****

面向对象程序

课程设计报告

专业班级： 11J192

学 号： 20191000950

姓 名： 赵健欣

指导老师： 张剑波

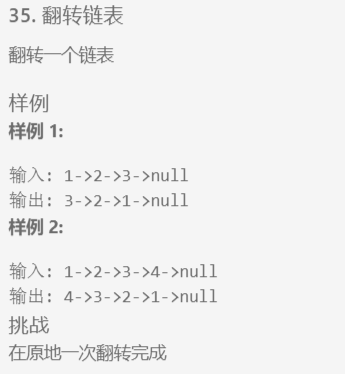
**中国地质大学地理与信息工程学院**

**2020年 8 月**

**一、链表**

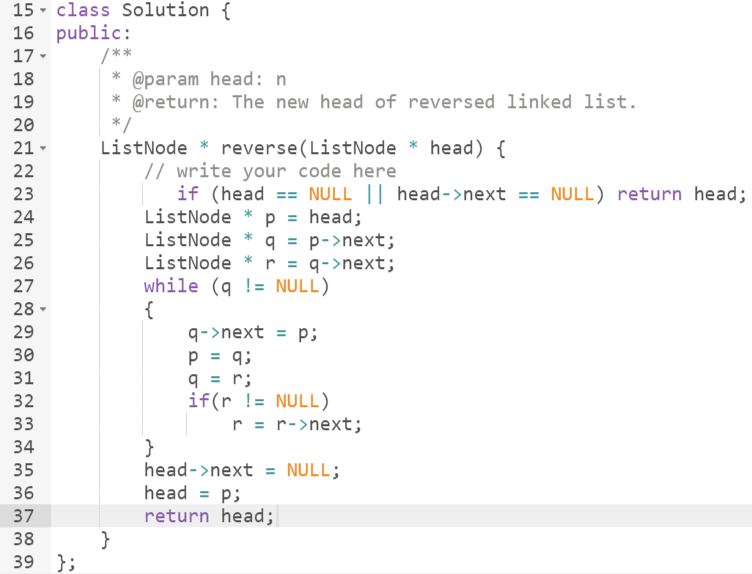
**1.**

**题目要求**

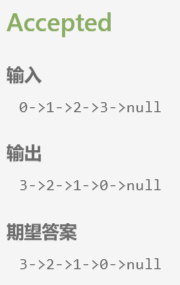


**源码和说明**

先判断链表是否为空或者只有一个元素，这两种情况是不需要将链表逆置的，接下来建立三个指针，从头依次往下，每个指针指向一个节点，再进行翻转。

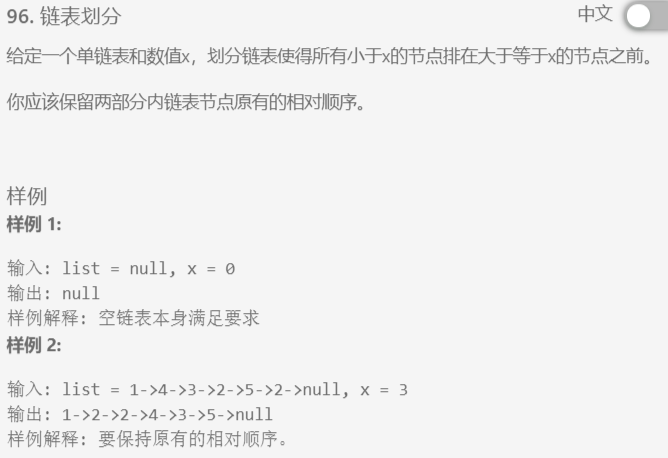


**测试结果**



**2.**

**题目要求**



**源码和说明**

先将node节点插入到小于x的链表的尾部，后再将node节点插入到大于等于x的链表的尾，接着判断返回的链表后进行拼接，最后将node节点插入到链表尾部

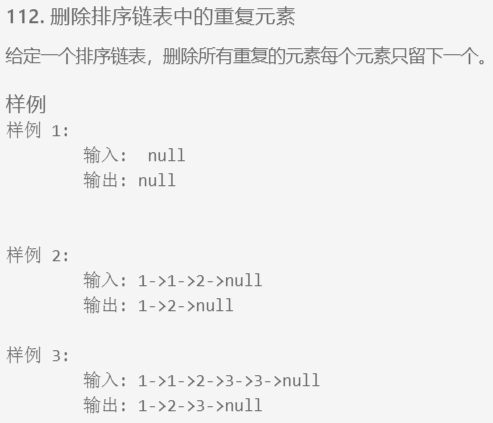


**测试结果**

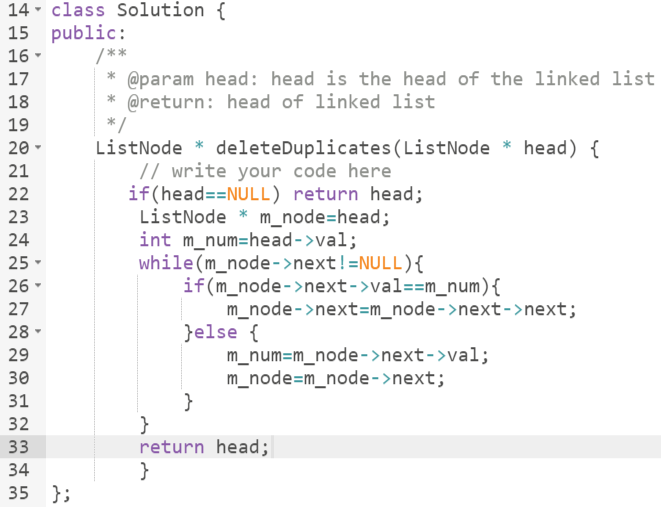


**3.**

**题目要求**



**源码和说明**

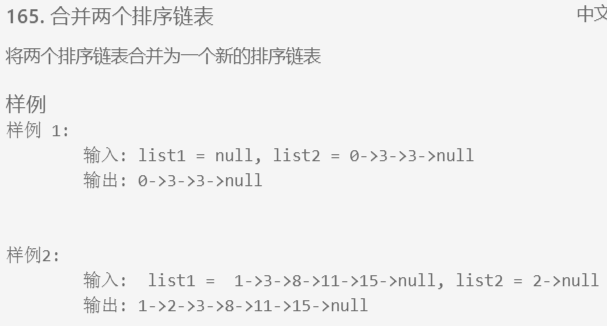


**测试结果**



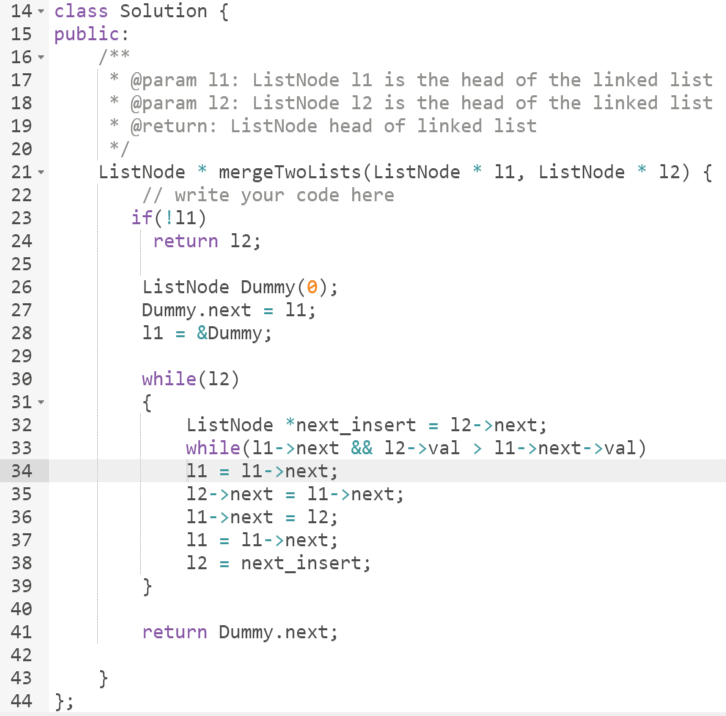
**4.**

**题目要求**



**源码和说明**

遍历l2链表，将链表每个结点逐个插入l1链表，然后保存l2下一个要插入结点的指针l1指针后移，接着找到l2当前结点插入位置插入结点l1指最后一个结点，l2置为下一个插入结点

