1. What is a Generation Operator?

As of now, we have discussed many LINQ Extension methods such as Select, Where, Any, etc. The Enumerable class also provides three Non-Extension Methods. They are as follows:

* Range()
* Repeat<T>()
* Empty<T>()

These three methods allow us to create some specific type of Array, Sequence, or Collection using a single expression instead of creating them manually and populating them using some loop. That means these three methods return a new sequence or collection that implements the IEnumerable<T> interface.

1. LINQ Range Method in C#:

The LINQ Range method is a static method of the Enumerable class that generates a sequence of integral numbers within a specified range. It is particularly useful when you need a simple sequence of numbers without manually initialing an array or list. The Range method takes two parameters: the start value and the count of sequential integers to generate.

The LINQ Range Method in C# generates a sequence of integral (integer) numbers within a specified range. The following is the signature of this method.

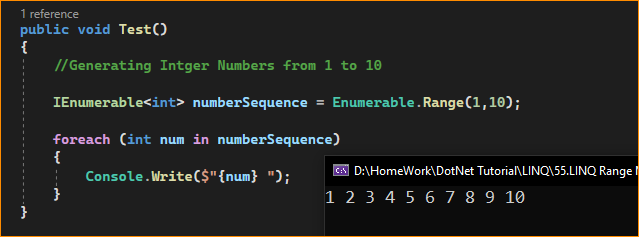


This method takes 2 integer parameters. The first parameter (i.e., int start) specifies the value of the first integer in the sequence. The second parameter (i.e., int count) specifies the number of sequential integers to be generated. The return type of this method is IEnumerable<int>, which will contain a range of sequential integer numbers.

Note: When the count value is less than 0, or when the start + count – 1 value is larger than the System.Int32.MaxValue, then it will throw **ArgumentOutOfRangeException**.

1. Example to Understand LINQ Range Method in C#:

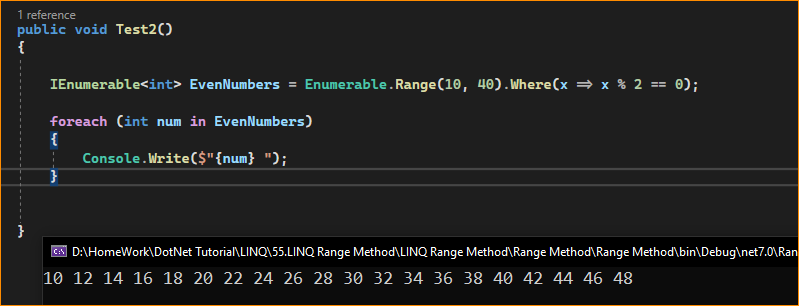
Let us see an example of the LINQ Range Method in C#. In the following example, we generate a sequence of integer numbers from 1 to 10 using the Range Method.



Once you run the application, it will print the values from 1 to 10 as expected.

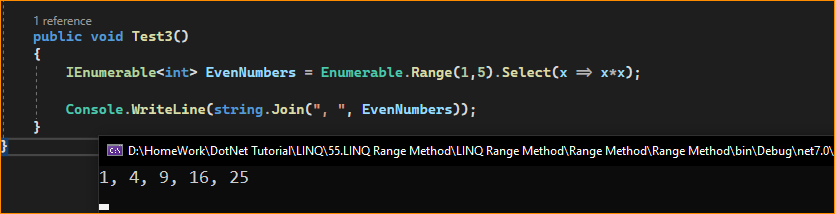
1. Example to Understand LINQ Range Method with Where Extension Method in C#:

Let us see an example to understand how to use the LINQ Range Method with the LINQ Where Extension Method. The following example prints all the even numbers which are present between 10 and 40. Here, first, we need to generate the sequence using the Range method, and then we need to filter the data using the Where extension method.



