[Click on the logo to find **Problem Statement**]

## Intuition

The algorithm uses two pointers, a slow pointer (tortoise) and a fast pointer (hare), both initially set to the head of the linked list. The pointers traverse the list at different speeds, with the fast pointer moving twice as fast as the slow one. If there is a cycle in the linked list, the fast pointer will eventually catch up to the slow one, detecting the cycle. On the other hand, if the linked list is acyclic, the fast pointer will reach the end. The function returns `True` if a cycle is detected and `False` otherwise. Additionally, the code includes initial checks for cases where the linked list is empty or has only one node, in which case it is trivially cycle-free.

## Approach

1. **Initialization:**
    - Initialize both pointers (slow and fast) to the head of the linked list.
2. **Traversal:**
    - Use a while loop to iterate through the linked list.
    - In each iteration, move the slow pointer one step and the fast pointer two steps.
3. **Cycle Detection:**
    - If there is a cycle in the linked list, the fast pointer will eventually catch up to the slow pointer.
    - Check if the slow pointer is equal to the fast pointer. If true, return `True` as a cycle is detected.
4. **Termination:**
    - If the fast pointer reaches the end of the linked list (i.e., becomes `None`), return `False` as no cycle is found.

## Complexity

- Time complexity: O(n)
- Space complexity: O(1)

## Code

```python
# Definition for singly-linked list.
# class ListNode:
#     def __init__(self, x):
#         self.val = x
#         self.next = None

class Solution:
    def hasCycle(self, head: Optional[ListNode]) -> bool:
        if head is None or head.next is None:
            return False

        slow = head
        fast = head
        while fast is not None and fast.next is not None:
            fast = fast.next.next
            slow = slow.next
            if slow == fast:
                return True
        return False
```

If you want to see more solutions to coding problems, you can visit:

GitHub