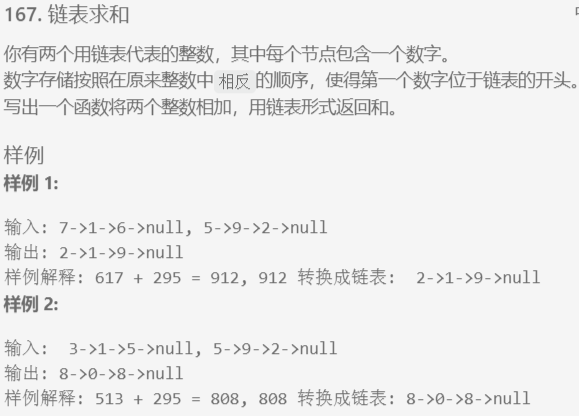


**测试结果**



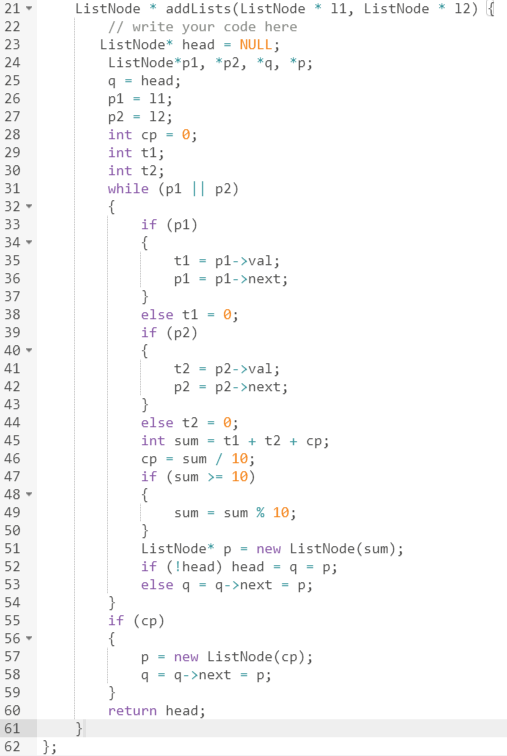
**5.**

**题目要求**



**源码和说明**

当前位相加 考虑进位

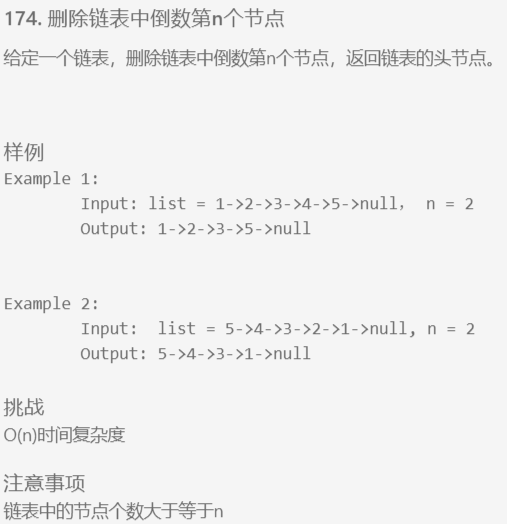


**测试结果**



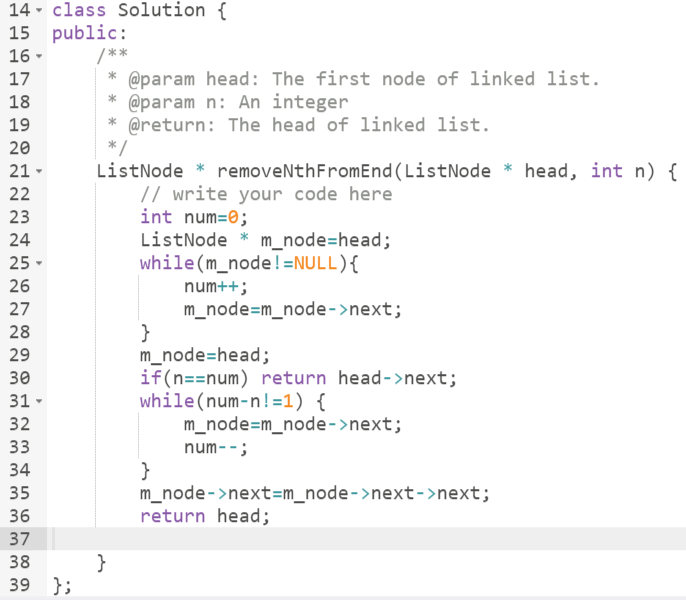
**6.**

**题目要求**

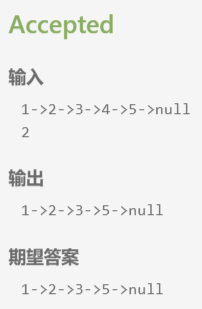


**源码和说明**

可以通过计数的方法获得答案

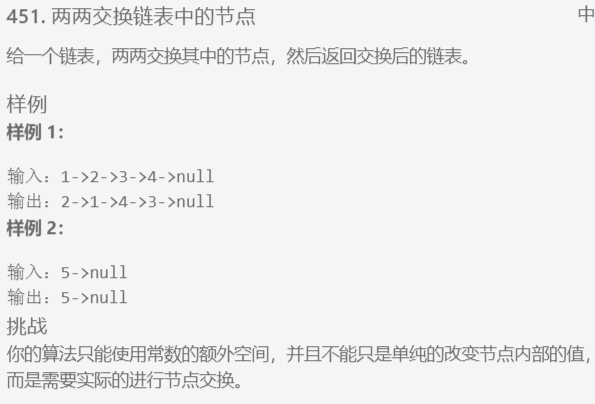


**测试结果**



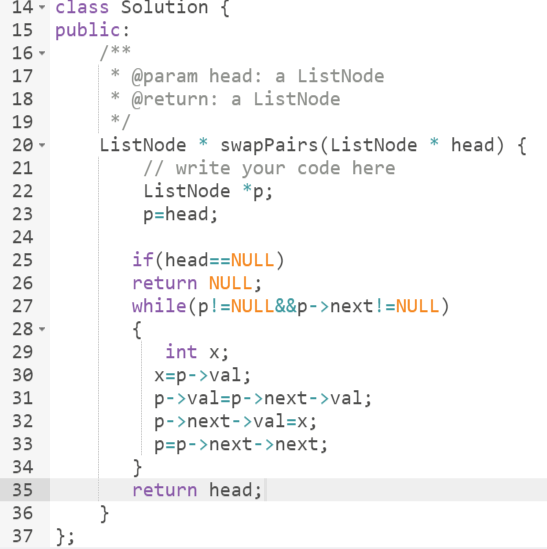
**7.**

**题目要求**

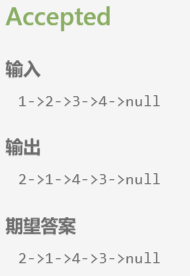


**源码和说明**

直接交换链表中节点的值，**在进行循环时要一次移动两个指针**。

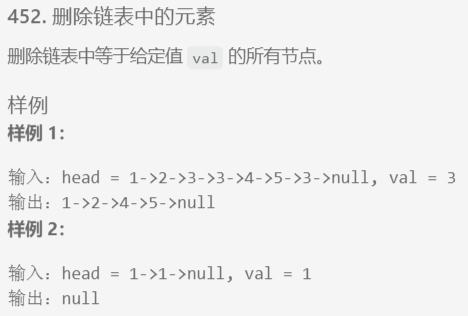


