Inserting a Node Into a Sorted Doubly Linked List ☆

Your Inserting a Node Into a Sorted Doubly Linked List submission got 5.00 points. Share Tweet ✕

Try the next challenge

Problem Submissions Leaderboard Editorial 🔒

Given a reference to the head of a doubly-linked list and an integer, $data$, create a new DoublyLinkedListNode object having data value $data$ and insert it at the proper location to maintain the sort.

Example

$head$ refers to the list $1 \leftrightarrow 2 \leftrightarrow 4 \rightarrow NULL$

$data = 3$

Return a reference to the new list: $1 \leftrightarrow 2 \leftrightarrow 3 \leftrightarrow 4 \rightarrow NULL$.

Function Description

Complete the sortedInsert function in the editor below.

sortedInsert has two parameters:

- DoublyLinkedListNode pointer head: a reference to the head of a doubly-linked list
- int data: An integer denoting the value of the $data$ field for the DoublyLinkedListNode you must insert into the list.

Returns

- DoublyLinkedListNode pointer: a reference to the head of the list

Note: Recall that an empty list (i.e., where $head = \texttt{NULL}$) and a list with one element are sorted lists.

Input Format

The first line contains an integer t, the number of test cases.

Each of the test case is in the following format:

- The first line contains an integer n, the number of elements in the linked list.
- Each of the next n lines contains an integer, the data for each node of the linked list.
- The last line contains an integer, $data$, which needs to be inserted into the sorted doubly-linked list.

Constraints

- $1 \leq t \leq 10$
- $1 \leq n \leq 1000$
- $1 \leq DoublyLinkedListNode.data \leq 1000$

Sample Input

```
1
4
1
3
4
10
5
```

Sample Output

```
1 3 4 5 10
```

Explanation

The initial doubly linked list is: $1 \leftrightarrow 3 \leftrightarrow 4 \leftrightarrow 10 \rightarrow NULL$.

The doubly linked list after insertion is: $1 \leftrightarrow 3 \leftrightarrow 4 \leftrightarrow 5 \leftrightarrow 10 \rightarrow NULL$

Change Theme C++

```cpp
DoublyLinkedListNode* sortedInsert(DoublyLinkedListNode* head, int data) {
    DoublyLinkedListNode* node = new DoublyLinkedListNode(data);
    if(!head) return node;  //if empty list then return node
    if(head-> data> data){   // data is less then 1st element in list
        head->prev = node;
        node-> next = head;
        return node;    }
    DoublyLinkedListNode* temp = head;
    while(temp != NULL){  // can't handle 1st element because 1st if creates connetion to prev_node->next (temp->prev = prev_node) which is Null in case data is smallest ele.
        if(data <= temp->data){
            temp->prev->next= node;
            node->prev = temp->prev;
            node->next=temp;
            break;}
        else if(temp->next == NULL){ //if data is the largest element
            temp->next = node;
            node->prev = temp;
            break;        }
        else{
            temp = temp->next;  }
    }
    return head;  //https://www.youtube.com/watch?v=DGez4qfCGMA&ab_channel=nexTRIE
}

```

Line: 96 Col: 21

Upload Code as File ☐ Test against custom input Run Code Submit Code

You have earned 5.00 points!

These points will also count towards your progress in the Problem Solving Badge.

47% 330/475

Problem Solving ★★★

Congratulations

You solved this challenge. Would you like to challenge your friends? Next Challenge

Earn a certificate in Problem Solving

Kudos on your progress! Take the HackerRank Skills Certification test and enrich your profile Get Certified

✓ Test case 0

Compiler Message

Success

✓ Test case 1

✓ Test case 2 🔒

✓ Test case 3 🔒

✓ Test case 4 🔒

✓ Test case 5 🔒

✓ Test case 6 🔒

Input (stdin) Download

```
1
4
1
3
4
10
5
```

Expected Output Download

Contest Calendar | Blog | Scoring | Environment | FAQ | About Us | Support | Careers | Terms Of Service | Privacy Policy | Request a Feature