Aula Prática 11

1. Implemente os seguintes métodos estáticos da classe `ReflectionUtils`:
   a. a) `bool ImplementsInterface(Type t, Type tIntf)` o qual verifica se o tipo representado por `t` implementa a interface representada por `tIntf`.
   b. b) `bool IsDisposablePattern(Type t)` o qual verifica se o tipo representado por `t` implementa o padrão *Disposable*, isto é, se contém todos os métodos necessários à concretização deste padrão.
   c. c) `bool isBoxedInstance(object o)` que determina se o objecto o corresponde à versão *boxed* de uma instância de tipo valor.
   d. `bool isCompatibleWithDelegate(Type t, MethodInfo mi)` que retorna *true* se o método `mi` tiver uma assinatura compatível com o tipo *delegate* representado por `t`.

2. Implemente o método `public static MethodInfo[] GetCompatibleMethods(object o, Type t)` da classe `DelegateUtils` que recebe como parâmetros o objecto `o` e representante `t` de um tipo delegate, retorna um *array* com os representantes dos métodos de `o` que são compatíveis com o tipo *delegate* representado por `t`.

3. Pretende-se desenvolver um sistema para realizar testes unitários sobre métodos públicos de qualquer tipo. O sistema assume que todos os métodos públicos a testar estão anotados com o atributo `MethodTestAttribute`.
   a. Implemente o atributo `MethodTestAttribute`, o qual só poderá ser aplicado uma vez a métodos e que recebe no construtor uma `String` com o valor que o método deverá retornar.
   b. Implemente o método estático `boolean IsMethodToTest(MethodInfo mi)` que retorna `true` se o método representado por `mi` for relevante para o sistema de testes. Para um método ser relevante para o sistema de testes tem de estar anotado com `MethodTestAttribute`, não pode ter parâmetros e o retorno não pode ser `void` nem *array* ou *delegate*.
   c. Implemente o tipo valor genérico `Pair<T,W>` o qual agrega um `T` e um `W` e expõe propriedades para leitura e escrita destes membros.
   d. Implemente o método estático `Pair<MethodInfo, boolean>[] testAllMethods(Object obj)` da classe `UnitTest`, que recebe o objecto a testar. O método retorna um *array* onde cada entrada é um par com o representante do método testado e o resultado do teste (`true` se o valor de retorno é o esperado, `false` caso contrário). Assuma que os tipos dos valores retornados redefinem o método `ToString()` de `Object`. Na classe `MethodInfo` o método de instância `Object Invoke(Object thisRef, Object[] parameters)` realiza a chamada ao método representado pela instância de `MethodInfo`.