(23CSE111)

object-Oriented Programing

LAB MANUAL

CSE-1st YEAR 2nd SEMESTER (2025)

SUBMITTED From: SUBMITTED To:

NAME	G. Balaji Ajay	**NAME**	**Dr.Raj Kumar Batchu**
ROLL NO	**AV.SC.U4CSE24124**	**DEPARTMAENT**	**Ogject-Oriented_programing**
SECTION	**CSE-B**	**DESIGNATION**	**PROFESSOR**

MARKS	
DATE	
SIGNATURE	

INDEX

	constructor for my class that increments the count variables each time an object of my class is created variable each time an object of myclass is created. Finally print the final values of count and pie variables.			
week5.	1. Create a calculator using the operations including addition, subtraction, multiplication and division using multilevel inheritance and display the desired output. 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 and bikes and they need a program to store details about each variable such as brand and speed. • Cars should have an additional property: numbers of doors, seating capacity. • 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 & indicates when a vehicle is starting. • Every class should have 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)		12/3/2025	

	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 & bike sub classed. Finally display its details			
week6.	Write a Java program to create a vehicle class with a method displayInfo(). Override this method in the car subclass to provide specific information about a car, model,fuel type, and colour using the constructor Write a Java program to create a vehicle class with a method displayInfo(). Override this method in the car subclass to provide specific information about a car, model,fuel type, and colour using the constructor 1. Create a Java program for the scenario. A college is developing an automated admission system that verifies student eligibility for undergraduate (UG) and postgraduate(PG) programs. Each program has diterent eligibility criteria based on the student's percentage in their previous qualification. i) UG admissions require a minimum of 60%		26/3/2025	

	ii) PG admissions require a minimum of 70% Create a Java program for the scenario. A college is developing an automated admission system that verifies student eligibility for undergraduate (UG) and postgraduate(PG) programs. Each program has diterent eligibility criteria based on the student's percentage in their previous qualification. i) UG admissions require a minimum of 60% ii) PG admissions require a minimum of 70% 3. Write a Java Program to create a Calculator class with overloaded methods to perform addition: Take the integer values a and b from the user. i) Add two integers ii) Add two doubles iii) Add three integer 4. Write a Java Program to create a shape class with a method calculateArea() that is 5. overloaded for diterent shapes(e.g., Square, Rectangle). Then create a subclass Circle that overrides the calculateArea() method for a circle.			

week7.	1. .Write java Program to create an Abstract class Animal with an abstract method called sound().Create subclass lion and tiger that extends the animal class and implements the sound() method to make a specific sound for each animal. 2. write a java program to create an abstract class Shape3d with abstract method calculateVolume() and Calculate SurfaceArea() Create subclasses Sphere and Cube that extends the shape3d class and implements the respective method to calculate the volume and surface area of type shape 3. Write a Java program using an abstract class to define a method for printing patterns.		2/4/2025	

	1. Create an abstract class named PatternPrinter with: An abstract method printPattern(int n) A concrete method to display the pattern title			

	2. Implement 2 subclasses: Subclass 1 (Star Pattern): Prints a right-angled triangle of stars (*) Subclass 2 (Number Pattern): Prints a right-angled triangle of numbers In the main method, create objects of both subclasses and print the pattern for a given number of rows.			
week8.	1. Write a java program to create an interface shape with getPerimeter() method. Create three classes Rectangle, Circle, and Triangle that implement the shape interface implement the getPerimeter() method for each of the three classes.		9/4/2025	

	2. Write a java program to create an interface playable with a method play that takes no arguments and returns play that takes no arguments and returns void. Create 3 classes Football,			
	Volleyball, Basketball that implements the playable interface and override the play method to play the respective sports.			
week9.	1. Write a java program to create a method that takes integer as parameter and throws an exception if the number is even. 2. Write a java program to create a method that reads a file and throws an exception if the file is not found 3. Write a java program to handle arithematic exception using try catch and finally import java.util.Scanner; 4. Write a java program to simulate a university system using inner classes ✓ Create an outer class namedd University with a variable UniversityName ✓ Inside it defgine two non-static in classes 1. Department- With variable like deptName and deptCode and a method to display department details. 2. Student-Variable like stdName and stdCode and a method to display Student details. 3. Create an object for each class and call their methods to display their		16/4/2025	
	details and with the university name.			

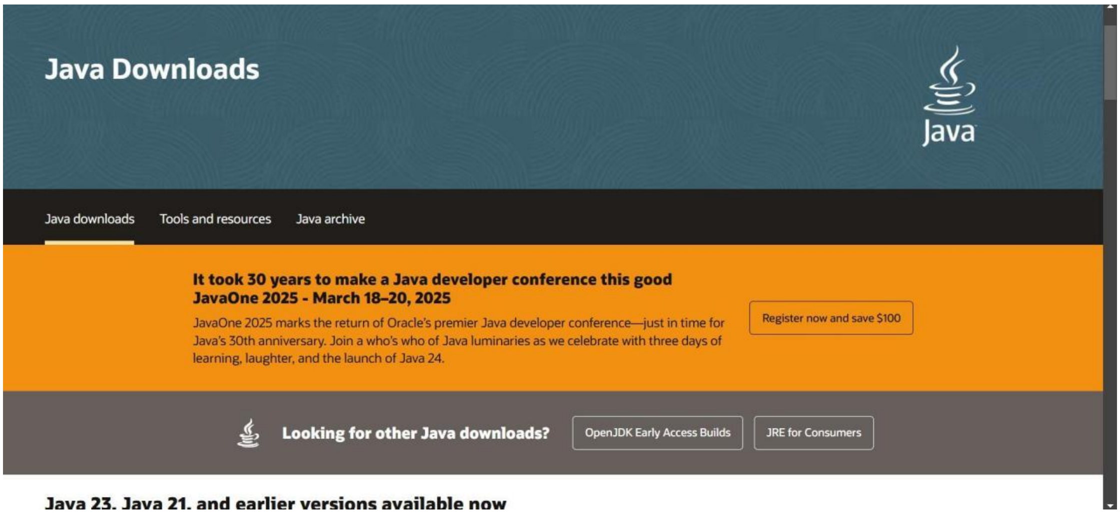
week10.	1. Write a java program to generate a password for a student using his/her initials and age. The password displayed should the string consists of first displayed should the string consists of first character of firstname , middlename, lastname with age. 2.Design and implement a Java program that will do the following operations to this string, "Welcome! You are practicing Strings concept" - Convert all alphabets to capital letters and printout the result - Convert all alphabets to lowercase lettes & printout the result - Printout the length of the String - Print out the index of Course. 3. Implement a java progam using the below array methods - Sorting the elements (numbers & Strings) of an array.		30/4/2025	

	- Convert the array elements into string. - Fill then part of an array. - Copy the elements of one array into another. 4. Implement a java program using the below Array List methods - Insert an element at particular Index in the array list. - Modify the element in the arraylist. - Remove an element from the arraylist. - Clear the elements from the arraylist.			

WEEK-1:

Aim: How to install jdk and first program on   printing student details.

Step-1: Download JDK-21 from oracle website



Java Downloads

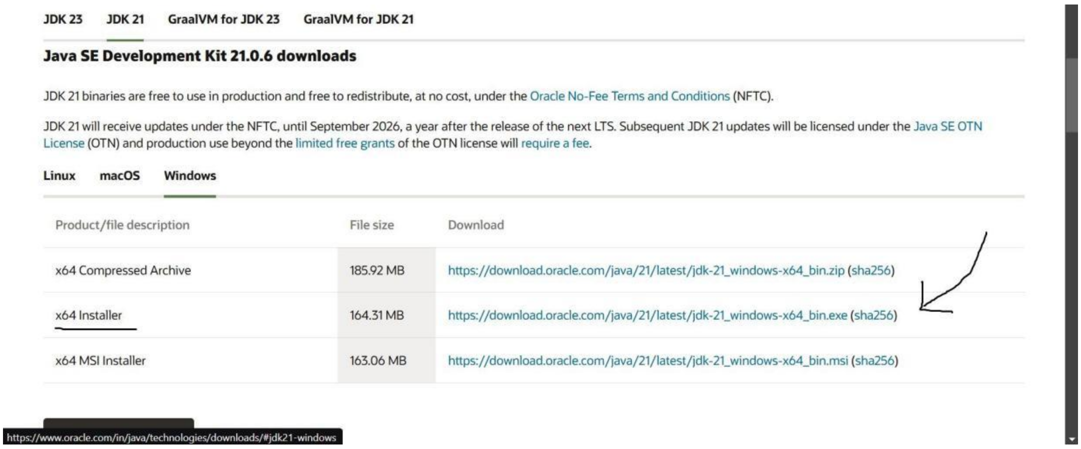
It took 30 years to make a Java developer conference this good
JavaOne 2025 - March 18–20, 2025

JavaOne 2025 marks the return of Oracle's premier Java developer conference—just in time for Java's 30th anniversary. Join a who's who of Java luminaries as we celebrate with three days of learning, laughter, and the launch of Java 24.

Looking for other Java downloads?

Java 23, Java 21, and earlier versions available now

Step-2: Install the JDK-21 with accepting terms and conditions according to the respective windows.



Java SE Development Kit 21.0.6 downloads

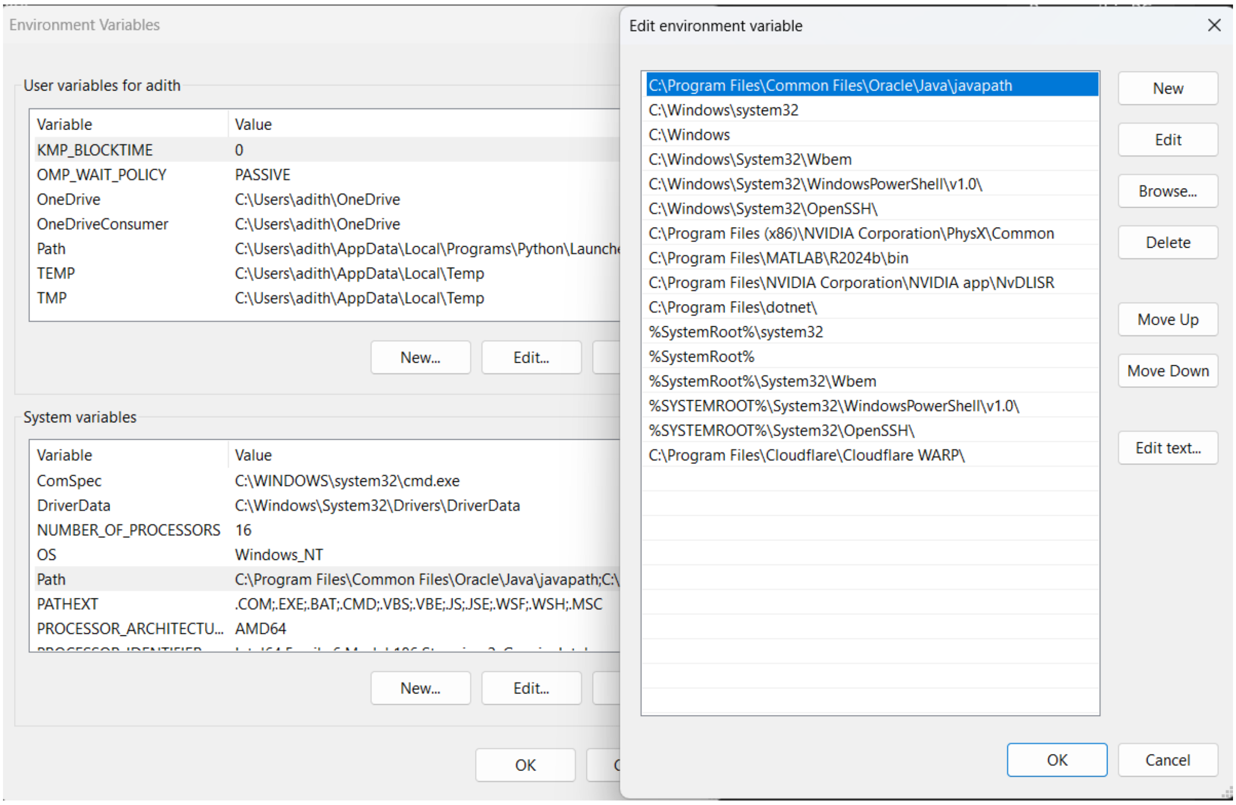
JDK 21 binaries are free to use in production and free to redistribute, at no cost, under the Oracle No-Fee Terms and Conditions (NFTC).

