Assignments on Annotations

1) Create a custom annotation called **@Test** which can be only applied on a method implying that the following method is a test-case. (Is it possible to restrict the annotation to be applied only on a non-static method?)

2) Build a custom annotation called **@Info**, which can be used by developers on a class, a property, or a method. The developer can provide the following information when using this annotation:

   a) AuthorID: <<Developers ID>> - (Mandatory Input)
   b) Author: <<Developers name>> - (Optional Input)
   c) Supervisor: <<Developers Supervisor>> - (Optional Input)
   d) Date: <<"String Date">> - (Mandatory Input)
   e) Time: <<"String Time">> - (Mandatory Input)
   f) Version: <<Numerical Version >> - (Mandatory Input)
   g) Description: <<Description of the class, method, or property >> - (Optional Input)

3) Create a custom annotation called **@Execute** to be applied on methods. Placing the **@Execute** method on a method implies that method should be invoked using Reflection API (*Invoking the method using Reflection API is out of scope of this assignments*). The annotation **@Execute** should have an optional property "sequence" which can be given values such as 1, 2, 3… in the order of priority. In case the sequence property is not used the API may invoke methods in random order.

   E.g.

   Class MyClass{

   @Execute(Sequence=2)
   Public void myMethod1(){
   }

   @Execute(Sequence=1)
   Public void myMethod2(){
   }

   @Execute(Sequence=3)
   Public void myMethod3(){
   }
   }

   Note: The above annotation tells the system that the invocation should be in the order: myMethod2 first, followed by myMethod1 and finally myMethod3