Lab6

W.A.C Fernando (26545)

1.	public interface MyFirstInterface {
	   int x = 10; // Variable declaration with or without public static final keywords.
	
	   void display(); // Abstract method declaration.
	}
	
	public interface MyFirstInterface {
	   public static final int x = 10; // Variable declaration with public static final keywords.
	
	   void display(); // Abstract method declaration.
	}
	
	public interface MyFirstInterface {
	   int x = 10; // Variable declaration without public static final keywords.
	
	   void display(); // Abstract method declaration.
	}
	
	1. There is no practical difference between these two approaches because interface variables are implicitly public, static, and final. When you declare a variable inside an interface, it is by default considered as public static final, regardless of whether you explicitly specify those keywords or not. So, both of the above declarations for the variable "x" are equivalent.
	
	public interface MyFirstInterface {
	   abstract void display(); // Abstract method declaration with abstract keyword.
	}
	
	public interface MyFirstInterface {
	   void display(); // Abstract method declaration without abstract keyword (implicit).
	}

2. In both cases, the method display() is an abstract method. The interface itself is implicitly abstract since it contains at least one abstract method. An abstract method is a method without a method body (implementation). Any class that implements this interface must provide a concrete implementation of the abstract method.

```
public class InterfaceImplemented implements
MyFirstInterface {

   // Implementing the abstract method from the
interface
   @Override
   public void display() {
      // Trying to change the value of 'x'
      x = 20; // This will result in a compilation
error.
      System.out.println("Value of x: " + x);
   }

   public static void main(String[] args) {
      InterfaceImplemented obj = new
InterfaceImplemented();
      obj.display();
   }
}
```

3. The reason why you cannot change the value of "x" is that interface variables are implicitly considered as public static final, which means they are constants and their values cannot be modified once assigned. The final keyword ensures that the value remains constant throughout the program execution.

   Thus, any attempt to change the value of "x" inside the display() method or any other method of the implementing class will lead to a compilation error. The variable "x" will always have the value assigned to it in the interface, which is 10 in this case.