



# Linked Lists: Detect a Cycle

by [harsha\\_s](#)

Problem

Submissions

Leaderboard

Discussions

Editorial

Check out the resources on the page's right side to learn more about linked lists. The video tutorial is by Gayle Laakmann McDowell, author of the best-selling interview book [Cracking the Coding Interview](#).

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 in the editor below. 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

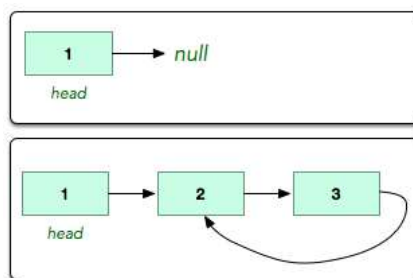
- $0 \leq \text{list size} \leq 100$

## 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

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

Submissions: 18091

Max Score: 25

Difficulty: Easy

Rate This Challenge:




Need Help?

7:43

Linked Lists

More

Current Buffer (saved locally, editable)  

Java 7



```
1  /*
2  Detect a cycle in a linked list. Note that the head pointer may be 'null' if the list is empty.
3
4  A Node is defined as:
5      class Node {
6          int data;
7          Node next;
8      }
9  */
10
11 boolean hasCycle(Node head) {
12     Node fast, slow;
13
14
15     if (head == null)
16         return false;
17
18     fast = head.next;
19     slow = head;
20
21     while (fast != null && fast.next != null && slow != null) {
22
23         if (fast == slow) {
24             return true;
25         }
26
27         fast = fast.next.next;
28         slow = slow.next;
29
30     }
31
32     return false;
33 }
34
35
36
```

Line: 12 Col: 5

 Upload Code as File☐ Test against custom input

Run Code

Submit Code

Congrats, you solved this challenge!

✓ Test Case #0

✓ Test Case #1

Next Challenge

Copyright © 2017 HackerRank. All Rights Reserved

Join us on IRC at [#hackerrank](#) on freenode for hugs or bugs.

[Contest Calendar](#) | [Interview Prep](#) | [Blog](#) | [Scoring](#) | [Environment](#) | [FAQ](#) | [About Us](#) | [Support](#) | [Careers](#) | [Terms Of Service](#) | [Privacy Policy](#) | [Request a Feature](#)