Print in Reverse

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

Given a pointer to the head of a singly-linked list, print each $data$ value from the reversed list. If the given list is empty, do not print anything.

Example

$head*$ refers to the linked list with $data$ values $1 \rightarrow 2 \rightarrow 3 \rightarrow NULL$

Print the following:

```
3
2
1
```

Function Description

Complete the *reversePrint* function in the editor below.

reversePrint has the following parameters:

- *SinglyLinkedListNode pointer head:* a reference to the head of the list

Prints

The $data$ values of each node in the reversed list.

Input Format

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

The input of each test case is as follows:

- The first line contains an integer n, the number of elements in the list.
- Each of the next *n* lines contains a data element for a list node.

Constraints

- $1 \leq n \leq 1000$
- $1 \leq list[i] \leq 1000$, where $list[i]$ is the i^{th} element in the list.

Sample Input

```
3
5
16
12
4
```

```
2
5
3
7
3
9
5
5
1
18
3
13
```

Sample Output

```
5
2
4
12
16
9
3
7
13
3
18
1
5
```

Explanation

There are three test cases. There are no blank lines between test case output.

The first linked list has 5 elements: $16 \rightarrow 12 \rightarrow 4 \rightarrow 2 \rightarrow 5$. Printing this in reverse order produces:

```
5
2
4
12
16
```

The second linked list has 3 elements: $7 \rightarrow 3 \rightarrow 9 \rightarrow NULL$. Printing this in reverse order produces:

```
9
3
7
```

The third linked list has 5 elements: $5 \rightarrow 1 \rightarrow 18 \rightarrow 3 \rightarrow 13 \rightarrow NULL$. Printing this in reverse order produces:

```
13
3
18
1
5
```

Contest ends in 3 hours

Submissions: 177
Max Score: 40
Difficulty: Easy

Rate This Challenge:
☆☆☆☆☆

More

C++

```
#include ↔

using namespace std;

class SinglyLinkedListNode {
    public:
        int data;
        SinglyLinkedListNode *next;

        SinglyLinkedListNode(int node_data) {
            this->data = node_data;
            this->next = nullptr;
        }
};

class SinglyLinkedList {
    public:
        SinglyLinkedListNode *head;
        SinglyLinkedListNode *tail;

        SinglyLinkedList() {
            this->head = nullptr;
            this->tail = nullptr;
        }

        void insert_node(int node_data) {
            SinglyLinkedListNode* node = new SinglyLinkedListNode(node_data);

            if (!this->head) {
                this->head = node;
            } else {
                this->tail->next = node;
            }

            this->tail = node;
        }
};

void print_singly_linked_list(SinglyLinkedListNode* node, string sep) {
    while (node) {
        cout << node->data;

        node = node->next;

        if (node) {
            cout << sep;
        }
    }
}

void free_singly_linked_list(SinglyLinkedListNode* node) {
    while (node) {
        SinglyLinkedListNode* temp = node;
        node = node->next;

        free(temp);
    }
}

/*
 * Complete the 'reversePrint' function below.
 *
 * The function accepts INTEGER_SINGLY_LINKED_LIST llist as parameter.
 */
```

```

/*
 * For your reference:
 *
 * SinglyLinkedListNode {
 *     int data;
 *     SinglyLinkedListNode* next;
 * };
 *
 */

void reversePrint(SinglyLinkedListNode* llist) {
    if (llist == NULL) {
        return;
    } else {
        reversePrint(llist->next);
        cout << llist_count->data << endl;
    }
}
```

```
int main()
{
    int tests;
    cin >> tests;
    cin.ignore(numeric_limits<streamsize>::max(), '\n');

    for (int tests_itr = 0; tests_itr < tests; tests_itr++) {
        SinglyLinkedList* llist = new SinglyLinkedList();

        int llist_count;
        cin >> llist_count;
        cin.ignore(numeric_limits<streamsize>::max(), '\n');

        for (int i = 0; i < llist_count; i++) {
            int llist_item;
            cin >> llist_item;
            cin.ignore(numeric_limits<streamsize>::max(), '\n');

            llist->insert_node(llist_item);
        }

        reversePrint(llist->head);
    }

    return 0;
}

```

Line: 26 Col: 1

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

Interview Prep | Blog | Scoring | Environment | FAQ | About Us | Support | Careers | Terms Of Service | Privacy Policy |