Move last element to front of a given Linked List

Write a C function that moves last element to front in a given Singly Linked List. For example, if the given Linked List is 1->2->3->4->5, then the function should change the list to 5->1->2->3->4.

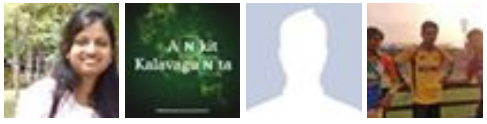
Algorithm:
Traverse the list till last node. Use two pointers: one to store the address of last node and other for address of second last node. After the end of loop do following operations.
i) Make second last as last (secLast->next = NULL).
ii) Set next of last as head (last->next = *head_ref).
iii) Make last as head (*head_ref = last)

```
/* Program to move last element to front in a given linked list */
#include<stdio.h>
#include<stdlib.h>

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

/* We are using a double pointer head_ref here because we change
   head of the linked list inside this function.*/
void moveToFront(struct node **head_ref)
{
  /* If linked list is empty, or it contains only one node,
    then nothing needs to be done, simply return */
  if(*head_ref == NULL || (*head_ref)->next == NULL)
    return;

  /* Initialize second last and last pointers */
```

```
  struct node *secLast = NULL;
  struct node *last = *head_ref;

  /*After this loop secLast contains address of second last
  node and last contains address of last node in Linked List */
  while(last->next != NULL)
  {
    secLast = last;
    last = last->next;
  }

  /* Set the next of second last as NULL */
  secLast->next = NULL;

  /* Set next of last as head node */
  last->next = *head_ref;

  /* Change the head pointer to point to last node now */
  *head_ref = last;
}

/* UTILITY FUNCTIONS */
/* Function to add a node at the begining of 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)
{
  while(node != NULL)
  {
    printf("%d ", node->data);
    node = node->next;
```

Geometric Algorithms

Popular Posts

```
    }
}

/* Druver program to test above function */
int main()
{
  struct node *start = NULL;

  /* The constructed linked list is:
   1->2->3->4->5 */
  push(&start, 5);
  push(&start, 4);
  push(&start, 3);
  push(&start, 2);
  push(&start, 1);

  printf("\n Linked list before moving last to front ");
  printList(start);

  moveToFront(&start);

  printf("\n Linked list after removing last to front ");
  printList(start);

  getchar();
}
```

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

Please write comments if you find any bug in above code/algorithm, or find other 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

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

13 Comments GeeksforGeeks

Sort by Newest

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 · 1 hour ago

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

Join the discussion…

wishall • a day ago

Bug:head node link should be made pointing 2 NULL,,,,
(*head_ref)->next=NULL; before *head_ref=last;

  • Reply • Share ›

Akash Panda • a month ago

```
void MoveLastToFront(struct node **head)

{

struct node *current=*head;

if(current==NULL || current->next==NULL)

return;

while(current->next->next!=NULL)

{

current=current->next;

}

struct node *temp=current->next;

current->next=NULL;

temp->next=*head;

*head=temp;

}
```

Find subarray with given sum · 2 hours ago

• Reply • Share ›

Himanshu Dagar • 3 months ago

even we can do it with a single pointer by keep track of forward nodes frm curr

• Reply • Share ›

Guest • 4 months ago

void move_last_node_to_beg(struct node **head)
{
struct node **temp=&((*head)->link); //temp holds address of link part of 1st n
if(*temp!=NULL) //this is just to handle the case that the 1ST node itself is not
{ while(((*temp)->link)!=NULL) //find the address present in the link field of 1st
and check if that nodes link field is null then quit the loop
, temp=&((*temp)->link); //this is basically to make temp to hold the next node'
//finally temp will hold address of the last but 1 nodes's link field's address...bc
nodes link field contains null
(*temp)->link=*head; //now change the address of the present in the last node
*head=*temp; //head node now points to where earlier the last but 1 node's link
node
*temp=NULL; //the last but 1 node's link field now contains null
}
}

• Reply • Share ›

adithya • 2 years ago

```
 /* Make last node first */
void reverse(node **head) {
        node *temp,*temp1;
        temp=*head;
        temp1=*head;
        temp=temp->link;
        while(temp1->link!=NULL) {
```

```
        while(temp1->link!=NULL) {
                temp=temp->link;
                temp1=temp1->link->link;
        }
        temp1->link=*head;
        *head=temp1;
        temp->link=NULL;
        return;
}
```


adithya • 2 years ago

```
 /* Function for making lastnode first*/
void reverse(node **head) {
        node *temp,*temp1;
        temp=*head;
        temp1=*head;
        temp=temp->link;
        while(temp1->link!=NULL) {
                temp=temp->link;
                temp1=temp1->link->link;
        }
        temp1->link=*head;
        *head=temp1;
        temp->link=NULL;
        return;
}
```


Venki • 3 years ago

Function to move last node to start of the list with only one crawl pointer. Com

```
 void moveToFront(struct node **head_ref)
{
    /* Proceed only when list is valid (efficient code) */
    if( *head_ref && (*head_ref)->next )
    {
        struct node *ite = *head_ref;

        /* Move to second last node */
        while( ite && ite->next && ite->next->next )
        {
            ite = ite->next;
        }

        /* Make the list circular */
        ite->next->next = *head_ref;
        /* Set up new head */
        *head_ref = ite->next;
        /* Break the loop */
        ite->next= NULL;
    }
}
```

1 ^ v • Reply • Share ›

Murali S Iyengar ➦ Venki • 4 months ago

@Venki
The check "ite && ite->next" in the while loop is redundant as you have >next in "if" at the beginning.

The while loop may be changed to

```
while (ite->next->next)
{
ite = ite->next;
}
```

1 · Reply · Share ›

renu Venki · 7 months ago

awesome!!!

· Reply · Share ›

Coder Venki · 11 months ago

Nice approach really good Venki :)

· Reply · Share ›

Soumya Sengupta Venki · a year ago

@venki-great iterative code.....enjoyed it...

```
/* Paste your code here (You may delete these lines if not wri
```

· Reply · Share ›

Sambasiva · 4 years ago

```
void moveToFront(struct node **head_ref)
{
    struct node *p = *head_ref;
    if(!p || !p->next) return;
    for(;p->next->next; p = p->next);
    p->next->next = *head_ref;
    *head_ref = p->next;
    p->next = NULL;
}
```

}

  • Reply • Share ›

Sam • 4 years ago

Below is C# version

```
public static LinkedList MoveLastItemToFirst(LinkedList head)
        {
            LinkedList last = null;
            LinkedList secondLast = null;
            LinkedList cur = head;

            while (null != cur)
            {
                secondLast = last;
                last = cur;
                cur = cur.Next;
            }

            if (null != last)
            {
                secondLast.Next = null;
                last.Next = head;
                head = last;
            }

            return head;
        }
```