Linq-запросы.

1. Что такое linq.
2. Два варианта синтаксиса.
3. Отложенное выполнение.
4. Провайдеры данных
5. Стратегии композиций
6. Стратегии проекций

1. Спросить студентов: Что такое Linq?

Linq(**Language Integrated Query**) – это синтаксис в .net-языках, спроектированный для работы с источниками данных, так как коллекции, базы данных, XML-документы и пр.

Запрос Linq – это выражение, которое производит некоторые операции над выборкой данных.

Два варианта синтаксиса:

1. Fluent API/Метод-синтаксис (пример)
2. Query Syntax/Декларативный синтаксис
   1. Сравнение двух подходов:
      1. Декларативный синтаксис более читаем на коротких сценариях, Fluent API – на более длинных.
      2. Стиль написания: декларативный синтаксис больше похож на SQL, метод-синтаксис — на вызовы методов C#.
      3. Декларативный синтаксис поддерживает не все возможности LINQ. В частности нет поддержки работы с:
         1. Методами агрегирования Sum(), Count(), Max(), Min(), Average()
         2. Методами пагинации Skip(), SkipWhile(), Take(), TakeWhile()
         3. Методы для работы с порядком элементов (ThenBy(), ThenByDescending())
         4. Методы для работы с множествами (Distinct(), Union(), Intersect(), Except())
         5. В декларативном синтаксисе поддерживается join и group by, но операция группового соединения (GroupJoin) доступна только через метод-синтаксис.
         6. Все асинхронные версии методов LINQ: ToList(), ToDictionary()
         7. Экзистенциональные запросы: Any(), All(), Contains()
      4. Fluent API не поддерживает запросы с ключевым словом: let, into

Отложенное выполнение

**Отложенное выполнение (Deferred Execution)** в LINQ — это механизм, при котором запрос не выполняется в момент его создания, а откладывается до тех пор, пока к данным не будет фактического обращения. Этот подход позволяет повысить эффективность работы с данными, так как запросы выполняются только тогда, когда это действительно необходимо. К примеру, если будет разрыв по времени в выполнении цикла (foreach) состояние коллекции может быть различным в каждый момент времени.

**Методы с отложенным выполнением:**

* Where()
* Select()
* OrderBy()
* GroupBy()
* Join()
* Take()
* Skip()

Эти методы формируют запрос, но не запускают его выполнение до фактического обращения к данным.

**Методы с немедленным выполнением:**

Существуют методы, которые вызывают **немедленное выполнение** запроса и загрузку данных в память. В основном это агрегатные функции и методы возвращения коллекции. Эти методы преобразуют запрос в конкретные данные:

* ToList() — немедленно выполняет запрос и возвращает результаты в виде списка.
* ToArray() — немедленно выполняет запрос и возвращает результаты в виде массива.
* First() / FirstOrDefault() — выполняет запрос и возвращает первый элемент.
* Count() — выполняет запрос и возвращает количество элементов.
* Sum(), Max(), Min() — агрегатные функции, которые немедленно выполняют запрос.

Стриминговое чтение данных(???)

**Провайдеры данных**

**Провайдеры данных LINQ** — это компоненты, которые позволяют выполнять запросы LINQ к различным источникам данных, таким как базы данных, коллекции в памяти, XML-документы и другие источники.

Пример:

1. Коллекция в памяти(IEnumerable)
2. Набор данных из Database (IQueryable)
3. Набор данных из XML
4. Etc

**Три части операций запроса**

1. Источник данных
2. Текст запроса
3. Выполнение запроса

При использовании декларативного синтаксиса всегда используются 3 предложения:

From – где указывается источник данных;

Where – фильтры;

Select – возвращаемые элементы данных.

Выражение должно начинаться с ключевого слова [from](https://learn.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/from-clause) и должно заканчиваться [select](https://learn.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/select-clause) или [group](https://learn.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/group-clause). Между 1 и 2 предложением могут находится [where](https://learn.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/where-clause), [orderby](https://learn.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/orderby-clause), [join](https://learn.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/join-clause), [let](https://learn.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/let-clause), и может быть дополнительный from, может быть into, join, group.

Выражение может возвращать 2 типа: var или IEnumerable<T>

Три основных возможных сценария linq:

1. Выборка данных без изменения элементов (сортировка).
2. Выборка данных с трансформацией.
3. Извлечение информации о последовательности (агрегатные функции).

Описать все ключевые слова let, from, select и пр.( <https://learn.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/query-keywords>)

**Стандартные методы расширения**

Привести пример запроса с декларативным синтаксисом и с синтаксисом fluent api.

Сказать, что методы linq являются именно методами расширения.

<https://vk.com/wall-61281696_13069>

**Анонимные типы**

Привести пример анонимного типа. Заодно рассказать про анонимные типы.

https://learn.microsoft.com/en-us/dotnet/csharp/fundamentals/types/anonymous-types

var v = new { Amount = 108, Message = "Hello" };

**Лямбда выражения**