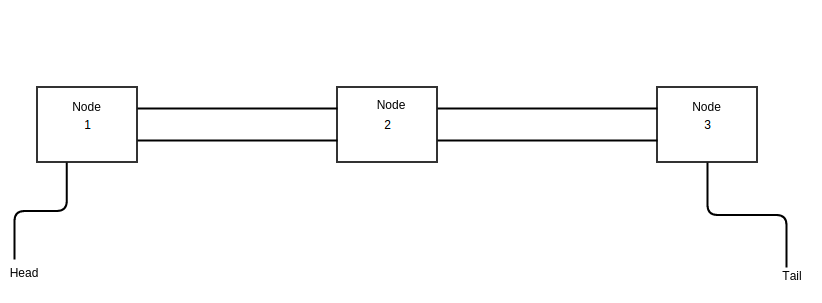
**Vector and List in C++**

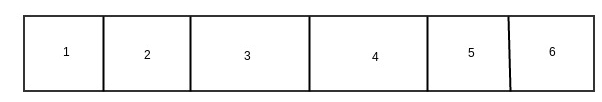
Both vector and list are sequential containers of C++ Standard Template Library.

 But there are many differences between them because of their internal implementation i.e.

**List** stores elements at non-contiguous memory location i.e. it internally uses a doubly linked list i.e.

[](https://thispointer.com/wp-content/uploads/2016/05/linkedlist.png)

Whereas, **vector** stores elements at contiguous memory locations like an array i.e.

[](https://thispointer.com/wp-content/uploads/2016/06/array.jpg)

### 1.) Insertion and Deletion

Insertion and Deletion in List is very efficient as compared to vector because to insert an element in list at start, end or middle, internally just a couple of pointers are swapped.

Whereas, in vector insertion and deletion at start or middle will make all elements to shift by one. Also, if there is insufficient contiguous memory in vector at the time of insertion, then a new contiguous memory will be allocated and all elements will be copied there.

**So, insertion and deletion in list is much efficient than vector in c++.**

### 2.) Random Access:

As List is internally implemented as doubly linked list, therefore no random access is possible in List. It means, to access 15th element in list we need to iterate through first 14 elements in list one by one.

Whereas, vector stores elements at contiguous memory locations like an array. Therefore, in vector random access is possible i.e. we can directly access the 15th element in vector using operator [] i.e.

|  |  |
| --- | --- |
| 1  2 | std::vector<int> vec(20);  vec[15] = 10; |

**So, we can not use std::list with some of the STL algorithm that needs the random access operators like std::sort.**

### 3.) Iterator Invalidation

Deleting or Inserting an element in List does not invalidate any iterator because during insertion and deletion no element is moved from its position only a couple pointers are changed.

Whereas, in vector insertion and deletion can invalidate the iterators. For more details about vector Iterator Invalidation, check this article i.e.

### 4.) Special Member functions

As std::list do not provide random access, there many STL algorithms that uses Random Access Iterators can not be used with List. Hence std::list provides some extra functions for Sorting, Splicing, Removing elements and identifying unique elements.

Vector provides the random access and hence can be used with STL algorithms that uses Random Access Iterators.