

问题一：拷贝以后，新链表需要确定每个random，需要自己取链表中查找，时间复杂度为O(N)，如果每个random都要去处理就是O(N\*N)，会大大的降低效率。

问题二：如果链表中有多个相同值的结点，那么会使得问题变得更加的复杂。针对这个题而言，如果有多个结点的值为7，那么原链表random指向7的结点还要确定是指向第几个值为7的结点。

所以我们想出了一种方法：

1、把拷贝结点链接在原结点的后面（目的是更好的找到random指向的结点,链接到原结点的后面就使得拷贝结点的random就是原结点random->next的指向）

2、处理拷贝结点的random

3、把拷贝结点拆解下来，链接到一起

代码：

struct Node\* copyRandomList(struct Node\* head) {

    //1.拷贝结点放在原结点的后面,建立起对应关系

    struct Node\* cur = head;

    while(cur)

    {

        struct Node\* next = cur->next;

        struct Node\* copy = (struct Node\*)malloc(sizeof(struct Node));

        copy->val = cur->val;

        cur->next = copy;

        copy->next = next;

        cur = next;

    }

    //2.处理copy结点的random

    cur = head;

    while(cur)

    {

        struct Node\* copy = cur->next;  //上面的copy是局部变量,已经销毁了

        if(cur->random == NULL)

        {

            copy->random = NULL;

        }

        else

        {

            copy->random = cur->random->next;

        }

        cur = copy->next;

    }

    //3.将拷贝结点拆解下来,再链接到一起

    cur = head;

    struct Node\* copyHead, \*copyTail;

    copyHead = copyTail = (struct Node\* )malloc(sizeof(struct Node));

    while(cur)

    {

        struct Node\* copy = cur->next;

        struct Node\* next = copy->next;

        //尾插

        copyTail->next = copy;

        copyTail = copyTail->next;

        cur->next = next;

        cur = cur->next;

    }

    struct Node\* guard = copyHead;

    copyHead = copyHead->next; //哨兵位的头结点

free(guard);

guard = NULL;

    return copyHead;

}