# 初级算法

这一系列卡片将帮助您掌握算法及数据结构，并提高您的编程能力。

编程能力就像任何其他技能一样，也是一个可以通过刻意练习大大提高的。

大多数经典面试题目都有多种解决方案。 为了达到最佳的练习效果，我们强烈建议您至少将此清单里的题目练习两遍，如果可以的话，三遍会更好。

在第二遍练习时，你可能会发现一些新的技巧或新的方法。 到第三遍的时候，你会发现你的代码要比第一次提交时更加简洁。 如果你达到了这样的效果，那么恭喜你，你已经掌握了正确的练习方法！

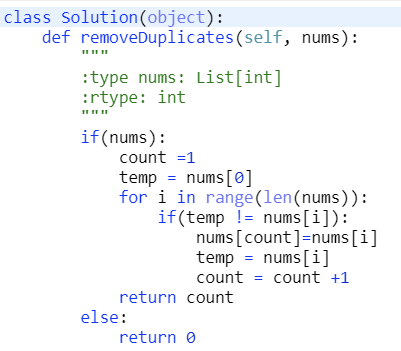
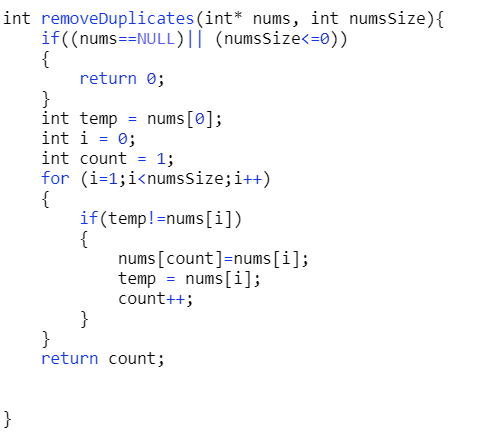
记住：刻意练习并不意味着寻找答案并记住它，这种练习方法不是长久之计。 在没有参考答案情况下，越能自主解决问题，才越能提高自身能力。

## 数组

### 从排序数组中删除重复项

给定一个排序数组，你需要在[原地](http://baike.baidu.com/item/%E5%8E%9F%E5%9C%B0%E7%AE%97%E6%B3%95" \t "_blank)删除重复出现的元素，使得每个元素只出现一次，返回移除后数组的新长度。

不要使用额外的数组空间，你必须在[原地](https://baike.baidu.com/item/%E5%8E%9F%E5%9C%B0%E7%AE%97%E6%B3%95" \t "_blank)修改输入数组并在使用 O(1) 额外空间的条件下完成

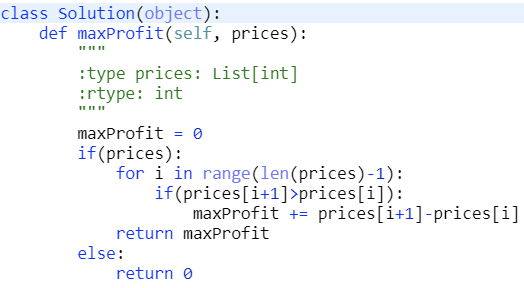
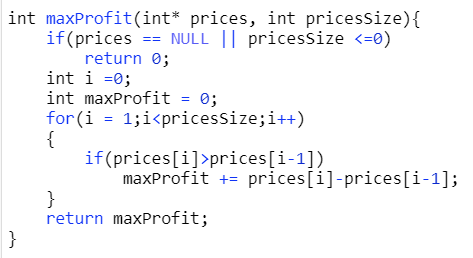


### 买卖股票的最佳时机

给定一个数组，它的第 *i* 个元素是一支给定股票第 *i* 天的价格。

设计一个算法来计算你所能获取的最大利润。你可以尽可能地完成更多的交易（多次买卖一支股票）。

**注意：**你不能同时参与多笔交易（你必须在再次购买前出售掉之前的股票）。



### 旋转数组

给定一个数组，将数组中的元素向右移动 k 个位置，其中 k 是非负数。

示例 1:

**输入:** [1,2,3,4,5,6,7] 和 *k* = 3

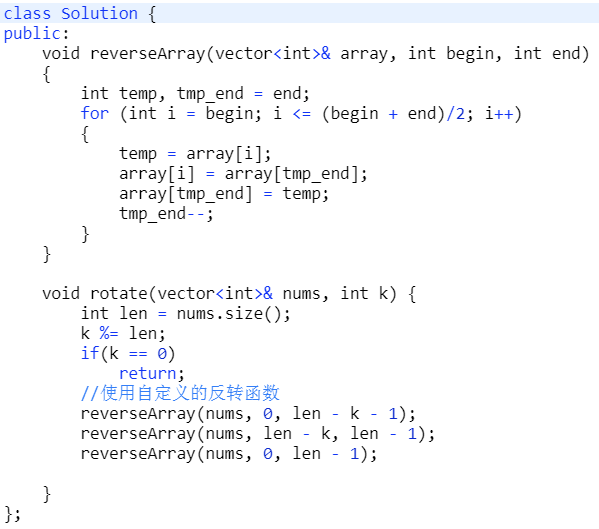
**输出:** [5,6,7,1,2,3,4]

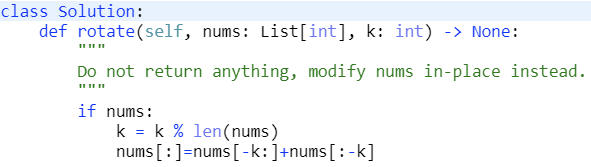
**解释:**

向右旋转 1 步: [7,1,2,3,4,5,6]

向右旋转 2 步: [6,7,1,2,3,4,5]

