This challenge is part of a tutorial track by MyCodeSchool

Given pointers to the head nodes of 2 linked lists that merge together at some point, find the node where the two lists merge. The merge point is where both lists point to the same node, i.e. they reference the same memory location. It is guaranteed that the two head nodes will be different, and neither will be NULL. If the lists share a common node, return that node's $data$ value.

Note: After the merge point, both lists will share the same node pointers.

Example

In the diagram below, the two lists converge at Node `x`:

```
[List #1] a--->b--->c
                    \
                     x--->y--->z--->NULL
                    /
    [List #2] p--->q
```

Function Description

Complete the findMergeNode function in the editor below.

findMergeNode has the following parameters:

- SinglyLinkedListNode pointer head1: a reference to the head of the first list
- SinglyLinkedListNode pointer head2: a reference to the head of the second list

Returns

- int: the $data$ value of the node where the lists merge

Input Format

Do not read any input from stdin/console.

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

Each of the test cases is in the following format:

The first line contains an integer, $index$, the node number where the merge will occur.

The next line contains an integer, $list1_count$ that is the number of nodes in the first list.

Each of the following $list1_count$ lines contains a $data$ value for a node. The next line contains an integer, $list2_count$ that is the number of nodes in the second list.

Change Theme Language C

```c
#include <assert.h>...

// Complete the findMergeNode function below.

/*
 * For your reference:
 *
 * SinglyLinkedListNode {
 *     int data;
 *     SinglyLinkedListNode* next;
 * };
 *
 */
int findMergeNode(SinglyLinkedListNode* head1, SinglyLinkedListNode* head2) {
while(head1){
        SinglyLinkedListNode *tmp = head1->next;
        head1->next = NULL;
        head1 = tmp;
    }

    while(head2){
        if(head2->next == NULL){
            return head2->data;
        }
        head2 = head2->next;
    }
    return 0;

}

int main()...
```

Line: 94 Col: 2

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

Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

Sample Test case 0

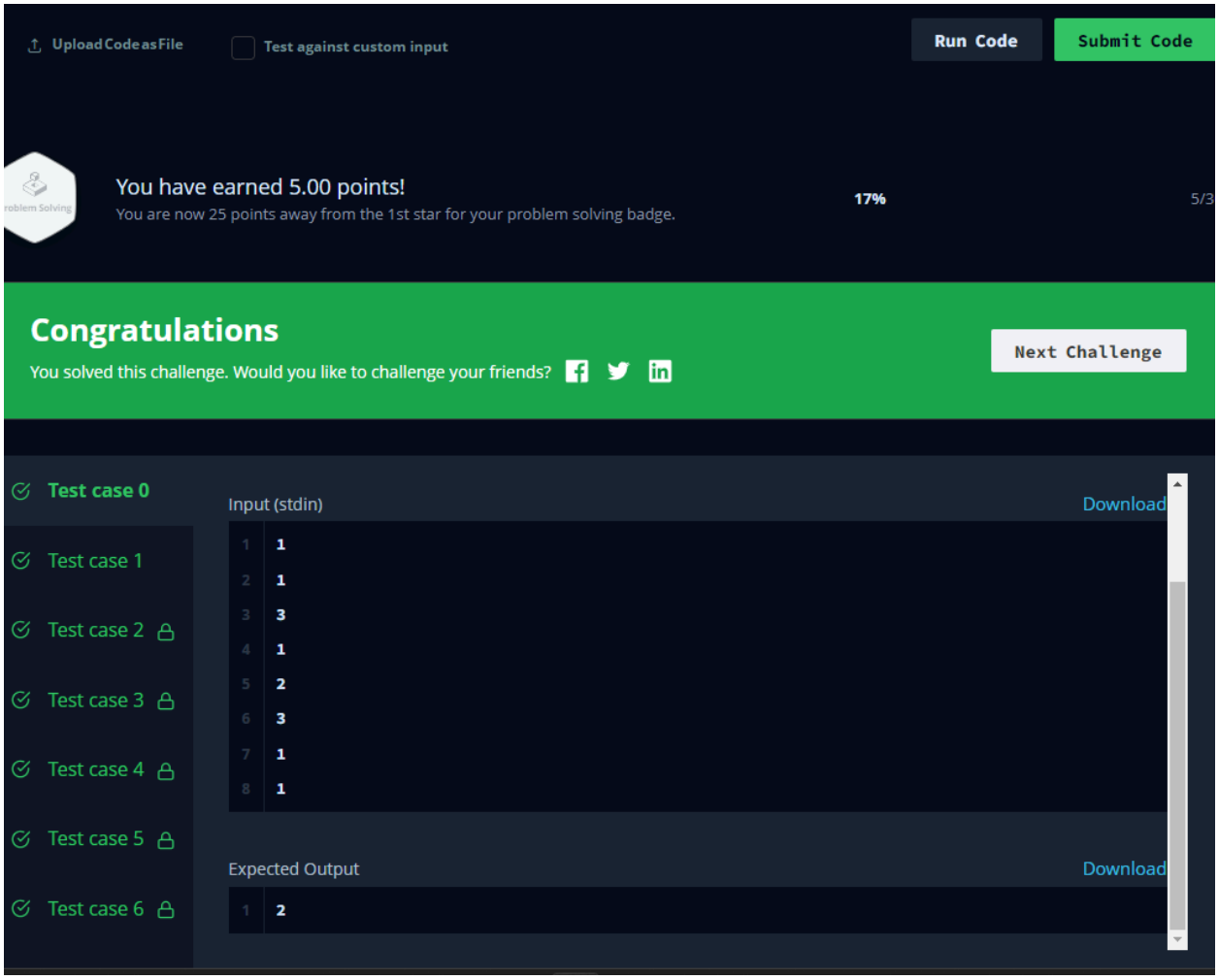
Sample Test case 1

Input (stdin) Download

```
1
1
3
1
2
3
1
1
```

Your Output (stdout)

```
2
```


Upload Code as File
Test against custom input
Run Code
Submit Code
You have earned 5.00 points!
You are now 25 points away from the 1st star for your problem solving badge.
17%
5/3
Congratulations
You solved this challenge. Would you like to challenge your friends?
Next Challenge
Test case 0
Test case 1
Test case 2
Test case 3
Test case 4
Test case 5
Test case 6
Input (stdin)
Download
1 1
2 1
3 3
4 1
5 2
6 3
7 1
8 1
Expected Output
Download
1 2