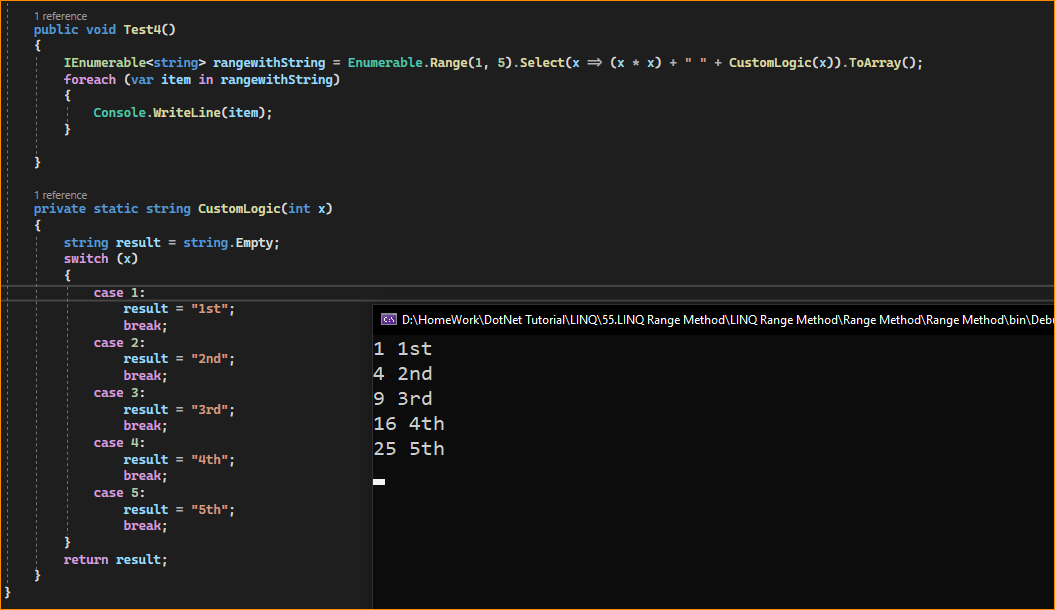
1. Example to use Range Method with Select Extension Method in C#:

The following example shows how to generate a sequence of integers from 1 to 5 and then return the square of each number.



1. Example to Understand LINQ Range Method with String Type

In the following example, first, we generate a sequence and pass the sequence to our CustomLogic method. Then, we merge the sequence with the return value from the CustomLogic method and return the result as IEnumerable<string>.



Note: This method is implemented by using deferred execution. The immediate return value is an object that stores all the information that is required to perform the action. The query represented by this method is not executed until the object is enumerated by calling its GetEnumerator method directly or using a for each loop.

1. When to Use the LINQ Range Method in C#?

The LINQ Range method is a convenient way to generate a sequence of integers, which can be particularly useful in a variety of scenarios:

**Loops**: When you need a simple for-loop-like iteration in LINQ query syntax and don’t want to use the traditional for or for each loop construct.

**Generating** **Sequences**: Quickly create a sequence of numbers for testing or prototyping or when the exact numbers aren’t important, just because you have a sequence.

**Indexing**: When you need a sequence of indices to iterate over another collection or to perform an operation at each index.

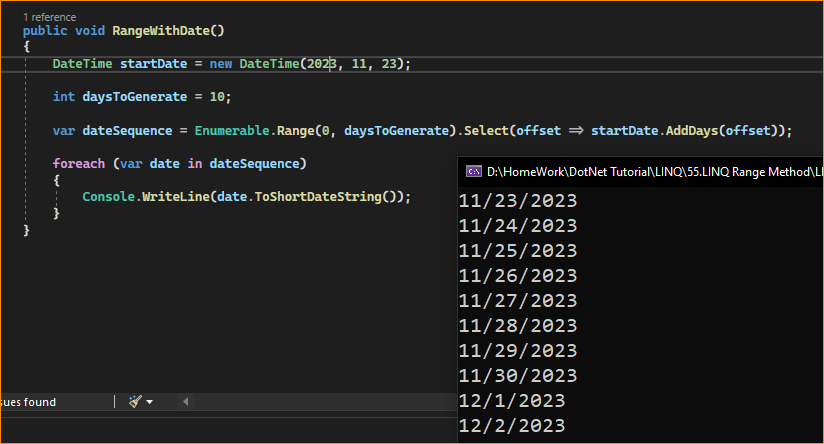
**Batch** **Processing**: For batch operations where you need to perform an action repeatedly a certain number of times.

**Data** **Population**: When populating data structures with sample data, you might want a quick way to insert a range of numbers.

**Scheduling**: In scheduling scenarios, you might generate a sequence representing days, weeks, or other time periods for further calculations.

**Pagination**: To calculate page numbers for a paging system in a web or desktop application.

1. Here’s an example of using Range in combination with Select to create a sequence of dates:



In this example, Range generates a sequence of offsets, which are then transformed into a sequence of dates starting from a startDate.

It’s important to understand that Range generates a sequence of memory integers, so working with a large range could lead to high memory consumption. Always ensure that the count of numbers you are generating is within the bounds of what your application can handle regarding performance and memory usage.