Day 24: More Review + More Linked Lists!

by harsha_s

Problem Submissions Leaderboard Discussions

Welcome to Day 24! Review everything we've learned so far by coding a Hangman Game, or just jump right into the problem.

Given a pointer to the *head* node of a *linked list* whose *data* elements are in non-decreasing order, you must delete any duplicate nodes and print the updated list.

Code handling I/O is provided in the editor. Complete the *removeDuplicates(Node)* function.

Note: The *head* pointer may be null, indicating that the list is empty. Be sure to reset your *next* pointer when performing deletions to avoid breaking the list.

Input Format

The first line contains N, the number of nodes to be inserted.
The N subsequent lines each contain an integer describing the *data* for a node being inserted at the list's tail; the lines of *data* will always be in non-decreasing order.

Output Format

Print the *data* for your list of ascending nodes as a single line of space-separated integers.

Sample Input

```
6
1
2
2
3
3
4
```

Sample Output

```
1 2 3 4
```

Explanation

N = 6, and our non-decreasing list is $\{1, 2, 2, 3, 3, 4\}$. The *data* values 2 and 3 each have a duplicate, so we remove the two duplicate nodes and print our updated (ascending) list:

```
1 2 3 4
```

Submissions: 3026
Max Score: 100
Difficulty: Easy

More

Current Buffer (saved locally, editable) C++

```
#include <cstddef>
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;
class Node
{
    public:
        int data;
        Node *next;
        Node(int d){
            data=d;
            next=NULL;
        }
};
class Solution{
    public:
          Node* removeDuplicates(Node *head)
          {
            //Write your code here
          }
          Node* insert(Node *head,int data)
          {
               Node* p=new Node(data);
               if(head==NULL){
                   head=p;

               }
               else if(head->next==NULL){
                   head->next=p;

               }
               else{
                   Node *start=head;
                   while(start->next!=NULL){
                       start=start->next;
                   }
                   start->next=p;

               }
                return head;


          }
          void display(Node *head)
          {
                  Node *start=head;
                    while(start)
                    {
                        cout<<start->data<<" ";
                        start=start->next;
                    }
           }
};

int main()
{
    Node* head=NULL;
    Solution mylist;
    int T,data;
    cin>>T;
    while(T-->0){
        cin>>data;
        head=mylist.insert(head,data);
    }
    head=mylist.removeDuplicates(head);

    mylist.display(head);

}
```

Line: 6 Col: 1