**测试结果**

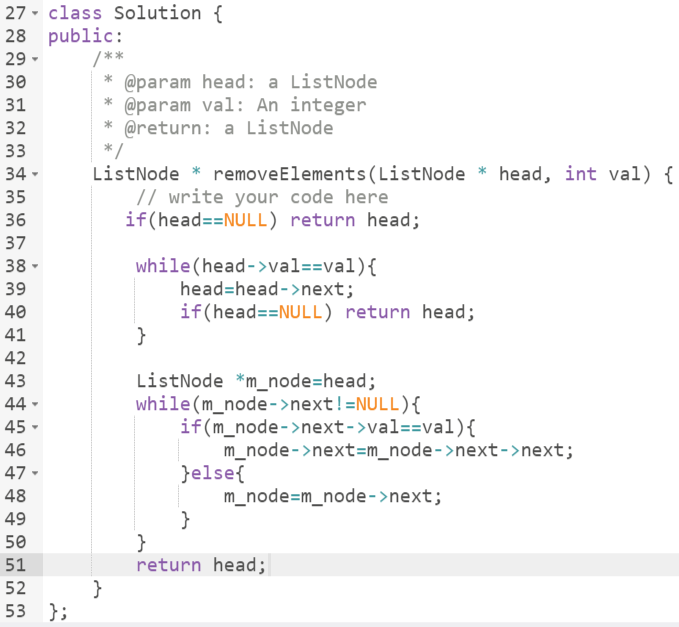


**8.**

**题目要求**



**源码和说明**

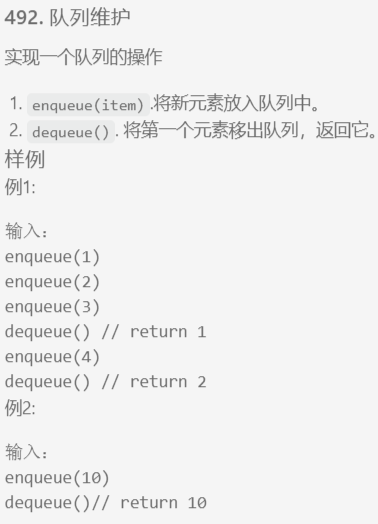


**测试结果**

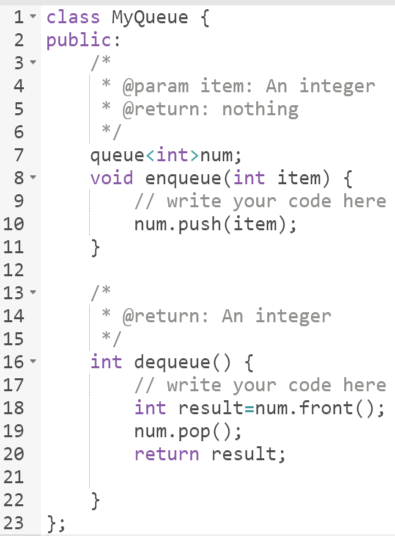


**9.**

**题目要求**



**源码和说明**

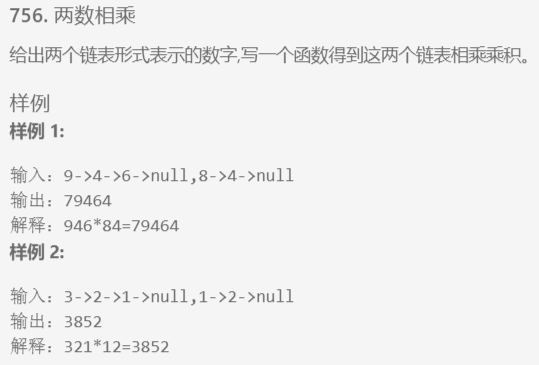


**测试结果**

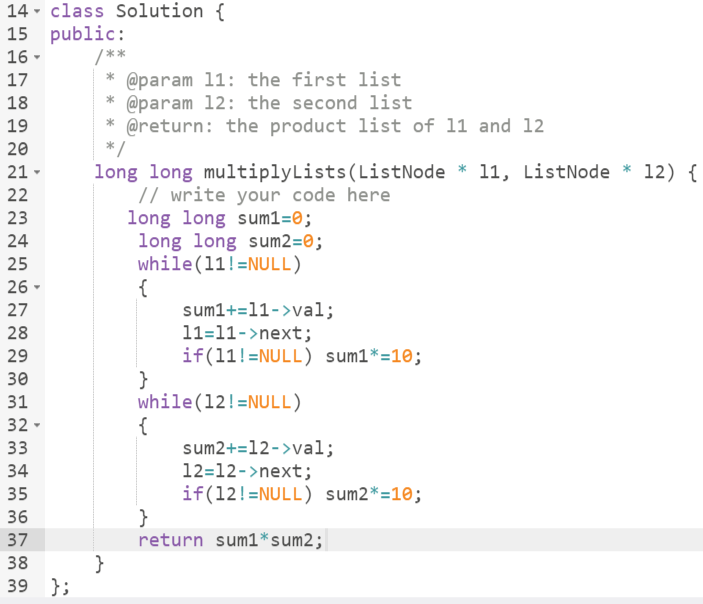


**10.**

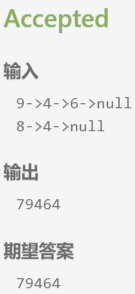
**题目要求**



**源码和说明**

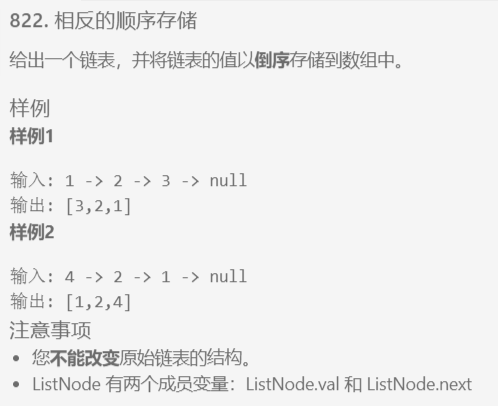


**测试结果**



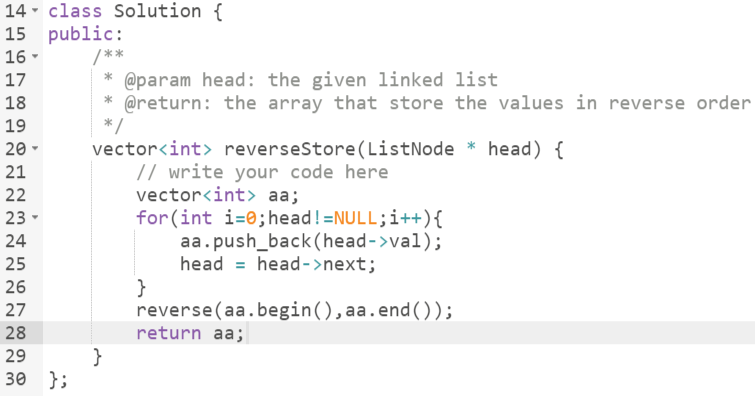
**11.**

**题目要求**

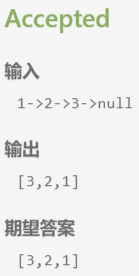


**源码和说明**

把链表数据存到vector里面，之后reverse，返回，就可以了。

