LeetCode 第 21 号问题：合并两个有序链表

> *本文首发于公众号「图解面试算法」，是 图解 LeetCode 系列文章之一。*
>
> *同步博客：https://www.algomooc.com*

题目来源于 LeetCode 上第 21 号问题：合并两个有序链表。题目难度为 Easy，目前通过率为 45.8% 。

题目描述

将两个有序链表合并为一个新的有序链表并返回。新链表是通过拼接给定的两个链表的所有节点组成的。

示例：

```
输入：1->2->4, 1->3->4
输出：1->1->2->3->4->4
```

题目解析

一般方案

1.1 解题思想

> *（1）对空链表存在的情况进行处理，假如 pHead1 为空则返回 pHead2 ，pHead2 为空则返回 pHead1。（两个都为空此情况在pHead1为空已经被拦截） （2）在两个链表无空链表的情况下确定第一个结点，比较链表1和链表2的第一个结点的值，将值小的结点保存下来为合并后的第一个结点。并且把第一个结点为最小的链表向后移动一个元素。 （3）继续在剩下的元素中选择小的值，连接到第一个结点后面，并不断next将值小的结点连接到第一个结点后面，直到某一个链表为空。 （4）当两个链表长度不一致时，也就是比较完成后其中一个链表为空，此时需要把另外一个链表剩下的元素都连接到第一个结点的后面。*

1.2 代码实现

```
ListNode* mergeTwoOrderedLists(ListNode* pHead1, ListNode* pHead2){
    ListNode* pTail = NULL;//指向新链表的最后一个结点 pTail->next去连接
    ListNode* newHead = NULL;//指向合并后链表第一个结点
    if (NULL == pHead1){
        return pHead2;
    }else if(NULL == pHead2){
        return pHead1;
    }else{
        //确定头指针
        if ( pHead1->data < pHead2->data){
            newHead = pHead1;
            pHead1 = pHead1->next;//指向链表的第二个结点
        }else{
            newHead = pHead2;
            pHead2 = pHead2->next;
        }
        pTail = newHead;//指向第一个结点
        while ( pHead1 && pHead2) {
            if ( pHead1->data <= pHead2->data ){
                pTail->next = pHead1;
                pHead1 = pHead1->next;
            }else {
```

```
                pTail->next = pHead2;
                pHead2 = pHead2->next;
            }
            pTail = pTail->next;

        }
        if(NULL == pHead1){
            pTail->next = pHead2;
        }else if(NULL == pHead2){
            pTail->next = pHead1;
        }
        return newHead;
}
```

2 递归方案

2.1 解题思想

> *（1）对空链表存在的情况进行处理，假如 pHead1 为空则返回 pHead2 ，pHead2 为空则返回 pHead1。 （2）比较两个链表第一个结点的大小，确定头结点的位置 （3）头结点确定后，继续在剩下的结点中选出下一个结点去链接到第二步选出的结点后面，然后在继续重复（2 ） （3） 步，直到有链表为空。*

2.2 代码实现

```
ListNode* mergeTwoOrderedLists(ListNode* pHead1, ListNode* pHead2){
    ListNode* newHead = NULL;
    if (NULL == pHead1){
        return pHead2;
    }else if(NULL ==pHead2){
        return pHead1;
    }else{
        if (pHead1->data < pHead2->data){
            newHead = pHead1;
            newHead->next = mergeTwoOrderedLists(pHead1->next, pHead2);
        }else{
            newHead = pHead2;
            newHead->next = mergeTwoOrderedLists(pHead1, pHead2->next);
         }
        return newHead;
    }
}
```