# **GeeksforGeeks**

A computer science portal for geeks

Login	
-------	--

Home	Algorithms	DS	GATE	Interv	view Corner	Q&A	С	C++	Java	Books	Contribute	Ask a Q	About
Array	Bit Magic	C/C+	+ Arti	cles	<b>GFacts</b>	Linked L	ist	MCQ	Misc	Outpu	t String	Tree	Graph

## Remove duplicates from a sorted linked list

Write a removeDuplicates() function which takes a list sorted in non-decreasing order and deletes any duplicate nodes from the list. The list should only be traversed once.

For example if the linked list is 11->11->11->21->43->60 then removeDuplicates() should convert the list to 11->21->43->60.

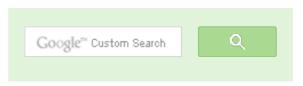
### Algorithm:

Traverse the list from the head (or start) node. While traversing, compare each node with its next node. If data of next node is same as current node then delete the next node. Before we delete a node, we need to store next pointer of the node

### Implementation:

Functions other than removeDuplicates() are just to create a linked linked list and test removeDuplicates().

```
/*Program to remove duplicates from a sorted linked list */
#include<stdio.h>
#include<stlib.h>
/* Link list node */
struct node
    int data;
    struct node* next;
};
/* The function removes duplicates from a sorted list */
void removeDuplicates(struct node* head)
  /* Pointer to traverse the linked list */
```





53,528 people like GeeksforGeeks.









## Interview Experiences

Advanced Data Structures

Dynamic Programming

**Greedy Algorithms** 

Backtracking

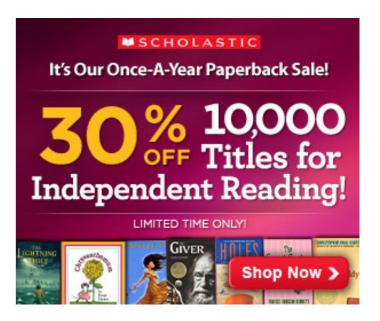
Pattern Searching

Divide & Conquer

Mathematical Algorithms

Recursion

```
struct node* current = head;
  /* Pointer to store the next pointer of a node to be deleted*/
  struct node* next next;
  /* do nothing if the list is empty */
  if(current == NULL)
     return:
  /* Traverse the list till last node */
  while(current->next != NULL)
   /* Compare current node with next node */
    if(current->data == current->next->data)
       /*The sequence of steps is important*/
      next next = current->next->next;
      free(current->next);
      current->next = next next;
    else /* This is tricky: only advance if no deletion */
      current = current->next;
/* UTILITY FUNCTIONS */
/* Function to insert a node at the beginging of the linked list */
void push(struct node** head ref, int new data)
    /* allocate node */
    struct node* new node =
            (struct node*) malloc(sizeof(struct node));
    /* put in the data */
    new node->data = new data;
    /* link the old list off the new node */
    new node->next = (*head ref);
    /* move the head to point to the new node */
    (*head ref) = new node;
/* Function to print nodes in a given linked list */
void printList(struct node *node)
```



## Popular Posts

All permutations of a given string

Memory Layout of C Programs

Understanding "extern" keyword in C

Median of two sorted arrays

Tree traversal without recursion and without stack!

Structure Member Alignment, Padding and

Data Packing

Intersection point of two Linked Lists

Lowest Common Ancestor in a BST.

Check if a binary tree is BST or not

Sorted Linked List to Balanced BST

```
while (node!=NULL)
  printf("%d ", node->data);
  node = node->next;
/* Drier program to test above functions*/
int main()
  /* Start with the empty list */
  struct node* head = NULL;
  /* Let us create a sorted linked list to test the functions
  Created linked list will be 11->11->11->13->13->20 */
  push (&head, 20);
  push (&head, 13);
  push (&head, 13);
  push (&head, 11);
  push (&head, 11);
  push(&head, 11);
 printf("\n Linked list before duplicate removal ");
 printList(head);
  /* Remove duplicates from linked list */
 removeDuplicates(head);
 printf("\n Linked list after duplicate removal ");
 printList(head);
  getchar();
```

Time Complexity: O(n) where n is number of nodes in the given linked list.