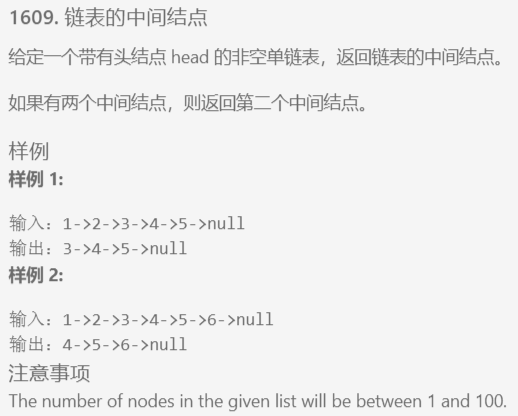


**测试结果**



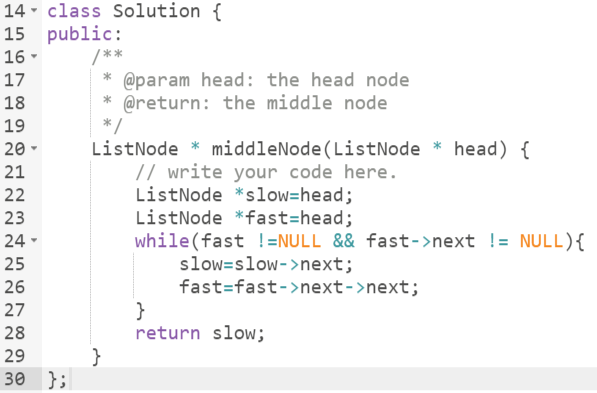
**12.**

**题目要求**

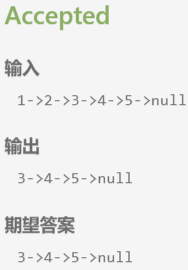


**源码和说明**

利用快慢指针解决这类问题，但是要注意快指针的中止条件



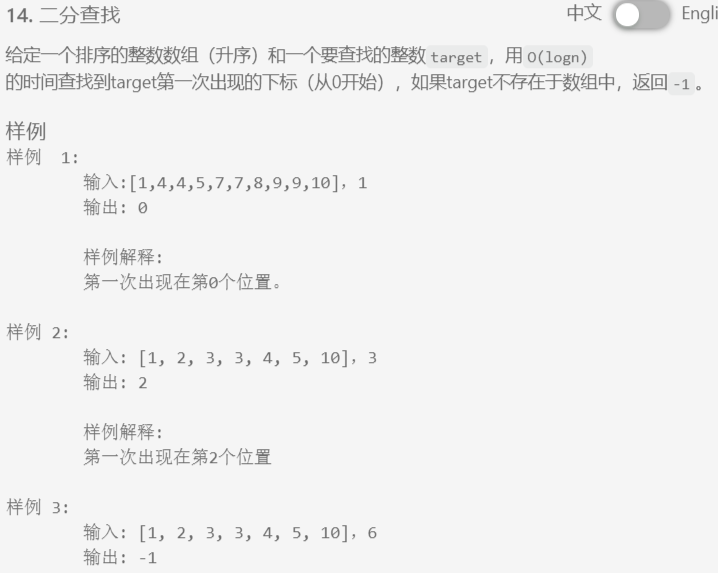
**测试结果**



**二、数组**

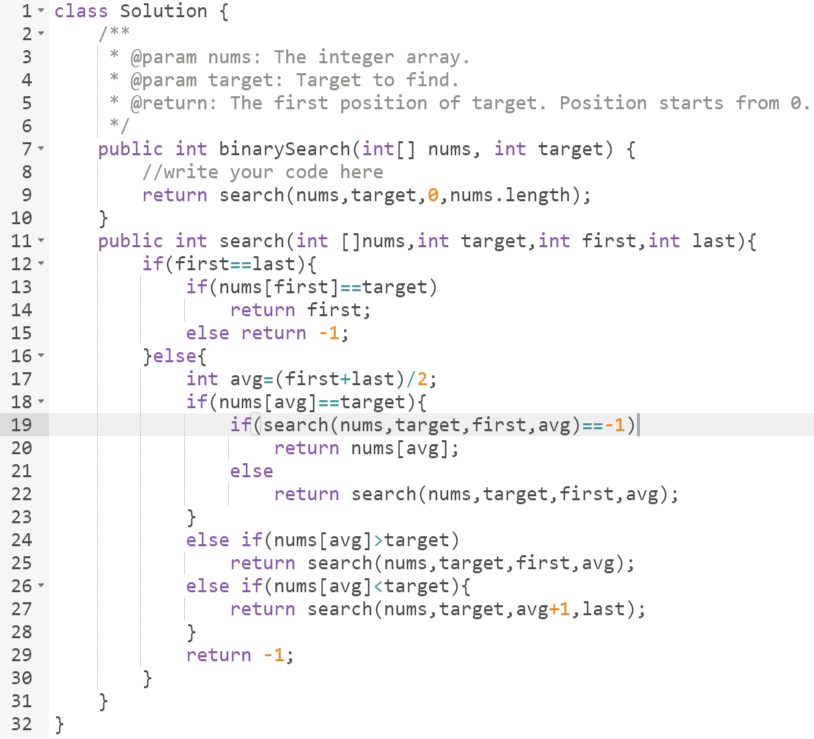
**1.**

**题目要求**

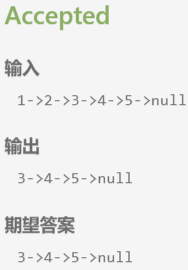


**源码和说明**

使用递归，每次都找到中间那个数进行判断，直到区间第一个数的下标等于最后一个数的下标。判断的时候，对中间的数进行判断，分为3种情况，一种是刚好等于我们要找的数，这时候不能够直接返回下标，因为我们不确定这个数的前面是否存在着同样的数，所以，要对这个数前面的数进行查找，找到就返回前面的，没找到就返回中间数的下标，当中间数小于目标数时，查找后一半，当中间数大于目标数的时候，查找前面一半。递归直到区间只有一个数。