JDK 21 will receive updates under the NFTC, until September 2026, a year after the release of the next LTS. Subsequent JDK 21 updates will be licensed under the Java SE OTN License (OTN) and production use beyond the limited free grants of the OTN license will require a fee.

Product/file description	File size	Download
x64 Compressed Archive	185.92 MB	https://download.oracle.com/java/21/latest/jdk-21_windows-x64_bin.zip (sha256)
x64 Installer	164.31 MB	https://download.oracle.com/java/21/latest/jdk-21_windows-x64_bin.exe (sha256)
x64 MSI Installer	163.06 MB	https://download.oracle.com/java/21/latest/jdk-21_windows-x64_bin.msi (sha256)

https://www.oracle.com/in/java/technologies/downloads/#jdk21-windows

Step-3:Setting up environmental variables.



User variables for adith

Variable	Value
KMP_BLOCKTIME	0
OMP_WAIT_POLICY	PASSIVE
OneDrive	C:\Users\adith\OneDrive
OneDriveConsumer	C:\Users\adith\OneDrive
Path	C:\Users\adith\AppData\Local\Programs\Python\Launche
TEMP	C:\Users\adith\AppData\Local\Temp
TMP	C:\Users\adith\AppData\Local\Temp

System variables

Variable	Value
ComSpec	C:\WINDOWS\system32\cmd.exe
DriverData	C:\Windows\System32\Drivers\DriverData
NUMBER_OF_PROCESSORS	16
OS	Windows_NT
Path	C:\Program Files\Common Files\Oracle\Java\javapath;C:\
PATHEXT	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC
PROCESSOR_ARCHITECTU...	AMD64

```
C:\Program Files\Common Files\Oracle\Java\javapath
C:\Windows\system32
C:\Windows
C:\Windows\System32\Wbem
C:\Windows\System32\WindowsPowerShell\v1.0\
C:\Windows\System32\OpenSSH\
C:\Program Files (x86)\NVIDIA Corporation\PhysX\Common
C:\Program Files\MATLAB\R2024b\bin
C:\Program Files\NVIDIA Corporation\NVIDIA app\NvDLISR
C:\Program Files\dotnet\
%SystemRoot%\system32
%SystemRoot%
%SystemRoot%\System32\Wbem
%SYSTEMROOT%\System32\WindowsPowerShell\v1.0\
%SYSTEMROOT%\System32\OpenSSH\
C:\Program Files\Cloudflare\Cloudflare WARP\
```

*Windows c -> C-drive -> program files ->Java ->JDK-21->select bin

*Select and open environmental variable in search bar-> either select system variables or user variables-> select path-> click edit->New-> paste the bin-> finish the setup(apply the changes).

~for verifying the installed version

Open cmd-> type java --version

~command propt

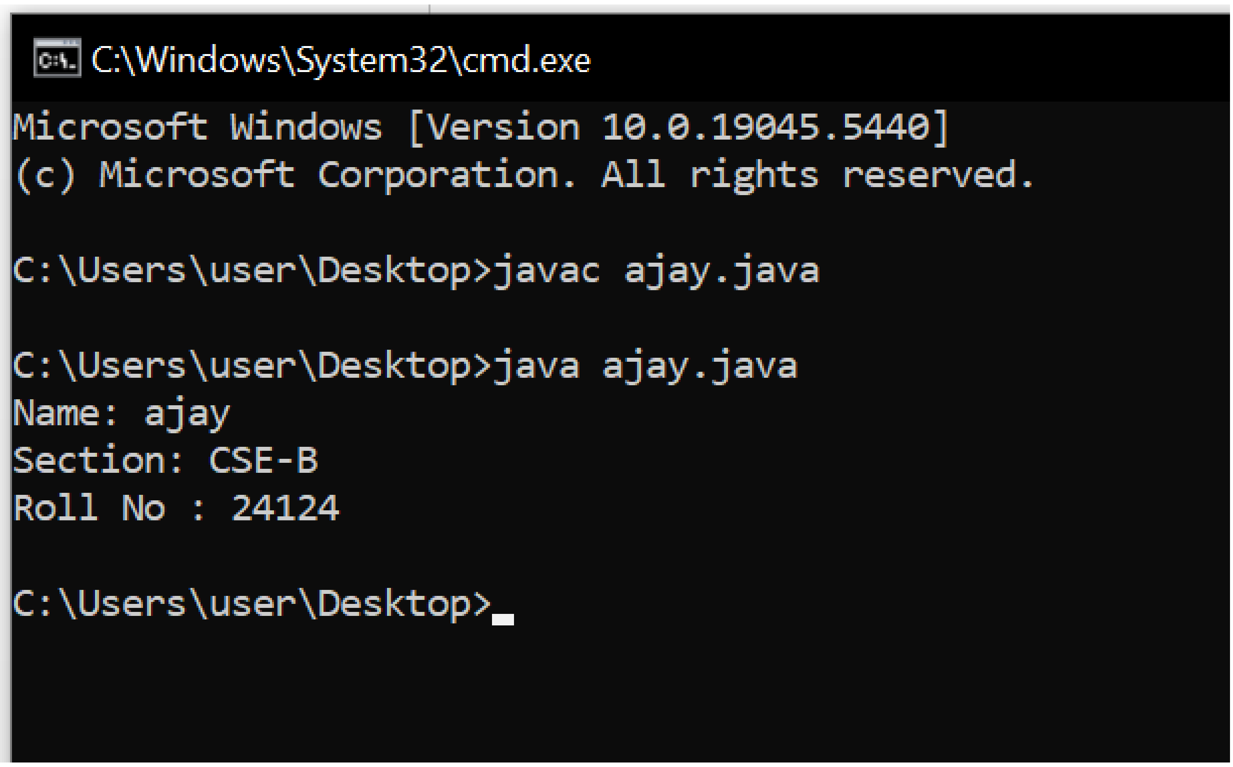
Javac filename.java ->compiling.

Java filename.java ->displaying

PROGRAM-1(Rectified):

```
class Main
{
  public static void main(String[] args)
  {
    System.out.println("Name: ajay");
    System.out.println("Section: CSE-B");
    System.out.println("Roll No : 24124");
  }
}
```

Output:



```
Microsoft Windows [Version 10.0.19045.5440]
(c) Microsoft Corporation. All rights reserved.

C:\Users\user\Desktop>javac ajay.java

C:\Users\user\Desktop>java ajay.java
Name: ajay
Section: CSE-B
Roll No : 24124

C:\Users\user\Desktop>
```

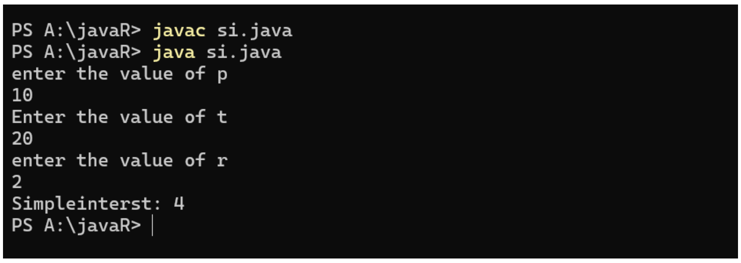
WEEK-2:

PROGRAM-1:

Aim: Write a java program for SI

```
import java.util.Scanner;
public class si {
    public static void main(String[] args) {
        int p, t, r, Simpleinterest;
        Scanner num = new Scanner(System.in);
        System.out.println("enter the value of p");
        p = num.nextInt();
        System.out.println("Enter the value of t");
        t = num.nextInt();
        System.out.println("enter the value of r");
        r = num.nextInt();
       Simpleinterest = (p * t * r) / 100;
        System.out.println("Simpleinterest: " + Simpleinterest);
    }
}
```

Output:



```
PS A:\javaR> javac si.java
PS A:\javaR> java si.java
enter the value of p
10
Enter the value of t
20
enter the value of r
2
Simpleinterst: 4
PS A:\javaR> |
```

ERROR TABLE*:*

Code Error	**Code rectification**
1.Giving space between next and Double. 2.Not giving parenthesis after closing the input.	1.Should not give space between next and Double. 2.We must put parenthesis after closing the input

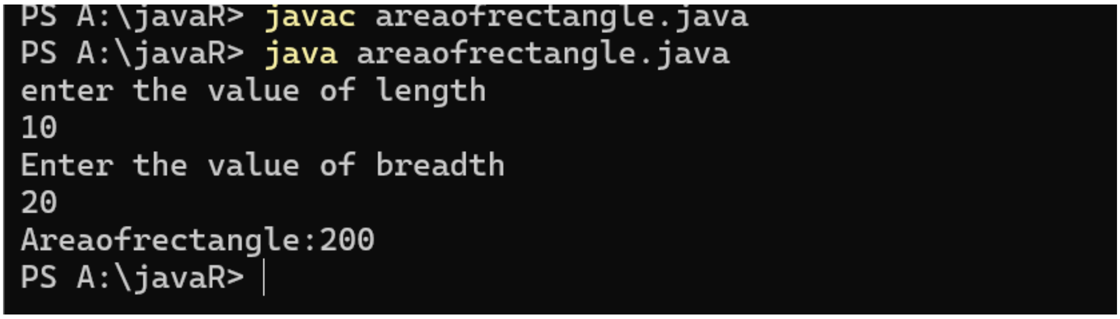
Important Points:

1.Usethecorrectformula:
Simple Interest = (Principal × Rate × Time) / 100
2.Use double type:
To handle decimal values for principal, rate, and interest.
3.Input values clearly:
Principal (P), Rate (R), and Time (T) must be entered correctly in the right units (usually in years for T and annual rate for R).

PROGRAM-2:
Aim: Write a program in java for area of rectangle.

```
Cimport java.util.Scanner;

public class areaofrectangle {
    public static void main(String[] args) {
        int l, b, Areaofrectangle;
        Scanner measurement = new Scanner(System.in);
        System.out.println("enter the value of length");
        l = measurement.nextInt();
        System.out.println("Enter the value of breadth");
        b = measurement.nextInt();
        Areaofrectangle = l * b;
        System.out.println("Areaofrectangle: " + Areaofrectangle);
    }
}
```

Output:



```
PS A:\javaR> javac areaofrectangle.java
PS A:\javaR> java areaofrectangle.java
enter the value of length
10
Enter the value of breadth
20
Areaofrectangle:200
PS A:\javaR>
```

ERROR TABLE:

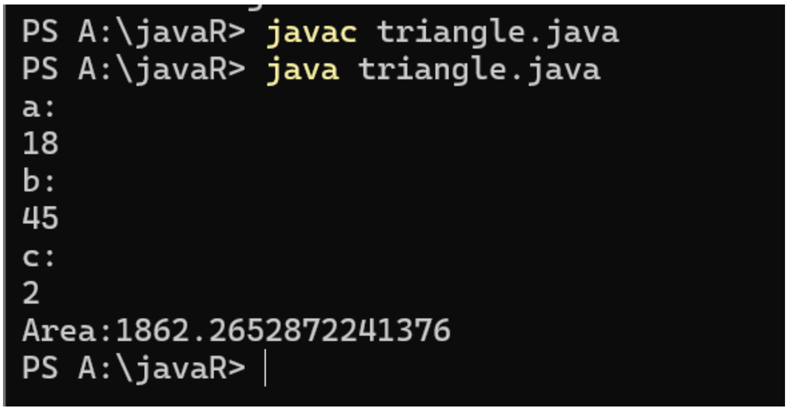
Code Error	Code rectification
1.While using for iteration, not giving the conditions correctly. 2.Declaring the data type as double instead of int.	1.We should give iterative statements correctly. 2.We should give the data type as int for integers.

PROGRAM-3:

Aim: Write a program in java for area oftriangle using heron's formula.

```
import java.util.Scanner;

public class triangle {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        System.out.println("a: ");
        double a = input.nextDouble();
        System.out.println("b: ");
        double b = input.nextDouble();
        System.out.println("c: ");
        double c = input.nextDouble();
        double s = (a + b + c) / 2;
        double area = Math.sqrt(s * (s - a) * (s - b) * (s - c));
        System.out.println("Area: " + area);
        input.close();
    }
}
```

Output:



```
PS A:\javaR> javac triangle.java
PS A:\javaR> java triangle.java
a:
18
b:
45
c:
2
Area:1862.2652872241376
PS A:\javaR>
```

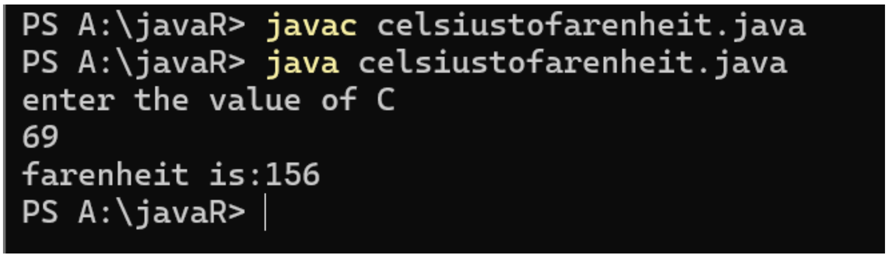
ERROR TABLE:

Code Error	Code rectification
1.While printing the variable not giving + sign. 2.Not closing the scanner.	1.We should give correct indentation. 2.Closing the scanner is must.

