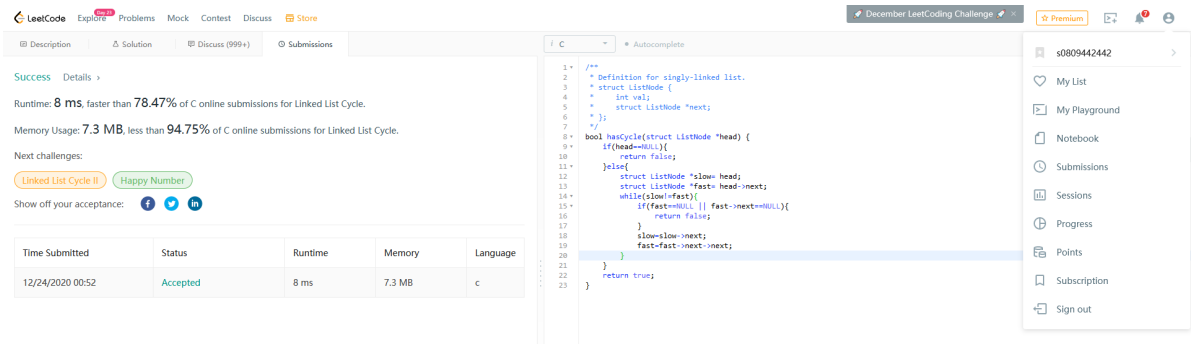
截圖



Source Code

```c
bool hasCycle(struct ListNode *head) {
    if(head==NULL){
        return false;
    }else{
        struct ListNode *slow= head;
        struct ListNode *fast= head->next;
        while(slow!=fast){
            if(fast==NULL || fast->next==NULL){
                return false;
            }
            slow=slow->next;
            fast=fast->next->next;
        }
    }
    return true;
}
```

解釋

本題要求查看一個Linking List的實作是否有循環，首先開頭為NULL先判掉(Line2~4)，必定沒有循環，再來有循環的情況下，如同再跑操場，假設有兩個人在跑步速度不一樣，那就一定會有相遇的一天，要是沒有循環，一定會跑到底。

因此實作上設定兩個struct ListNode的pointer，一個為開頭slow，一個為開頭下一個fast(Line 5)，當slow不等於fast時(Line7)，也就是說他們還沒相遇，那就繼續做，slow會指向下一個Node(Line11)，而fast會指向下一個的下一個(Line12)，也就是說slow的速度是1，fast的速度是2，必定會找到相遇的點，當然沒有循環的情況下，由於fast比較快，因此要不是fast撞到最後一個NULL，不然就是fast的下一個撞到NULL(因為fast跳著做)，這兩種情況下就代表Linking List沒有循環，return false(Line8~10)。