```
/*
给定一个链表，两两交换其中相邻的节点，并返回交换后的链表。
你不能只是单纯的改变节点内部的值，而是需要实际的进行节点交换。
示例:
给定 1->2->3->4, 你应该返回 2->1->4->3.
来源：力扣（LeetCode）
链接：https://leetcode-cn.com/problems/swap-nodes-in-pairs
著作权归领扣网络所有。商业转载请联系官方授权，非商业转载请注明出处。
*/
```

分析:

- head->node1->node2->node3->....
- 要交换head和node1
  - 返回的节点temp需要指向node1,head.next指向node2,node1.next指向node1 ，得到:node1->head->node2->node3->...
  - 然后要交换node2和node3,得保存head(也就是要交换的node2和node3的前驱节点),让head.next = node3,node2.next = node3.next,node3.next要指向原型node2, 所以还需要一个temp指针指向原来的前一个元素(head)..
  - 两个两个交换，直到剩余1个节点或没有节点剩下.

方法一:C++_迭代法

```
/**
 * Definition for singly-linked list.
 * struct ListNode {
 *     int val;
 *     ListNode *next;
 *     ListNode(int x) : val(x), next(NULL) {}
 * };
 */

class Solution
{
public:
    ListNode* swapPairs(ListNode* head)
    {
        ListNode    bhead(0)        ;   /*方便利用bhead.next做返回值*/
        ListNode*   p1      = NULL  ;   /*两个节点的头节点*/
        ListNode*   p2      = NULL  ;   /*两个节点的尾节点*/
        ListNode*   temp    = NULL  ;   /*临时变量,保存当前要交换的两个节点的前驱节点*/

        bhead.next      =   head;
        do
        {
            p1 = bhead.next;
            if (p1 && p1->next)
            {
                p2 = p1->next;
            }
```

```
            /*链表只有一个节点或者没有节点,直接返回*/
            else
            {
                break;
            }
            temp = &bhead;
            while(1)
            {
                temp ->next =   p2          ;   /*修改"头"指针指向*/
                p1->next    =   p2->next    ;   /*交换两个节点*/
                p2->next    =   p1          ;
                temp        =   p1          ;   /*保存下次交换节点的头前节点*/
                p1          =   p1->next    ;

                if(p1 == NULL)
                {
                    break;
                }
                else
                {
                    if(p1->next == NULL)
                    {
                        break;
                    }
                    p2 = p1->next;
                }
            }
        }while (0);
        return bhead.next;
    }
};
/*
执行结果:
通过
显示详情
执行用时 :0 ms, 在所有 cpp 提交中击败了100.00%的用户
内存消耗 :8.5 MB, 在所有 cpp 提交中击败了89.28%的用户
*/
```

AlimyBreak
2019.10.17