PROGRAM-4(a):
Aim: Write a program in java for converting temperature from celsius to fahrenite.

```
import java.util.Scanner;
public class celsiustofarenheit {
   public static void main(String[] args) {
      int C, F;
      Scanner num = new Scanner(System.in);
      System.out.println("enter the value of C");
      C = num.nextInt();
      F = (C * 9 / 5) + 32;
      System.out.println("farenheit is: " + F);
   }
}
```

OUTPUT:

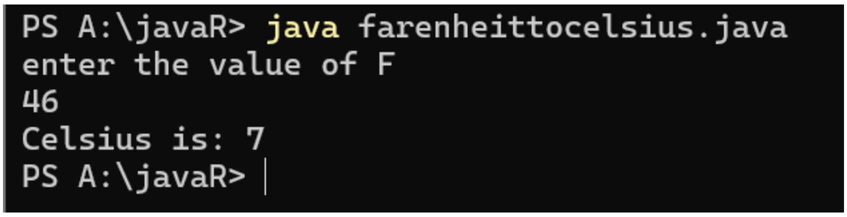


```
PS A:\javaR> javac celsiustofarenheit.java
PS A:\javaR> java celsiustofarenheit.java
enter the value of C
69
farenheit is:156
PS A:\javaR>
```

PROGRAM-4(b):
Aim: Write a program in java for converting temperature from fahrenite to celsius.

```
import java.util.Scanner;

public class farenheittocelsius {
    public static void main(String[] args) {
        int C, F;
        Scanner num = new Scanner(System.in);

        System.out.println("enter the value of F");
        F = num.nextInt();

        C = (F - 32) * 5 / 9;
        System.out.println("Celsius is: " + C);
    }
}
```

Output:



```
PS A:\javaR> java farenheittocelsius.java
enter the value of F
46
Celsius is: 7
PS A:\javaR>
```

ERROR TABLE:

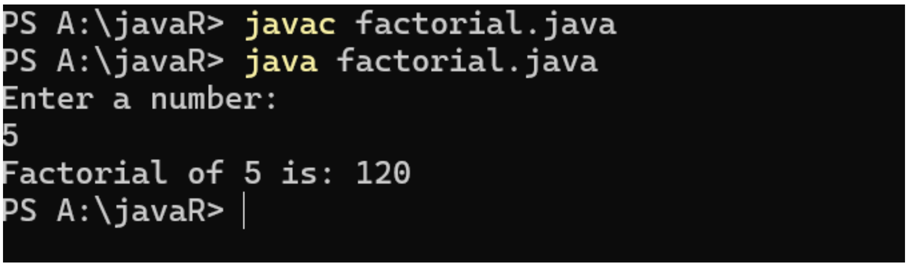
Code Error	**Code rectification**
1.While printing the variable not giving + sign. 2.Not closing the scanner.	1.We should give correct indentation. 2.Closing the scanner is must.

PROGRAM-5:

Aim: Write a program in java for factorial of a number.

```
import java.util.Scanner;
public class Factorial {
   public static void main(String[] args) {
      int n, factorial = 1;
      Scanner num = new Scanner(System.in);
      System.out.println("Enter a number:");
      n = num.nextInt();
      for (int i = 1; i <= n; i++) {
         factorial *= i;
      }

      System.out.println("Factorial of " + n + " is: " + factorial);
   }
}
```

OUTPUT:



```
PS A:\javaR> javac factorial.java
PS A:\javaR> java factorial.java
Enter a number:
5
Factorial of 5 is: 120
PS A:\javaR> |
```

ERROR TABLE:

Code Error	**Code rectification**
1.While using for iteration, not giving the conditions correctly. 2.Declaring the data type as double instead of int.	1.We should give iterative statements correctly. 2.We should give the data type as int for integers.

Important Points:

1.**Use a loop to avoid repetitive code** – A for loop is ideal for calculating factorials from 1 to n efficiently.

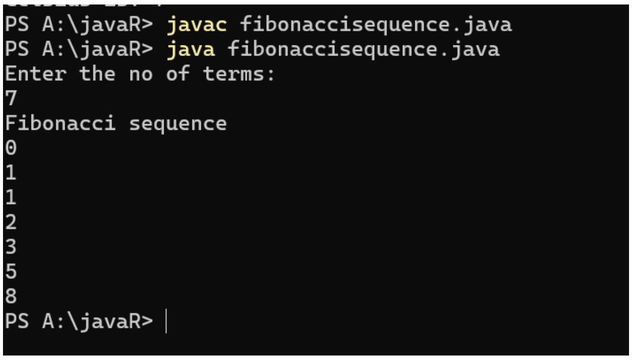
2.**Use long or BigInteger** – Factorials grow quickly, so use long for moderate values and BigInteger for very large ones.

3.**Build each factorial from the previous one** – Instead of recalculating from scratch, compute n! as (n-1)! × n to save time.

PROGRAM-6:
Aim: Write a program in java for fibonacci series.

```
import java.util.Scanner;

public class fibonaccisequence {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);

      System.out.println("Enter the no of terms:");
      int n = scanner.nextInt();

      int a = 0, b = 1;
      System.out.println("Fibonacci sequence");

      for (int i = 1; i <= n; i++) {
         System.out.println(a + " ");
         int nextTerm = a + b;
         a = b;
         b = nextTerm;
      }

      scanner.close();
   }
}
```

OUTPUT:



```
PS A:\javaR> javac fibonaccisequence.java
PS A:\javaR> java fibonaccisequence.java
Enter the no of terms:
7
Fibonacci sequence
0
1
1
2
3
5
8
PS A:\javaR>
```

ERROR TABLE:

Code Error	**Code rectification**
1.Giving space between next and Double. 2.Not giving parenthesis after closing the input.	1.Should not give space between next and Double. 2.We must put parenthesis after closing the input.

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 these methods named start(),stop(),service()
4.Create the objects named car, car1,car2
Class diagram:

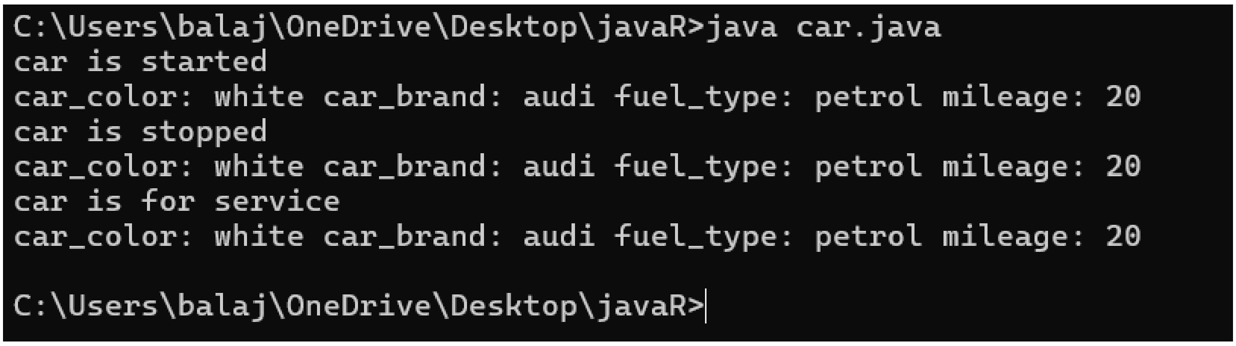
```
      car
----------------------


car_color:string
car_brand:string

fuel_type:string
milage:double
----------------------


+start():void
+stop():void
+service():void
```

CODE:

```
class Car {
   private String car_color;
   private String car_brand;
   private String fuel_type;
   private String mileage;
   public void start() {
      System.out.println("car is started");
   }

   public void stop() {
      System.out.println("car is stopped");
   }

   public void service() {
      System.out.println("car is for service");
   }
```

```
 public static void main(String args[]) {
       Car car = new Car();
       car.car_color = "white";
       car.car_brand = "audi";
       car.fuel_type = "petrol";
       car.mileage = "20";
       car.start();
       System.out.println("car_color: " + car.car_color + " car_brand: " + car.car_brand +
                   " fuel_type: " + car.fuel_type + " mileage: " + car.mileage);

       Car car1 = new Car();
       car1.car_color = "white";
       car1.car_brand = "audi";
       car1.fuel_type = "petrol";
       car1.mileage = "20";
       car1.stop();
       System.out.println("car_color: " + car1.car_color + " car_brand: " + car1.car_brand +
                   " fuel_type: " + car1.fuel_type + " mileage: " + car1.mileage);

       Car car2 = new Car();
       car2.car_color = "white";
       car2.car_brand = "audi";
       car2.fuel_type = "petrol";
       car2.mileage = "20";
       car2.service();
       System.out.println("car_color: " + car2.car_color + " car_brand: " + car2.car_brand +
                   " fuel_type: " + car2.fuel_type + " mileage: " + car2.mileage);
   }
}
```

OUTPUT:



```
C:\Users\balaj\OneDrive\Desktop\javaR>java car.java
car is started
car_color: white car_brand: audi fuel_type: petrol mileage: 20
car is stopped
car_color: white car_brand: audi fuel_type: petrol mileage: 20
car is for service
car_color: white car_brand: audi fuel_type: petrol mileage: 20

C:\Users\balaj\OneDrive\Desktop\javaR>
```

Error table:

S.no	Error name	Cause of error	Rectification
1	Syntax Error	Missing '{'	'{' added
2	Compile time Error	Mispelled Variable call	Rectified with Correct variable name
3	Case sensitive error	Uppercase and lowercase	rectified

IMPORTANT POINTS:

1. Before calling the function we should write the method properly.
2. Here, the "public void start( )" indicates that we are writing a method to call the function.
3. When we call a certain method, the process inside it will be printed as an output of the code.

PROGRAM-2:

Aim: To create a class BankAccount with methods deposit() and withdraw() create two subclasses savingsaccount and checkingaccount override the withdraw method in each subclass to impose different withdrawal limits and fees

Class diagram:

BankAccount

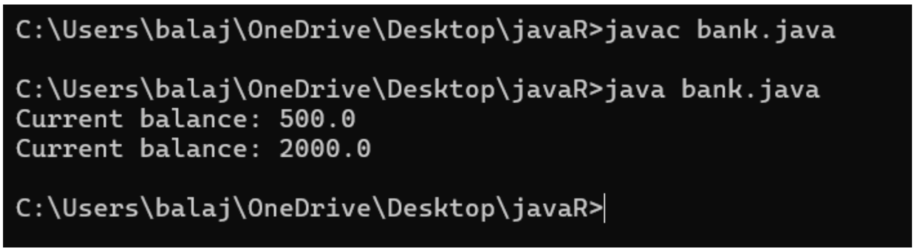
-balance: double

+BankAccount(intialBalance: double)
+deposit(amount: double):void
+withdraw(amount: double):void

CODE:

```
class BankAccount {
    protected String accountHolder;
    protected double balance;
    protected int accountNumber;

    public BankAccount(String accountHolder, int accountNumber, double balance) {
        this.accountHolder = accountHolder;
        this.accountNumber = accountNumber;
this.balance = balance;
    }

    public void withdrawal(double amount) {
        if (amount <= balance) {
            balance = balance - amount;
            System.out.println("Current balance: " + balance);
        } else {
            System.out.println("Insufficient funds");
        }
    }

    public void deposit(double amount) {
        balance = balance + amount;
        System.out.println("Current balance: " + balance);
    }

    public static void main(String[] args) {
        BankAccount BA = new BankAccount("Abdul", 24248, 1000);
        BA.withdrawal(500);
        BA.deposit(1500);
    }
}
```

OUTPUT:



```
C:\Users\balaj\OneDrive\Desktop\javaR>javac bank.java

C:\Users\balaj\OneDrive\Desktop\javaR>java bank.java
Current balance: 500.0
Current balance: 2000.0

C:\Users\balaj\OneDrive\Desktop\javaR>
```

Error table:

S.no	**Error name**	**Error name**	**Rectification**
1	**Name Error**	**Undefined name**	**Correct variable Name replaced**
2	**Syntax Error**	**Missing Parenthesis**	**Parenthesis Added**
3	**Logical Error**	**Incorrect Condition**	**Condition Rectified**

IMPORTANT POINTS:

1. The condition inside the if statement must be correct.
2. It explains that if the withdrawal money is less than the money in the bank account, then we can withdraw the amount.

