print out the length of the string d)print out the index of concept.**

**CLASS DIAGRAM:**

****

**CODE:**

**class StringOperations {**

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

**String text = "Welcome! You are practicing strings concepts";**

**String upper = text.toUpperCase();**

**System.out.println("Uppercase : " + upper);**

**String lower = text.toLowerCase();**

**System.out.println("Lowercase : " + lower);**

**int length = text.length();**

**System.out.println("Length of the string: " + length);**

**int index = text.indexOf("concept");**

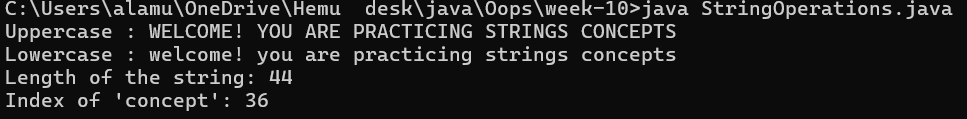
**if (index != -1) {**

**System.out.println("Index of 'concept': " + index);**

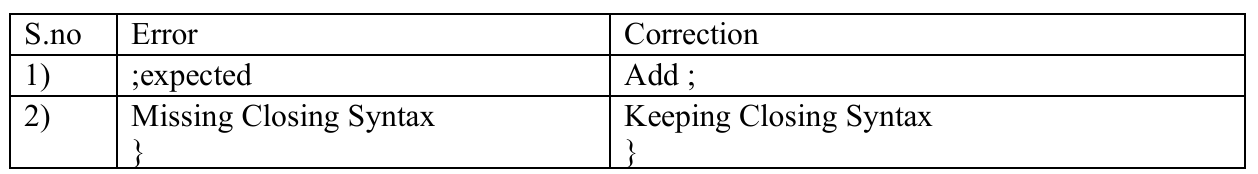
**} else {**

**System.out.println("'concept' not found in the string.");}}}**

**OUTPUT:**

****

**ERRORS:**

****

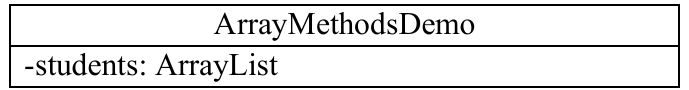
**PROGRAM-03**

**AIM: Implement a java program using the below array methods**

**a) sorting the elements of an array.**

**b) convert the elements into string c)fill the part of the array d)copy the elements of one array into another.**

**CLASS DIAGRAM:**

****

**CODE:**

**import java.util.Arrays;**

**public class ArrayMethodsDemo {**

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

**int[] originalArray = {1, 3, 5, 7, 9, 11, 13, 15};**

**Arrays.sort(originalArray);**

**System.out.println("Sorted array: " + Arrays.toString(originalArray));**

**String arrayAsString = Arrays.toString(originalArray);**

**System.out.println("Array as string: " + arrayAsString);**

**Arrays.fill(originalArray, 1, 4, 0);  // from index 1 (inclusive) to 4 (exclusive)**

**System.out.println("Array after partial fill (index 1-3 with 0): " + Arrays.toString(originalArray));**

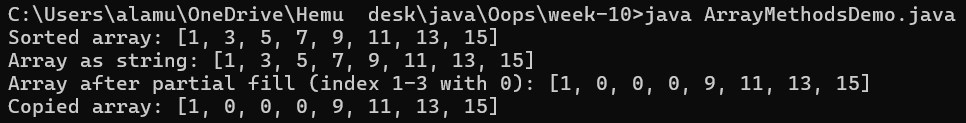
**int[] copiedArray = Arrays.copyOf(originalArray, originalArray.length);**

**System.out.println("Copied array: " + Arrays.toString(copiedArray));**

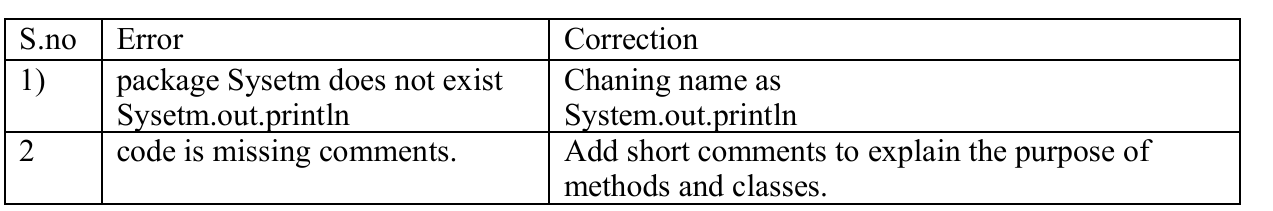
**}**

**}**

**OUTPUT:**

****

**ERRORS:**

****

**PROGRAM-04**

**AIM: Aim: Implement a java program using below ArrayList methods**

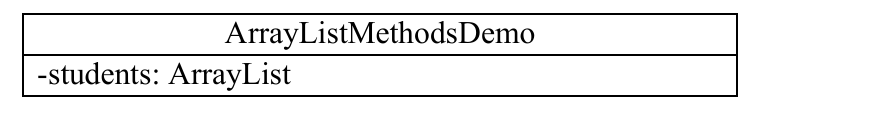
**- Insert an element at particular index in the array list**

**a) modify an element in the array list**

**b)Access an element from the array list**

**c)Remove an element from the array list d)clear the elements from the array list**

**CLASS DIAGRAM:**

****

**CODE:**

**import java.util.ArrayList;**

**public class ArrayListMethodsDemo {**

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

**ArrayList<String> students = new ArrayList<>();**

**students.add("classic");**

**students.add("Hemu");**

**students.add("Siva");**

**students.add("Manikanta");**

**System.out.println("Original List: " + students);**

**students.add(2, "Ramu");**

**System.out.println("After inserting 'Ramu' at index 2: " + students);**

**students.set(1, "Divi");**

**System.out.println("After modifying index 1 to 'Divi': " + students);**

**String studentAt3 = students.get(3);**

**System.out.println("Element at index 3: " + studentAt3);**

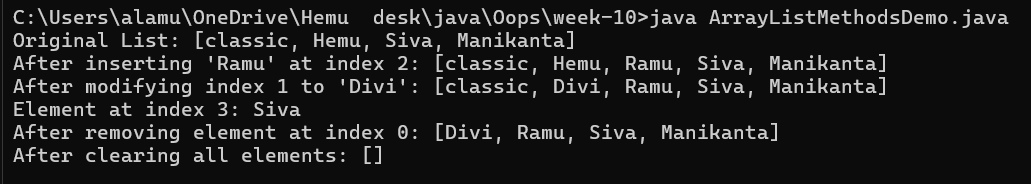
**students.remove(0);**

**System.out.println("After removing element at index 0: " + students);**

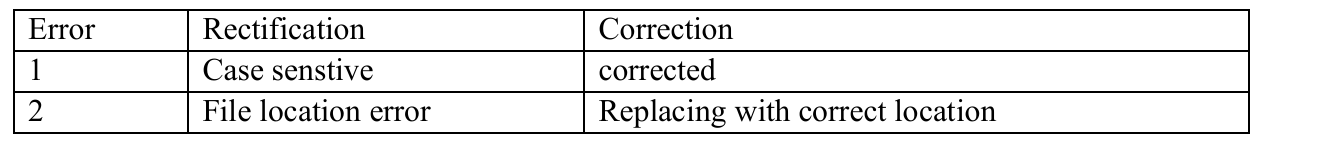
**students.clear();**

**System.out.println("After clearing all elements: " + students);}}**

**OUTPUT:**

****

**ERRORS:**

****