6. [3] Considere a classe `Properties` com o método estático `Setter` que produz uma nova instância do tipo `IAction<T>` --- `interface IAction<T>{void Set(T val);}` -- que realiza **todas** as validações indicadas nos *custom attributes* da propriedade especificada e em caso de sucesso afecta a respectiva propriedade. Caso alguma validação falhe dá a excepção `IllegalArgumentException`.
   Usando a API de `System.Reflection` implemente uma solução incluindo um dos *custom attributes* do exemplo seguinte.
   Por exemplo, dada a classe de domínio `Stock` considere a seguinte utilização do método `Setter`.

```
public class Stock {
  ...
  [Min(73)]
  [Max(121)]
  virtual int Quote { get; set; }
  [Max(58)]
  virtual int Price { get; set; }
  void AddRate(double tax) {... }
}
```

```
Stock apple = new Stock("Apple", "Dow Jones");
IAction<int> quoteSetter = Properties.Setter<int>(apple, "Quote");
IAction<int> priceSetter = Properties.Setter<int>(apple, "Price");
quoteSetter.Set(30);  // IllegalArgumentException
quoteSetter.Set(200); // IllegalArgumentException
quoteSetter.Set(100); // Ok
priceSetter.Set(30);  // Ok
```