Государственный Университет Молдовы

Факультет Математики и Информатики

Департамент Информатики

Реферат по предмету

“Технологии веб программирования”

на тему

“**Concatenation and Aggregation Operations**”

(Concat, MinBy, Sum)

Выполнил студент группы MIA2102:

Паненко Е.

Кишинёв 2021

**Содержание**

[**Введение** 2](#_Toc103177122)

[**Concatenation Operation** 2](#_Toc103177123)

[**MinBy** 3](#_Toc103177124)

[**Sum** 3](#_Toc103177125)

[**Заключение** 5](#_Toc103177126)

[**Библиограция** 5](#_Toc103177127)

# **Введение**

Целью данной работы является освоение операторов конкатенации и аггрегации в языке C#, а так же презентации о проделанной работе другим студентам. Данные операции предоставляют удобный способ по объединению перечеслений, а так же получение результата различных операций по отношению к перечеслению. В примерах с перечеслениями будет использован класс Книга с полями название и количество страниц.

# **Concatenation Operation**

Операция конкатенации используется при необходимости объединить два перечесление в одно. Синтаксис команды выглядит следующим образом:

IEnumerable<TSource> Concat<TSource> (this System.Collections.Generic.IEnumerable<TSource> first, System.Collections.Generic.IEnumerable<TSource> second);

Пример работы метода:

var l1 = new List<Book>() { b1, b2 };

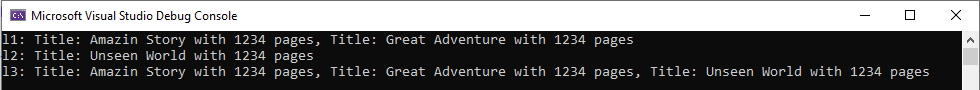
var l2 = new List<Book>() { b3 };

Console.WriteLine("l1: " + ArrayStringResult(l1));

Console.WriteLine("l2: " + ArrayStringResult(l2));

var l3 = l1.Concat(l2);

Console.WriteLine("l3: " + ArrayStringResult(l3));



# **MinBy**

Данная операция используется для нахождения минимума коллекции по кастомно указанному критерию. Синтаксис команды выглядит следующим образом:

MinBy<TSource,TKey>(IEnumerable<TSource>, Func<TSource,TKey>)

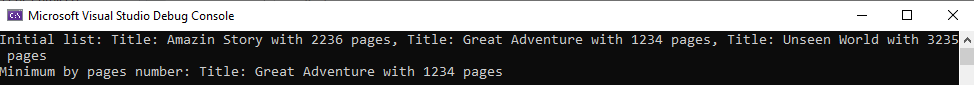
Пример работы метода:

var bookList = new List<Book>() { b1, b2, b3 };

Console.WriteLine("Initial list: " + ArrayStringResult(bookList));

Console.WriteLine("Minimum by pages number: "

+ bookList.MinBy(book => book.Pages));



# **Sum**

Группа методов принимающих численные в качестве входной коллекции для вычисления общей суммы. Ниже представлена перегруженная группа методов:

|  |  |
| --- | --- |
| [Sum(IEnumerable<Decimal>)](https://docs.microsoft.com/en-us/dotnet/api/system.linq.enumerable.sum?view=net-6.0#system-linq-enumerable-sum(system-collections-generic-ienumerable((system-decimal)))) | Computes the sum of a sequence of [Decimal](https://docs.microsoft.com/en-us/dotnet/api/system.decimal?view=net-6.0) values. |
| [Sum(IEnumerable<Double>)](https://docs.microsoft.com/en-us/dotnet/api/system.linq.enumerable.sum?view=net-6.0#system-linq-enumerable-sum(system-collections-generic-ienumerable((system-double)))) | Computes the sum of a sequence of [Double](https://docs.microsoft.com/en-us/dotnet/api/system.double?view=net-6.0) values. |
| [Sum(IEnumerable<Int32>)](https://docs.microsoft.com/en-us/dotnet/api/system.linq.enumerable.sum?view=net-6.0#system-linq-enumerable-sum(system-collections-generic-ienumerable((system-int32)))) | Computes the sum of a sequence of [Int32](https://docs.microsoft.com/en-us/dotnet/api/system.int32?view=net-6.0) values. |
| [Sum(IEnumerable<Int64>)](https://docs.microsoft.com/en-us/dotnet/api/system.linq.enumerable.sum?view=net-6.0#system-linq-enumerable-sum(system-collections-generic-ienumerable((system-int64))))  [Sum(IEnumerable<Single>)](https://docs.microsoft.com/en-us/dotnet/api/system.linq.enumerable.sum?view=net-6.0#system-linq-enumerable-sum(system-collections-generic-ienumerable((system-single)))) | Computes the sum of a sequence of [Int64](https://docs.microsoft.com/en-us/dotnet/api/system.int64?view=net-6.0) values.  Computes the sum of a sequence of [Single](https://docs.microsoft.com/en-us/dotnet/api/system.single?view=net-6.0) values. |

Пример работы методов:

var decimals = new List<Decimal>() { 123, 321, 345 };

var doubles = new List<Double>() { 1.2, 2.4, 3.5 };

var int32s = new List<Int32>() { 1, 2, 3 };

var int64s = new List<Int64>() { 10, 11, 13 };

var singles = new List<Nullable<Single>>() { .2f, .1f, .4f };

Console.WriteLine("Sum of decimals: " + ArrayStringResult(decimals) + " equals " + decimals.Sum());

