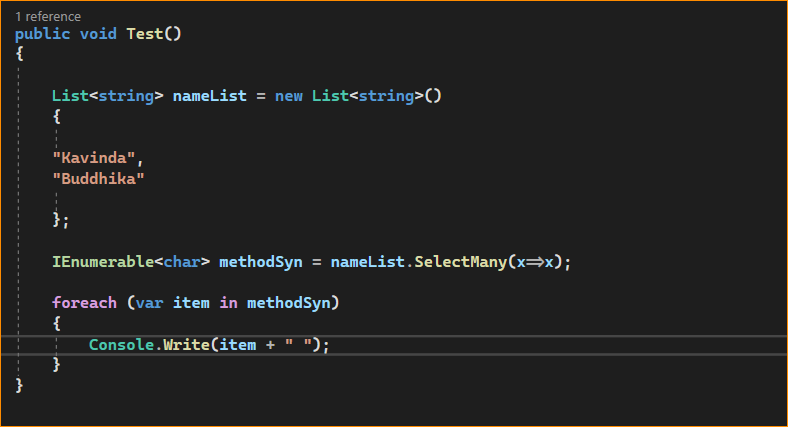
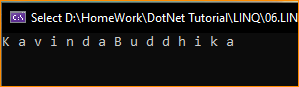
1. What is Linq SelectMany Method?

The SelectMany Method in LINQ is used to project each element of a sequence or collection or data source to an IEnumerable<T> type and then flatten the resulting sequences into one sequence. That means the SelectMany Projection Method combines the records from a sequence of results and then converts it into one result.

1. Example to Understand LINQ SelectMany Projection Method using C#:

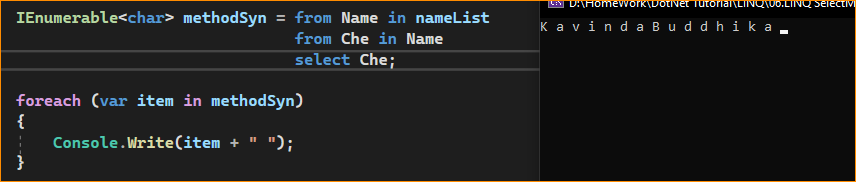
Let us understand the LINQ SelectMany Method with an example using C#. In the below example, we are using the SelectMany Method to flatten the resulting sequences into one sequence. Here, you can see that the SelectMany method returns an IEnumerable<char>. This is because the SelectMany method returns all the elements from the sequence. Here the nameList is the sequence or collection or the data source. And the sequence contains two strings. And we know the string is a collection of characters. So, the SelectMany method fetches all the characters from the above two strings and then converts them into one sequence i.e. IEnumerable<char>. In the below example, we are using LINQ Method Syntax i.e. using the LINQ SelectMany Method.





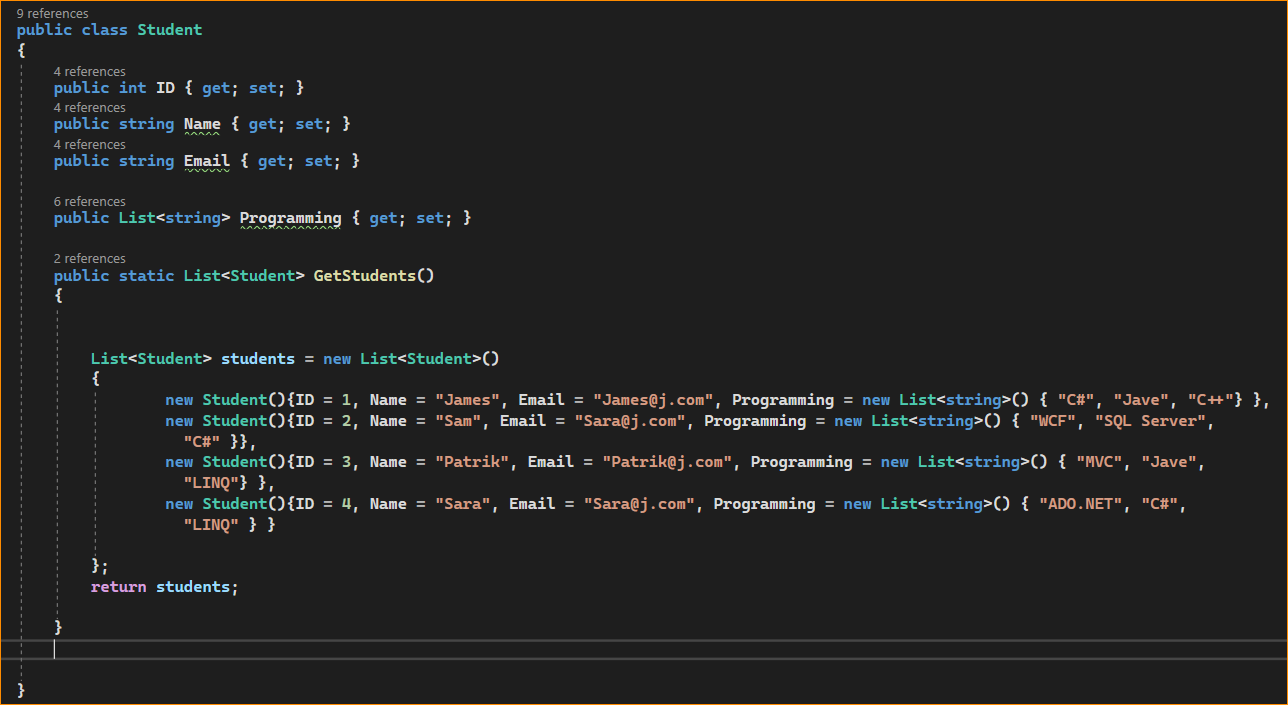
1. LINQ SelectMany Using Query Syntax in C#:

