Nombre:	**Nota**
Curso:	

Tiempo estimado	Tipo de actividad
2 horas	**Evaluación**

Unión Europea
Fondo Europeo de Desarrollo Regional
Una manera de hacer Europa

GENERALITAT VALENCIANA
Conselleria d'Educació, Cultura i Esport

IES Mare Nostrum

Normas

- **El código entregado tiene que compilar sin problemas**
- **Se tiene que proporcionar un main que pruebe todas las funciones implementadas, si no, no se corregirá el examen**
- **Existen algunos errores (comentados en clase) que hará que la nota máxima de este examen sea de 4**
- **La nota mínima en esta prueba para poder hacer media con el resto de las pruebas es de 4 sobre 10**
- **No puedes hacer trampas**

Leyenda

- **(A)** → Si no implementa alguno de estos puntos, la nota del exámen será de 0.
- **(B)** → Si no se implementa alguno de estos puntos, se bajará 4 puntos sobre 10 al examen
- **(C)** → Si no se implementa alguno de estos puntos, se bajará 3 puntos sobre 10 al examen
- **(D)** → Si no se implementa alguno de estos puntos, se bajará 2 puntos sobre 10 al examen
- Cada fallo penalizará en función de la gravedad del fallo

Implementa una clase que sea una lista hecha con genéricos.

- La clase se llamará ExList
- **(A)** La lista se implementará usando un array. El número de elementos que tiene el array no tiene por qué coincidir con el número de elementos de la lista.
- Las properties que hay que implementar son:
  - **(A)** Count → Devuelve el número de elementos que hay en la lista.
  - **(A)** Capacity → Devuelve la capacidad interna de la lista.
  - **(A)** First → Devuelve el primer elemento de la lista, si no se puede, hay que lanzar una excepción.
  - **(A)** Last → Devuelve el último elemento de la lista, si no se puede, hay que lanzar una excepción.
  - **(D)** Reversed → Devuelve una copia de la lista pero con los elementos en el orden inverso.
- Los métodos a implementar son:
  - **(A)** GetElementAt(index) → Devuelve el elemento que está en la posición "index". En caso de que no se pueda, se debe devolver un valor por defecto.
  - **(A)** SetElementAt(index, element) → Pone (remplaza) el elemento en la posición "index".
  - **(A)** Add(element) → Añade un elemento al final de la lista.
  - **(B)** RemoveAt(index) → Quita de la lista el elemento que está en la posición "index". **Esta función no crea un array nuevo.**
  - **(A)** Clear() → Quita todos los elementos de la lista. **Esta función no crea un array nuevo.**
  - **(A)** Insert(index, element) → Inserta el nuevo elemento en la posición especificada. Por ejemplo, si inserto "hola" en la posición 2, y luego llamo a GetElementAt(2) me devolverá hola. Esta función hará que la lista tenga un elemento más.
  - **(C)** IndexOf(element) → Devuelve la posición en la que se encuentra un elemento.
  - **(C)** Contains(element) → Dice si un elemento está en la lista.
  - **(C)** IndexOf(delegate) → Devuelve la posición en la que se encuentra un elemento.
  - **(C)** Contains(delegate) → Dice si un elemento está en la lista.
  - **(C)** Visit(delegate) → Pasa por todos los elementos de la lista invocando a una lambda que se le pasa como parámetro con cada elemento recorrido.
  - **(D)** Sort(delegate) → Ordena la lista con el criterio de un delegado que se le tiene que pasar como parámetro.
  - **(B)** Filter(delegate) → Devuelve una ExList con los elementos que superen el filtro que es un delegado.
  - **(D)** Reverse() → Le da la vuelta a la lista.
  - **(D)** Clone() → Devuelve una copia de la lista.
  - **(A)** ToArray() → Devuelve un array (una copia, no el array interno) con todos los elementos de la lista y sólo con los elementos de la lista.