**Colecciones**

**¿Que son las colecciones?**

Son clases que pertenecen al namespace System.Collection.Generic.

Estas clases permiten almacenar elementos.

Son clases genéricas.

No tienen las limitaciones de los arrays, sin embargo, consumen mayores recursos que estos.

**Importancia de las Colecciones**

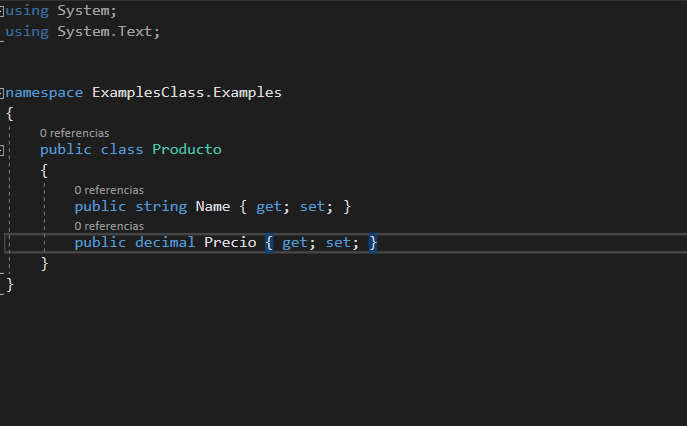
Las colecciones en C# facilitan la gestión, es decir el almacenamiento, la recuperación, el ordenamiento y operaciones de manipulación de grupos o conjuntos de datos.

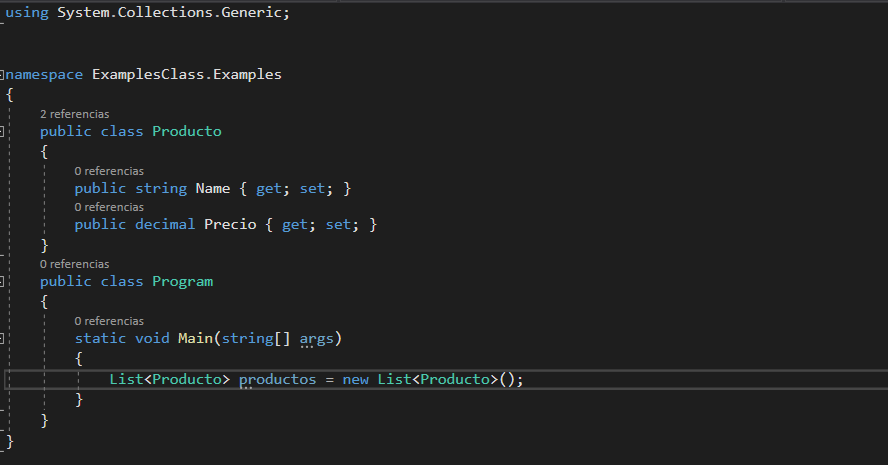
**Colecciones más utilizadas:**

* **List<T>:** Son parecidos a los arrays, pero con métodos adicionales para agregar, eliminar, ordenar y buscar etc.
* **Queue<T>:** Las “Colas”. Un elemento entra y otro sale. El primero en entrar es el primero en salir.
* **Stack<T>:** Parecidos a las Queue, pero con la diferencia que el primero en entrar es el último en salir.
* **Dictionary<TKey,TValue>:** Almacena elementos con estructura clave-valor.