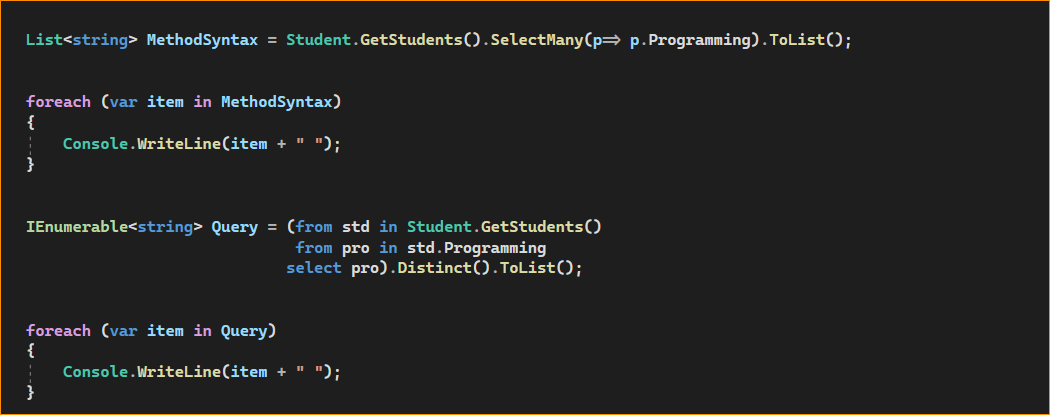
The most important point that you need to remember is there is no such SelectMany Operator available in LINQ to write Query Syntax. But we can achieve this by writing multiple “from clauses” in the query as shown in the below example. In the below example, we are fetching all the strings from the nameList collection to str object and then we are using another from clause to fetch all the characters from the str object and then we are projecting the characters which are going to contain all the characters available in both strings.



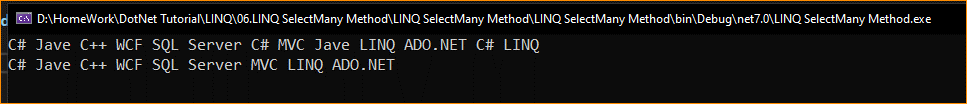
1. Example to Understand LINQ SelectMany Projection Method with Complex Data Type in C#

Let us create a class file with the name Student.cs and then copy and paste the following code into it. As you can see, we have created the following Student class with four properties. Further, if you notice the Programming property of the Student class returns List<string>. Here we have also created one method which will return the List of students which will be going to act as our data source.





Result –



Second result – Removed duplicate Element

The Distinct() method will remove duplicate elements from the source collection and return a new sequence with only the unique elements. It does so by comparing elements for equality based on their default equality comparer, which is determined by the element type. If you want to use a custom equality comparison, you can provide an IEqualityComparer<T> as an argument to the Distinct() method.

