Ejercicios Clases Anónimas y Métodos Extensores

Descargar estos ejercicios

Índice

Ejercicio 1

Crea una clase Libro con los siguientes Propiedades públicas `Titulo`, `Autor`, `Editorial`, `NumPaginas` (short), `ISBN` y `Reseña` (de tipo `string`) y no definas un constructor para la misma.

> 📌 **Nota:** Las propiedades deben ser de solo lectura pero deben poder asignarse al menos una vez. Por lo cual el '*setter*' lo definiremos con la palabra reservada `init`. Ej. `string Titulo { get; init; }`

Dada la siguiente definición de una lista de objetos libro …

```
List<Libro> libros = new List<Libro>
{
    new Libro
    {
        Titulo = "Don Quijote de la Mancha",
        Autor = "Miguel de Cervantes",
        Editorial = "Editorial EDAF, S.A",
        NumPaginas = 765,
        ISBN = "9788441405298",
        Reseña = @"El libro, sinopsis… Nos presentan a este personaje como un loco
                   trastornado a causa de las novelas de caballerías, pero,
                   ¿Quién dice que el señor Quijana era sólo eso?
                   ¿Por algún motivo será la cumbre de la literatura española verdad?
                   Y aquí se plantea la duda héroe o simplemente viejo loco."
    },
    ...
};
```

Termina de definir la lista de libros utilizando la siguiente sintaxis utilizada en el ejemplo con los siguientes datos ...

Libro 2

- **Título**: El camino
- **Autor**: Miguel Delibes
- **Editorial**: Espasa
- **Páginas**: 187
- **ISBN**: 9788467023664

- **Reseña**: Una de las más importantes obras de Miguel Delibes cuenta la historia de un niño , Daniel el Mochuelo, que tiene que trasladarse a la ciudad para cursar bachillerato. Una noche antes de partir Daniel recordará todo lo que le ha ocurrido en este lugar, sus amigos, sus peripecias, y descubrirá que su camino está en esa aldea, unido a lo que ha sido hasta ese momento su vida. Nostálgica novela realista a través de la cual podemos aprender que nunca sabemos lo que tenemos hasta que se nos ha escapado.

Libro 3

- **Título**: Cien años de soledad
- **Autor**: Gabriel García Márquez
- **Editorial**: Alfaguara
- **Páginas**: 562
- **ISBN**: 9788420471839
- **Reseña**: Muchos años después, frente al pelotón de fusilamiento, el coronel Aureliano Buendía había de recordar aquella tarde remota en que su padre lo llevó a conocer el hielo.» Con estas palabras empieza una novela ya legendaria en los anales de la literatura universal, una de las aventuras literarias más fascinantes de nuestro s iglo.Millones de ejemplares de Cien años de soledad leídos en todas las lenguas y el premio Nobel de Literatura coronando una obra que se había abierto paso «boca a boca» -como gusta decir el escritor-son la más palpable demostración de que la aventura fabulosa de la familia Buendía - Iguarán, con sus milagros, fantasías, obsesiones, tragedias, incestos, adulterios, rebeldías, descubrimientos y condenas, representaba al mismo tiempo el mito y la historia, la tragedia y el amor del mundo entero.

Libro 4

- **Título**: La Regenta
- **Autor**: Leopoldo Alas Clarín
- **Editorial**: Crítica
- **Páginas**: 182
- **ISBN**: 9788484326977
- **Reseña**: En La Regenta, sin lugar a dudas una de las cumbres de la novela realista, Leopoldo Alas alcanzó a cifrar de forma inolvidable uno de los motivos que obsesionaron a la narrativa europea de la segunda mitad del siglo XIX: el retrato de un carácter femenino que se debate entre el deseo y su represión , y que sufre, en este caso, las asechanzas de un galán y de un cura.La peripecia tiene como trasfondo la magistraly despiadada descripción del entorno de Ana Ozores, esa Vetusta murmuradora y provinciana en la que toda vanidad e hipocresía tiene su asiento. José Luis Gómez, tras

un minucioso análisis de las primeras ediciones de la obra, sigue el texto de la tercera(1901), revisada por Clarín y publicada poco antes de su muerte. El prólogo de S. Beser al autor y su novela en el contexto de la creación europea y española de la época, mientras que la anotación facilita la comprensión de cada uno de los pormenores del rico universo clariniano.

Libro 5

- **Título**: Los mejores cuentos de Clarín
- **Autor**: Leopoldo Alas Clarín
- **Editorial**: De Vecchi
- **Páginas**: 145
- **ISBN**: 9788431533441
- **Reseña**: Una cuidadosa selección que nos muestra la riqueza de los recursos estilísticos de este gran escritor del s. XIX.En el volumen se incluyen: Doña Berta, Benedictino, Manín de Pepa José, Zurita, Cambio de luz, y la Conversión de Chiripa.

Define la misma lista cómo un **array de tipos anónimos** …

```
var listaInmutable = new[] { … }
```

¿Qué diferencia hay entre uno y otro?

Explícalo con un comentario y pon un ejemplo de la diferencia en el programa principal.

✅ Ejercicio 2

Vamos a crear una clase denominada `Biblioteca` que tendrá como propiedades: un `Nombre` de tipo string y una **lista genérica de libros** `Libros`.

- Su constructor recibirá el nombre y la lista de libros que hemos creado en el ejercicio anterior.

  > 📌 **Nota:** Usa la clase `Libro` creada en el ejercicio anterior, pero cambiando la clase a tipo **record**.