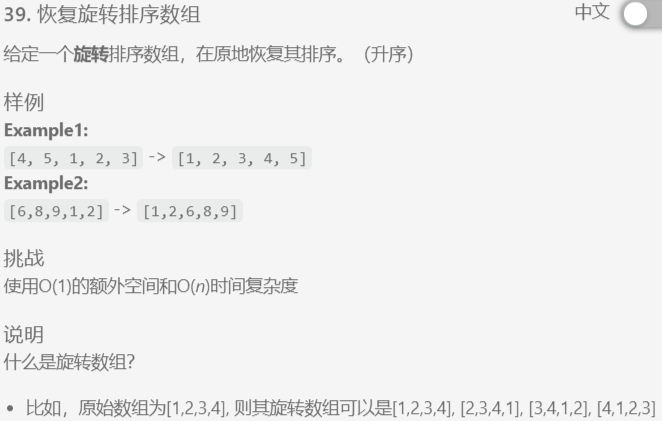


**测试结果**



**2.**

**题目要求**



**源码和说明**

一、直接多次循环

二、假设1前面有k个数,找到1的位置后, 将其前面的k个数增加到数组末尾,然后数据统一前移一次,再去除后面的k个位置

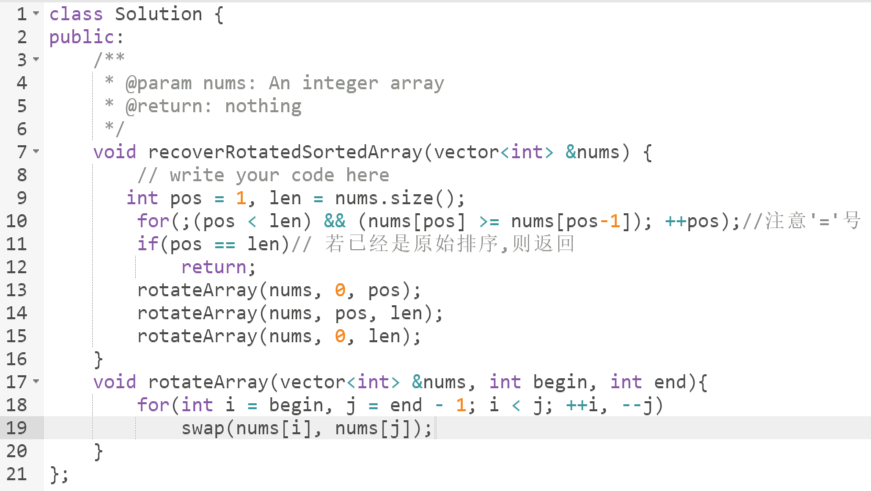
三、(也是代码中采用的方法)