Week-4

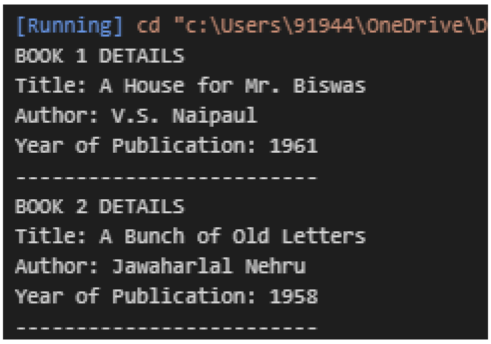
Program 1:

AIM : Write a java program with class named book the class should contain various attributes such as title ,author, year of publication .It should also contain a constructor with parameters with initializes title ,author and year of publication . create a method which displays the details of the book (display the details of two book i.e, create two books and objects with details).

Code:

```
class Book {
  String title;
  String author;
  int year;
  public Book(String title, String author, int year) {
    this.title = title;
    this.author = author;
    this.year = year;
  }
  public void details() {
    System.out.println("Title: " + title);
    System.out.println("Author: " + author);
    System.out.println("Year of Publication: " + year);
    System.out.println("-------------------------");

  public static void main(String[] args) {
    Book b1 = new Book("A House for Mr. Biswas", "V.S. Naipaul", 1961);
    Book b2 = new Book("A Bunch of Old Letters", "Jawaharlal Nehru", 1958);
    System.out.println("BOOK 1 DETAILS");
    b1.details();
    System.out.println("BOOK 2 DETAILS");
    b2.details();
  }
}
```

Output:



```
[Running] cd "c:\Users\91944\OneDrive\D
BOOK 1 DETAILS
Title: A House for Mr. Biswas
Author: V.S. Naipaul
Year of Publication: 1961
-------------------------
BOOK 2 DETAILS
Title: A Bunch of Old Letters
Author: Jawaharlal Nehru
Year of Publication: 1958
-------------------------
```

Error table:

S.No	Error type	Reason for error	Rectification
1	syntax error	Forgot to keep } At last	} is added
2	Logical error	Incorrect logic	Correct logic

Class diagram:

Book
title : String author : String year : int
+ book(title: String, author: String, year: int) + details(): void

IMPORTANT POINTS:

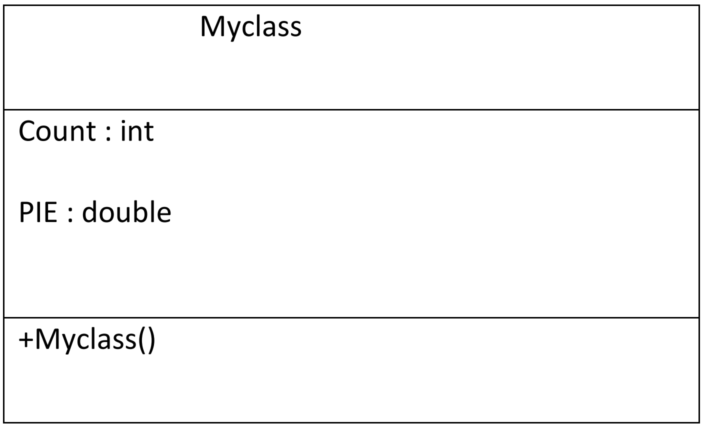
1. While defining two classes for a code, we must be sure that we save both the classes in separate files.
2. While defining a method we should also define a function to call that method.

Program 2:

AIM :

Write a java program to create a class named myclass with a static variable count of int type and initialize to zero and a constant variable pie of double data type ,initialize to 3.1415 as attributes of that class now define a constructor for my class that increments the count variables each time an object of my class is created variable each time an object of myclass is created. Finally print the final values of count and pie variables.

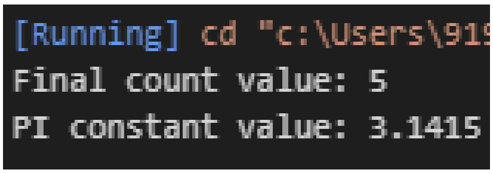
Class diagram:



Code:

```
class MyClass {
   static int count = 0;
   final double PI = 3.1415;

   public MyClass() {
      count++;
   }

   public static void main(String[] args) {
      MyClass obj1 = new MyClass();
      MyClass obj2 = new MyClass();
      MyClass obj3 = new MyClass();
      MyClass obj4 = new MyClass();
      MyClass obj5 = new MyClass();

      System.out.println("Final count value: " + count);
      System.out.println("PI constant value: " + obj1.PI);
   }
}
```

Output:



```
[Running] cd "c:\Users\91
Final count value: 5
PI constant value: 3.1415
```

Error table :

S.No	Error type	Reason for error	Rectification
1	syntax error	String forgot in main function	String is added
2	Logical error	Incorrect logic	Correct logic

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

1) Create a calculator using the operations including addition, subtraction, multiplication and division using multilevel in heritance and display the desired output.
Write your code in VS CODE and execute

Class Diagram:

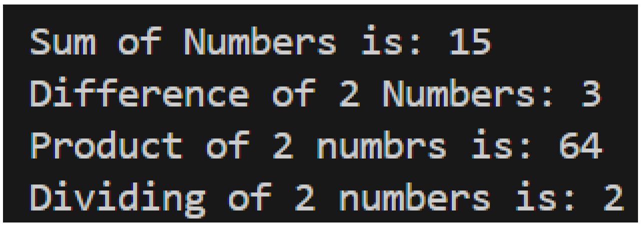
```
+-----------------------------+
|        Calculator        |
+-----------------------------+
| + add(double, double): void |
| + subs(double, double): void|
+-----------------------------+
          ▲
+-----------------------------+
|       Calculator1        |
+-----------------------------+
| + mul(double, double): void |
+-----------------------------+
          ▲
+-----------------------------+
|       Calculator2        |
+-----------------------------+
| + div(double, double): void |
+-----------------------------+
+-----------------------------+
|          Cals          |
+-----------------------------+
| + main(String[] args): void |
+-----------------------------+
```

CODE:

```
class easy {
   void add(int a, int b) {
      System.out.println("Sum of Numbers is: " + (a + b));
   }
   void subtract(int a, int b) {
      System.out.println("Difference of 2 Numbers: " + (a - b));
   }
}
class hard extends easy {
   void product(int a, int b) {
      System.out.println("Product of 2 numbers is: " + (a * b));
   }
}
```

```
class ultra extends hard {
   void divide(int a, int b) {
      if (b != 0) {
         System.out.println("Dividing of 2 numbers is: " + (a / b));
      } else {
         System.out.println("Denominator must not be zero");
      }
   }
}
class Calc {
   public static void main(String[] args) {
      ultra d = new ultra();
      d.add(6, 9);
      d.subtract(9, 6);
      d.product(23, 3);
      d.divide(4, 2);
   }
}
```

OUTPUT:



```
Sum of Numbers is: 15
Difference of 2 Numbers: 3
Product of 2 numbrs is: 64
Dividing of 2 numbers is: 2
```

Error table :

S.NO	Error Name	Error Rectification
1	Syntax/ Compilation Error	Absence of Semicolon
2	Closing Brackets	Need to Close the brackets
3	Class Name Error	Give the class name correctly
4	Constructor Calling	Call the constructor correctly

Important points:

In order to do this, we have to use inheritance concept. Here

we used the multi-inheritance concept.

2) 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.

i. Cars should have an additional property: number of doors, Seating capacity.

ii. Bikes should have a property indicating whether they have gears or not.

iii. The system should also include a function to display details about each vehicle and indicate when a vehicle is starting.

iv. Each class should have a constructor.

Questions:

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?

a. Truck should include and additional property capacity (in tons).

b. Create a showTruck() 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 subclasses. Finally display the details.

- Write your code in VS CODE and execute

- Important Points:

- **Understand the calling of a Constructor**
- **Giving class name correctly**

```
}
```

Class Diagram:

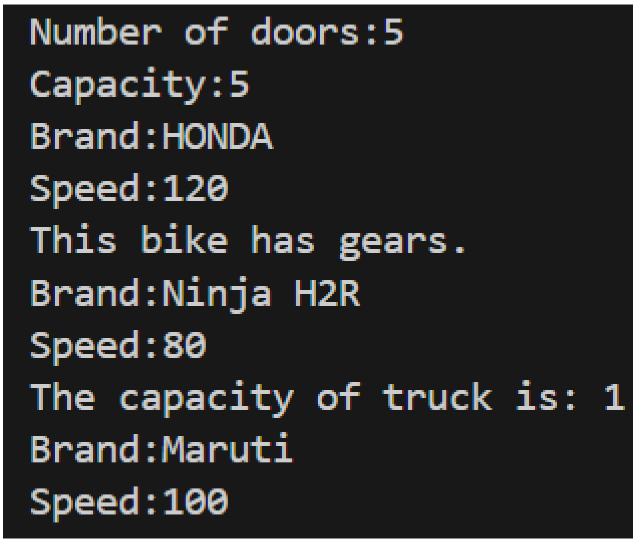
```
            +-----------------------------+
            |         Vehicle           |
            +-----------------------------+
            | - brand: String            |
            | - speed: int               |
            +-----------------------------+
            | + Details(): void          |
            +-----------------------------+
                ▲       ▲       ▲
   +-----------------+  +-----------------+  +-----------------+
   |     Cars       |  |     Bikes      |  |     Trucks     |
   +-----------------+  +-----------------+  +-----------------+
   | - doors: int    |  | - gears: boolean |  | - tons: int      |
   | - capacity: int  |  +-----------------+  +-----------------+
   | + cardetails()   |  | + bikedetails()  |  | + truckdetails() |
   +-----------------+  +-----------------+  +-----------------+
           +-----------------------------+
            |          Rent            |
            +-----------------------------+
            | + main(String[]): void      |
            +-----------------------------+

                                   -
```

CODE:

```
class Vehicle {
   String brand;
   int speed;

   Vehicle(String brand, int speed) {
      this.brand = brand;
      this.speed = speed;
   }

   void Details() {
        System.out.println("Brand: " + brand);
        System.out.println("Speed: " + speed);
    }
}


class CARS extends Vehicle {
    int doors;
    int capacity;

    public CARS(String brand, int speed, int doors, int capacity) {
        super(brand, speed);
        this.doors = doors;
        this.capacity = capacity;
    }

    void cardetails() {
        System.out.println("Number of doors: " + doors);
        System.out.println("Capacity: " + capacity);
    }
}

class Bikes extends Vehicle {
    Boolean gears;

    Bikes(String brand, int speed, Boolean gears) {
        super(brand, speed);
        this.gears = gears;
    }
    void bikedetails() {
        if (gears == true)
            System.out.println("This bike has gears.");
        else
            System.out.println("This bike does not have gear system.");
    }
}

class Vehicle {
    String brand;
    int speed;

    Vehicle(String brand, int speed) {
        this.brand = brand;
        this.speed = speed;
    }

    void Details() {
        System.out.println("Brand: " + brand);
        System.out.println("Speed: " + speed);
    }
}
class cars extends Vehicle {
    int doors;
    int capacity;

    public CARS(String brand, int speed, int doors, int capacity) {
        super(brand, speed);
        this.doors = doors;
        this.capacity = capacity;
```

```
  }

    void cardetails() {
        System.out.println("Number of doors: " + doors);
        System.out.println("Capacity: " + capacity);
    }
}

class Bikes extends Vehicle {

 Boolean gears;

    Bikes(String brand, int speed, Boolean gears) {
        super(brand, speed);
        this.gears = gears;
    }

    void bikedetails() {
        if (gears == true)
            System.out.println("This bike has gears.");
        else
            System.out.println("This bike does not have gear system.");
    }
}

class Trucks extends Vehicle {
    int tons;
    Trucks(String brand, int speed, int tons) {
        super(brand, speed);
        this.tons = tons;
 }
    void truckdetails() {
        System.out.println("The capacity of truck is: " + tons);
    }
}

class Rent {
    public static void main(String[] args) {



  CARS c = new CARS("HONDA", 120, 5, 5);
        c.cardetails();
        c.Details();

        Bikes b = new Bikes("Ninja H2R", 80, true);
        b.bikedetails();
        b.Details();

        Trucks t = new Trucks("Maruti", 100, 1);
        t.truckdetails();
        t.Details();
    }
```

OUTPUT:



```
Number of doors:5
Capacity:5
Brand:HONDA
Speed:120
This bike has gears.
Brand:Ninja H2R
Speed:80
The capacity of truck is: 1
Brand:Maruti
Speed:100
```

Errors:

S.NO	Error Name	Error Rectification
1	Syntax/ Compilation Error	Absence of Semicolon
2	Closing Brackets	Need to Close the brackets
3	Class Name Error	Give the class name correctly
4	Constructor Calling	Call the constructor correctly

Important points:

Multi-inheritance: It is one of the types of the inheritance where subclass 2 inherits subclass1 and subclass1 inherits superclass.