### References:

cslibrary.stanford.edu/105/LinkedListProblems.pdf

Custom market research at scale.

Get \$75 off

Google consumer surveys

## Find + Remove Duplicates



\$0 License and free support Clean up your duplicate data today!



## Related Tpoics:

- Given a linked list, reverse alternate nodes and append at the end
- Pairwise swap elements of a given linked list by changing links
- Self Organizing List | Set 1 (Introduction)
- Merge a linked list into another linked list at alternate positions
- QuickSort on Singly Linked List
- Delete N nodes after M nodes of a linked list
- Design a stack with operations on middle element
- Swap Kth node from beginning with Kth node from end in a Linked List









Writing code in comment? Please use ideone.com and share the link here.

21 Comments GeeksforGeeks





## **Recent Comments**

Abhi You live US or India?

Google (Mountain View) interview · 49 minutes ago

Aman Hi, Why arent we checking for conditions...

Write a C program to Delete a Tree. 1 hour ago

kzs please provide solution for the problem...

Backtracking | Set 2 (Rat in a Maze) 1 hour ago

#### Sanjay Agarwal bool

tree::Root to leaf path given sum(tree...

Root to leaf path sum equal to a given number · 1

hour ago

**GOPI GOPINATH** @admin Highlight this sentence "We can easily...

Count trailing zeroes in factorial of a number  $\cdot$  1

hour ago

**newCoder3006** If the array contains negative numbers also. We...

Find subarray with given sum  $\cdot$  2 hours ago

## Sort by Newest ▼



Join the discussion...



### Aman • 6 days ago

Solution with less number of variables:

void removeDuplicates(Node\* head)

```
{
```

```
while(head->next != NULL)
```

```
{
```

if(head->data == head->next->data)

{

Node\* temp = head->next;

head->next = head->next->next;

temp = NULL;

}

see more



### Kunal Arora • 3 months ago

We can even use hash table to delete duplicate entries in linked list...

1.) We will map every value of linked list to hash table.

AdChoices D

- ► Linked List
- ► C++ Code
- ▶ Key Duplicates

AdChoices [⊳

- ► C++ Linked List
- ▶ Key Duplicates
- ▶ Java Array

AdChoices [>

- ► C++ Linked List
- ► Duplicates Remover
- ▶ Node

- 2.) While mapping we can compare if the two values hash to same location in the two values in linked list and delete one of them.
- 3.) Arrange pointers of the linked list.

Advantage: if there are millions and more elements and few of them are duplical above solution.

Disadvantage:extra space for hash table implementation @GeeksforGeeks please correct me if i am wrong...



Jonathan Chen → Kunal Arora • a month ago

How is it more optimized? I don't think it is.

The original strategy traverses the list only once. In order to remove du every item at least once.

Your strategy goes through every element once as well. However, the auxiliary hashtable which is worse in terms of space complexity.

If there are millions more elements, your strategy has to traverse throu the above approach does.



Kunal Arora → Jonathan Chen • a month ago

Yes, you are write it is not much optimized then above algorithmeven mentioned this disadvantage in my post.... but it can come



Himanshu Dagar • 3 months ago can refer to the below code also

http://ideone.com/BOzFhK

```
1 ^ Reply · Share >
```



setu • 5 months ago

If there are millions of data in it and one number is duplicate somewhere in mic one then would take lot of time while compilation. please provide a solution for



neo • 6 months ago

@GeeksforGeeks team the condition in the while statement should contain cu current=current->next in else part ,so if current is null then current->next in wh correct me if i am wrong



**Jayanth** → neo • 6 months ago

I dont think it is necessary as when "current->next" itself is not null curr check is necessary...in fact that check is implied...can u post some tes am not seeing something u hav...