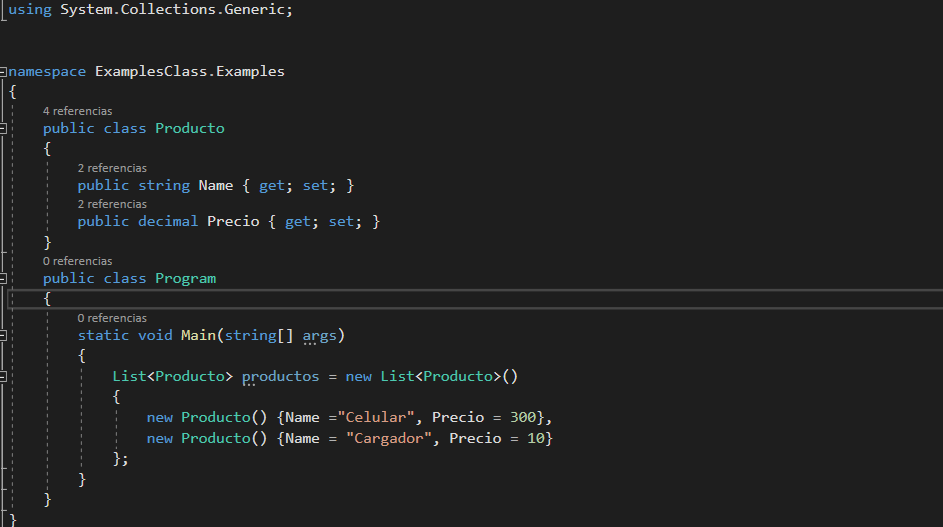
**Ejemplo:**

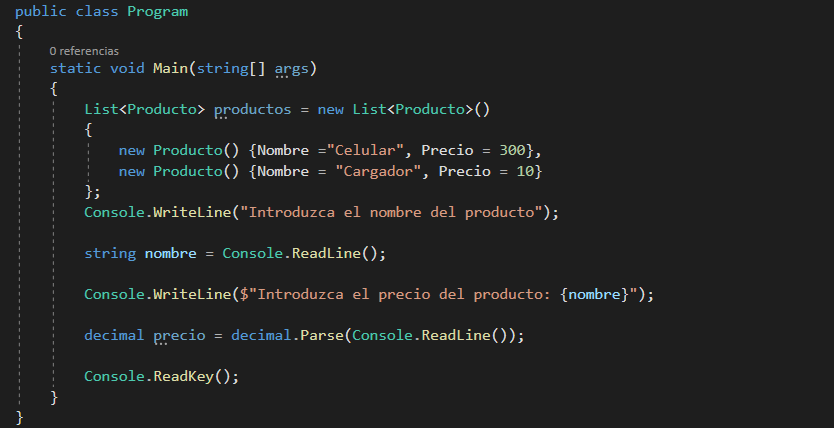
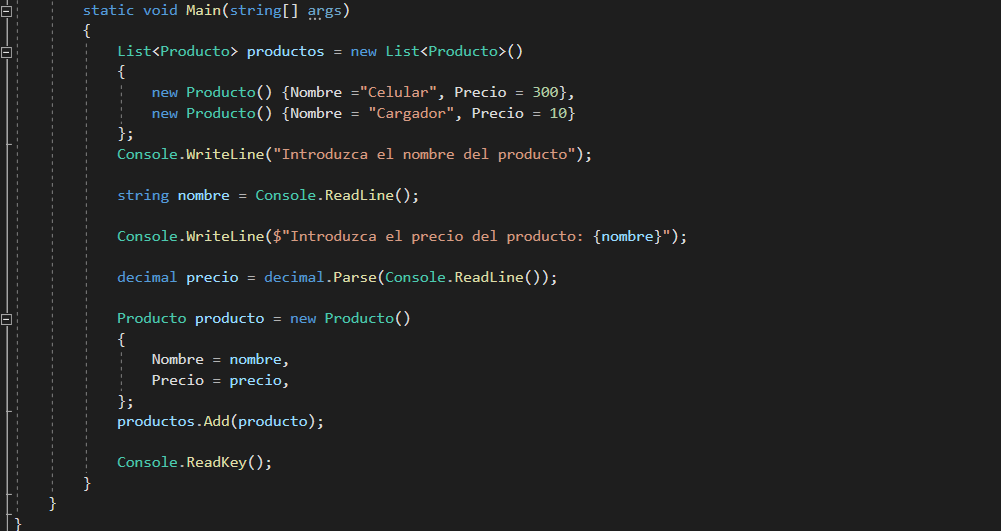
Imagínate que creas un programa donde te almacene los nombres y los precios de los productos de una tienda.

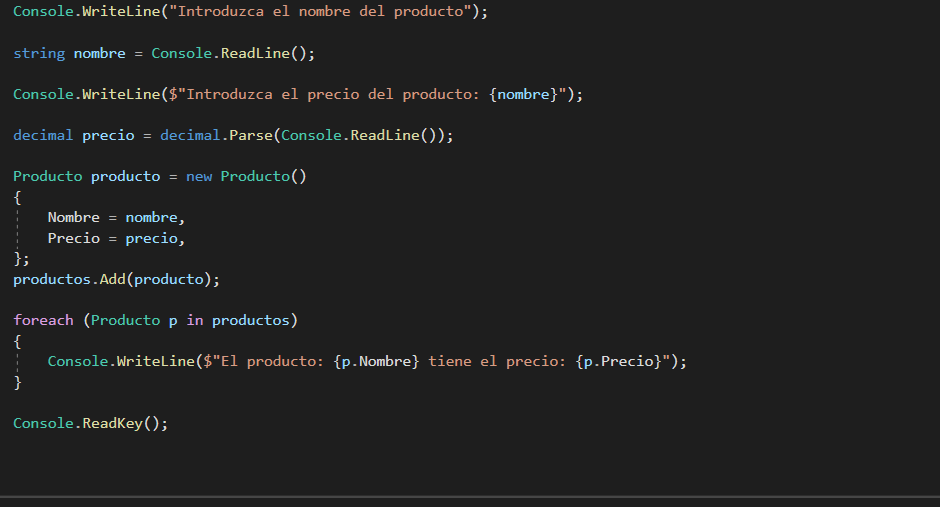
* Primero se crearía la clase entidad:
* Luego importamos el namespace “System.Collection.Generic.” para utilizar la colección “List”. Declaramos en el método main de nuestra clase funcional la colección “List<T>” de tipo “Producto” con el nombre “productos”:



* Lo inicializaremos con unos elementos integrados:



* Luego haríamos que el usuario pueda meter unos nuevos productos haciendo lo siguiente:
* Nosotros tendríamos que registrar ese nuevo producto con el método “Add()” que trae la colección “List”, ejemplo:
* Creamos un objeto de tipo “Producto” llamado “producto” y lo inicializamos con el nombre y el precio que el usuario ingresó, y utilizamos el método “Add()” para meterlo en la colección list “productos”.
* Ahora simplemente crearíamos un “foreach” el cual le mostrará los productos al usuario.



