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. It is guaranteed that the two head Nodes will be different, and neither will be NULL.

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

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

Complete the `int findMergeNode(SinglyLinkedListNode* head1, SinglyLinkedListNode* head2)` method so that it finds and returns the data value of the Node where the two lists merge.

Input Format

Do not read any input from stdin/console.

The `findMergeNode(SinglyLinkedListNode,SinglyLinkedListNode)` method has two parameters, $head1$ and $head2$, which are the non-null head Nodes of two separate linked lists that are guaranteed to converge.

Constraints

The lists will merge.
$head1, head2 \neq null$.
$head1 \neq head2$.

Output Format

Do not write any output to stdout/console.

Each Node has a data field containing an integer. Return the integer data for the Node where the two lists merge.

Sample Input

The diagrams below are graphical representations of the lists that input Nodes $headA$ and $headB$ are connected to. Recall that this is a method-only challenge; the method only has initial visibility to those 2 Nodes and must explore the rest of the Nodes using some algorithm of your own design.

Test Case 0

```
 1
  \
   2--->3--->NULL
  /
 1
```

Test Case 1

```
1--->2
      \
       3--->Null
      /
     1
```

Sample Output

```
2
3
```

Explanation

Test Case 0: As demonstrated in the diagram above, the merge Node's data field contains the integer 2.

Test Case 1: As demonstrated in the diagram above, the merge Node's data field contains the integer **3**.