

Add two numbers represented by linked lists | Set 2

Given two numbers represented by two linked lists, write a function that returns sum list. The sum list is linked list representation of addition of two input numbers. It is not allowed to modify the lists. Also, not allowed to use explicit extra space (Hint: Use Recursion).

Example

Input:

First List: 5->6->3 // represents number 563
Second List: 8->4->2 // represents number 842

Output

Resultant list: 1->4->0->5 // represents number 1405

We have discussed a solution [here](#) which is for linked lists where least significant digit is first node of lists and most significant digit is last node. In this problem, most significant node is first node and least significant digit is last node and we are not allowed to modify the lists. Recursion is used here to calculate sum from right to left.

Following are the steps.

- 1) Calculate sizes of given two linked lists.
- 2) If sizes are same, then calculate sum using recursion. Hold all nodes in recursion call stack till the rightmost node, calculate sum of rightmost nodes and forward carry to left side.
- 3) If size is not same, then follow below steps:
 -a) Calculate difference of sizes of two linked lists. Let the difference be *diff*
 -b) Move *diff* nodes ahead in the bigger linked list. Now use step 2 to calculate sum of smaller list and right sub-list (of same size) of larger list. Also, store the carry of this sum.
 -c) Calculate sum of the carry (calculated in previous step) with the remaining left sub-list of

Google™ Custom Search



GeeksforGeeks



53,527 people like [GeeksforGeeks](#).



[Interview Experiences](#)

[Advanced Data Structures](#)

[Dynamic Programming](#)

[Greedy Algorithms](#)

[Backtracking](#)

[Pattern Searching](#)

[Divide & Conquer](#)

[Mathematical Algorithms](#)

[Recursion](#)

larger list. Nodes of this sum are added at the beginning of sum list obtained previous step.

Following is C implementation of the above approach.

```
// A recursive program to add two linked lists

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

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

typedef struct node node;

/* A utility function to insert a node at the beginning 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;
}

/* A utility function to print linked list */
void printList(struct node *node)
{
    while (node != NULL)
    {
        printf("%d ", node->data);
        node = node->next;
    }
    printf("\n");
}

// A utility function to swap two pointers
void swapPointer( node** a, node** b )
```

Free Online Database

 zoho.com/creator/online-database

Create database apps in minutes.
Just Drag-&-drop. Try NOW!



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

```

{
    node* t = *a;
    *a = *b;
    *b = t;
}

/* A utility function to get size of linked list */
int getSize(struct node *node)
{
    int size = 0;
    while (node != NULL)
    {
        node = node->next;
        size++;
    }
    return size;
}

// Adds two linked lists of same size represented by head1 and head2 a:
// head of the resultant linked list. Carry is propagated while return
// the recursion
node* addSameSize(node* head1, node* head2, int* carry)
{
    // Since the function assumes linked lists are of same size,
    // check any of the two head pointers
    if (head1 == NULL)
        return NULL;

    int sum;

    // Allocate memory for sum node of current two nodes
    node* result = (node *)malloc(sizeof(node));

    // Recursively add remaining nodes and get the carry
    result->next = addSameSize(head1->next, head2->next, carry);

    // add digits of current nodes and propagated carry
    sum = head1->data + head2->data + *carry;
    *carry = sum / 10;
    sum = sum % 10;

    // Assign the sum to current node of resultant list
    result->data = sum;

    return result;
}

```



```
// This function is called after the smaller list is added to the bigger
// lists's sublist of same size. Once the right sublist is added, the
// must be added to left side of larger list to get the final result.
void addCarryToRemaining(node* head1, node* cur, int* carry, node** re
{
    int sum;

    // If diff. number of nodes are not traversed, add carry
    if (head1 != cur)
    {
        addCarryToRemaining(head1->next, cur, carry, result);

        sum = head1->data + *carry;
        *carry = sum/10;
        sum %= 10;

        // add this node to the front of the result
        push(result, sum);
    }
}
```

```
// The main function that adds two linked lists represented by head1 and head2
// The sum of two lists is stored in a list referred by result
void addList(node* head1, node* head2, node** result)
{
    node *cur;

    // first list is empty
    if (head1 == NULL)
    {
        *result = head2;
        return;
    }

    // second list is empty
    else if (head2 == NULL)
    {
        *result = head1;
        return;
    }

    int size1 = getSize(head1);
    int size2 = getSize(head2);

    int carry = 0;

    // Add same size lists
```

705



Subscribe

Recent Comments

Abhi You live US or India?