Here Vehicle is the super class or parent class and remaining cars, bikes, trucks are the subclasses or child classes.

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 about a car, model, fuel type, and color using the constructor**

- Write your code in VS CODE and execute

- Important Points:

- Understand the calling of a Constructor
- Giving class name correctly
- Give the parameters Correctly

Class Diagram:

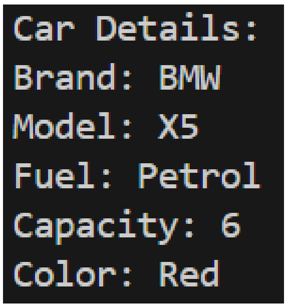
```
    +------------------+
    |    Vehicle     |
    +------------------+
    | - brand: String  |
    | - speed: int     |
    +------------------+
    | + Vehicle()      |
    | + displayInfo()  |
    +------------------+
          ▲
    +------------------------------+
    |          Car           |
    +------------------------------+
    | - model: String            |
    | - company: String           |
    | - mileage: int             |
    | - fuelType: String          |
    +------------------------------+
    | + Car(brand, speed, ...)     |
    | + displayInfo()            |
    +------------------------------+
     Main Method:
+----------------------------+
|        Help           |
+----------------------------+
| + main(String[] args)     |
+----------------------------+
```

CODE:

```
class Vehicle {
   String Brand;
   String model;
   String fuel;
   String color;
   int capacity;

   Vehicle(String Brand, String model, String fuel, int capacity, String color) {
      this.Brand = Brand;
      this.model = model;
      this.fuel = fuel;
      this.capacity = capacity;
      this.color = color;
   }

   void displayInfo(String Brand, String model, String fuel, int capacity, String color) {
      System.out.println("Vehicle Details: ");
      System.out.println("Brand: " + Brand);
      System.out.println("Model: " + model);
      System.out.println("Fuel: " + fuel);
      System.out.println("Capacity: " + capacity);
      System.out.println("Color: " + color);
   }
}

class Car extends Vehicle {

   Car(String Brand, String model, String fuel, int capacity, String color) {
      super(Brand, model, fuel, capacity, color);
   }

   void displayInfo() {
      System.out.println("Car Details: ");
      System.out.println("Brand: " + Brand);
      System.out.println("Model: " + model);
      System.out.println("Fuel: " + fuel);
      System.out.println("Capacity: " + capacity);
      System.out.println("Color: " + color);
   }
}


System.out.pritnln(“color:”+ color);
     }
}
class Week6_1 {
   public static void main(String[] args) {
      // Creating an instance of Car
      Car car1 = new Car("BMW", "X5", "Petrol", 6, "Red");
```

```
        car1.displayInfo();  // Display car details
    }
}
```

OUTPUT:



```
Car Details:
Brand: BMW
Model: X5
Fuel: Petrol
Capacity: 6
Color: Red
```

Errors:

S.NO	Error Name	Error Rectification
1	Syntax/ Compilation Error	Absence of Semicolon
2	Closing Brackets	Need to Close the brackets
3	Class Name Error	Give the class name correctly
4	Constructor Calling	Call the constructor correctly

Important points:
In order to do this, we have to use inheritance concept. Here
we used the multi-inheritance concept

2) Create a Java program for the scenario.
A college is developing an automated admission system that verifies student eligibility for undergraduate (UG) and postgraduate(PG) programs. Each program has different eligibility criteria based on the student's percentage in their previous qualification.
i) UG admissions require a minimum of 60%
ii) PG admissions require a minimum of 70%

- Write your code in VS CODE and execute
- Important Points:

- **Understand the calling of a Constructor**
- **Giving class name correctly**
- **Give the parameters Correctly**

Class Diagram:

```
    +-------------------------------+
    |        AdmissionSystem        |
    +-------------------------------+
    | - studentName: String         |
    | - percentage: double          |
    +-------------------------------+
    | + AdmissionSystem(name, %)    |
    | + checkEligibility()          |
    +-------------------------------+
              ▲          ▲
  +------------------+  +------------------+
  |  UGAdmission     |  |  PGAdmission     |
  +------------------+  +------------------+
  | + UGAdmission(...)|  | + PGAdmission(...)|
  | + checkEligibility()| | + checkEligibility()|
  +------------------+  +------------------+
```

Main Method:

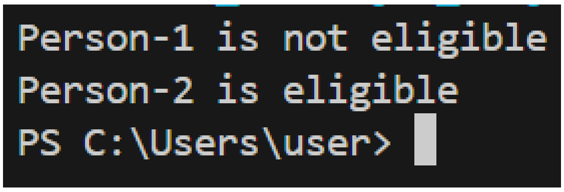
```
+-------------------------------+
|           Main                |
+-------------------------------+
| + main(String[] args)         |
+-------------------------------+
```

CODE:

```
class College {
   String name;
   int percentage;

   void geteligibility(String name, int percentage) {
      this.name = name;
      this.percentage = percentage;
   }
}

class UG extends College {
   void geteligibility(String name, int percentage) {
      if (percentage >= 60) {
         System.out.println(name + " is eligible");
      } else {
         System.out.println(name + " is not eligible");
```

```
        }
      }
  }

  class PG extends College {
      void geteligibility(String name, int percentage) {
          if (percentage >= 70) {
              System.out.println(name + " is eligible");

   } else {
              System.out.println(name + " is not eligible");
          }
      }
  }

  class week6_2 {
      public static void main(String[] args) {
          UG ug = new UG();
          ug.geteligibility("Person-1", 40);

          PG pg = new PG();
          pg.geteligibility("Person-2", 80);
      }
  }
```

OUTPUT:



```
Person-1 is not eligible
Person-2 is eligible
PS C:\Users\user>
```

Errors:

S.NO	Error Name	Error Rectification
1	Syntax/ Compilation Error	Absence of Semicolon
2	Closing Brackets	Need to Close the brackets
3	Class Name Error	Give the class name correctly
4	Constructor Calling	Call the constructor correctly

Important points:

Super keyword is used take the method,variable,constructor from the super class.

3) Write a Java Program to create a Calculator class with overloaded methods to perform addition: Take the integer values a and b from the user.

i) Add two integers
ii) Add two doubles
iii) Add three integers

- Write your code in VS CODE and execute
- Important Points:

- **Understand the calling of a Constructor**
- **Giving class name correctly**
- **Give the parameters Correctly**

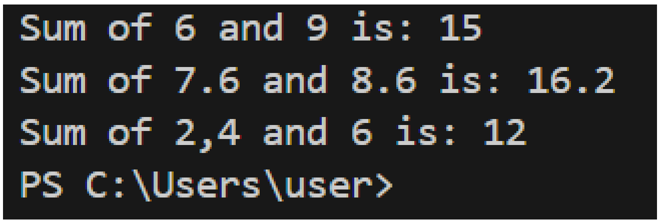
Class Diagram:

```
+-----------------------------------+
|     Calculatoroverloading         |
+-----------------------------------+
| + add(int a, int b): int          |
| + add(double a, double b): double |
| + add(int a, int b, int c): int   |
+-----------------------------------+
            ▲
+-----------------------------+
|          loading            |
+-----------------------------+
| + main(String[] args): void |
+-----------------------------+
```

CODE:

```
class Calc {
   public int add(int a, int b) {
      return a + b;
   }

   public double add(double a, double b) {
      return a + b;
   }

   public int add(int a, int b, int c) {
      return a + b + c;
   }
}

class week6_3 {
    public static void main(String[] args) {
        Calc C1 = new Calc();
        System.out.println("Sum of 6 and 9 is: " + C1.add(6, 9));
        System.out.println("Sum of 7.6 and 8.6 is: " + C1.add(7.6, 8.6));
        System.out.println("Sum of 2, 4 and 6 is: " + C1.add(2, 4, 6));
    }
}
```

OUTPUT:



```
Sum of 6 and 9 is: 15
Sum of 7.6 and 8.6 is: 16.2
Sum of 2,4 and 6 is: 12
PS C:\Users\user>
```

Errors table :

S.NO	Error Name	Error Rectification
1	Syntax/ Compilation Error	Absence of Semicolon
2	Closing Brackets	Need to Close the brackets
3	Class Name Error	Give the class name correctly
4	Constructor Calling	Call the constructor correctly

Important points:

We should carefully pass the double and integer and diterent types of input to an constructor when creating an object to access the diterent constructors based on the parameter.

4) Write a Java Program to create a shape class with a method calculateArea() that is overloaded for different shapes(e.g., Square, Rectangle). Then create a subclass Circle that overrides the calculateArea() method for a circle.

- Write your code in VS CODE and execute

- Important Points:

- Understand the calling of a Constructor
- Giving class name correctly
- Give the parameters Correctly

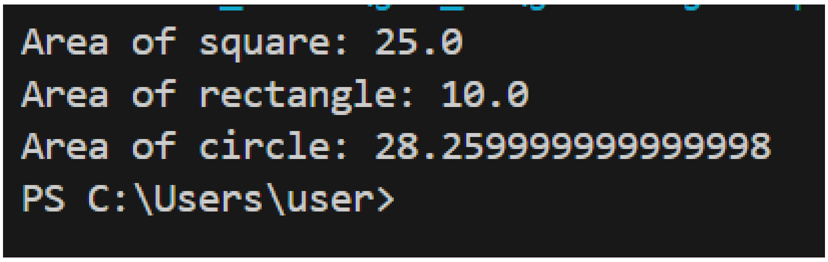
Class Diagram:

```
+---------------------------+
      |         Shape          |
      +---------------------------+
      | + calculatearea(int, int): void     |  ← Rectangle
      | + calculatearea(int): void         |  ← Square
      +---------------------------+
      +---------------------------+
      |         Circle         |
      +---------------------------+
      | + calculatearea(double, double): void |  ← Circle
      +---------------------------+
      +---------------------------+
      |         Areas2          |
      +---------------------------+
      | + main(String[] args): void        |
      +---------------------------+
```

CODE:

```
class Shape {
   double calculateArea(double side) {
      return side * side;
   }
   double calculateArea(double width, double height) {
      return width * height;
   }
}
class Circle extends Shape {

   double calculateArea(double radius) {
      return 3.14 * radius * radius;
   }
}
class Week6_4 {
   public static void main(String[] args) {
      Shape S1 = new Shape();
      System.out.println("Area of square: " + S1.calculateArea(5));
      System.out.println("Area of rectangle: " + S1.calculateArea(2, 5));
      Circle C1 = new Circle();
      System.out.println("Area of circle: " + C1.calculateArea(3));
   }
}
```

OUTPUT:



```
Area of square: 25.0
Area of rectangle: 10.0
Area of circle: 28.259999999999998
PS C:\Users\user>
```

Errors:

S.NO	Error Name	Error Rectification
1	Syntax/ Compilation Error	Absence of Semicolon
2	Closing Brackets	Need to Close the brackets
3	Class Name Error	Give the class name correctly
4	Constructor Calling	Call the constructor correctly

Important points:

In this program we use both method overloading andoverriding to calculate area of diterent shapes

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-

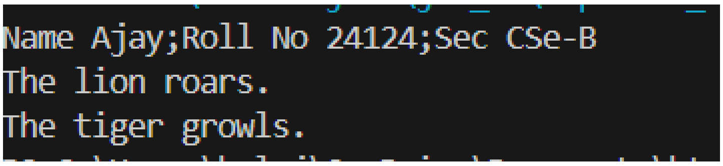
```
            +------------------------------------+
            |  <<abstract>>   Animal     |
            +------------------------------------+
            | - makeSound()              |
            +------------------------------------+
                              ▲
                              |
            +---------------+--------------------+
            |                                    |
  +--------------------+           +----------------------+
  |      Tiger      |              |      Lion         |
  +--------------------+           +----------------------+
  | +makeSound()                   | +makeSound()
  +--------------------+           +----------------------+

                 +-----------------------------+
                 |       AnimalTest        |          +-----------------------------+
                 | +main(String[] args)  |
                 +-----------------------------+
```

