# First / Last List – Data Structures Exam

This document describes an **exam problem** for the ["Data Structures" course @ Software University](https://softuni.bg/trainings/1147/Data-Structures-June-2015).

## Problem Description

You need to implement a **“first / last list” data structure** that performs fast enough the following operations:

* **Add(*element*)**
  + **Adds an element** to the structure (duplicate elements are accepted)
* **Count**
  + Returns the **number of elements** in the structure
* **First(*count*)**
  + Returns the **earliest added *count* elements** (in the order of addition from first to last)
  + Throws an **exception** in case the structure holds less than *count* elements
* **Last(*count*)**
  + Returns the **latest added *count* elements** (in the order of addition from last to first)
  + Throws an **exception** in case the structure holds less than *count* elements
* **Min(*count*)**
  + Returns the **smallest *count* elements** (ordered from the smallest to the largest as first criteria and by the order of adding as second criteria)
  + Throws an **exception** in case the structure holds less than *count* elements
  + Note: the elements should be **comparable** in order to find the minimal element
* **Max(*count*)**
  + Returns the **biggest *count* elements** (ordered from the biggest to the smallest as first criteria and by the order of adding as second criteria)
  + Throws an **exception** in case the structure holds less than *count* elements
  + Note: the elements should be **comparable** in order to find the maximal element
* **Remove-All(*element*)**
  + **Removes all elements** that are equal to the passed element
  + Returns the **number of removed elements** or 0 when no elements are matched
  + Note: the elements should be **comparable** in order to find which pairs of elements are equal
* **Clear()**
  + **Removes all elements** from the data structure

### Input and Output

You are given a **Visual Studio C# project skeleton** (unfinished project) / **Eclipse Java project** / **Visual Studio C++ project** holding the interface IFirstLastList, the unfinished class FirstLastList and **tests** covering its **functionality** and its **performance**.

Your task is to **finish this class** to make the tests run correctly.

* You are **not allowed to change the tests**.
* You are **not allowed to change the interface**.

### Interface IFirstLastList

The interface IFirstLastList in C# looks like the code below:

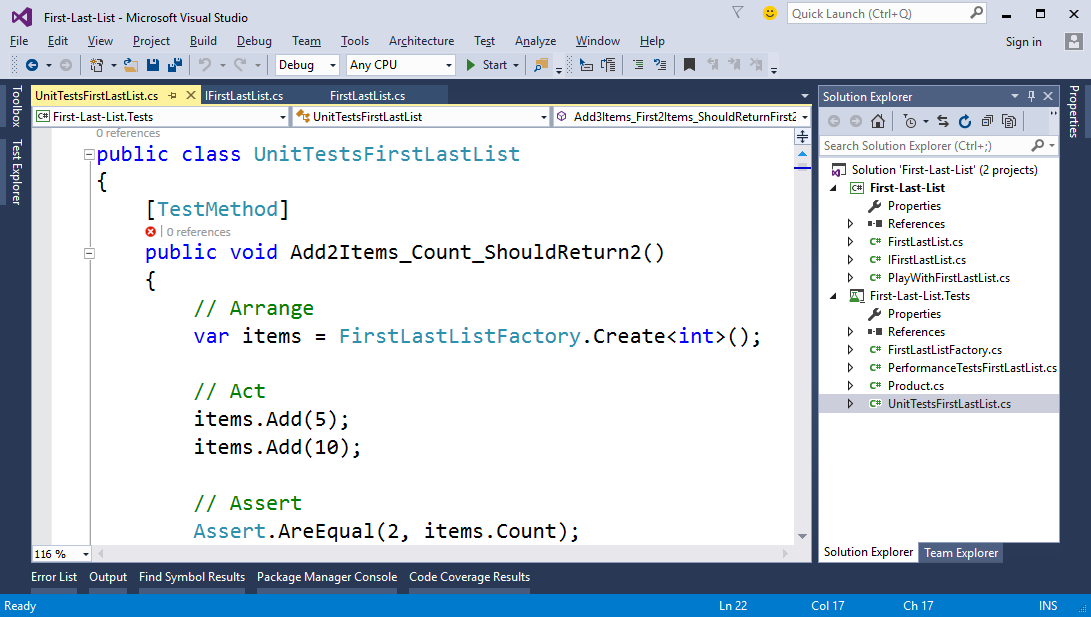
|  |
| --- |
| public interface IFirstLastList<T> where T : IComparable<T>  {  void Add(T element);  int Count { get; }  IEnumerable<T> First(int count);  IEnumerable<T> Last(int count);  IEnumerable<T> Min(int count);  IEnumerable<T> Max(int count);  void Clear();  int RemoveAll(T element);  } |

The interface IFirstLastList in Java looks like the code below:

|  |
| --- |
| **public** **interface** IFirstLastList<T **extends** Comparable<T>> {  **void** add(T element);  **int** getCount();  Iterable<T> first(**int** count);  Iterable<T> last(**int** count);  Iterable<T> min(**int** count);  Iterable<T> max(**int** count);  **void** clear();  **int** removeAll(T element);  } |

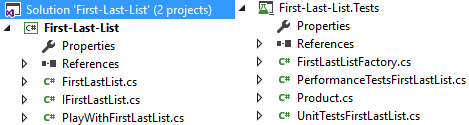
### Project Skeleton

The project skeleton opens correctly in **Visual Studio 2015** but can be open in other Visual Studio versions as well and also can run in **SharpDevelop** and **Xamarin Studio**.



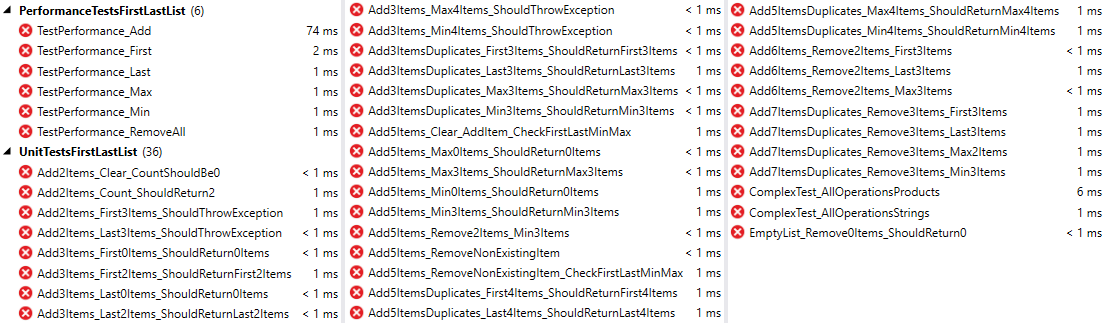
### Project Assets

The C# project holds the following assets:



### Unit Tests

The C# project holds many automated **tests** for the data structure **functionality** and its **performance**:



### Evaluation

* **Passed tests** give **50%** of the score for this problem.
* When **all tests pass** (with no exception), this gives the other **50%** of the score.
* Note: the performance tests are given 3 times more execution time than needed (for average computer from 2015). This is to ensure the tests will pass on a slower computer. Still, on very old computer, you may need to manually increase the timeouts.

### Submissions

Submit as solution your project (source code + libraries) as a ZIP archive.