**1. Что такое LINQ?**

**LINQ (Language Integrated Query)** — это технология в C#, которая позволяет выполнять запросы к различным источникам данных (массивы, списки, базы данных, XML и т.д.) в синтаксисе языка C#. LINQ делает работу с коллекциями данных удобной, универсальной и типобезопасной.

Пример простого запроса LINQ к массиву:

csharp

Копировать код

int[] numbers = { 1, 2, 3, 4, 5 };

var evenNumbers = from n in numbers

where n % 2 == 0

select n;

**2. Разница между отложенными и не отложенными операциями LINQ to Object**

* **Отложенные операции** (deferred execution) — это операции, результат которых вычисляется только в момент обращения к результатам. Примеры: Where, Select, OrderBy. Запрос не выполняется до тех пор, пока не будет произведена итерация (например, с помощью foreach).

Пример:

csharp

Копировать код

var query = numbers.Where(n => n > 2); // Запрос не выполнится до использования

* **Не отложенные операции** (immediate execution) — это операции, которые вычисляются сразу при их вызове и возвращают конкретный результат. Примеры: Count, Sum, First, ToList.

Пример:

csharp

Копировать код

int count = numbers.Count(); // Запрос выполнится немедленно

**3. Что такое лямбда-выражения?**

**Лямбда-выражения** — это краткий способ написания анонимных функций (функций без имени). Они обычно используются в LINQ для упрощения записи условий или операций над элементами коллекции.

Пример лямбда-выражения:

csharp

Копировать код

Func<int, bool> isEven = n => n % 2 == 0;

Лямбда-выражение может быть использовано в LINQ:

csharp

Копировать код

var evenNumbers = numbers.Where(n => n % 2 == 0);

**4. Группы операций в LINQ to Object**

Операции LINQ можно разделить на несколько групп:

* **Фильтрация**: Where
* **Проекции**: Select, SelectMany
* **Упорядочивание**: OrderBy, ThenBy
* **Агрегации**: Sum, Min, Max, Average, Count
* **Группировка**: GroupBy
* **Соединение коллекций**: Join, GroupJoin
* **Множество**: Distinct, Union, Except, Intersect
* **Ограничение и пропуск**: Take, Skip
* **Генерация**: Range, Repeat, Empty
* **Операции над элементами**: First, Last, Single, ElementAt

**5. Операция Where в LINQ to Object**

Операция Where используется для фильтрации элементов последовательности на основе условия.

Пример:

csharp

Копировать код

var evenNumbers = numbers.Where(n => n % 2 == 0);

**6. Операция Select**

Операция Select используется для проекции каждого элемента коллекции в новый вид. Она может преобразовывать элементы исходной коллекции.

Пример:

csharp

Копировать код

var squares = numbers.Select(n => n \* n);

**7. Операции Take, Skip**

* **Take** — возвращает первые N элементов коллекции.

Пример:

csharp

Копировать код

var firstThree = numbers.Take(3);

* **Skip** — пропускает первые N элементов коллекции и возвращает остальные.

Пример:

csharp

Копировать код

var afterThree = numbers.Skip(3);

**8. Операция Concat**

Операция Concat объединяет два источника данных.

Пример:

csharp

Копировать код

var combined = numbers.Concat(new[] { 6, 7 });

**9. Операция OrderBy**

Операция OrderBy сортирует элементы коллекции по возрастанию на основе заданного критерия.

Пример:

csharp

Копировать код

var sortedNumbers = numbers.OrderBy(n => n);

**10. Операция Join**

Операция Join используется для объединения двух коллекций на основе общего ключа.

Пример:

csharp

Копировать код

var people = new[]

{

new { Name = "John", City = "New York" },

new { Name = "Jane", City = "London" }

};

var cities = new[]

{

new { City = "New York", Country = "USA" },

new { City = "London", Country = "UK" }

};

var query = people.Join(cities, p => p.City, c => c.City, (p, c) => new { p.Name, c.Country });

**11. Операции Distinct, Union, Except, Intersect**

* **Distinct** — возвращает уникальные элементы коллекции.

Пример:

csharp

Копировать код

var uniqueNumbers = numbers.Distinct();

* **Union** — объединяет два множества, возвращая уникальные элементы.

Пример:

csharp

Копировать код

var union = numbers.Union(new[] { 6, 7 });

* **Except** — возвращает элементы из первой коллекции, которых нет во второй.

Пример:

csharp

Копировать код

var except = numbers.Except(new[] { 1, 2 });

* **Intersect** — возвращает элементы, общие для обеих коллекций.

Пример:

csharp

Копировать код

var intersect = numbers.Intersect(new[] { 1, 2, 3 });

**12. Операции First, Last, Any, All, Contains**

* **First** — возвращает первый элемент последовательности.

Пример:

csharp

Копировать код

var first = numbers.First();

* **Last** — возвращает последний элемент последовательности.

Пример:

csharp

Копировать код

var last = numbers.Last();

* **Any** — проверяет, содержит ли последовательность хотя бы один элемент, удовлетворяющий условию.

Пример:

csharp

Копировать код

bool hasEven = numbers.Any(n => n % 2 == 0);

* **All** — проверяет, все ли элементы удовлетворяют условию.

Пример:

csharp

Копировать код

bool allEven = numbers.All(n => n % 2 == 0);

* **Contains** — проверяет, содержит ли последовательность конкретный элемент.

Пример:

csharp

Копировать код

bool containsThree = numbers.Contains(3);

**Операции Count, Sum, Min, Max, Average**

* **Count** — возвращает количество элементов в коллекции.

Пример:

csharp

Копировать код

int count = numbers.Count();

* **Sum** — возвращает сумму всех элементов.

Пример:

csharp

Копировать код

int sum = numbers.Sum();

* **Min** — возвращает минимальный элемент.

Пример:

csharp

Копировать код

int min = numbers.Min();

* **Max** — возвращает максимальный элемент.

Пример:

csharp

Копировать код

int max = numbers.Max();

* **Average** — возвращает среднее значение элементов.

Пример:

csharp

Копировать код

double average = numbers.Average();