A Linked List implementation of a List:

List.cpp

```
1  #pragma once
2
3  template <typename T>
4  class List {
5    public:
…      /* ... */
19
20   private:
21     class ListNode {
22       public:
23         T & data;
24         ListNode * next;
25         ListNode(T & data) :
               data(data), next(nullptr) { }
26     };
27
28     ListNode *head_;
…      /* ... */
…  };
```

_index():

List.hpp

```
57  template <typename T>
58  typename List<T>::ListNode *&
       List<T>::_index(unsigned index) {
59
60
61
62  }
63  template <typename T>
64  typename List<T>::ListNode *& List<T>::_index(unsigned
    index, ListNode *& root){
65
66
67
68
69
70
71
72
73  }
```

What is the return type of **_index**?

Building functionality with _index():

List.hpp

```
48  template <typename T>
49  T & List<T>::operator[](unsigned index) {
50
51
52
53
54
    }
```

List.hpp

```
90  template <typename T>
91  T & List<T>::insert(const T & t, unsigned index) {
92
93
94
95
96
    }
```

List.hpp

```
103  template <typename T>
104  T List<T>::remove(unsigned index) {
105
106
107
108
109
     }
```