Day 24: More Linked Lists

by harsha_s

Objective

Check out the Tutorial tab for learning materials and an instructional video!

Task

A *Node* class is provided for you in the editor. A *Node* object has an integer data field, $data$, and a Node instance pointer, $next$, pointing to another node (i.e.: the next node in a list).

A *removeDuplicates* function is declared in your editor, which takes a pointer to the $head$ node of a linked list as a parameter. Complete *removeDuplicates* so that it deletes any duplicate nodes from the list and returns the head of the updated list.

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

You do not need to read any input from stdin. The following input is handled by the locked stub code and passed to the *removeDuplicates* function:
The first line contains an integer, N, the number of nodes to be inserted.
The N subsequent lines each contain an integer describing the $data$ value of a node being inserted at the list's tail.

Constraints

- The data elements of the linked list argument *will always be* in non-decreasing order.

Output Format

Your *removeDuplicates* function should return the head of the updated linked list. The locked stub code in your editor will print the returned list to stdout.

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 values 2 and 3 both occur twice in the list, so we remove the two duplicate nodes. We then return our updated (ascending) list, which is $\{1, 2, 3, 4\}$.

Easy Submitted 23871 times
Max Score 30

Need Help?

View Tutorial

View Discussions

View Editorial Solution

View Top Submissions

Rate This Challenge:

Download problem statement

Download sample test cases

Suggest Edits

Current Buffer (saved locally, editable) Java 8

```
import ↔;
class Node{
    int data;
    Node next;
    Node(int d){
        data=d;
        next=null;
    }

}
class Solution
{
    public static Node removeDuplicates(Node head) {
      //Write your code here
        if (head.next == null) {
            return head;
        }
        if (head.data != head.next.data) {
            removeDuplicates(head.next);
        } else {
            head.next = head.next.next;
            removeDuplicates(head);
        }
        return head;


    }
    public static  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;
```

```
39              while(start.next!=null)
40                  start=start.next;
41              start.next=p;
42
43          }
44          return head;
45      }
46      public static void display(Node head)
47          {
48                Node start=head;
49                while(start!=null)
50                {
51                    System.out.print(start.data+" ");
52                    start=start.next;
53                }
54          }
55          public static void main(String args[])
56          {
57                Scanner sc=new Scanner(System.in);
58                Node head=null;
59                int T=sc.nextInt();
60                while(T-->0){
61                    int ele=sc.nextInt();
62                    head=insert(head,ele);
63                }
64                head=removeDuplicates(head);
65                display(head);
66
67           }
68      }
```

Line: 16 Col: 9

Upload Code as File Test against custom input Run Code Submit Code

Testcase 0 ✔

Congratulations, you passed the sample test case.

Click the **Submit Code** button to run your code against all the test cases.

Input (stdin)

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

Your Output (stdout)

```
1 2 3 4
```

Expected Output

```
1 2 3 4
```