GeeksforGeeks

A computer science portal for geeks

GeeksQuiz

Login

Home Algorithms DS GATE Interview Corner Q&A C C++ Java Books Contribute Ask a Q About

Array Bit Magic C/C++ Articles GFacts Linked List MCQ Misc Output String Tree Graph

Reverse a Linked List in groups of given size

Given a linked list, write a function to reverse every k nodes (where k is an input to the function).

```
Example:
Inputs:  1->2->3->4->5->6->7->8->NULL and k = 3
Output:  3->2->1->6->5->4->8->7->NULL.

Inputs:   1->2->3->4->5->6->7->80->NULL and k = 5
Output:  5->4->3->2->1->8->7->6->NULL.
```

Algorithm: *reverse(head, k)*
1) Reverse the first sub-list of size k. While reversing keep track of the next node and previous node. Let the pointer to the next node be *next* and pointer to the previous node be *prev*. See this post for reversing a linked list.
2) *head->next* = *reverse(next, k)* /* Recursively call for rest of the list and link the two sub-lists */
3) return *prev* /* *prev* becomes the new head of the list (see the diagrams of iterative method of this post) */

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

/* Link list node */
struct node
{
    int data;
    struct node* next;
};

/* Reverses the linked list in groups of size k and returns the
   pointer to the new head node. */
```

Interview Experiences
Advanced Data Structures
Dynamic Programming
Greedy Algorithms
Backtracking
Pattern Searching
Divide & Conquer
Mathematical Algorithms
Recursion
Geometric Algorithms

```
struct node *reverse (struct node *head, int k)
{
    struct node* current = head;
    struct node* next;
    struct node* prev = NULL;
    int count = 0;

    /*reverse first k nodes of the linked list */
    while (current != NULL && count < k)
    {
       next  = current->next;
       current->next = prev;
       prev = current;
       current = next;
       count++;
    }

    /* next is now a pointer to (k+1)th node
       Recursively call for the list starting from current.
       And make rest of the list as next of first node */
    if(next !=  NULL)
    {  head->next = reverse(next, k); }

    /* prev is new head of the input list */
    return prev;
}

/* UTILITY FUNCTIONS */
/* Function to push a node */
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 linked list */
void printList(struct node *node)
```

Geometric Algorithms

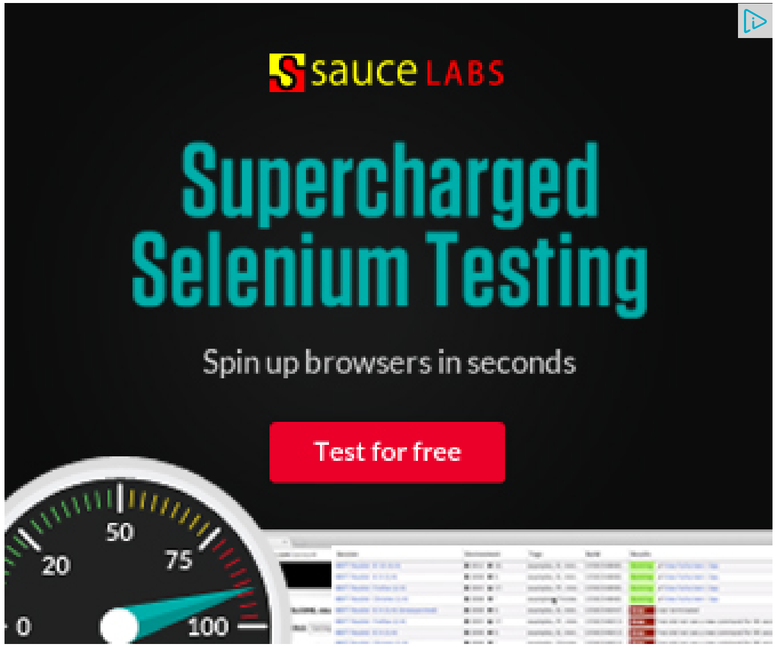
Popular Posts

- 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 function*/
int main(void)
{
    /* Start with the empty list */
    struct node* head = NULL;

     /* Created Linked list is 1->2->3->4->5->6->7->8 */
     push(&head, 8);
     push(&head, 7);
     push(&head, 6);
     push(&head, 5);
     push(&head, 4);
     push(&head, 3);
     push(&head, 2);
     push(&head, 1);

     printf("\n Given linked list \n");
     printList(head);
     head = reverse(head, 3);

     printf("\n Reversed Linked list \n");
     printList(head);

     getchar();
     return(0);
}
```

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

Please write comments if you find the above code/algorithm incorrect, 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

5 Tweet 0 0

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

705 Subscribe

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