

RemoveNode.cpp

```
1 /*****
2 * AUTHOR      : Amirarsalan Valipour & Ali Eshghi
3 * STUDENT ID  : 1103126 - 1112261
4 * LAB #10     : Creating an Ordered List
5 * CLASS      : CS 1B
6 * SECTION     : MW - 7:30 pm
7 * DUE DATE    : 11/05/2019
8 *****/
9 * RemoveNode
10 *****/
11 * This function Allows the user to input a name and removes the node from the
12 * list.
13 *****/
14
15 #include "MyHeader.h"
16
17 void RemoveNode(PersonNode *&head)
18 {
19     PersonNode *persPtr;    //Proc - stores the target name
20     PersonNode *ptr;        //Proc - search node
21     PersonNode *rmv;        //Proc - remove node
22     PersonNode node;        //Proc & In - passes the info into persPtr
23
24     bool found;             //Proc - condition value for the searched name
25
26     //CHECKS FOR EMPTY LIST
27
28     found = false;
29
30     if (head == NULL)
31     {
32         cout << endl;
33         cout << "Remove operation is not available for an empty list.";
34         cout << endl;
35     }
36
37     else
38     {
39         //NEW NODE
40
41         persPtr = new PersonNode;
42         ptr = head;
43
44         //INPUT
45
46         cout << endl;
47         cout << "Who would you like to remove? ";
48         getline(cin, node.name);
49
50         *persPtr = node;
51
52         //FIRST CASE
53         if (ptr -> next == NULL || ptr -> name == persPtr -> name)
54         {
55             head = ptr -> next;
```

RemoveNode.cpp

```

56     ptr -> next -> prev = NULL;
57     delete ptr;
58
59
60     cout << endl;
61     cout << "Removing " << persPtr -> name << '!';
62     cout << endl;
63 }
64
65 else
66 {
67     found = false;
68
69     //GOES THROUGH
70     while (ptr -> next != NULL && !found)
71     {
72         if (ptr -> name == persPtr -> name)
73         {
74             found = true;
75         }
76         else
77         {
78             ptr = ptr -> next;
79         }
80     }
81
82     //IF FOUND DELETE
83
84     if (found)
85     {
86         rmv = ptr;
87         ptr = ptr -> prev;
88         ptr -> next = rmv -> next;
89         rmv -> next -> prev = ptr;
90         delete rmv;
91
92         rmv = NULL;
93
94
95         cout << endl;
96         cout << "Removing " << persPtr -> name << '!';
97         cout << endl;
98     }
99
100    //DELETE OTHER
101    else if (ptr -> name == persPtr -> name)
102    {
103        rmv = ptr;
104        ptr = ptr -> prev;
105        ptr -> next = NULL;
106        delete rmv;
107
108        rmv = NULL;
109
110

```

RemoveNode.cpp

```
111         cout << endl;
112         cout << "Removing " << persPtr -> name << '!';
113         cout << endl;
114     }
115
116     //IF THEY DON'T EXIST
117     else
118     {
119         cout << endl;
120         cout << "I'm sorry, \"" << persPtr -> name;
121         cout << "\" was NOT found!";
122         cout << endl;
123     }
124 }
125
126 ptr = NULL;
127 }
128 }
129 }
130 }
```