Tecnología y paradigmas de programación. Laboratorio 4. Grupos 1, 2 y 3.

1. Implementar el tipo genérico Par<T>, con dos atributos de tipo T, derivando de `IComparable`. Implementar `CompareTo` de modo que realice una comparación lexicográfica (un ejemplo de ordenación lexicográfica es el orden alfabético):
   -Si el primer atributo del primer objeto es menor que el primer atributo del segundo, aquel va antes en la ordenación.
   -Si el primer atributo del primer objeto es mayor que el primer atributo del segundo, aquel va después en la ordenación.
   -Si el primer atributo del primer objeto es igual que el primer atributo del segundo, se procede análogamente con los segundos atributos.
   Ejemplos: ['a','x'] va antes que ['b','c'] pero después de ['a','d']

   Comprobar que se puede usar el método `Sort` de los ejemplos de teoría (`generics.bounded`), usándolo en una aplicación de consola con un array de `Par`. Para ello incorporar el fichero fuente `Algorithms.cs` al proyecto del ejercicio de modo que la solución sea autocontenida. Esto es: el fuente ha de estar en el árbol de directorios de la solución. Una opción es copiar y pegar el archivo con el explorador de archivos de Windows y después usar añadir item existente usando el explorador de soluciones de Visual Studio para añadir el fuente al proyecto de la biblioteca de clases en donde está `Par`. Observar como en la aplicación de consola dentro del mencionado proyecto se generan los elementos del array, encoger el rango para que sea más probable que se den pares con el primer elemento igual entre ellos. Si se quiere probarlo con letras, se puede convertir el entero que proporciona `random` a carácter, usando el rango de 65 a 90. Obviamente, para mostrar los elementos por la pantalla, es necesario implementar `ToString`.

   Implementar el método `Max (`añadiéndolo al fuente incorporado `Algorithms.cs)`, que recibe un array de `T` y devuelve el mayor elemento del array, según la ordenación definida para `T`. No usar `Sort` para implementarlo (no ordenar de menor a mayor y devolver el último), usar `CompareTo`. Probarlo con un array de `int` y con un array de `Par` de cualquier tipo.

2. Partiendo del ejemplo de teoría que se puede examinar en el proyecto `enumerables` dentro de la solución `generics` (el de la sucesión de Fibonacci), realizar un ejemplo análogo, que permita mostrar por la pantalla una secuencia de números primos de tamaño dado, usando `foreach` y usando el iterador correspondiente, deberían servir tal cual los métodos definidos en el proyecto mencionado. Probarlo mediante una aplicación de consola. Si el tamaño de la secuencia es 5, la lista de primos es [2, 3, 5, 7, 11] ya que el primer primo (en la posición 0) es 2. El nombre de la clase puede ser `Prime.`

NOTA: recordar entregar la tarea autónoma antes del siguiente laboratorio, en el examen de prácticas se usará.