三步翻转法：以{4,5,6,7,1,2,3}为例

1.  先找到1的位置.然后翻转{4,5,6,7}得到{7,6,5,4}

2. 翻转{1,2,3}得到{3,2,1}

3. 此时数组为：{7,6,5,4,3,2,1}, 将其翻转即得{1,2,3,4,5,6,7}

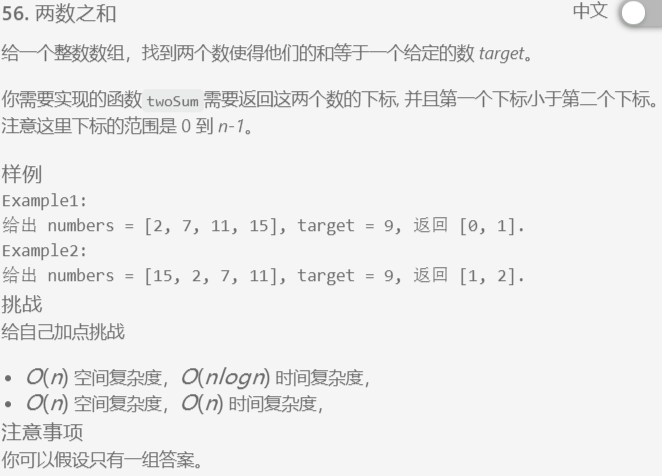


**测试结果**

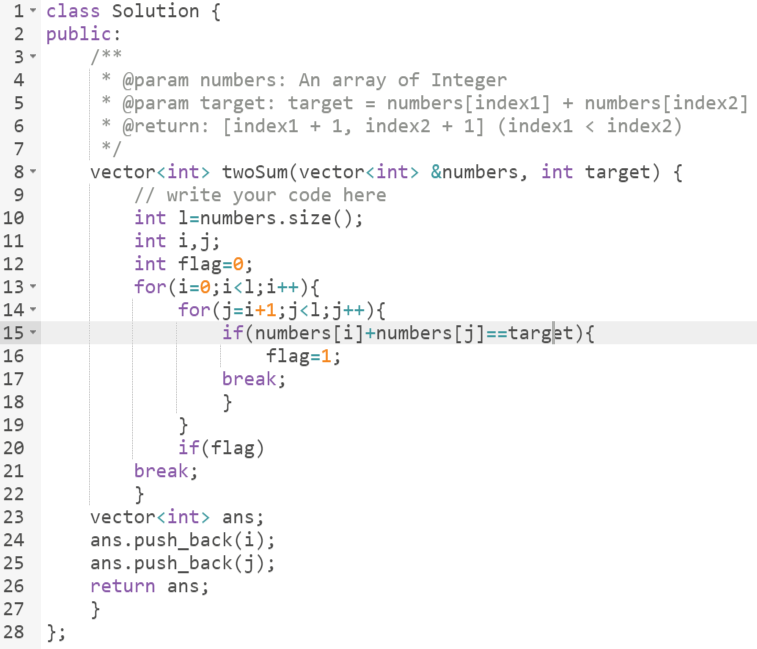


**3.**

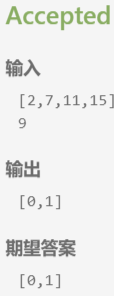
**题目要求**



**源码和说明**

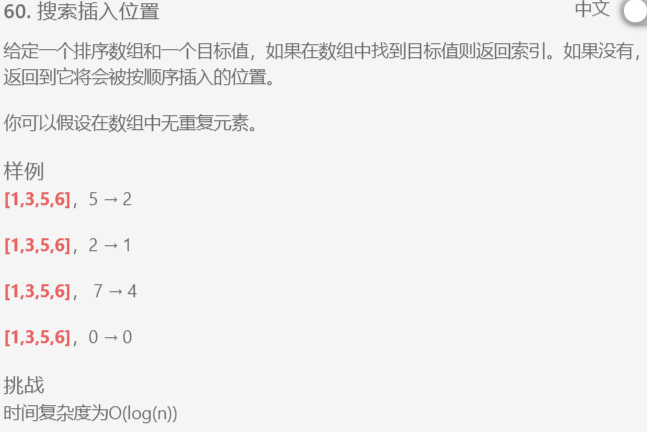


**测试结果**

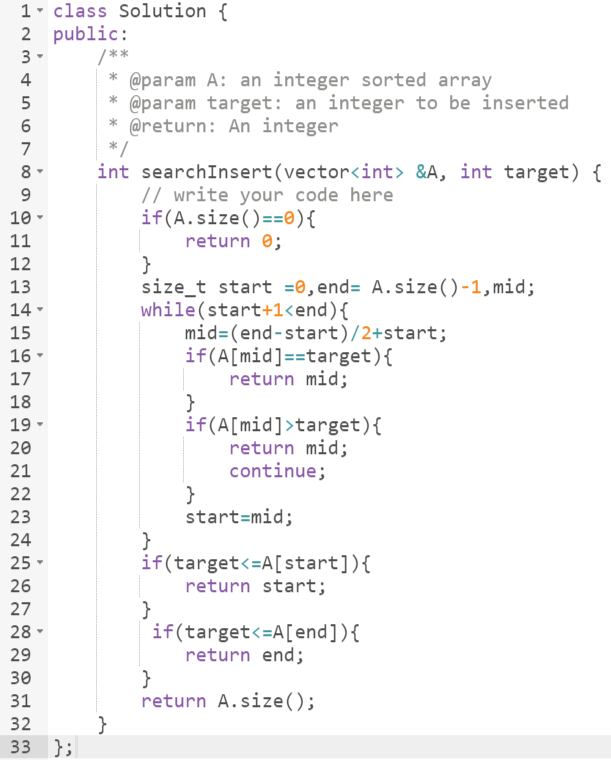


**4.**

**题目要求**



**源码和说明**

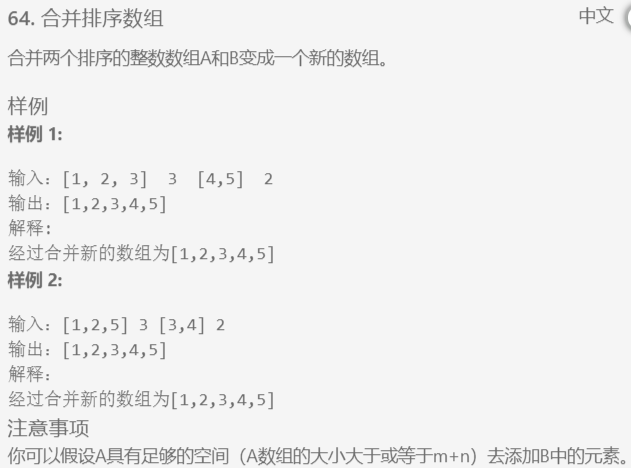


**测试结果**

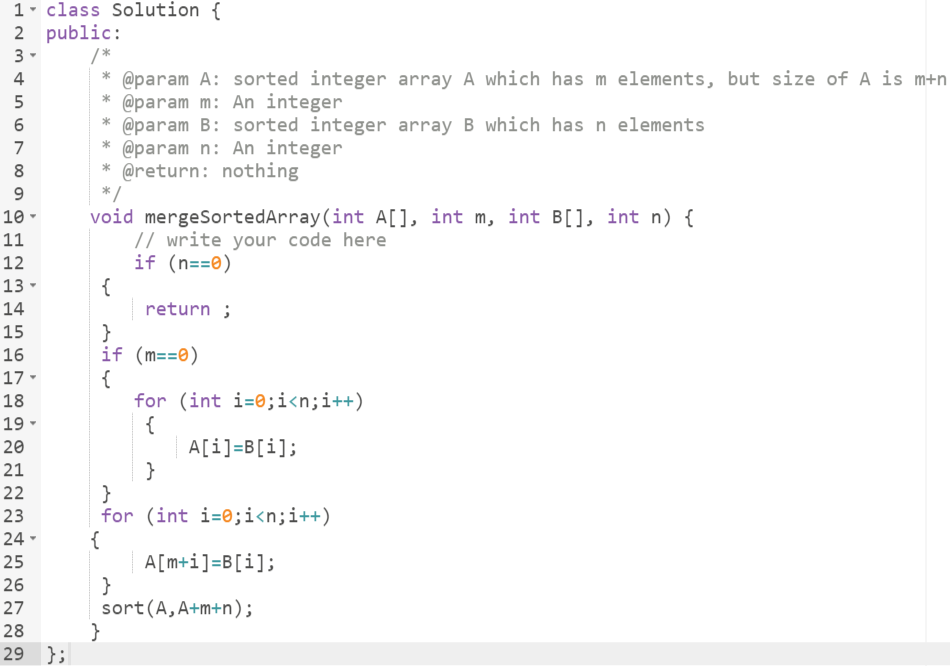


**5.**

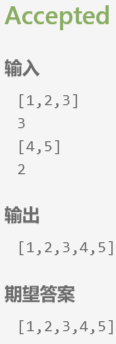
**题目要求**



**源码和说明**

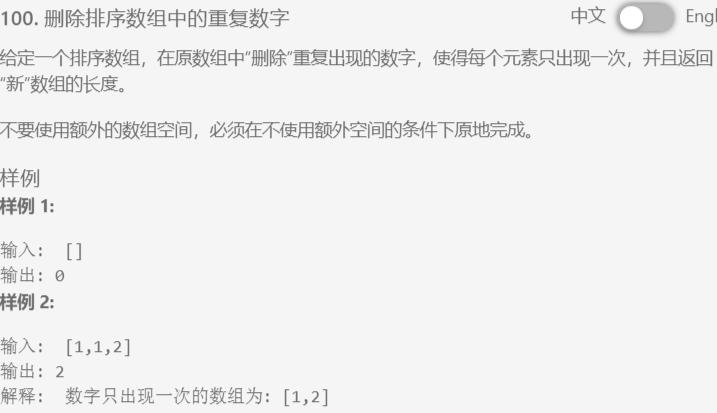


**测试结果**

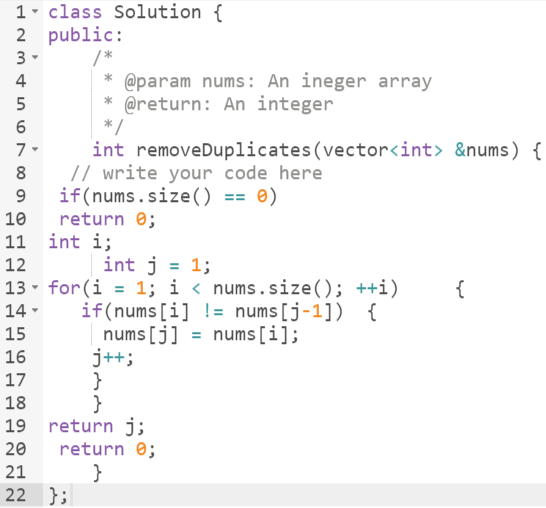


**6.**

**题目要求**



**源码和说明**

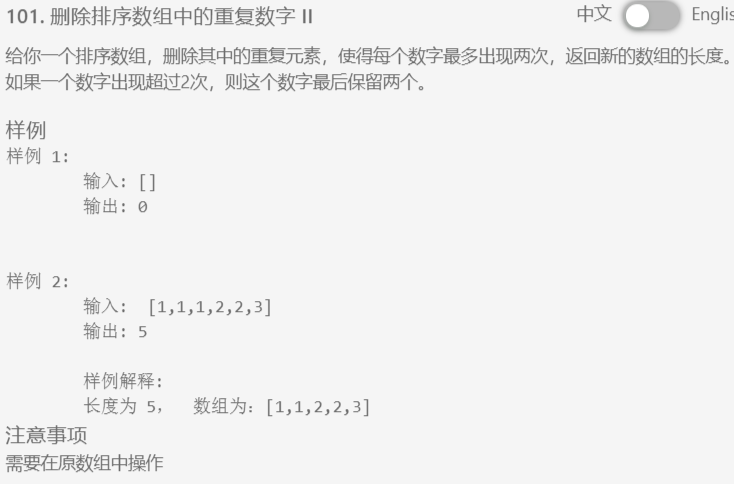


