**Modern & Effective C++: Homework**

Write your own implementation of a singly linked list (you can look at std::forward\_list [1] as a reference for what needs to be implemented), which has the following operations defined:

1. Basic list operations: constructing, inserting elements at the beginning of the list, iterating over the list, size of the list, removing elements.
2. Move operations (Move constructor, move assignment)
3. The method for inserting elements at the beginning must have an overload for an rvalue reference, and move objects correctly.
4. Constructing a list using a constructor that takes a std::initializer\_list.
5. The list must be templated, and work not only with primitive values, but also with objects (and correctly release them)
6. Ideally, you should strive for a class interface like std::forward\_list [1], but you do not need to implement absolutely all methods - a custom allocator, and emplace\_\* methods are not required (But it will not be a minus). sort, merge, unique do not need to be done.
7. BONUS POINTS: Implement the split\_when method, which allows you to split a list into two, which determines the position from which element to split, based on the bool value returned from the passed predicate function [2]. Pass predicate for example as std::function:

// Predicate type

**template**<**typename** T>

**using** SplitPredicate = std::function<**bool**(**const** T& value)>;

// Proposed function signature

**template**<**typename** T>

**class** **LinkedList**

{

// ... Other methods ...

LinkedList<T> split\_when(SplitPredicate<T> condition);

// ... Other methods ...

};

// Expected usage

LinkedList<**int**> original = { **1**, **42**, **3** };

**auto** tail = original.split\_when(

[](**const** **int**& value)){ **return** value == **42**; }

);

// original now is { 1 }

// tail now is { 42, 3 }

Nodes:

* Memory leaks/memory mishandling are also evaluated; if smart pointers (std::unique\_ptr) are correctly used to work with memory, it will be a plus.
* Be sure to check your work for compilability/workability before submitting.
* It is recommended to use Visual Studio 2019 for development. But if you really want to and there is no way out at all, you can use anything :)

Links:

* + - 1. <https://en.cppreference.com/w/cpp/container/forward_list>
      2. <https://stackoverflow.com/questions/3230944/what-does-predicate-mean-in-the-context-of-computer-science>