Insert a Node at the Tail of a Linked List

by harsha_s

Problem Submissions Leaderboard Discussions Editorial

This challenge is part of a tutorial track by MyCodeSchool and is accompanied by a video lesson.

You are given the pointer to the head node of a linked list and an integer to add to the list. Create a new node with the given integer. Insert this node at the tail of the linked list and return the head node. The given head pointer may be null, meaning that the initial list is empty.

Input Format

You have to complete the `Node* Insert(Node* head, int data)` method. It takes two arguments: the head of the linked list and the integer to insert. You should **not** read any input from the stdin/console.

Output Format

Insert the new node at the tail and just `return` the head of the updated linked list. Do **not** print anything to stdout/console.

Sample Input

NULL, data = **2**
2 --> NULL, data = **3**

Sample Output

```
2 -->NULL
2 --> 3 --> NULL
```

Explanation

1. We have an empty list, and we insert **2**.
2. We start with a **2** in the tail. When **3** is inserted, **3** then becomes the tail.

Video lesson

Submissions: 68745
Max Score: 5
Difficulty: Easy

Rate This Challenge:

More

Current Buffer (saved locally, editable) C++

```
/*
  Insert Node at the end of a linked list
  head pointer input could be NULL as well for empty list
  Node is defined as
  struct Node
```

```
  {
     int data;
     struct Node *next;
  }
*/
Node* Insert(Node *head,int data)
{
    struct Node* buffer = (struct Node*)malloc(sizeof(Node));
    buffer->data = data;
    buffer->next = NULL;
    if (!head)
        return buffer;
    struct Node* answer = head;
    while (answer)
    {
        if (answer->next  == NULL)
        {
            answer->next = buffer;
            return head;
        }
        answer = answer->next;
    }
    return head;
}

```

Line: 14 Col: 25

Upload Code as File

Run Code | Submit Code

Testcase 0 ✔

Congratulations, you passed the sample test case.

Click the **Submit Code** button to run your code against all the test cases.

Input (stdin)

```
4
2
3
4
1
```

Your Output (stdout)

```
Right Answer!
```

Expected Output

```
Right Answer!
```