画解剑指 Offer - LeetBook - 力扣（LeetCode）全球极客挚爱的技术成长平台

leetcode.cn/leetbook/read/illustrate-lcof/xztfus

剑指 Offer 06. 从尾到头打印链表 - 解决方案

题目描述

输入一个链表的头节点，从尾到头反过来返回每个节点的值（用数组返回）。

示例 1：

```
输入：head = [1,3,2]
输出：[2,3,1]
```

限制：

`0 <= 链表长度 <= 10000`

解题方案

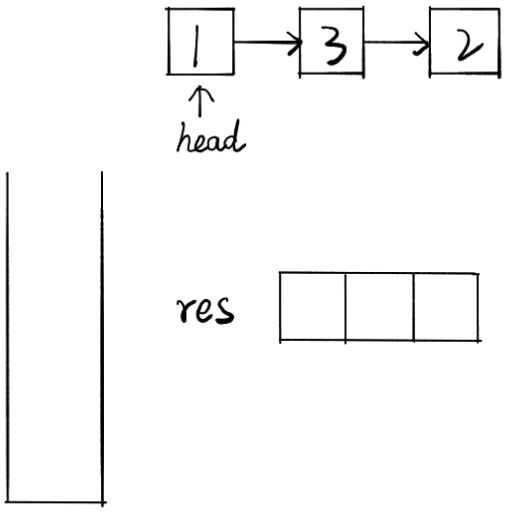
思路

- 标签：栈
- 栈的特点是先进后出，因为题目要求从尾到头打印元素，所以符合栈的特性
    - 先遍历一遍链表，将链表中的元素存入到栈中
    - 再不断弹出栈内元素，将弹出元素存放到结果数组中
- 也有使用递归来进行解题的，在此提出一个思考，递归和栈的关系是什么？其实递归的本质也是在使用栈，只不过是程序调用栈，因为没有显式在代码中体现出来，所以常常被忽略了
- 时间复杂度：O(n)O(n)，空间复杂度：O(n)O(n)

代码

```
/**
 * Definition for singly-linked list.
 * public class ListNode {
 *     int val;
 *     ListNode next;
 *     ListNode(int x) { val = x; }
 * }
 */
class Solution {
     public int[] reversePrint(ListNode head) {
        Stack<ListNode> stack = new Stack<ListNode>();
        ListNode pointer = head;
        while (pointer != null) {
            stack.push(pointer);
            pointer = pointer.next;
        }
        int length = stack.size();
        int[] res = new int[length];
        for (int i = 0; i < length; i++) {
            res[i] = stack.pop().val;
        }
        return res;
    }
}
```

画解



1 / 8

© 本 LeetBook 由「力扣」和作者共同制作和发行，版权所有侵权必究。本节内容是否对您有帮助？ 5

讨论区

共 4 个讨论

最热

笨熊 L5
来自广东2021-07-04

还有一种不需要反转链表的方法，直接直到数组长度后，在从头遍历链表，然后从数组尾巴，也就是从后往前赋值，这样就直接相当于逆转了，时间击败100%，时间复杂度O(n)，空间除了需要装填答案的数组，空间复杂度O(1)

```
public int[] reversePrint(ListNode head){
    int sum = 0;
    ListNode node = head;
    while(node != null){
        sum++;
        node = node.next;
    }
    int[] bak = new int[sum];
    int index = 0;
    while(head != null){
        bak[sum-1-index] = head.val;
        head = head.next;
        index++;
    }
    return bak;
}
```

6

笨熊 L5
来自广东2021-07-04

欸 我就是不用container/list来模拟栈，欸 就是玩 Go 递归法 时间击败100.00%,空间击败24.62% 「代码块」 Go 迭代法 时间击败62.96%，空间击败96.44% 「代码块」

递归会用到系统的栈，和直接用栈没有区别

笨熊 L5
来自广东2021-07-04

不用栈也行，直接一次对链表节点遍历，边遍历边用一个变量计数，为了之后存放答案，然后遍历时候把指针反向修改，遍历完成，声明数组，然后就可以回头遍历依次添加答案就行 轻松100%击败，可以时间复杂度O(n) 空间复杂度O(1)【除了需要保存答案的数组外，这个是没办法避免的】
原地选择next指针画画图就OK了

```
public int[] reversePrint(ListNode head){
        int length = 0,i = 0;
        ListNode temp = null;
        ListNode pNode = head;
        while(pNode != null){
            pNode = pNode.next;
            head.next = temp;
            temp = head;
            head = pNode;
            length++;
        }
        int[] ans = new int[length];
        head = temp;
        while(head != null){
            ans[i++] = head.val;
            head = head.next;
        }
        return ans;
    }
```

tour1st

L1
来自河北2021-06-07

欸 我就是不用container/list来模拟栈，欸 就是玩
Go 递归法 时间击败100.00%,空间击败24.62%

```
func reversePrint(head *ListNode) []int {
    if head == nil {
        return nil
    }
    return append(reversePrint(head.Next), head.Val)
}
```

Go 迭代法 时间击败62.96%，空间击败96.44%

```
func reversePrint(head *ListNode) []int {
    var cnt int
    for p := head; p != nil; p = p.Next{
        cnt++
    }
    ans := make([]int, cnt)
    for p := head; p != nil; p = p.Next {
        ans[cnt-1] = p.Val
        cnt--
    }
    return ans
}
```