Console.WriteLine("Sum of doubles: " + ArrayStringResult(doubles) + " equals " + doubles.Sum());

Console.WriteLine("Sum of int32s: " + ArrayStringResult(int32s) + " equals " + int32s.Sum());

Console.WriteLine("Sum of int64s: " + ArrayStringResult(int64s) + " equals " + int64s.Sum());

Console.WriteLine("Sum of nullable singles: " + ArrayStringResult(singles) + " equals " + singles.Sum());

Следующая группа перегруженных методов для суммирования аналогична предыдущей за тем исключениям, что позволяет значениям перечесления хранить в себе null элементы. Сигнатура методов выглядит следующим образом:

|  |  |
| --- | --- |
| [Sum(IEnumerable<Nullable<Decimal>>)](https://docs.microsoft.com/en-us/dotnet/api/system.linq.enumerable.sum?view=net-6.0#system-linq-enumerable-sum(system-collections-generic-ienumerable((system-nullable((system-decimal)))))) | Computes the sum of a sequence of nullable [Decimal](https://docs.microsoft.com/en-us/dotnet/api/system.decimal?view=net-6.0) values. |
| [Sum(IEnumerable<Nullable<Double>>)](https://docs.microsoft.com/en-us/dotnet/api/system.linq.enumerable.sum?view=net-6.0#system-linq-enumerable-sum(system-collections-generic-ienumerable((system-nullable((system-double)))))) | Computes the sum of a sequence of nullable [Double](https://docs.microsoft.com/en-us/dotnet/api/system.double?view=net-6.0) values. |
| [Sum(IEnumerable<Nullable<Int32>>)](https://docs.microsoft.com/en-us/dotnet/api/system.linq.enumerable.sum?view=net-6.0#system-linq-enumerable-sum(system-collections-generic-ienumerable((system-nullable((system-int32)))))) | Computes the sum of a sequence of nullable [Int32](https://docs.microsoft.com/en-us/dotnet/api/system.int32?view=net-6.0) values. |
| [Sum(IEnumerable<Nullable<Int64>>)](https://docs.microsoft.com/en-us/dotnet/api/system.linq.enumerable.sum?view=net-6.0#system-linq-enumerable-sum(system-collections-generic-ienumerable((system-nullable((system-int64))))))  [Sum(IEnumerable<Nullable<Single>>)](https://docs.microsoft.com/en-us/dotnet/api/system.linq.enumerable.sum?view=net-6.0#system-linq-enumerable-sum(system-collections-generic-ienumerable((system-nullable((system-single)))))) | Computes the sum of a sequence of nullable [Int64](https://docs.microsoft.com/en-us/dotnet/api/system.int64?view=net-6.0) values.  Computes the sum of a sequence of nullable [Single](https://docs.microsoft.com/en-us/dotnet/api/system.single?view=net-6.0) values. |

Пример работы метода:

var decimals = new List<Nullable<Decimal>>() { 123, null, 345 };

var doubles = new List<Nullable<Double>>() { 1.2, 2.4, null };

var int32s = new List<Nullable<Int32>>() { null, 2, null, 4 };

var int64s = new List<Nullable<Int64>>() { 10, null, 13 };

var singles = new List<Nullable<Single>>() { .2f, null, .4f };

Console.WriteLine("Sum of nullable decimals: " + ArrayStringResult(decimals) + " equals " + decimals.Sum());

Console.WriteLine("Sum of nullable doubles: " + ArrayStringResult(doubles) + " equals " + doubles.Sum());

Console.WriteLine("Sum of nullable int32s: " + ArrayStringResult(int32s) + " equals " + int32s.Sum());

Console.WriteLine("Sum of nullable int64s: " + ArrayStringResult(int64s) + " equals " + int64s.Sum());

Console.WriteLine("Sum of nullable singles: " + ArrayStringResult(singles) + " equals " + singles.Sum());Graphical user interface, text

Description automatically generated

# **Заключение**

В ходе выполнения работы были освоены 4 метода, которые облегчают работу с перечислениями, путём имплементации тривиальных задач и предоставления интерфейсов для кастомизации поведения в случае необходимости пользователя. В работе были рассмотренны операции позволяющие объединять два перечисления в одно, вычислять минимальное значение в коллекции используя кастомное условие, а так же нахождения суммы коллекций содержащих численные значения, которые в свою очередь могут принимать null значения.

# **Библиограция**

* [**https://docs.microsoft.com/en-us/dotnet/api/system.linq.enumerable.concat?view=net-6.0**](https://docs.microsoft.com/en-us/dotnet/api/system.linq.enumerable.concat?view=net-6.0)
* [**https://docs.microsoft.com/en-us/dotnet/api/system.linq.enumerable.minby?view=net-6.0#system-linq-enumerable-minby-2(system-collections-generic-ienumerable((-0))-system-func((-0-1)))**](https://docs.microsoft.com/en-us/dotnet/api/system.linq.enumerable.minby?view=net-6.0#system-linq-enumerable-minby-2(system-collections-generic-ienumerable((-0))-system-func((-0-1))))
* [**https://docs.microsoft.com/en-us/dotnet/api/system.linq.enumerable.sum?view=net-6.0**](https://docs.microsoft.com/en-us/dotnet/api/system.linq.enumerable.sum?view=net-6.0)
* [**https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/builtin-types/nullable-value-types**](https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/builtin-types/nullable-value-types)
* [**https://docs.microsoft.com/en-us/dotnet/api/system.single?view=net-6.0**](https://docs.microsoft.com/en-us/dotnet/api/system.single?view=net-6.0)