CODE:

```
 abstract class Animal {
public abstract void sound();
}

class Lion extends Animal {

@Override

public void sound() {
System.out.println("The lion roars.");

}

}
```

```
 class Tiger extends Animal {

@Override


System.out.println("The tiger growls.");

}
}

public class q17 {

public static void main(String[] args) {

   System.out.println("Name Ajay;Roll No 24124;Sec CSe-B");

Lion lion = new Lion();

Tiger tiger = new Tiger();

lion.sound();

tiger.sound();

}

      }
```

output:



```
Name Ajay;Roll No 24124;Sec CSe-B
The lion roars.
The tiger growls.
```

ERRORS:

S.No	Error type	Reason for error	Rectification
1	Syntax error	Abstract key word is missed before method	Abstract keyword is added
2	Logical error	Incorrect logic in subclass method	Corrected logic in subclass method

Important Points:

1. Abstract Class Usage: The Animal class is declared as abstract, meaning it cannot be instantiated directly and must be extended. It defines a common structure (method sound()) that all animal types must implement.
2. Polymorphism: The use of the abstract class as a reference type (Animal lion = new Lion();) showcases polymorphism, allowing us to call the overridden sound() method

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-

```
      +----------------------------------------------+
      |                  Shape3D               |
      +---------------------------------------------+
      | +calculateVolume(): double            |
      | +calculateSurfaceArea(): double     |
      +---------------------------------------------+
                           ▲
                           |
       +---------------------+----------------------+
       |                                            |
  +------------------------------+        +------------------------------+
  |          Sphere              |        |          Cube              |
  +-------------------------------+        +------------------------------+
| - radius: double              |        | - sideLength: double    |
| +calculateVolume()        |        |+calculateVolume()       |
| +calculateSurfaceArea() |        |+calculateSurfaceArea()|
+--------------------------------+        +-------------------------------+


      +------------------------------+
      |       Shape3DTest        |
      +-------------------------------+        | +main(String[] args)     |        +----------------------------
---+
```

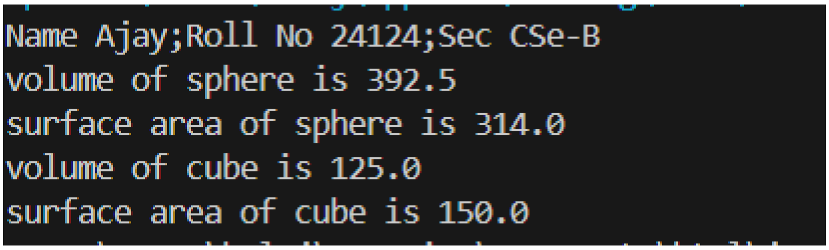
CODE:

```
abstract class area{
   abstract double   calculatevolume();

   abstract double  calculatesurfacearea();
}
```

```
class sphere extends area{
   double radius;
   double calculatevolume(){

      return 4/3*3.14*radius*radius*radius;

      }

      double calculatesurfacearea(){

         return 4*3.14*radius*radius;

         }
   }


  class cube extends area{

      double side;

      double calculatevolume(){

         return side*side*side;

         }
         double calculatesurfacearea(){

            return 6*side*side;
            }
            }
class q18{

   public static void main(String[] args) {

      System.out.println("Name Ajay;Roll No 24124;Sec CSe-B");

      sphere s=new sphere()

      s.radius=5;

      System.out.println("volume of sphere is "+s.calculatevolume());

      System.out.println("surface area of sphere is "+s.calculatesurfacearea());

      cube c=new cube();

      c.side=5;

      System.out.println("volume of cube is "+c.calculatevolume());
```

```
    System.out.println("surface area of cube is "+c.calculatesurfacearea());

    }


  }
```

output:



```
Name Ajay;Roll No 24124;Sec CSe-B
volume of sphere is 392.5
surface area of sphere is 314.0
volume of cube is 125.0
surface area of cube is 150.0
```

ERROR:

S.No	Error type	Reason for error	Rectification
1	Syntax error	package is missed before abstract class	Package is imported
2	Logical error	Incorrect logic in subclass method	Corrected logic in subclass method

Important Points:

1. Abstract Method Implementation: The Shape3D class declares two abstract methods: calculateVolume() and calculateSurfaceArea(). All subclasses must implement these methods, ensuring a consistent interface for 3D shapes.
2. Encapsulation of Properties: Sphere and Cube each manage their own properties (radius and side), which supports encapsulation and makes the program more modular and scalable.

3)Aim : Write a java program using an abstract class to define a method for pattern printing Create an abstract class named pattern printer 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 triangles of increasing numbers.
In the main() method, create Objects

```
Star Pattern                    Number pattern
*                               1
**                              1 2
***                      1 2 3
****                            1 2 3 4
*****                           1 2 3 4 5
```

CLASS DIAGRAM-

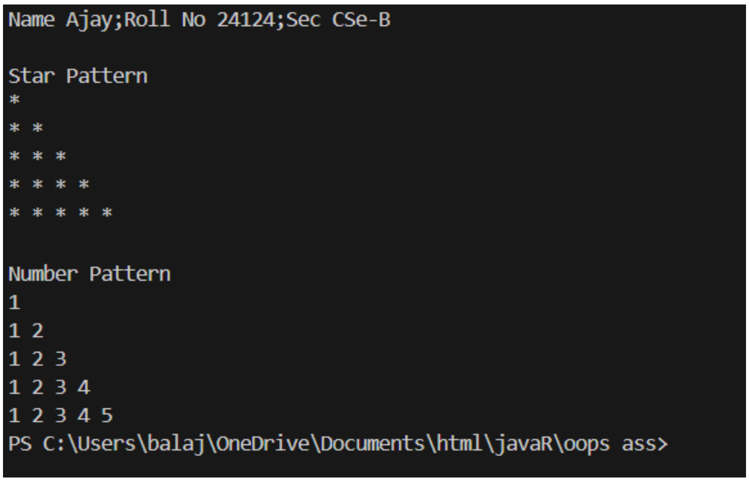
```
         +-------------------------------------------+
         |              PatternPrinter               |
         +-------------------------------------------+
         | +printPattern(n: int): void               |
         +-------------------------------------------+
                              ▲
                              |
               +----------------+------------------+
               |                                   |
+------------------------------+         +---------------------------+
|        StarPattern           |         |  NumberPattern            |
+-------------------------------+        +---------------------------+
| +printPattern(n: int)         |         |+printPattern(n: int) |
+-------------------------------+        +----------------------------+

      +---------------------------------+
      |      PatternPrinterTest         |
      +----------------------------------+
      | +main(String[] args): void|
      +----------------------------------+
```

CODE:

```
abstract class PatternPrinter {
   abstract void printPattern(int n);

   void displayTitle(String title) {
      System.out.println(title);
   }
}

class StarPattern extends PatternPrinter {
   void printPattern(int n) {
      for (int i = 1; i <= n; i++) {
         for (int j = 1; j <= i; j++) {
            System.out.print("* ");
         }
```

```
  System.out.println();
      }
    }
}

class NumberPattern extends PatternPrinter {
    void printPattern(int n) {
        for (int i = 1; i <= n; i++) {
            for (int j = 1; j <= i; j++) {
                System.out.print(j + " ");
            }
            System.out.println();
        }
    }
}

 class pat {
    public static void main(String[] args) {


        StarPattern sp = new StarPattern();
        NumberPattern np = new NumberPattern();
        System.out.println("Name Ajay;Roll No 24124;Sec CSe-B");
        System.out.println("    ");
        sp.displayTitle("Star Pattern");
        sp.printPattern(5);

        System.out.println("    ");

        np.displayTitle("Number Pattern");
        np.printPattern(5);
    }
}
```

output:



```
Name Ajay;Roll No 24124;Sec CSe-B

Star Pattern
*
* *
* * *
* * * *
* * * * *

Number Pattern
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
PS C:\Users\balaj\OneDrive\Documents\html\javaR\oops ass>
```

ERROR:

S.No	**Error type**	**Reason for error**	**Rectification**
1	Syntax error	For loop increment condition is missed in subclass method	Increment condition is added in subclass method
2	Logical error	Incorrect logic in subclass method	Corrected logic in subclass method

Importat Points:

1. Abstraction with Polymorphism: The PatternPrinter abstract class enforces a contract (printPattern(int n)) that all subclasses must implement. This ensures a consistent interface while allowing different behaviors (e.g., star vs number pattern).

WEEK-8

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

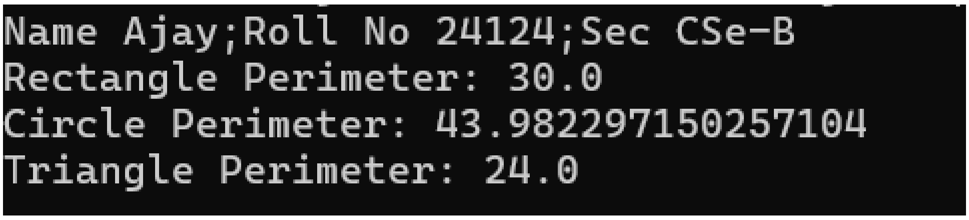
CLASS DIAGRAM-

```
                  +--------------------------------+
                  |           Shape2              |
                  +--------------------------------+
                  | +getPerimeter(): double|
                  +--------------------------------+
                                ▲
                          |
              +---------------------+---------------------------+
              |                     |                           |
     +---------------------+  +---------------------+   +----------------------+
     |  Rectangle      |  |      Circle         |   |   Triangle         |
     +---------------------+  +---------------------+   +----------------------+
     | -width: double  |  | -radius: double   |    | -a: double          |
     | -height: double |  | +getPerimeter() |   | -b: double          |
     | +getPerimeter()|  +----------------------+  | -c: double          |
     +---------------------+                           | +getPerimeter() |
                                                        +---------------------

+              +-------------------------------------------+

             |            Shape2Factory               |
             +-------------------------------------------+
             | +createRectangle(w,h):

Rectangle |              | +createCircle(r):

Circle                 |

             | +createTriangle(a,b,c): Triangle     |
             +--------------------------------------------+


             +-----------------------------------------+
             |               Shape2                   |
             +-----------------------------------------+
             | +main(String[] args): void           |
             | +getPerimeter(Shape2): double  |

             +-----------------------------------------+
```

Code:

```
interface Shape {
 double getPerimeter();
 }
 class Rectangle implements Shape {
 private double length;
 private double width;public Rectangle(double length, double width) {
 this.length = length;
 this.width = width;
 }
 @Override
 public double getPerimeter() {
 return 2 * (length + width);
 }
 }
 class Circle implements Shape {
 private double radius;
 public Circle(double radius) {
 this.radius = radius;
 }
 @Override
 public double getPerimeter() {
 return 2 * Math.PI * radius;
 }
 }
 class Triangle implements Shape {
 private double side1;
 private double side2;
 private double side3;
 public Triangle(double side1, double side2, double side3) {
 this.side1 = side1;
 this.side2 = side2;
 this.side3 = side3;
 }
 @Override
 public double getPerimeter() {
 return side1 + side2 + side3;
 }
 }
 public class q20 {
 public static void main(String[] args) {
  System.out.println("Name Ajay;Roll No 24124;Sec CSe-B");

 Rectangle rectangle = new Rectangle(10, 5);Circle circle = new Circle(7);
 Triangle triangle = new Triangle(6, 8, 10);
 System.out.println("Rectangle Perimeter: " +
   rectangle.getPerimeter());
   System.out.println("Circle Perimeter: " +
   circle.getPerimeter());
   System.out.println("Triangle Perimeter: " +

 triangle.getPerimeter());
   }
   }
```

Output:



```
Name Ajay;Roll No 24124;Sec CSe-B
Rectangle Perimeter: 30.0
Circle Perimeter: 43.982297150257104
Triangle Perimeter: 24.0
```

Error:

S.No	Error type	Reason for error	Rectification
1	Syntax error	Used extends keyword instead of implements keyword in inheritance	Implements key word is added in inheritance

Important Points:

○ Interface: Defines a contract for classes that implement it.

○ Implementing an Interface: Classes must provide implementations for all methods declared in the interface.

○ Polymorphism: Objects of different classes can be treated uniformly through the Shape interface.

Aim: Write a Java program to create an interface Playable with a method play() that takes no arguments and returns void. Create three classes Football, Volleyball, and Basketball that implement the Playable interface and override the play() method to play the respective sports.

