141 Linked List Cycle

2018年4月3日 13:12

Question:

Given a linked list, determine if it has a cycle in it.
Follow up:
Can you solve it without using extra space?

来自 <https://leetcode.com/problems/linked-list-cycle/description/>

给定一个链表，判断链表中否有环。
补充：
你是否可以不用额外空间解决此题。

Solution for Python3:

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

class Solution1(object):
    def hasCycle(self, head):
        """
        :type head: ListNode
        :rtype: bool
        """
        p, q = head, head
        while q and q.next:
            p = p.next
            q = q.next.next
            if p == q:
                return True
        return False

```

```python
# Hash Tabel
class Solution2(object):
    def hasCycle(self, head):
        """
        :type head: ListNode
        :rtype: bool
        """
        D = {}
        while head:
            if head in D:
                return True
            else:
                D[head] = 1
            head = head.next
        return False
```

Solution for C++:

```cpp
/**
 * Definition for singly-linked list.
 * struct ListNode {
 *     int val;
 *     ListNode *next;
 *     ListNode(int x) : val(x), next(NULL) {}
 * };
 */
class Solution1 {
public:
    bool hasCycle(ListNode *head) {
        if (!head) {
            return false;
        }
        ListNode *F = head, *S = head;
        while (S && S->next) {
            F = F->next;
            S = S->next->next;
            if (F == S) {
                return true;
            }

        }
        return false;
    }
};

class Solution2 {
public:
    bool hasCycle(ListNode *head) {
        set<ListNode*> s;
        while (head) {
            if (s.count(head)) {
                return true;
            } else {
                s.insert(head);
            }
            head = head->next;
        }
        return false;
    }
};
```