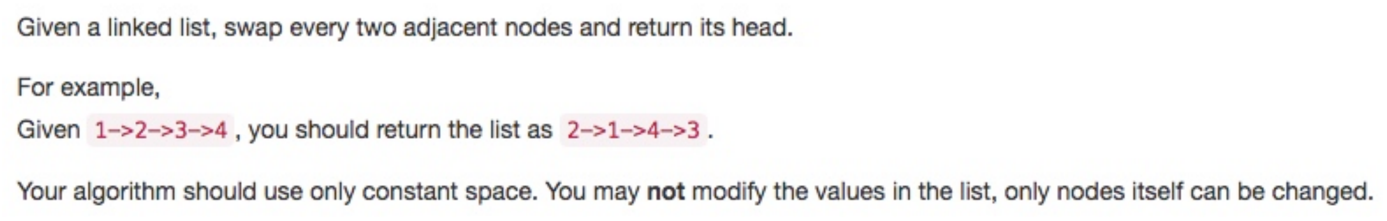
024. Swap Node in Pairs

024 Swap Nodes in Pairs

- **Linked List**

Description



Given a linked list, swap every two adjacent nodes and return its head.

For example,
Given `1->2->3->4`, you should return the list as `2->1->4->3`.

Your algorithm should use only constant space. You may **not** modify the values in the list, only nodes itself can be changed.

1. Thought line

2. Linked List

```
/**
 * 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* dummyHead = new ListNode (0);
        dummyHead->next = head;
        ListNode* ptrBeforeOdd = dummyHead;
        ListNode* ptrOdd = dummyHead;
        ListNode* ptrEven = dummyHead;
        while (ptrBeforeOdd !=nullptr){
            // Only when odd and even both exist, continue
            if (ptrOdd->next != nullptr && ptrOdd->next->next !=nullptr){
                ptrOdd = ptrOdd->next;
                ptrEven = ptrOdd->next;
                ptrOdd->next = ptrEven->next;
                ptrEven->next = ptrOdd;
                ptrBeforeOdd->next = ptrEven;
                ptrBeforeOdd = ptrEven = ptrOdd;
            }
            else break;
        }
        return dummyHead->next;
    }
};
```