Google (Mountain View) interview · 44 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

AdChoices

► [Linked List](#)

► [C++ Code](#)

► [Programming C++](#)

AdChoices

► [Java Array](#)

```

if (size1 == size2)
    *result = addSameSize(head1, head2, &carry);

else
{
    int diff = abs(size1 - size2);

    // First list should always be larger than second list.
    // If not, swap pointers
    if (size1 < size2)
        swapPointer(&head1, &head2);

    // move diff. number of nodes in first list
    for (cur = head1; diff--; cur = cur->next);

    // get addition of same size lists
    *result = addSameSize(cur, head2, &carry);

    // get addition of remaining first list and carry
    addCarryToRemaining(head1, cur, &carry, result);
}

// if some carry is still there, add a new node to the front of
// the result list. e.g. 999 and 87
if (carry)
    push(result, carry);
}

```

// Driver program to test above functions

```

int main()
{
    node *head1 = NULL, *head2 = NULL, *result = NULL;

    int arr1[] = {9, 9, 9};
    int arr2[] = {1, 8};

    int size1 = sizeof(arr1) / sizeof(arr1[0]);
    int size2 = sizeof(arr2) / sizeof(arr2[0]);

    // Create first list as 9->9->9
    int i;
    for (i = size1-1; i >= 0; --i)
        push(&head1, arr1[i]);

    // Create second list as 1->8
    for (i = size2-1; i >= 0; --i)
        push(&head2, arr2[i]);
}

```

[► Numbers Number](#)

AdChoices 

[► Null Pointer](#)

[► Memory Array](#)

[► C++ Example](#)

```

    addList(head1, head2, &result);

    printList(result);

    return 0;
}

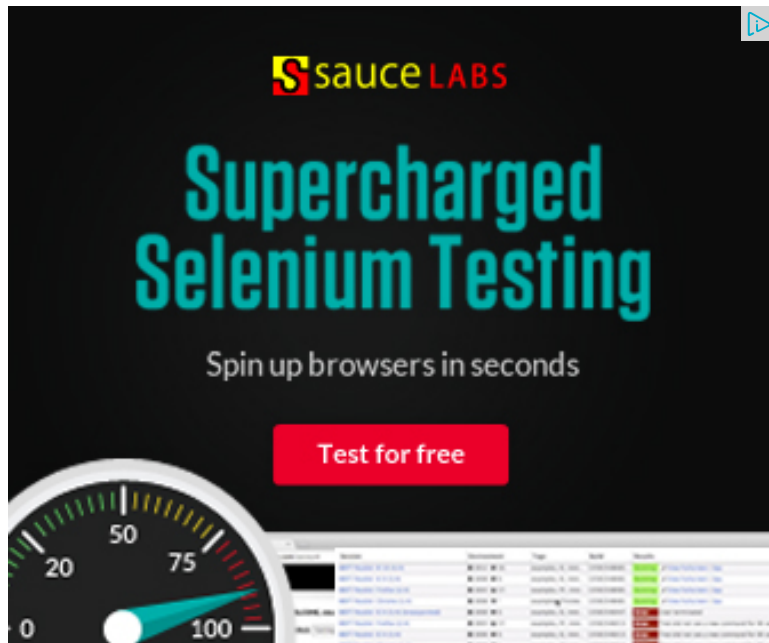
```

Output:

```
1 0 1 7
```

Time Complexity: $O(m+n)$ where m and n are the sizes of given two linked lists.

Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.



Related Topics:

- 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



19



Tweet

2



1

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

@geeksforgeeks, **Some rights reserved**

Contact Us!

Powered by **WordPress** & **MooTools**, customized by geeksforgeeks team