Q)

Declare an interface called “MyFirstInterface”. Declare integer type variable called “x”. Declare an abstract method called “display()”.

1. Try to declare the variable with/without public, static and final keywords. Is there any difference between these two approaches? Why?
2. Declare the abstract method with/without abstract keyword. Is there any difference between these two approaches? Why?
3. Implement this into a class called “IntefaceImplemented” . Override all the abstract methods. Try to change the value of x inside this method and print the value of x. Is it possible for you to change x? why?

ANSWER-

public interface MyFirstInterface

{

int x = 10;

void display();

}

1. Variable declaration with/without public, static, and final keywords:

- Public: If you declare the variable as public, it will be accessible from other classes that implement the interface. Other classes can access and modify the variable.

- Static: If you declare the variable as static, it belongs to the class rather than an instance of the class. All instances of the class will share the same value of the variable.

- Final: If you declare the variable as final, its value cannot be changed once assigned. It acts as a constant.

Using or omitting these keywords in the interface will not affect the behavior of the variable. By default, interface variables are public, static, and final. So, the variable `x` in the interface `MyFirstInterface` will have these characteristics.

2. Abstract method declaration with/without the abstract keyword:

- Abstract: When you declare a method as abstract, it means that the method doesn't have an implementation in the interface itself. The implementation is left to the classes that implement the interface.

If you omit the `abstract` keyword, the method will still be treated as abstract by default since it is declared within an interface. However, explicitly including the `abstract` keyword is considered good practice for clarity and readability.

3.

public class InterfaceImplemented implements MyFirstInterface

{

public void display()

{

x = 20; // Error: Cannot assign a value to a final variable x

System.out.println(x);

}

}

In this example, when you try to change the value of `x` inside the `display()` method and print its value, you will encounter an error. This is because the variable `x` in the interface is declared as `final`, which means its value cannot be changed once assigned. Therefore, you cannot modify the value of `x` within the implementing class.