

Add two numbers represented by linked lists | Set 1

Given two numbers represented by two lists, write a function that returns sum list. The sum list is list representation of addition of two input numbers.

Example 1

Input:

First List: 5->6->3 // represents number 365
Second List: 8->4->2 // represents number 248

Output

Resultant list: 3->1->6 // represents number 613

Example 2

Input:

First List: 7->5->9->4->6 // represents number 64957
Second List: 8->4 // represents number 48

Output

Resultant list: 5->0->0->5->6 // represents number 65005

Solution

Traverse both lists. One by one pick nodes of both lists and add the values. If sum is more than 10 then make carry as 1 and reduce sum. If one list has more elements than the other then consider remaining values of this list as 0. Following is C implementation of this approach.

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

/* Linked list node */
```

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](#)

[Geometric Algorithms](#)

```

struct node
{
    int data;
    struct node* next;
};

/* Function to create a new node with given data */
struct node *newNode(int data)
{
    struct node *new_node = (struct node *) malloc(sizeof(struct node)
    new_node->data = data;
    new_node->next = NULL;
    return new_node;
}

/* Function to insert a node at the beginning of the Doubly Linked Lis
void push(struct node** head_ref, int new_data)
{
    /* allocate node */
    struct node* new_node = newNode(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;
}

/* Adds contents of two linked lists and return the head node of result
struct node* addTwoLists (struct node* first, struct node* second)
{
    struct node* res = NULL; // res is head node of the resultant list
    struct node *temp, *prev = NULL;
    int carry = 0, sum;

    while (first != NULL || second != NULL) //while both lists exist
    {
        // Calculate value of next digit in resultant list.
        // The next digit is sum of following things
        // (i) Carry
        // (ii) Next digit of first list (if there is a next digit)
        // (ii) Next digit of second list (if there is a next digit)
        sum = carry + (first? first->data: 0) + (second? second->data: 0);

        // update carry for next calculation
        carry = (sum >= 10)? 1 : 0;
    }
}

```



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

```

// update sum if it is greater than 10
sum = sum % 10;

// Create a new node with sum as data
temp = newNode(sum);

// if this is the first node then set it as head of the result.
if(res == NULL)
    res = temp;
else // If this is not the first node then connect it to the r
    prev->next = temp;

// Set prev for next insertion
prev = temp;

// Move first and second pointers to next nodes
if (first) first = first->next;
if (second) second = second->next;
}

if (carry > 0)
    temp->next = newNode(carry);

// return head of the resultant list
return res;
}

```

```

// A utility function to print a linked list

```

```

void printList(struct node *node)
{
    while(node != NULL)
    {
        printf("%d ", node->data);
        node = node->next;
    }
    printf("\n");
}

```

```

/* Drier program to test above function */

```

```

int main(void)
{
    struct node* res = NULL;
    struct node* first = NULL;
    struct node* second = NULL;

```

```

// create first list 7->5->9->4->6
push(&first, 6);

```

HIGH-PERFORMANCE COMPUTING ON A UNIVERSITY BUDGET

The quest to find tailored server capabilities on a shoestring


To conduct successful university research and crunch massive amounts of data, you need the highest-performing hardware on the market. Unfortunately, the price tag of high performance can be daunting for researchers who rely on grants to fund their projects.

There are many off-the-shelf server options available, of course, and your department may even have a standardized, prescribed server.

The best bang for your buck, however, may be to configure a custom server to fit your exact high-performance computing needs.

Unsure of how to get the best performance out of your application? Do you need assistance answering any of these questions?

Download To See Entire Infographic



WHAT NETWORKING INTERFACES ARE NEEDED?

☐ 1GbE ☐ InfiniBand
☐ 10GbE ☐ Unsure
☐ 40GbE

HOW MANY CPU CORES DO YOU WANT?

☐ 4 ☐ 16
☐ 8 ☐ Unsure
☐ 12

HOW MUCH STORAGE DO YOU WANT?

☐ 1TB ☐ 1PB
☐ Unsure

WHAT OPERATING SYSTEM ARE YOU RUNNING?

☐ Linux Variant
☐ Windows Server
☐ VMware
☐ Unsure

Recent Comments

Abhi You live US or India?

[Google \(Mountain View\) interview](#) · 43 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

```
push(&first, 4);
push(&first, 9);
push(&first, 5);
push(&first, 7);
printf("First List is ");
printList(first);

// create second list 8->4
push(&second, 4);
push(&second, 8);
printf("Second List is ");
printList(second);

// Add the two lists and see result
res = addTwoLists(first, second);
printf("Resultant list is ");
printList(res);

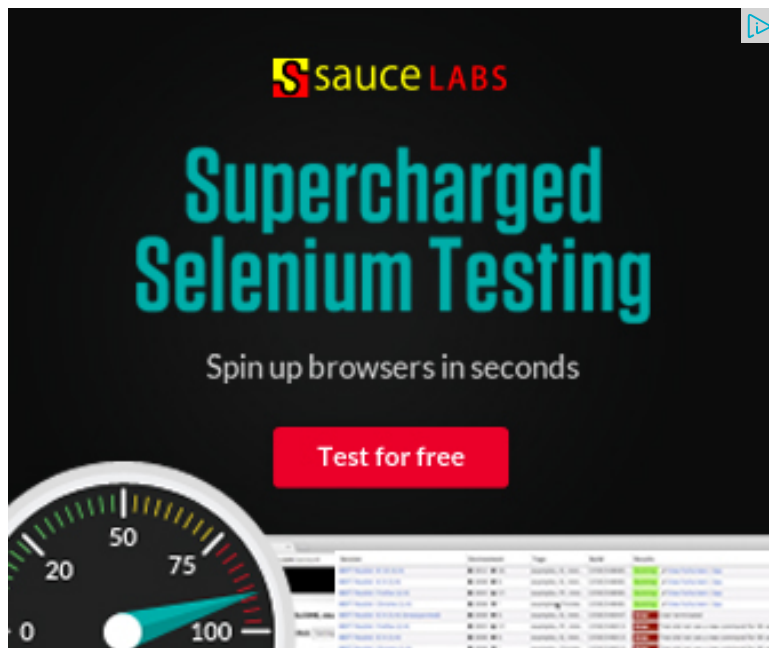
return 0;
}
```

Output:

```
First List is 7 5 9 4 6
Second List is 8 4
Resultant list is 5 0 0 5 6
```

Time Complexity: $O(m + n)$ where m and n are number of nodes in first and second lists respectively.

Please write comments if you find the above codes/algorithms incorrect, or find other ways to solve the same problem.



AdChoices

► [C++ Vector](#)

► [C++ Code](#)

► [Linked List](#)

AdChoices

► [Programming C++](#)

► [Numbers Up](#)

► [Numbers Number](#)

AdChoices

► [Numbers Up](#)

► [4 Digit Numbers](#)

► [Numbers Number](#)

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



6



0



0

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