**测试结果**



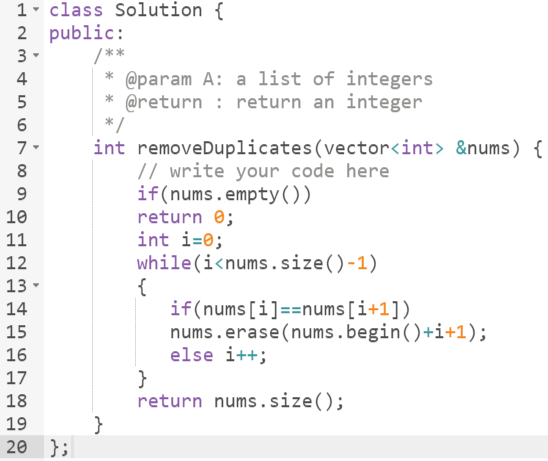
**7.**

**题目要求**



**源码和说明**

新建一个变量储存出现次数即可允许出现两次重复将如何处理。

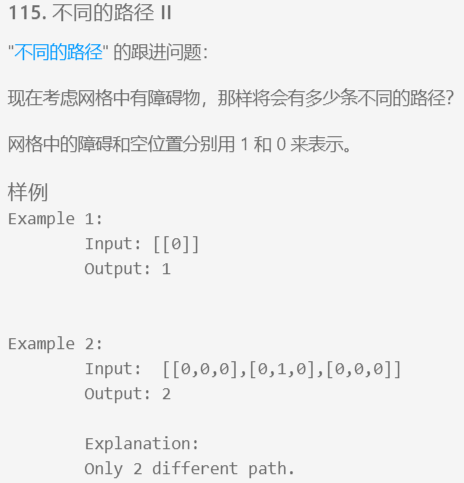


**测试结果**



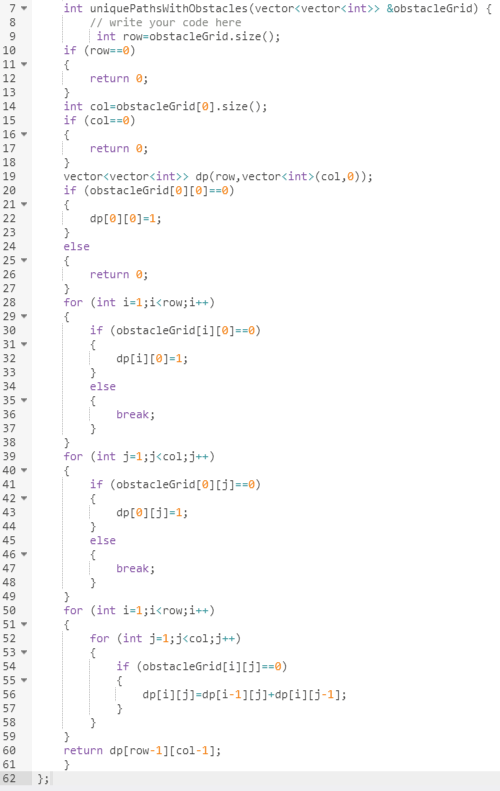
**8.**

**题目要求**



**源码和说明**

方法同不同的路劲类似，只不过要注意有障碍物的情况。有障碍物的地方dp值为0，这里尤其注意如果第一个点（0,0）就是障碍物，则整个grid无法走通，直接return 0即可。同理，初始化第一行、第一列时，如果遇到一个障碍物，后面的dp值全为0。

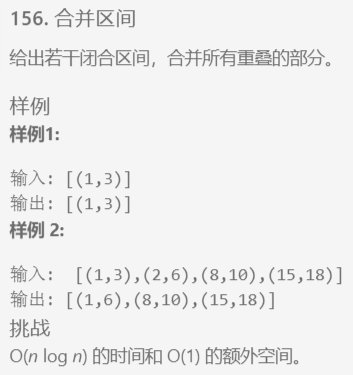


**测试结果**



**9.**

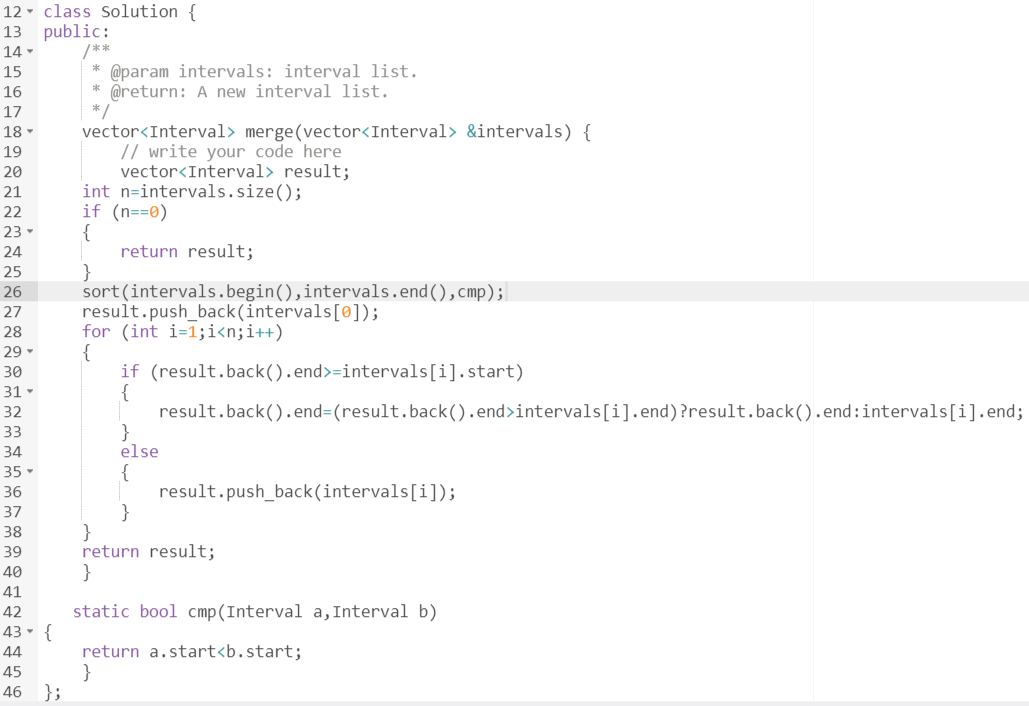
**题目要求**



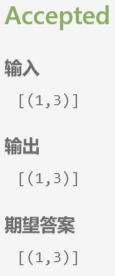
**源码和说明**

可使用自带的sort函数，排序的是区间的start参数，按照从小到大的顺序。

排序后是合并，将intervals的第一个区间（start最小区间）压入result中。然后是下标从1开始遍历intervals，如果当前区间的start小于等于结果中最后一个区间的end，两个区间有重叠，结果中区间的end取二者中较大值，否则，将当前区间压入结果数组中。

