

```

1  #include <iostream>
2  using namespace std;
3  struct Node {
4      int data;
5      Node* next;
6  };
7  Node* removeDuplicates(Node* head) {
8
9
10     if (head == NULL) {
11         return NULL; // Just return the empty list
12     }
13     // We start at the beginning (the head).
14     Node* current = head;
15     // If current->next is NULL, we are at the end, so we stop.
16     while (current->next != NULL) {
17         // Is the data in my current node (e.g., 3)
18         if (current->data == current->next->data) {
19
20
21             current->next = current->next->next;
22         } else {
23
24
25             current = current->next;
26         }
27     }
28
29     return head;
31 }
32
33
34
35 Node* createNode(int data) {
36     Node* newNode = new Node();
37     newNode->data = data;
38     newNode->next = NULL;
39     return newNode;
40 }
41
42
43 void push(Node** head_ref, int new_data) {
44     Node* new_node = createNode(new_data);
45     Node* last = *head_ref;
46
47     if (*head_ref == NULL) {
48         *head_ref = new_node;
49         return; // And we're done
50     }
51
52     // Otherwise, loop until we find the *very last* node
53     while (last->next != NULL) {
54         last = last->next; // Keep moving to the next node
55     }
56
57     // We're at the end. Make the last node's 'next' pointer point to our new node.
58     last->next = new_node;
59 }
60 // A helper function to print all the data in the list
61 void printList(Node* node) {
62     // Loop while our node pointer isn't NULL (which is at the end)
63     while (node != NULL) {
64         cout << node->data << " -> ";
65         node = node->next;
66     }

```

```

67     cout << "NULL" << endl;
68 }
69 int main() {
70     // We always start with an empty list (head is NULL)
71     Node* head = NULL;
72
73     int n; // This variable will hold how many numbers the user wants
74     int data; // This variable will hold each number as the user types it
75
76     // Ask the user how many numbers they want to type
77     cout << "How many numbers do you want to add to the list? ";
78     cin >> n;
79
80     // Remind the user of the problem's rule
81     cout << "Great. Please enter the " << n << " numbers in sorted order, pressing Enter after each one:"
<< endl;
82
83     // This loop will run 'n' times
84     for (int i = 0; i < n; i++) {
85         cout << "Enter number " << (i + 1) << ": ";
86         cin >> data; // Get the number from the keyboard
87         // Add the number they typed to the end of our list
88         push(&head, data);
89     }
90
91     // Now, we show them the list they just built
92     cout << "\nOriginal List you entered: " << endl;
93     printList(head); // Print the list before any changes
94
95     // this is where we call your function
96     head = removeDuplicates(head);
97     cout << "\nList after removing duplicates: " << endl;
98     printList(head);
99     return 0;
100 }

```