Experiment Number :11 **Date:**

Implementation of multiple inheritance in Java using Interface

Aim:

Write a Java program to implement multiple inheritance using Interface.

PRE LAB EXERCISE

QUESTIONS

1. Why Java does not support multiple inheritance using classes like that of C?

Java does not support multiple inheritance using classes to avoid the **Diamond Problem**.

In multiple inheritance, if two parent classes have the same method, the child class cannot decide which method to inherit. This creates ambiguity and confusion.

To avoid this complexity and maintain simplicity, Java allows:

- Multiple inheritance using **interfaces**
- Not using multiple classes

2. What is the primary purpose of an interface in Java?

A. To store data using instance variables

B. To define a contract of methods a class must implement

C. To allow creation of objects

D. To improve runtime performance

Correct Answer:

To define a contract of methods a class must implement.

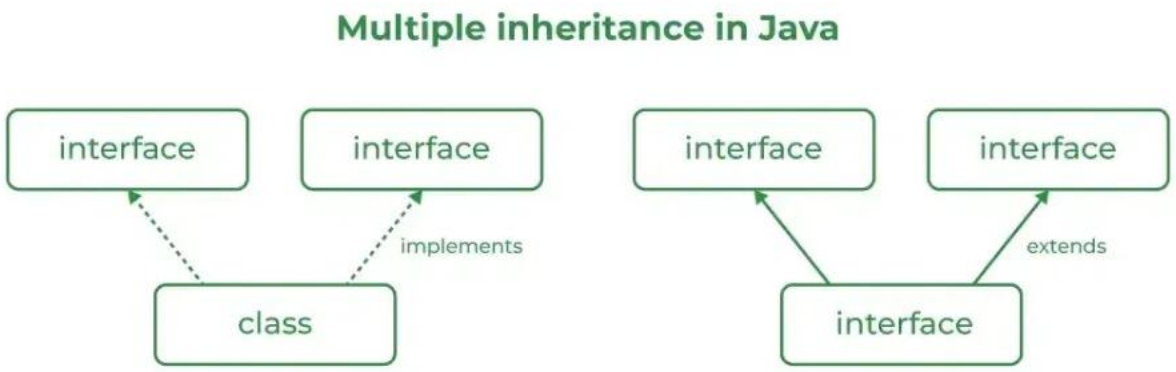
An interface specifies method declarations that a class must implement.

It ensures that different classes follow the same structure.

IN LAB EXERCISE

Objective

To demonstrate how an interface in Java defines constants and abstract methods, which are implemented by a class.



Source Code

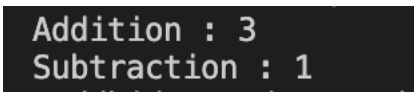
```
import java.io.*;
  // Add interface
interface Add{
    int add(int a,int b);
  }
// Sub interface
interface Sub{
          int sub(int a,int b);
  }
// Calculator class implementing Add and Sub
class Cal implements Add , Sub
  {
// Method to add two numbers
```

```
        public int add(int a,int b){
        return a+b;
    }
        // Method to sub two numbers
        public int sub(int a,int b){
        return a-b;
    }
}
class Example{
    // Main Method
    public static void main (String[] args){

        // instance of Cal class
        Cal x = new Cal();
        System.out.println("Addition : " + x.add(2,1));
        System.out.println("Substraction : " + x.sub(2,1));
    }
}
```

Outputs

Addition : 3
Subtraction : 1



POST LAB EXERCISE

1. Can a functional interface extend another interface?

Yes, a functional interface **can extend another interface**,

but it must still have **only one abstract method** in total.

If extending another interface results in more than one abstract method,

then it will no longer be a functional interface.

2. Which feature was introduced in interfaces starting from Java 8?
    A. Constructors
    B. Private methods
    C. Default and static methods
    D. Final classes

Correct Answer:

Default and static methods

ASSESSMENT

Description	Max Marks	Marks Awarded
Pre Lab Exercise	**5**	
In Lab Exercise	**10**	
Post Lab Exercise	**5**	
Viva	**10**	
Total	**30**	

Faculty Signature	