**Guest** • 8 months ago

```
struct treeNode * removeDuplicateFrmOrderedLinkList(struct treeNode*
{
    struct treeNode *next=root;
    struct treeNode *prev=NULL;
    while(next!=NULL)
        prev=next;
        next=next->next;
```

```
struct treeNode *current=next;
             next=next->next;
             free(current);
         prev->next=next;
     }
     return root;
Adarsh • 10 months ago
  #include <stdio.h>
 #include <string.h>
  typedef struct node
     int data;
     struct node *next;
 }node;
 void insert(node **head_ref, int val)
 {
     node *curr;
     node *temp;
     temp = (node *)malloc(sizeof(node));
     temp->data = val;
```

```
temp->next = NULL;
      curr = *head_ref;
                                                     see more

✓ • Reply • Share ›
ultimate_coder • 11 months ago
   /^{\star} Paste your code here (You may delete these lines if not writing co
trilok → ultimate_coder • 11 months ago
       /* The function removes duplicates from a sorted list */
       void removeDuplicates(struct node* head)
       /* Pointer to traverse the linked list */
       struct node* current = head;
       /* Pointer to store the next pointer of a node to be deleted*/
       struct node* dup node;
       /* do nothing if the list is empty */
       if(current == NULL)
       return;
       /* Traverse the list till last node */
       while(current->next != NULL)
       /* Compare current node with next node till duplicates exist*/
       while( (current->next != NULL) && (current->data == current->next->d
```

see more



Ujjwal ⋅ a year ago

How about using 2 pointers to solve this..!!

- -Both point to head initially.
- -Advance 2nd pointer, if same data, go on advancing 2nd pointer until we get r that node, make 1st node->next = 2nd node;

```
1st node = 2nd node;
```

Repeat this till we reach the end of the list..

Correct me if i m wrong.. :)



Deepak → Ujjwal • 10 months ago

Your code will remove duplicate nodes but it has wastage of space ber which have duplicate values.



**Pranav** ⋅ a year ago

```
/* Paste your code here (You may delete these lines if not writing co
void removeDuplicates(struct node* head)
  /* Pointer to traverse the linked list */
  struct node* p = head;
  struct node* t=NULL;
 while(p)
     t=p;
     if(!p->next)
        return;
     p=p->next;
     if( t->data == p->data )
```

```
t->next = p->next;
         free(p);
         p=t;
Sunny ⋅ a year ago
Similar code, but instead of next used prev.
[sourcecode language="C++"]
#include <iostream>
#include <stdlib.h>
using namespace std;
typedef struct link
int x;
struct link* next;
}node;
node* head;
void add(int a)
node* temp=new node;//(node*)malloc(sizeof(node));
temp->x=a;
temp->next=head:
                                              see more
```



Kunaal Ahuja • 2 years ago

This code removes duplicates only if they occur in succession. Otherwise it we

for eg.

122791229

for these set of inputs the code won't work

/st Paste your code here (You may delete these lines if not writing code



hARRY → Kunaal Ahuja · a year ago

Hello read the problem statement carefully!!



Tim ⋅ 2 years ago

```
j--;
```

```
see more
sourabhjakhar • 3 years ago
this code do not remove the dulpicat if present in the last node
Shekhu • 4 years ago
Java code:
[sourcecode language="java"]
public class ListNode
int data;
ListNode next;
public removeDuplicatesSorted( ListNode p )
while(p!=null)
while( p.next != null && p.data == p.next.data )
p.next = p.next.next;
p = p.next;
// this pair of loops is linear: in each while loop iteration we either
// (a) delete a node or (b) move to the next node.
```





Add Disqus to your site

@geeksforgeeks, Some rights reserved

Contact Us!

Powered by WordPress & MooTools, customized by geeksforgeeks team