CLASS DIAGRAM-

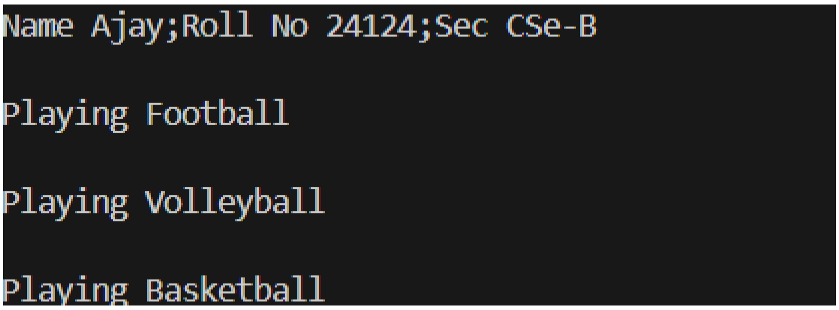
```
+---------------------------------------+

|     <<interface>> Playable      |

+---------------------------------------+

| +play(): void                   |

| +playDefault(): void            |                +---------------------------------------+

                 ▲
```

```
             +---------------------+-----------------------+
             |                     |                       |
    +-------------------+   +-------------------+    +-------------------+
    |    Football     |   |    Volleyball   |    | Basketball    |
    +-------------------+   +--------------------+   +-------------------+
     | +play(): void |   | +play(): void   |     | +play(): void  |
     +-------------------+   +-------------------+   +--------------------+                      +------------------
------------+
                      |     PlayableDemo       |
                      +------------------------------+
                    | +main(String[] args): void|                    +----------------------------------+
```

Code:

```
    interface Playable{

   public void play();
}

class Football implements Playable{
   @Override
   public void play(){
      System.out.println("Football is played between two teams of 11 players each, with the objective of scoring goals by kicking or heading the ball into the opposing team's net.");
   }
}

class Volleyball implements Playable{
   @Override
   public void play(){
      System.out.println("Volleyball is played between two teams of six players,where the teams try to score points by grounding a ball on the opponent's court, using a net and a maximum of three touches per side before sending the ball over. ");
   }
}

class Basketball implements Playable{
   @Override
   public void play(){
      System.out.println("Basketball is played between two teams of five players,where the teams try to score by shooting a ball through a hoop mounted high on a backboard, while preventing the opposing team from doing the same. ");
   }
}

public class play {
   public static void main(String[] args) {
        System.out.println(“Name Ajay;Roll No 24124;Sec CSe-B”);

      System.out.println();
      Football fb=new Football();
      fb.play();
      System.out.println();
      Volleyball vb=new Volleyball();
      vb.play();

 System.out.println();
      Basketball bb=new Basketball();
      bb.play();
   }
}
```

Output:



```
Name Ajay;Roll No 24124;Sec CSe-B

Playing Football

Playing Volleyball

Playing Basketball
```

ERROR:

S.No	Error type	Reason for error	Rectification
1	Syntax error	Error in calling play method in football class because object is not created for football class	Object created for football class

Important Points:

○ Interface: Defines a common action (play()) for different sports.

○ Implementing Interface: Classes provide specific implementations of how to play each sport.

○ Polymorphism: Different sports can be played through a common Playable interface

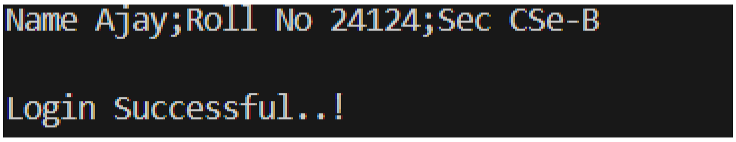
3) Aim: write a java program to implement a login system using interfaces.
CLASS DIAGRAM-

```
+-------------------------------------+
|           Interface              |
+-------------------------------------+
| +main(String[] args): void   |
| +checkNumber(int): void    |
+-------------------------------------+
```

Code:

```
interface LoginSystem {

    boolean Login(String ID, int pass);
}

class CollegePortal implements LoginSystem {
    public boolean Login(String ID, int pass) {
        if ((ID=="TEJA") && (pass==24138)){
            System.out.println("Login Successful..!");
            return true;
        }else {
            System.out.println("Invalid ID or Password");
            return false;
        }
    }
}
class LoginPortal {
    public static void main(String[] args) {
        CollegePortal CP = new CollegePortal();
        System.out.println("Name Ajay;Roll No 24124;Sec CSe-B");
        System.out.println("    ");


    }
}
```

output:



```
Name Ajay;Roll No 24124;Sec CSe-B

Login Successful..!
```

Error Table :

S.No	Error type	Reason for error	Rectification
1	Syntax error	Error in If statement condition	If statement condition is corrected

Important Points:

- Interface-Based Design: The Login interface abstracts the authentication mechanism.
- Loose Coupling: The LoginSystem depends on the Login interface, not the implementation (UserLogin), making it easily replaceable or extendable.

WEEK 9

Program 1

AIM: Write a Java program to create a method that takes integer as parameter and throws an exception if the number is even.

CLASS DIAGRAM-

```
+-------------------------------------+
|           PlayableTest              |
+-------------------------------------+
| +main(String[] args): void   |
| +checkNumber(int): void      |
+-------------------------------------+
```

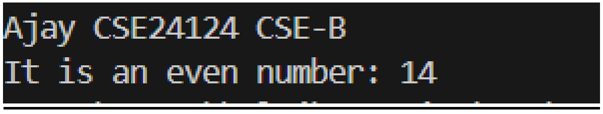
Code:

```
public class evennumexception {
    public static void Checknumber(int number)
    throws Exception {
        if (number % 2 == 0) {
            throw new Exception("It is an even number: " + number);
        } else {
            System.out.println("It is an odd number: " + number);
        }
    }
    public static void main(String[] args) {
        try {
            System.out.println("Ajay CSE24124 CSE-B");
            Checknumber(14);
        } catch (Exception e) {
            System.out.println(e.getMessage());
        }
    }
}
```

Error Table:

S.no	Expected Error	Reason
1	Types “throw” instead of “throws”	Replaced “throw” with “throws”
2	Missed “}”	Ending the main class and main method is required

output:



```
Ajay CSE24124 CSE-B
It is an even number: 14
```

IMPORTANT POINTS:

Class names should use PascalCase (e.g., EvenNumException)
Method names should use camelCase (e.g., checkNumber) Method throws an exception if the number is even
Uses try-catch block in main to handle the exception
Outputs message based on whether number is even or odd
Uses getMessage() to display the exception message

Program 2

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

CLASS DIAGRAM-

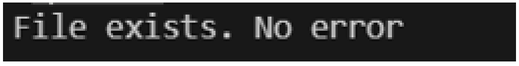
```
+---------------------------------------------+
|              FileReadExample               |
+---------------------------------------------+
|            (private attributes)            |
+---------------------------------------------+
| + readFile(fileName: String): void |
| + main(args: String[]): void          |
+---------------------------------------------+
```

Code:

```
import java.io.*;
public class filenotfound {
   public static void main(String[] args) {
      try {
         BufferedReader br = new BufferedReader(new
FileReader("C:/Users/prana/OneDrive/Desktop/1.txt"));
         String line;
         while ((line = br.readLine()) != null) {
            System.out.println(line);
         }
         br.close();
      } catch (FileNotFoundException e) {
         System.err.println("File not found: " + e.getMessage());
      } catch (IOException e) {
         System.err.println("Error reading file: " + e.getMessage());
      }
   }
}
```

If file exists:

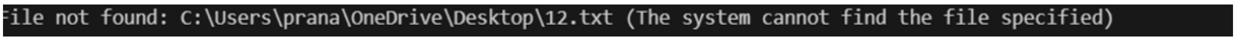
Output:



```
File exists. No error
```

If file does not exist:

Output:



```
ile not found: C:\Users\prana\OneDrive\Desktop\12.txt (The system cannot find the file specified)
```

Error Table:

S.no	Error	Reason
1	Did not change “/” to “\”	Replaced “/ with “\”
2	Missed “}”	Ending the main class and main method is required

IMPORTANT POINTS:

Check if the file exists at the specified path
Verify the file path is correct and has no typos
Ensure the file has the correct extension (e.g. not 12.txt.txt)
Use double backslashes if using backslash in the path
Make sure the file is not open or locked by another application

Program 3

AIM: Write a Java program to handle arithmetic exception using try, catch, and finally block.

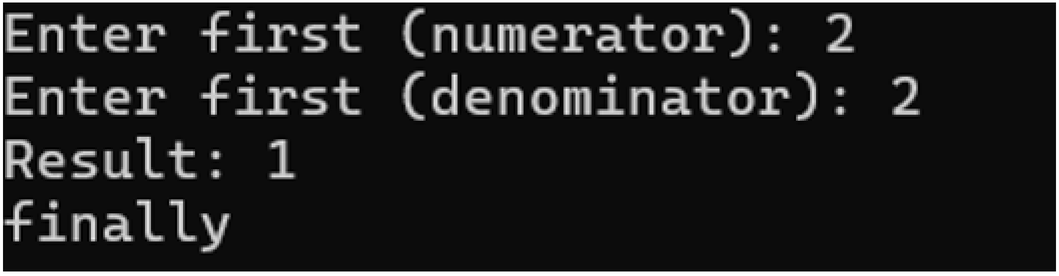
CLASS DIAGRAM-

```
+-----------------------------------------+
|  ArithmeticExceptionExample |
+-----------------------------------------+
| + main(args: String[]): void      |
+-----------------------------------------+
```

Code:

```
import java.util.Scanner;
public class airthmetic{
   public static void main(String[] args) {
      Scanner Scanner=new Scanner(System.in);
      try {
         System.err.print("Enter first (numerator): ");
         int a=Scanner.nextInt();
         System.err.print("Enter first (denominator): ");
         int b=Scanner.nextInt();
         int result=a/b;
         System.err.println("Result: "+result);
      }
      catch (Exception e) {
         System.err.println("An error occured");
      }
      finally{
         System.err.println("finally");
      }
      Scanner.close();

}
}
```

Output:



```
Enter first (numerator): 2
Enter first (denominator): 2
Result: 1
finally
```

ERROR TABLE:

S.no	Error	Reason
1.	Missed ";" in first line	Ended the statement with ";"

IMPORTANT POINTS:

Import Scanner class to read user input
Create a Scanner object to take input from the console
Use try block to wrap code that might throw an exception
Read numerator and denominator using scanner.nextInt()
Perform division inside try block (this might cause ArithmeticException)
Use catch block to handle exceptions and display an error message
catch block uses Exception class, which is a general exception handler
Use finally block to execute code regardless of whether an exception occurred

Program 4

AIM: Write a Java program to simulate a university system using inner class

- Create an outer class named University with a variable universityName
- Inside it define two non-static inner class:
    1. Department – with variables like deptName and deptCode and a method to display department details
    2. Student – with variables like studentName and rollNumber and a method to display student details
- Create an object for each class and call their methods to display the details along with the university name

CLASS DIAGRAM-

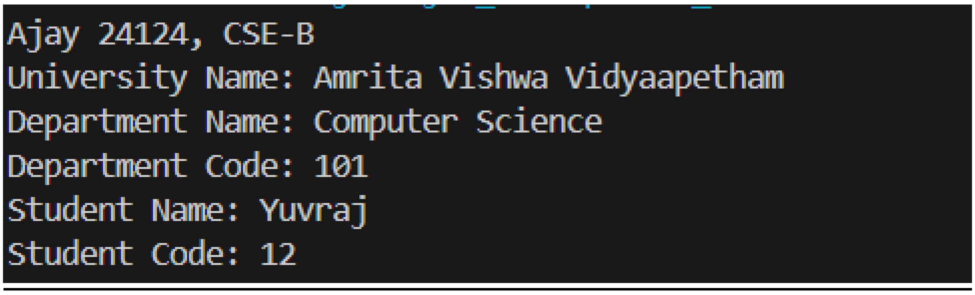
```
+---------------------------------------------+
|              University                     |
+---------------------------------------------+
| - universityName: String                    |
+---------------------------------------------+



| + University(name: String)                  |
| + displayUniversityDetails(): void          |
+---------------------------------------------+
                    |
+-------------------------------------------------------+ |                  Department                   |
+-------------------------------------------------------+
| - deptName: String                             |
| - deptCode: String                             |
+-------------------------------------------------------+
| + Department(name: String, code: String) |
| + displayDepartmentDetails(): void          |
+--------------------------------------------------------+
                              |
         +-------------------------------------------+
         |                 Student                 |
         +-------------------------------------------+
         | - studName: String                     |
         | - rollNo: int                          |
         +--------------------------------------------+
         | + Student(name: String, roll: int) |          | + displayStudentDetails(): void    |
+-------------------------------------------+
```

