PDGMS Malshani   28614

Practical 05

1) *

a)
```
public interface MyFirstInterface
 {
   int x = 10;

   void display();
}
```

b)
```
public interface MyFirstInterface {
   int x = 10; // Equivalent to "public static final int x = 10;"
}
```

c)
```
public class IfImplemented implements MyFirstInterface {
   @Override
   public void display() {
      x = 20; // This will result in a compilation error
      System.out.println("Value of x inside display(): " + x);
   }
}
```

When a class implements an interface, it must provide explicit (non-contextual) implementations for all abstracts declared in the interface. By default, the InterfaceImplemented class implements MyFirstInterface.
But since the x variable is final (always), you cannot change its value in the implementer class. Trying to change the value of x in the display() method shown in the code above will result in a compile error. An example message might look something like “Could not assign value to last type 'x'. This is because interface variables are immutable and their values cannot be changed after initialization.
In Java interfaces, variables can be used as constants (constant and final) and their values are set at compile time. so no attempt to change its value in the application class is allowed.