Delete alternate nodes of a Linked List

Given a Singly Linked List, starting from the second node delete all alternate nodes of it. For example, if the given linked list is 1->2->3->4->5 then your function should convert it to 1->3->5, and if the given linked list is 1->2->3->4 then convert it to 1->3.

Method 1 (Iterative)
Keep track of previous of the node to be deleted. First change the next link of previous node and then free the memory allocated for the node.

```
#include<stdio.h>
#include<stdlib.h>

/* A linked list node */
struct node
{
  int data;
  struct node *next;
};

/* deletes alternate nodes of a list starting with head */
void deleteAlt(struct node *head)
{
  if (head == NULL)
    return;

   /* Initialize prev and node to be deleted */
  struct node *prev = head;
  struct node *node = head->next;

  while (prev != NULL && node != NULL)
  {
     /* Change next link of previous node */
     prev->next = node->next;
```


```
      /* Free memory */
      free(node);

      /* Update prev and node */
      prev = prev->next;
      if(prev != NULL)
         node = prev->next;
   }
}

/* UTILITY FUNCTIONS TO TEST fun1() and fun2() */
/* Given a reference (pointer to pointer) to the head
  of a list and an int, push a new node on the front
  of the 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)
{
  while(node != NULL)
  {
    printf("%d ", node->data);
    node = node->next;
  }
}

/* Drier program to test above functions */
int main()
{
  int arr[100];

  /* Start with the empty list */
```


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

```
  struct node* head = NULL;

  /* Using push() to construct below list
    1->2->3->4->5  */
  push(&head, 5);
  push(&head, 4);
  push(&head, 3);
  push(&head, 2);
  push(&head, 1);

  printf("\n List before calling deleteAlt() ");
  printList(head);

  deleteAlt(head);

  printf("\n List after calling deleteAlt() ");
  printList(head);

  getchar();
  return 0;
}
```

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

Method 2 (Recursive)

Recursive code uses the same approach as method 1. The recursive coed is simple and short, but causes O(n) recursive function calls for a linked list of size n.

```
/* deletes alternate nodes of a list starting with head */
void deleteAlt(struct node *head)
{
  if (head == NULL)
    return;

  struct node *node = head->next;

  if (node == NULL)
    return;

  /* Change the next link of head */
  head->next = node->next;

  /* free memory allocated for node */
  free(node);

  /* Recursively call for the new next of head */
```

```
  deleteAlt(head->next);
}
```

Time Complexity: O(n)

Please write comments if you find the above code/algorithm incorrect, or find better ways to solve the same problem.

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

Recent Comments

Abhi You live US or India?

Google (Mountain View) interview · 50 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 · 1 hour ago

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

Find subarray with given sum · 2 hours ago

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

25 Comments GeeksforGeeks

Sort by Newest

Join the discussion…

AMIT JAMBOTKAR • 20 days ago

Java implementation is here

```
package deletealtrnatenodesoflinkedlist;

public class LinkedList<e> implements Cloneable {

Node<e> head = null;

// Adding at the End

class Node<t> {

T value;

Node<t> nextReference;

public Node(T value) {

this.value = value;

this.nextReference = null;
```

see more

• Reply • Share ›

Viswanath • 2 months ago

My implementation is better and shorter

Reply • Share ›

Himanshu Dagar • 3 months ago

For combined code can refer to below link

http://ideone.com/Om2sOW

Reply • Share ›

Marsha Donna • 4 months ago

is anything wrong in the following code:

```
void delete_alternate_node(struct node **headref)
{
struct node *cur=*headref;
struct node *temp=NULL;

while(cur!=NULL&&cur->link!=NULL)
{
temp=cur->link;
cur->link=cur->link->link;
free(temp);
cur=cur->link;
}
}
```

Reply • Share ›

Kartik Nagpal ➦ Marsha Donna • 3 months ago

http://ideone.com/u4fkZG

Here's a complete C++ code for the same, using your code for delete_ find it helpful.

Reply • Share ›

Marsha Donna ➦ Kartik Nagpal • 3 months ago

ya thanks it works fine..bt i havent chcked 4 all test cases

• Reply • Share ›

Kartik ➦ Marsha Donna • 3 months ago

all the relavent test cases here would be: 1) empty list, 2
list.
The above code passes all.

