▎1. Что такое LINQ?

LINQ (Language Integrated Query) — Набор языковых и платформенных средств для написания структурированных и безопасных в отношении типов запросов к локальным коллекциям объектов и удаленным источником данным (базы данных, документы XML и т.д.)

▎2. В чем разница между отложенными операциями и не отложенными операциями LINQ to Object?

• Отложенные операции (Deferred Execution) выполняются только тогда, когда к результату происходит обращение. Например, методы Where, Select и другие не выполняют запрос, пока не будет выполнен перебор результатов.

• Не отложенные операции (Immediate Execution) выполняются сразу, как только вызывается метод. К таким методам относятся ToList(), ToArray(), Count(), которые возвращают результаты немедленно.

▎3. Что такое лямбда-выражения?

Лямбда-выражения — это анонимные функции, которые могут содержать операторные выражения и использоваться для создания делегатов или выражений. Они позволяют писать более компактный и читаемый код, особенно в LINQ-запросах. Пример: x => x > 5 — это лямбда-выражение, которое принимает параметр x и возвращает true, если x больше 5.

▎4. Какие есть группы операции в LINQ to Object? Перечислите

Операции в LINQ to Object можно разделить на несколько групп:

• Операции фильтрации: Where

• Операции проекции: Select

• Операции сортировки: OrderBy, OrderByDescending

• Операции группировки: GroupBy

• Операции объединения: Concat, Union, Intersect, Except

• Операции преобразования: ToList(), ToArray()

• Операции агрегации: Count, Sum, Min, Max, Average

• Операции поиска: First, Last, Any, All, Contains

▎5. Как используется операция Where в LINQ to Object?

Операция Where используется для фильтрации последовательностей на основе заданного условия. Например:

var results = numbers.Where(x => x > 10);

Этот код вернет все элементы из коллекции numbers, которые больше 10.

▎6. Как используется операция Select?

Операция Select используется для проекции каждого элемента последовательности в новую форму. Например:

var results = numbers.Select(x => x \* 2);

Этот код умножает каждый элемент в коллекции numbers на 2.

▎7. Как используются операции Take, Skip?

• Take: выбирает указанное количество элементов из начала последовательности.

var firstThree = numbers.Take(3);

• Skip: пропускает указанное количество элементов и возвращает оставшиеся.

var skipFirstTwo = numbers.Skip(2);

▎8. Как используется операция Concat?

Операция Concat объединяет две последовательности в одну. Например:

var combined = firstList.Concat(secondList);

Этот код объединяет элементы из firstList и secondList.

▎9. Как используется операция OrderBy?

Операция OrderBy используется для сортировки последовательности по заданному ключу. Например:

var sorted = numbers.OrderBy(x => x);

Этот код сортирует элементы в порядке возрастания.

▎10. Как используется операция Join?

Операция Join используется для объединения двух последовательностей на основе общего ключа. Например:

var result = from a in collectionA

join b in collectionB on a.Key equals b.Key

select new { a, b };

▎11. Как используются операции Distinct, Union, Except и Intersect?

• Distinct: возвращает уникальные элементы последовательности.

var uniqueNumbers = numbers.Distinct();

• Union: объединяет две последовательности, исключая дубликаты.

var unionResult = listA.Union(listB);

• Except: возвращает элементы первой последовательности, которых нет во второй.

var exceptResult = listA.Except(listB);

• Intersect: возвращает общие элементы двух последовательностей.

var intersectResult = listA.Intersect(listB);

▎12. Как используются операции First, Last, Any, All и Contains?

• First: возвращает первый элемент последовательности или выбрасывает исключение, если последовательность пуста.

var firstElement = numbers.First();

• Last: возвращает последний элемент последовательности или выбрасывает исключение, если последовательность пуста.

var lastElement = numbers.Last();

• Any: проверяет, содержит ли последовательность хотя бы один элемент.

bool hasElements = numbers.Any();

• All: проверяет, соответствуют ли все элементы последовательности заданному условию.

bool allGreaterThanZero = numbers.All(x => x > 0);

• Contains: проверяет, содержит ли последовательность определенный элемент.

bool containsFive = numbers.Contains(5);

▎13. Как используются операции Count, Sum, Min и Max, Average?

• Count: возвращает количество элементов в последовательности.

int count = numbers.Count();

• Sum: возвращает сумму всех элементов в числовой последовательности.

int totalSum = numbers.Sum();

• Min: возвращает минимальное значение в числовой последовательности.

int minValue = numbers.Min();

• Max: возвращает максимальное значение в числовой последовательности.

int maxValue = numbers.Max();

• Average: возвращает среднее значение элементов в числовой последовательности.

double averageValue = numbers.Average();

Если у вас есть дополнительные вопросы или требуется более подробная информация по какой-либо теме, не стесняйтесь спрашивать!