Cycle Detection

A linked list is said to contain a *cycle* if any node is visited more than once while traversing the list.

Complete the function provided for you in your editor. It has one parameter: a pointer to a *Node* object named $head$ that points to the head of a linked list. Your function must return a boolean denoting whether or not there is a cycle in the list. If there *is* a cycle, return *true*; otherwise, return *false*.

Note: If the list is empty, $head$ will be *null*.

Input Format

Our hidden code checker passes the appropriate argument to your function. You are not responsible for reading any input from stdin.

Constraints

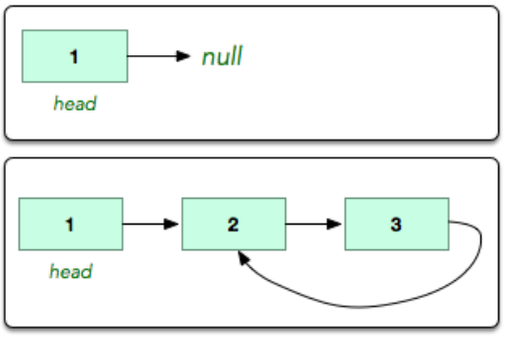
- $0 \leq$ $list\ size$ ≤ 100

Output Format

If the list contains a cycle, your function must return *true*. If the list *does not* contain a cycle, it must return *false*. The binary integer corresponding to the boolean value returned by your function is printed to stdout by our hidden code checker.

Sample Input

The following linked lists are passed as arguments to your function:



Sample Output

```
0
1
```

Explanation

1. The first list has no cycle, so we return *false* and the hidden code checker prints 0 to stdout.
2. The second list has a cycle, so we return *true* and the hidden code checker prints 1 to stdout.