

БКИТ РК №2

Задание:

Как использовать функции в Any и All в LINQ?

Ответ:

Методы All, Any позволяют определить, соответствует ли коллекция определенному условию, и в зависимости от результата они возвращают true или false.

1) **Метод All** проверяет, соответствуют ли все элементы условию. Например, узнаем, у всех ли пользователей возраст превышает 20 и имя начинается с буквы Т.

Пример:

```
List<User> users = new List<User>()
{
    new User { Name = "Tom", Age = 23 },
    new User { Name = "Sam", Age = 43 },
    new User { Name = "Bill", Age = 35 }
};
```

```
bool result1 = users.All(u => u.Age > 20); // true
if (result1)
    Console.WriteLine("У всех пользователей возраст больше 20");
else
    Console.WriteLine("Есть пользователи с возрастом меньше 20");

bool result2 = users.All(u => u.Name.StartsWith("T")); //false
if (result2)
    Console.WriteLine("У всех пользователей имя начинается с Т");
else
    Console.WriteLine("Не у всех пользователей имя начинается с Т");
```

Итог программы:

У всех пользователей возраст больше 20
Не у всех пользователей имя начинается с Т

Поскольку у всех пользователей возраст больше 20, то переменная result1 будет равна true. В то же время не у всех пользователей имя начинается с буквы Т, поэтому вторая переменная result2 будет равна false.

2) **Метод Any** действует подобным образом, только позволяет узнать, соответствует ли хотя бы один элемент коллекции определенному условию:

Пример:

```
bool result1 = users.Any(u => u.Age < 20); //false
```

```
if (result1)
```

```
    Console.WriteLine("Есть пользователи с возрастом меньше 20");
```

```
else
```

```
    Console.WriteLine("У всех пользователей возраст больше 20");
```

```
bool result2 = users.Any(u => u.Name.StartsWith("T")); //true
```

```
if (result2)
```

```
    Console.WriteLine("Есть пользователи, у которых имя начинается с Т");
```

```
else
```

```
    Console.WriteLine("Отсутствуют пользователи, у которых имя  
начинается с Т");
```

Итог программы:

```
    У всех пользователей возраст больше 20
```

```
    Есть пользователи, у которых имя начинается с Т
```

Первое выражение вернет false, поскольку у всех пользователей возраст больше 20. Второе выражение возвратит true, так как у нас есть в коллекции пользователь с именем Tom.

Отметим, что операция Any имеет два прототипа.

1.1 Пример первого прототипа:

```
public static bool Any<T> (  
    this IEnumerable<T> source);
```

Этот прототип операции Any вернет true, если входная последовательность source содержит любые элементы.

1.2 Пример второго прототипа:

```
public static bool Any<T> (  
    this IEnumerable<T> source,  
    Func<T, bool> predicate);
```

Второй прототип операции Any перечисляет входную последовательность и возвращает true, если хотя бы для одного элемента из входной последовательности вызов делегата метода predicate возвращает true.

Перечисление входной последовательности source прекращается, как только predicate вернет true.

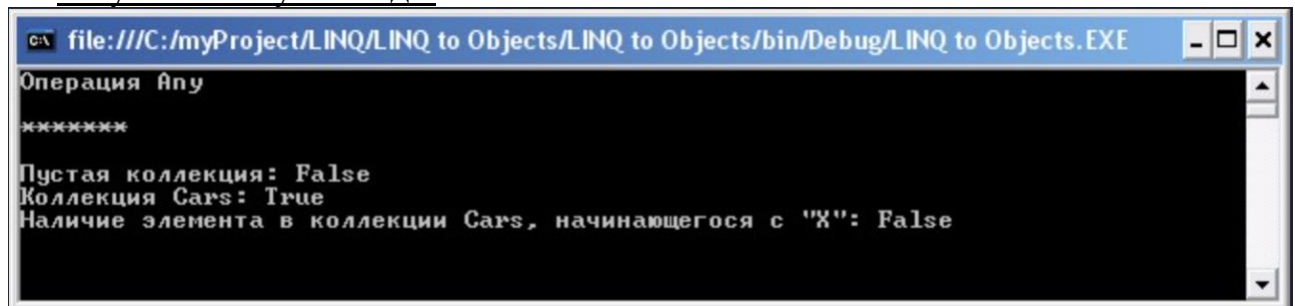
Ниже показан пример использования обоих прототипов.

Сначала воспроизводится пустая последовательность и проверяется с помощью операции Any. Затем проверяется массив Cars:

```
string[] cars = { "Alfa Romeo", "Aston Martin", "Audi", "Nissan", "Chevrolet", "Chrysler", "Dodge", "BMW", "Ferrari", "Bentley", "Ford", "Lexus", "Mercedes", "Toyota", "Volvo", "Subaru", "Жигули :)"};
```

```
bool anyNull = Enumerable.Empty<string>().Any();  
Console.WriteLine("Операция Any \n\n*****\n\nПустая коллекция: " + anyNull  
);  
bool anyCars = cars.Any();  
Console.WriteLine("Коллекция Cars: " + anyCars);  
// Используем второй прототип  
anyCars = cars.Any(s => s.StartsWith("X"));  
Console.WriteLine("Наличие элемента в коллекции Cars, начинающегося с \"X\":  
" + anyCars);
```

Результат запуска кода:



```
C:\ file:///C:/myProject/LINQ/LINQ to Objects/LINQ to Objects/bin/Debug/LINQ to Objects.EXE  
Операция Any  
*****  
Пустая коллекция: False  
Коллекция Cars: True  
Наличие элемента в коллекции Cars, начинающегося с "X": False
```