Code:

```
public class university {
   String UniversityName = "Amrita Vishwa Vidyaapetham";
   void name() {
      System.out.println("University Name: " + UniversityName);
   }
   class Department {
      String DeptName = "Computer Science";
      int Deptcode = 101;
      void Departmentinfo() {
         System.err.println("Department Name: " + DeptName);
         System.err.println("Department Code: " + Deptcode);
      }
   }
   class Student {
      String Studentname = "Yuvraj";
      int RollNumber = 12;
      void Studentinfo() {
         System.err.println("Student Name: " + Studentname);
         System.err.println("Student Code: " + RollNumber);
      }
   }


   public static void main(String[] args) {
     System.err.println("Ajay 24124, CSE-B ");
      university u = new university();
      u.name();
```

```
        university.Department dept = u.new Department();
        dept.Departmentinfo();
        university.Student student = u.new Student();
        student.Studentinfo();
    }
}
```

output:



```
Ajay 24124, CSE-B
University Name: Amrita Vishwa Vidyaapetham
Department Name: Computer Science
Department Code: 101
Student Name: Yuvraj
Student Code: 12
```

ERROR TABLE:

S.no	Error	Reason
1.	Missed “;” in 2nd line	Ended the statement with “;”

IMPORTANT POINTS:

The outer class is named University
It has a member variable universityName and a method to display it
Two non-static inner classes are defined inside the outer class: Department and Student
Inner class Department contains deptName, deptCode, and a method to display them
Inner class Student contains studentName, rollNumber, and a method to display them
Non-static inner classes require an instance of the outer class to be created
Inner class objects are created using outerClassInstance.new InnerClass() syntax

Week 10

Program 1

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

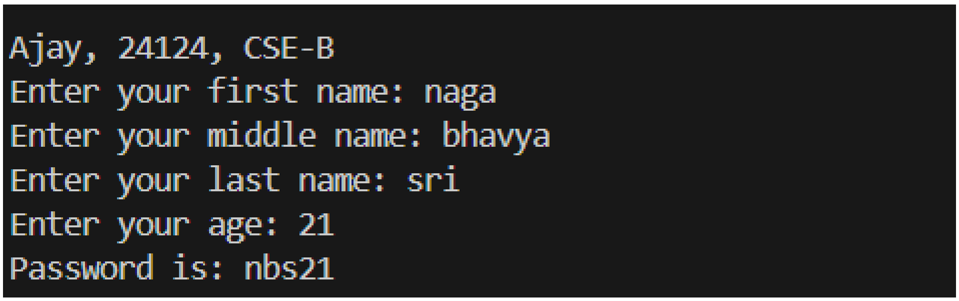
CLASS DIAGRAM-

```
+--------------------------------------+
|                Array5                |
+--------------------------------------+
| + main(args: String[]): void |
+-------------------------------------+
```

Code:

```
import java.util.Scanner;
class Password {
  public static void main(String[] args) {
    System.err.println("Ajay, 24124, CSE-B ");
    Scanner scanner = new Scanner(System.in);
    System.out.print("Enter your first name: ");
    String firstName = scanner.next();
    System.out.print("Enter your middle name: ");
    String middleName = scanner.next();
    System.out.print("Enter your last name: ");
    String lastName = scanner.next();
    System.out.print("Enter your age: ");
    int age = scanner.nextInt();
    char fname=firstName.charAt(0);

    char mname=middleName.charAt(0);
    char lname=lastName.charAt(0);
    System.err.println("Password is: "+fname+mname+lname+age);
  }
}
```

Output:



```
Ajay, 24124, CSE-B
Enter your first name: naga
Enter your middle name: bhavya
Enter your last name: sri
Enter your age: 21
Password is: nbs21
```

ERROR TABLE:

S.No	ERRORS	Error rectification
1.	Forget to create a scanner object.	Created a scanner object named “scanner”.

IMPORTANT POINTS:

The program is encapsulated in a class called Password (following Java naming conventions with a capitalized class name).

The program uses Scanner to accept user input for the first name, middle name, last name, and age

charAt(0) is used to get the first letter (initial) of each name.

The password is generated by concatenating the initials (first character) of the first, middle, and last names along with the user's age.

Program 2

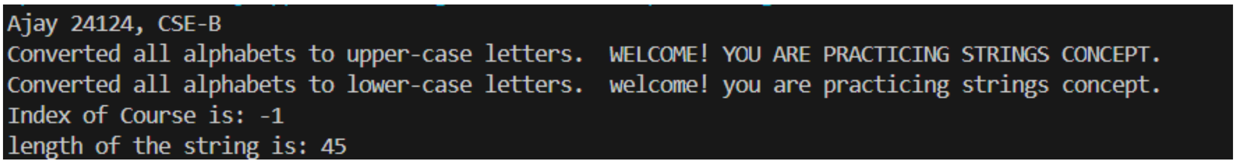
AIM: Design and implement a Java program that will do the following operations to this string "Welcome! You are practicing strings concept."

a. convert 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 Course.

Code:

```
class Strings{
   public static void main(String[] args) {
      System.out.println("Ajay 24124, CSE-B ");
      String string=" Welcome! You are practicing strings concept.";
      String UC=string.toUpperCase();
      System.err.println("Converted all alphabets to upper-case letters."+" "+UC);
      String LC=string.toLowerCase();
      System.err.println("Converted all alphabets to lower-case letters."+" "+LC);
      int index=string.indexOf("Course");
      System.err.println("Index of Course is: "+index);
      int length=string.length();
      System.err.println("length of the string is: "+length);
   }}
```

output:



```
Ajay 24124, CSE-B
Converted all alphabets to upper-case letters.  WELCOME! YOU ARE PRACTICING STRINGS CONCEPT.
Converted all alphabets to lower-case letters.  welcome! you are practicing strings concept.
Index of Course is: -1
length of the string is: 45
```

ERROR TABLE:

S.No	ERRORS	Error rectification
1.	Used “uppercase” instead of “UpperCase”	Replaced it.
2.	Used “lowercase” instead of “LowerCase”	Replaced it.

IMPORTANT POINTS:

Strings in Java are immutable.
Use toUpperCase() to convert all letters to uppercase.
Use toLowerCase() to convert all letters to lowercase.
Use length() to find the total number of characters in a string.
Use indexOf("substring") to find the starting index of a substring.

Program 3

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

a. Sorting the elements (numbers and Strings) of an array.
b. convert the array elements into string.
c. fill the part of an array.
d. copy the elements of one array into another.

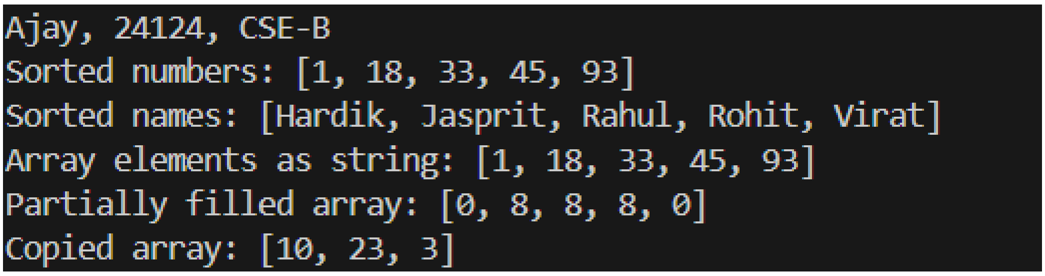
CLASS DIAGRAM-

```
+-----------------------------------------+
|        ArrayMethodsExample      |
+-----------------------------------------+
| + main(args: String[]): void       |
+------------------------------------------+
```

Code:

```
import java.util.Arrays;
public class pro3 {
   public static void main(String[] args) {
      System.out.println("Ajay, 24124, CSE-B ");
      int[] numbers = {45,33,18, 1, 93};
      String[] names = {"Rohit","Virat","Jasprit","Hardik","Rahul"};
      Arrays.sort(numbers);
      Arrays.sort(names);
      System.out.println("Sorted numbers: " + Arrays.toString(numbers));
      System.out.println("Sorted names: " + Arrays.toString(names));
      String numberString = Arrays.toString(numbers);
      System.out.println("Array elements as string: " + numberString);
      int[] filledArray = new int[5];
      Arrays.fill(filledArray, 1, 4, 8);
      System.out.println("Partially filled array: " + Arrays.toString(filledArray));
      int[] sourceArray = {10, 23, 3};
      int[] destinationArray = Arrays.copyOf(sourceArray, sourceArray.length);
      System.out.println("Copied array: " + Arrays.toString(destinationArray));
   }
}
```

output:



```
Ajay, 24124, CSE-B
Sorted numbers: [1, 18, 33, 45, 93]
Sorted names: [Hardik, Jasprit, Rahul, Rohit, Virat]
Array elements as string: [1, 18, 33, 45, 93]
Partially filled array: [0, 8, 8, 8, 0]
Copied array: [10, 23, 3]
```

ERROR TABLE:

S.No	ERRORS	Error rectification
1.	Did not place “;” after importing array package.	Added “;”

IMPORTANT POINTS:

Use Arrays.sort() to sort numeric and string arrays in ascending order.
Use Arrays.toString() to convert an array into a printable string format.
Use Arrays.fill(array, fromIndex, toIndex, value) to fill part of an array with a specific value.
Use Arrays.copyOf(originalArray, length) to copy one array into another.
The fromIndex in fill() is inclusive; toIndex is exclusive.
All arrays of primitive types are initialized with default values (0 for int).

Program 4

AIM: Implement a java program using the below Array List methods.

a. insert an element at particular index in the array list.
b. Modify an element in the array list.
c. Access an element from the array list.
d. Remove an element from the array list.
e. clear the elements from the array list.

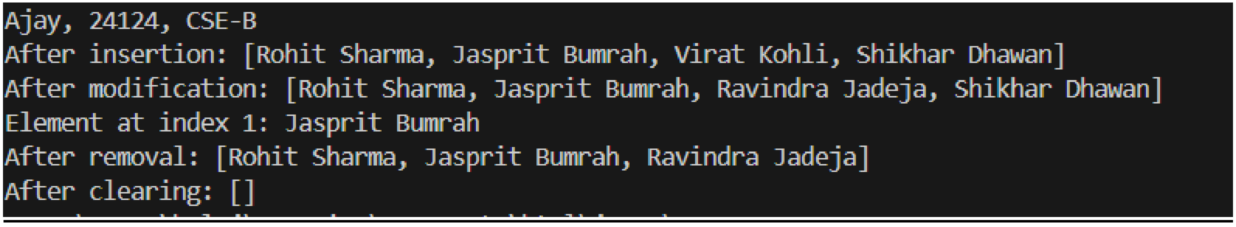
CLASS DIAGRAM-

```
+-------------------------------------+
|   ArrayListOperations        |
+-------------------------------------+
| + main(args: String[]): void |
+-------------------------------------+
```

Code:

```
import java.util.ArrayList;
public class pro4{
    public static void main(String[] args) {
        System.out.println("Ajay, 24124, CSE-B ");
        ArrayList<String> cricketers = new ArrayList<>();
        cricketers.add("Rohit Sharma");
        cricketers.add("Virat Kohli");
        cricketers.add("Shikhar Dhawan");
        cricketers.add(1, "Jasprit Bumrah");
        System.out.println("After insertion: " + cricketers);

        cricketers.set(2, "Ravindra Jadeja");
        System.out.println("After modification: " + cricketers);
        String cricketerAtIndex1 = cricketers.get(1);
        System.out.println("Element at index 1: " + cricketerAtIndex1);
        cricketers.remove(3);
        System.out.println("After removal: " + cricketers);
        cricketers.clear();
        System.out.println("After clearing: " + cricketers);
    }
}
```

Output:



```
Ajay, 24124, CSE-B
After insertion: [Rohit Sharma, Jasprit Bumrah, Virat Kohli, Shikhar Dhawan]
After modification: [Rohit Sharma, Jasprit Bumrah, Ravindra Jadeja, Shikhar Dhawan]
Element at index 1: Jasprit Bumrah
After removal: [Rohit Sharma, Jasprit Bumrah, Ravindra Jadeja]
After clearing: []
```

ERROR TABLE:

S. No	ERRORS	Error rectification
1.	Did not place “;” after importing array package.	Added “;”

IMPORTANT POINTS:

ArrayList Creation ArrayList<String> cricketers = new ArrayList<>();
Inserting an Element cricketers.add(1, "Jasprit Bumrah");
Modifying an Element cricketers.set(2, "Ravindra Jadeja");
Accessing an Element String cricketerAtIndex1 = cricketers.get(1);
Removing an Element cricketers.remove(3);
Clearing the List cricketers.clear()
.get(1);
Removing an Element cricketers.remove(3);
Clearing the List cricketers.clea