- Crea un método público denominado `BuscaPorISBN` que reciba una cadena con el ISBN y devuelva el primer libro con ese ISBN o null si no lo encuentra.
- Primero vamos a practicar con el concepto '*de tipos anónimos*' para guardar solo los datos referentes a un préstamo de un `Libro` en un fichero. **Sin definir una nueva clase o tipo Préstamo**.
  - Para ello, vamos a crear un método público en la clase Biblioteca denominado `Presta`. Este método recibirá dos `string`: `dni` del socio `ISBN` del libro a prestar.
  - Buscaremos el `ISBN` en la lista de libros y si lo encontramos, crearemos un nuevo objeto anónimo var préstamo con las propiedades `DNI`, `Titulo` e `ISBN`.

  > 📌 **Nota:** En caso de no existir el ISBN en la biblioteca generaremos una BliblitecaException con el mensaje correspondiente.

  - Por último, añadiremos a un fichero denominado **prestamos.txt** la cadena resultado de pasar a **ToString** este objeto anónimo seguida de un salto de línea.

  > 📌 **Nota:** Gestionaremos el acceso al fichero con una o más sentencias using.

- Crea otro método público `EstaPrestado` al que se le pasará el `ISBN` de un libro y devolverá un Booleano que nos indicará si el libro se encuentra prestado o no,

  > 📌 **Nota:** Gestionaremos el acceso al fichero con una o más sentencias `using`. Además, se puede usar, por ejemplo, el método `Contains` de la clase string.

- Crea un método público `CuentaLibrosConNumeroDePaginasMenorA` que reciba un valor entero y te devuelva la cantidad de líbros con un número de páginas menor a ese valor entero.
- Crea un método público `EliminaPorAutor` que reciba el nombre de un autor y borre de la biblioteca aquellos libros de ese autor.
- Redefine el método `ToString()` en `Libro` **creando un tipo anónimo** con `Libro`, `Autor`, e `ISBN` y devolviendo su ToString.

- Redefine un método `AutorTitulo` al que le llega un ISB y devuelve el libro de la biblioteca que coincida con el ISB, pero esta vez solo nos interesará sacar la información de Título y Autor, por lo que se creará un **Tipo Record** para hacerlo.
- Define la clase estática BibliotecaJson con un **método extensor de la clase Biblioteca** `Guarda` que reciba un `string` con el nombre del fichero y guarde la biblioteca en formato JSON. Deberás **añadir las anotaciones necesarias** a las propiedades de los `record` para que se genere correctamente el fichero `biblioteca.json` con el siguiente formato:

```
{
    "nombre": "EL RINCÓN DE LEER",
    "libros": [
        {
            "titulo": "Don Quijote de la Mancha",
            "autor": "Miguel de Cervantes",
            "editorial": "Editorial EDAF, S.A",
            "numPaginas": 765,
            "isbn": "9788441405298",
            "reseña": "El libro, sinopsis… "
        },
        ...
    ]
}
```

Crea una aplicación para probar estos métodos. Por ejemplo …

```
Biblioteca biblioteca = new Biblioteca("EL RINCÓN DE LEER", libros);
biblioteca.Presta("22111333", "9788420471839");
biblioteca.Presta("22111333", "9788431533441");
Console.WriteLine(biblioteca.EstaPrestado("9788420471839"));
Console.WriteLine(biblioteca.EstaPrestado("22111444"));
Console.WriteLine(biblioteca.BuscaPorISBN("9788431533441"));
Console.WriteLine(biblioteca.BuscaPorISBN("97884551533441"));
Console.WriteLine(biblioteca.CuentaLibrosConNumeroDePaginasMenorA(400));
Console.WriteLine(biblioteca.BuscaPorISBN("9788467023664"));
biblioteca.EliminaPorAutor("Miguel Delibes");
Console.WriteLine(biblioteca.BuscaPorISBN("9788467023664"));
Console.WriteLine(biblioteca.AutorTitulo("9788431533441"));
biblioteca.Guarda("biblioteca.json");
```

Ejercicio 3

Vamos a crear un método extensor de nuestra clase `Biblioteca` en un namespace llamado `BibliotecaExtensions`.
El método se llamará `ISBNS` y devolverá un array con los ISBN de los libros en nuestra biblioteca ordenados.

✅ Ejercicio 4

Vamos a crear un par de métodos extensores de la clase `List<T>` implementada por Microsoft en `System.Collections.Generic`.

Para ello define el namespace `System.Collections.Generic.ListExtensions` y dentro de él crea las clases que estimes oportunas para que ...

- Tener un método extensor `SecuenciaAleatoria` que reciba el número de elementos N a generar y devuelva una lista de elementos del mismo tipo con N elementos de la lista original escogidos de forma aleatoria y no repetidos.
- Tener un método extensor `SecuenciaAleatoriaConRepeticiones` que reciba el número de elementos N a generar y devuelva lo mismo que el anterior pero admitiendo repeticiones.

Prueba tus métodos extensores con el siguiente código ...

```
try
{
    List<int> numeros = new List<int> { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 };
    Console.WriteLine(string.Join(", ", numeros.SecuenciaAleatoria(5)));
    Console.WriteLine(string.Join(", ", numeros.SecuenciaAleatoriaConRepeticiones(5)));
    Console.WriteLine(string.Join(", ", numeros.SecuenciaAleatoriaConRepeticiones(15)));
    Console.WriteLine(string.Join(", ", numeros.SecuenciaAleatoria(15)));
}
catch (ListExtensionException e)
{
    Console.WriteLine(e.Message);
}
```

También deberemos tener en cuenta que ...

1. No se deben repetir las secuencias.
2. No debes repetir código en la implementación de tus métodos extensores.
3. `numeros.SecuenciaAleatoriaConRepeticiones(15)` no debe producir excepciones aunque la lista de números sea menor que la nueva secuencia porque admite repeticiones.
4. `numeros.SecuenciaAleatoria(15)` debe producir excepción porque tiene menos elementos en la secuencia a elegir que los solicitados.