**Expresiones Lambda**

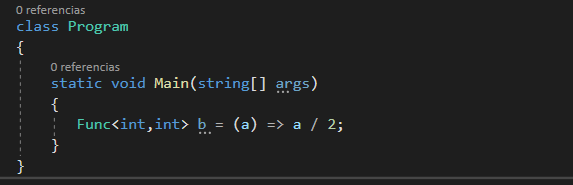
**¿Qué son?**

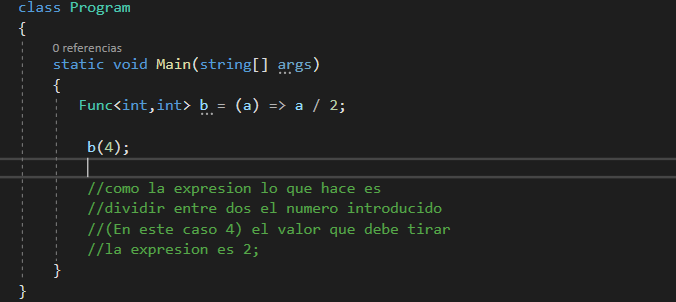
Una expresión lambda es una función anónima que puede contener expresiones e instrucciones

**Importancia de las expresiones Lambda**

Por medio de expresiones lambda podemos referenciar métodos anónimos o métodos sin nombre, lo que nos permite escribir códigos más claros y concisos que cuando usamos clases anónimas.

**Ejemplo:**

* Las expresiones lambda se inicializan con la palabra “Func” como se muestra a continuación:
* Primero se escribe dentro del “Func” el tipo de parámetro que va a recibir (en este caso “int”) y luego se escribe el tipo de parámetro que va a dar la expresión (de igual manera es “int”) a continuación se escribe el nombre con el que llamaremos a la expresión el cual es “b”, lo que está adentro del paréntesis es el nombre de la variable que vamos a recibir el cual es “a” y señalando con un “=>” escribimos lo que queremos que haga nuestra expresión con la variable “a”. un ejemplo de cómo función es el siguiente:



* Esta expresión arroja un número (en este caso un 2) el cual lo puede introducir tanto en un método como en un “Console.WriteLine()” para mostrárselo al usuario

**LINQ**

**¿Qué es LINQ?**

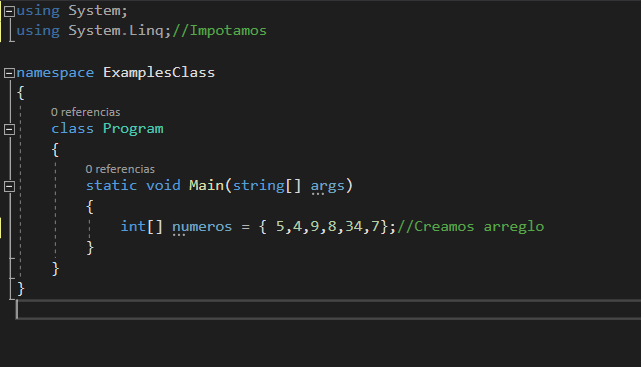
LINQ (Language Integrated Query) es una sintaxis de consulta uniforme en C # y VB.NET utilizada para guardar y recuperar datos de diferentes fuentes.

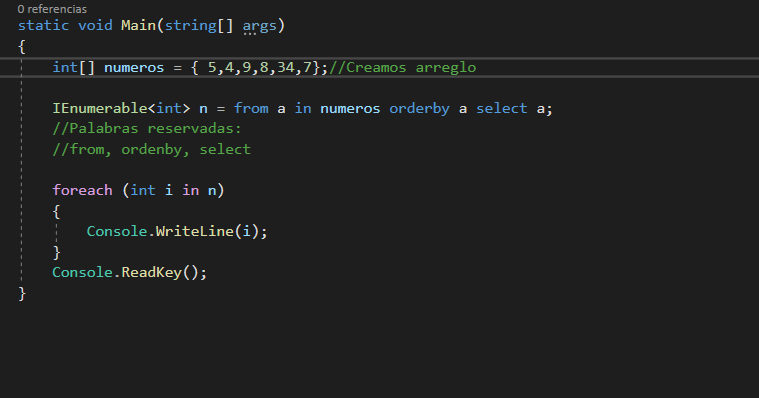
**Importancia de LINQ**

LINQ Proporciona unos métodos de manejo de datos con su sintaxi lo cual permite mayor facilidad para ordenar, guardar, recuperar y eliminar datos de un arreglo o de una colección

**Ejemplo:**

* Importamos el nameespace “System.Linq ”, vamos ordenar un arreglo de tipo “int” con unas palabras reservadas de LINQ



* ****Las palabras que ocuparemos son: from, ordenby, select
* ****Esto es lo que nos arroja el programa:
* Como puede observar el arreglo ya está ordenado sin necesidad de escribir un método que o un algoritmo de ordenamiento