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.

题目：就是要求两个相邻链子的单元相互交换。但是不能改变每个链节的值，只能移动链节来完成。

当做两个链节交互，只能从要交换的那个链节的前一个节点来操作。

找到要操作的链节之后就是先复制一个链节然后将复制的链节插入到后边，然后在删除之前复制的链节点。

例如

0🡪1🡪2🡪3 1

(1)复制1 0🡪1🡪2🡪3

(2)将复制的1插入到2的后边 0🡪1🡪2🡪1🡪3

(3)将复制前的那个1删除 0🡪2🡪1🡪3

在什么位置才执行交换工作呢？

我们设置一个计数位，0是第一次操作，下次操作为2，再次操作正好为4。所以每到偶数位的链节就进行操作。当末尾不够交换的时候不交换，保持原样即可。

代码：

public static ListNode swapPairs(ListNode head) {

int i=0;

ListNode temp;

ListNode p = new ListNode();

p=head;

while(p.next.next!=null){

if (i % 2 == 0) {

temp = new ListNode(p.next.val);

temp.next=p.next.next.next;

p.next.next.next=temp;

p.next=p.next.next;

}

i+=1;

p=p.next;

}

return head;

}

体会：就是熟悉链表的插入和删除操作的实现。