138. Copy List with Random Pointer

link

这个题原来看过, 第一遍是想用hashSet把Node存进去, 然后第二次用get, 但是不是spaceO(1)

第二次想是不是可以通过在链表里面插入节点, 然后通过modify节点, 使得新链表从老链表中分离出来.

第一次提交没有确定current.random != null

然后写了两个循环, 第一个循环已经改变了原指针, 所以

```
    public Node copyRandomList(Node head) {
        if(head == null) return null;
        //add a new node after each node.
        Node current = head;
        Node next = null;
        while(current != null){
            next = current.next;
            Node node = new Node(current.val);
            current.next = node;
            node.next = next;
            current = next;
        }
        current = head;
//这里开始报空指针, 但是不能在循环条件里面加, 得在循环体里面加.
        while(current.random != null && current.next.next != null){ //current.random
    != null;必须加在下面的if
            current.next.random = current.random.next;
            current = current.next.next;
        }
        current = head;
        Node newHead = current.next;
        while(current != null){
            current.next = current.next.next;
            current = current.next;
        }
//上一个while已经改了head.next 这里head.next.next就不会指向新创的节点了.
        current = newHead;
        while(current != null && current.next != null){
            current.next = current.next.next;
            current = current.next;
        }
//这里也忘记了把current.next重新置null
```

```
        return newHead;
    }
```

更正过后的答案.

```
    public Node copyRandomList(Node head) {
        if(head == null) return null;
        //add a new node after each node.
        Node current = head;
        Node next = null;
        while(current != null){
            next = current.next;
            Node node = new Node(current.val);
            current.next = node;
            node.next = next;
            current = next;
        }
        current = head;
        while(current != null){ //current.random != null;
            if(current.random != null)
                current.next.random = current.random.next;
            current = current.next.next;
        }
        current = head;
        Node newHead = current.next;
        Node newNode = null;
        while(current != null && current.next.next != null){
            newNode = current.next;
            next = current.next.next;
            newNode.next = next.next;
            current.next = next;
            current = next;
        }
//这里不能忘.
        current.next = null;
        return newHead;
    }
```