1 • Reply • Share ›

wakeup123 • 10 months ago

in method 1, why int arr[100]; in the driver portion of the code in the beginning

• Reply • Share ›

Adarsh • 10 months ago

```
 /* ================================================================
/*
/*   Filename.c
/*   (c) 2001 Author
/*
/*   Description
/*
/* ================================================================
#include<stdio.h>
#include<stdlib.h>
typedef struct node
{
  int data;
  struct node *next;
}node;
```

```
{
```

see more

• Reply • Share ›

vs • a year ago

For method 1
Inside the while loop
prev = prev->next; is not required
Suppose there are two nodes then
after during the first iteration of the loop
prev->next = node->next; sets the prev to NULL
Therefore, an error will occur while trying to access
next of prev.

Removing this line would make the code correct

• Reply • Share ›

Ankit ➚ vs • 11 months ago

Why do you say so ??!! when prev will get to NULL then simply the wh

• Reply • Share ›

yatharth.sharma • a year ago

```
node * recdel(node *t)
{
if(t==NULL) return NULL;
if(t->next==NULL)return t;
t->next=recdel(t->next->next);

}
```

Reply • Share ›

Nikin Kumar Jain • a year ago

```
 void delAlternateNode(node *sr)
{
        while(sr)
        {
                if(sr->next)
                {
                        node *temp = sr->next;
                        sr->next = sr->next->next;
                        delete temp;
                        sr = sr->next;
                }
                sr = sr->next;
        }
}
```

Reply • Share ›

Arindam Sanyal • a year ago

```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>

struct node{
int info;
struct node *link;.
};

struct node * addtoempty(struct node *, int);.
```

```
struct node * addtoend(struct node *, int);.
void display(struct node *);.
struct node *delalt(struct node *);.

void main(){
clrscr();
struct node *start=NULL;.
int num, d;
printf("n enter the number of nodes...");.
```

see more

Reply • Share ›

hARRY • a year ago

This complexity is o(n/2) not o(n) exactly...

Reply • Share ›

learning • 2 years ago

@Sambasiva shoudnt the for loop condition be this as when there are odd no. null pointer in ur solution which is incorrect i guess.

```
void deleteAlternateNode(list l)
{
Node *temp;

for( ; l && l->next ; l = l->next)
{
temp = l->next;
l->next = temp->next;
free(temp);
}
}
```

Reply • Share ›

learner • 2 years ago

```
while (prev != NULL && node != NULL)
 {
    /* Change next link of previous node */
    prev->next = node->next;

    /* Free memory */
    free(node);

    /* Update prev and node */
    prev = prev->next;
    if(prev != NULL) //statement 1
       node = prev->next;
  }
}
```

Why are we checking for prev!=NULL in statement 1. Is that necessary.If prev in turn fails to satisfy the condition in the while loop and thus the iteration stops

1 ▲ | ▼ • Reply • Share ›

Praveen ➦ learner • 2 years ago

It is necessary because if the linked list is of even length e.g. LL is 1->2 while loop prev is set to null but node is not set to null. So the condition as the while loop executes.

▲ | ▼ • Reply • Share ›

Venki • 3 years ago

Recursive method missing dual reference of head pointer. And similar modific against its NULL-ness.

▲ | ▼ • Reply • Share ›

Bandicoot • 3 years ago

Once again, Sambasiva's soln trumps both the solns that you gave above in te

Reply • Share ›

R.Srinivasan ➦ Bandicoot • 3 years ago

@Bandicoot

Sambasiva's program fails("runtime error") when the number of nodes temp=NULL(temp=l->next) when l becomes the last node and the illeg >next".(temp=NULL).

Reply • Share ›

Bandicoot ➦ R.Srinivasan • 3 years ago

My bad. You are right. Sambasiva should have added a if(!temp either temp or temp->next. Apart from that, his soln is good.

Reply • Share ›

R.Srinivasan • 3 years ago

```
void deletealtnode(struct node * head)
{
    struct node * current=head,*temp;
    while(current!=NULL && current->next!=NULL)
    {
           temp=current->next;
           current->next=temp->next;
           current=current->next;
           free(temp);
     }
}
```

Vinay Kumar • 4 years ago

```
typedef struct node
{
   int data;
   struct node *next;
}mynode;

void deleteAlt(mynode *head)
{
  /* Initialize prev and node to be deleted */
  mynode *current = head;
  mynode *nextPtr;

  while (current != NULL)
  {
     nextPtr = current->next;

     if(nextPtr)
```

see more

• Reply • Share ›

Techiee • 4 years ago

My 2 cents:

```
void deleteAlternate (Node *n) {
   while (n && n->next) {
       Node *nxt_nxt = n->next->next;
       delete (n->next);
       n->next = nxt_nxt;
```

```
        n = nxt_nxt;
    }
}
```