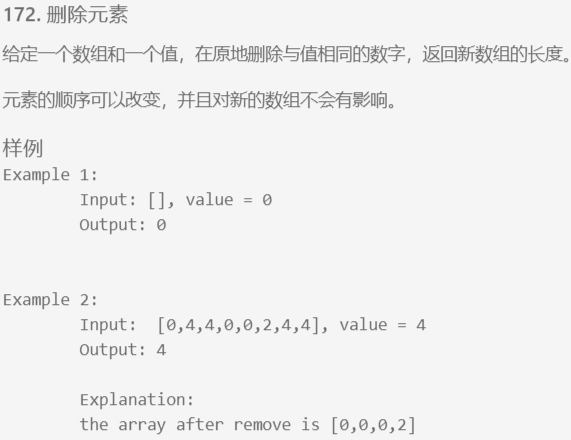


**测试结果**

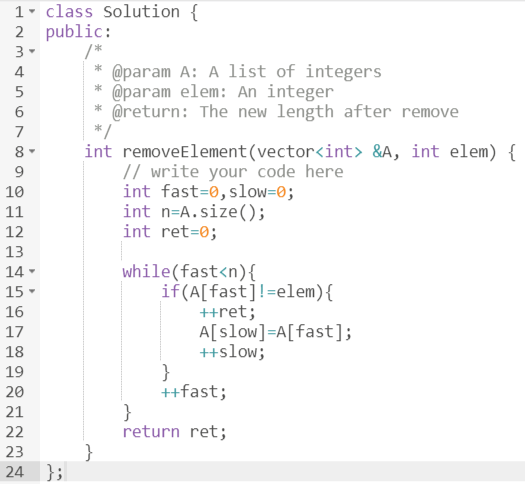


**9.**

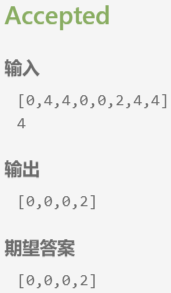
**题目要求**



**源码和说明**

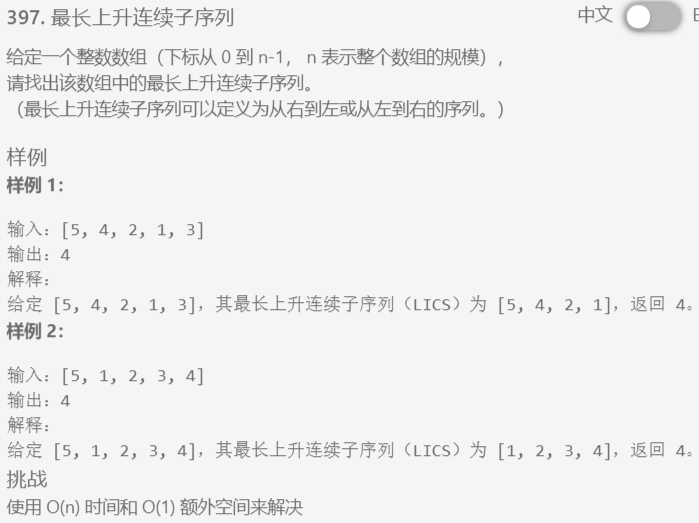


**测试结果**

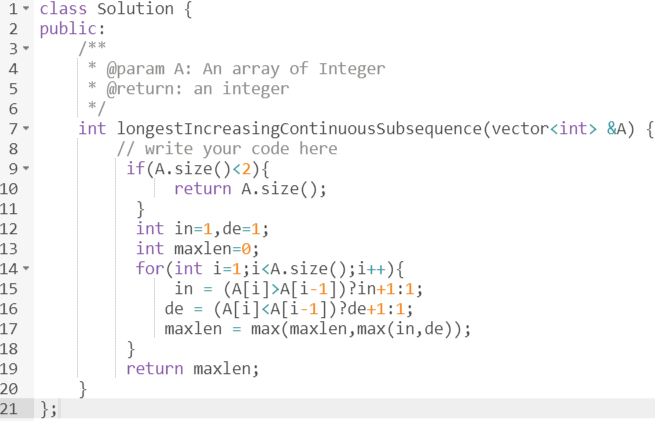


**10.**

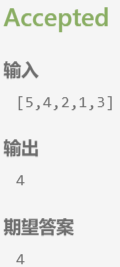
**题目要求**



**源码和说明**

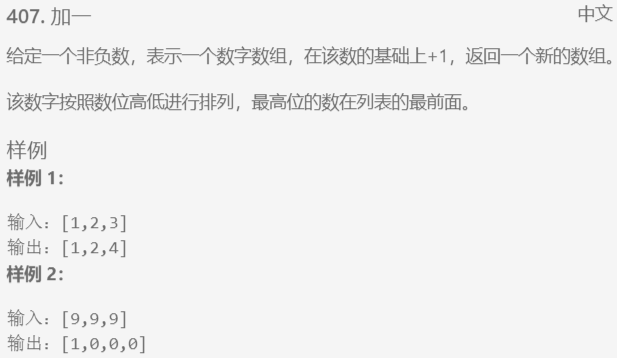


**测试结果**

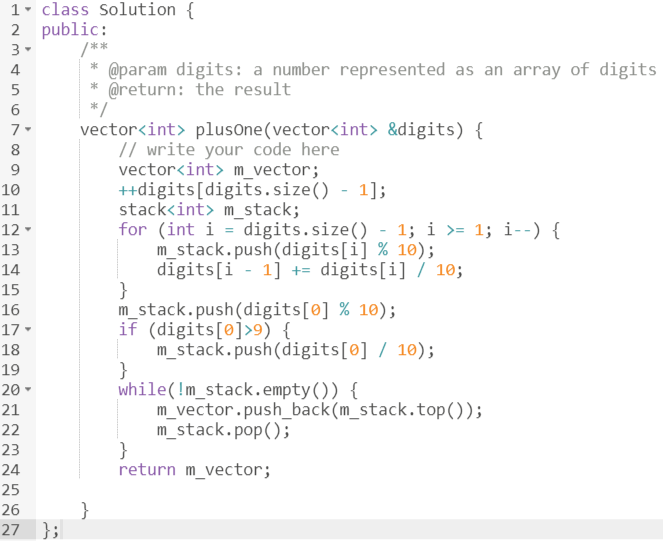


**11.**

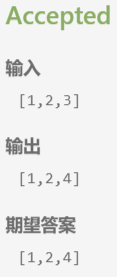
**题目要求**



**源码和说明**

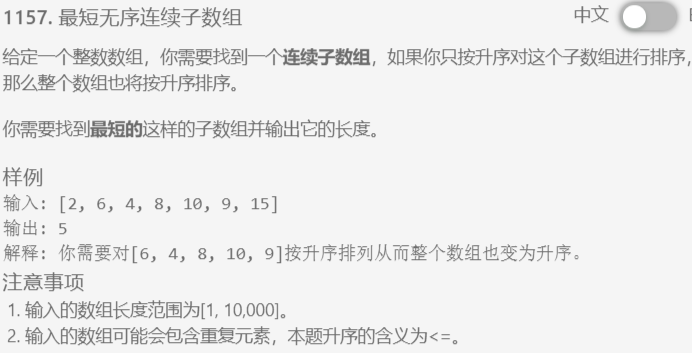


**测试结果**

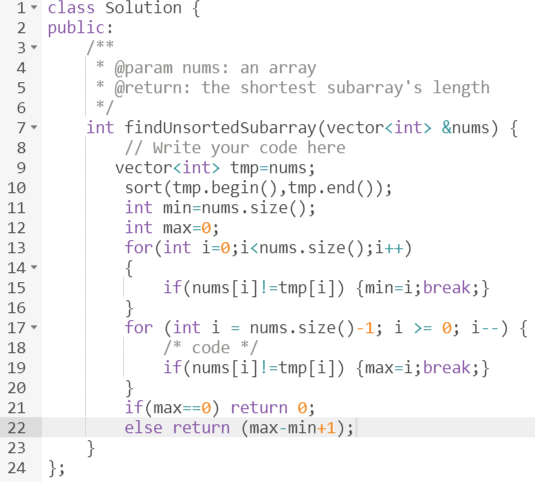


**12.**

**题目要求**



**源码和说明**



**测试结果**