向右旋转 3 步: [5,6,7,1,2,3,4]



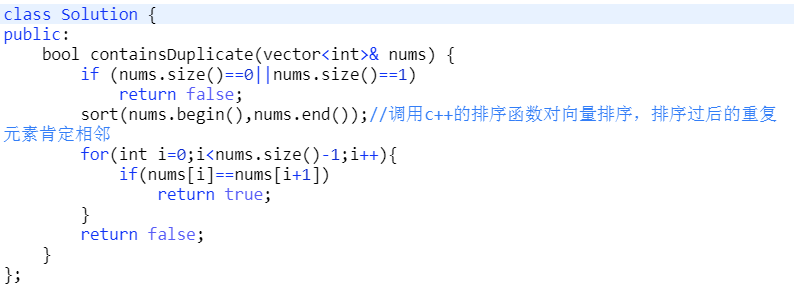


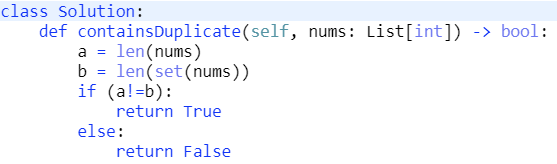
注意与回文的联系！python里对list和元祖有reserve操作！

### 存在重复

给定一个整数数组，判断是否存在重复元素。

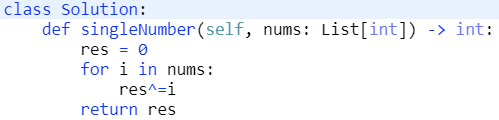
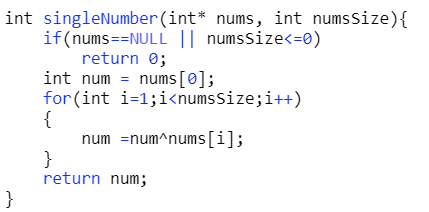
如果任何值在数组中出现至少两次，函数返回 true。如果数组中每个元素都不相同，则返回 false。





### 只出现一次的数字

给定一个**非空**整数数组，除了某个元素只出现一次以外，其余每个元素均出现两次。找出那个只出现了一次的元素。



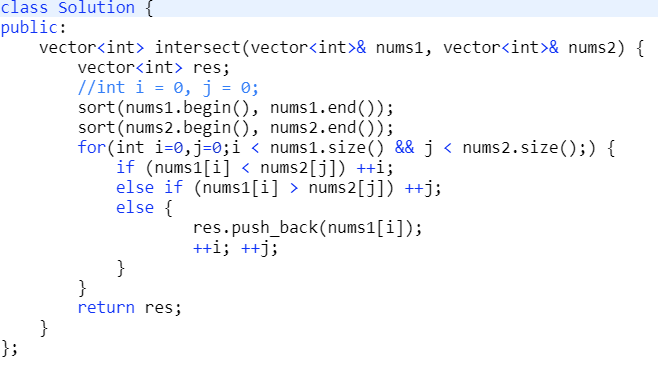
### 两个数组的交集

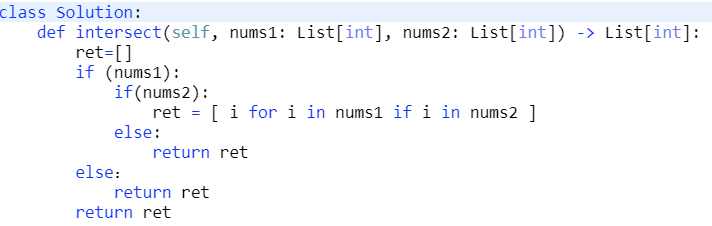
给定两个数组，编写一个函数来计算它们的交集。

**示例 1:**

**输入:** nums1 = [1,2,2,1], nums2 = [2,2]

**输出:** [2,2]



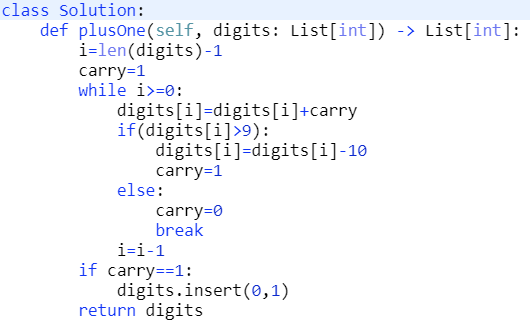
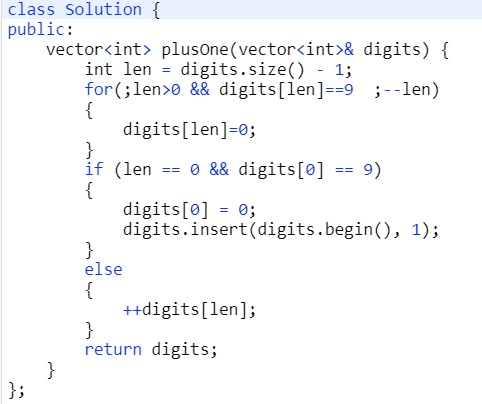


### 加一

给定一个由**整数**组成的**非空**数组所表示的非负整数，在该数的基础上加一。

最高位数字存放在数组的首位， 数组中每个元素只存储**单个**数字。

你可以假设除了整数 0 之外，这个整数不会以零开头。



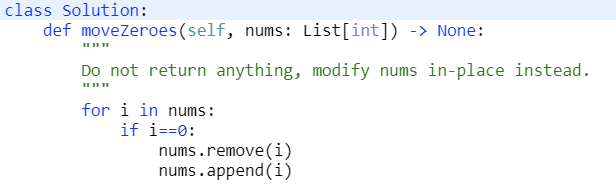
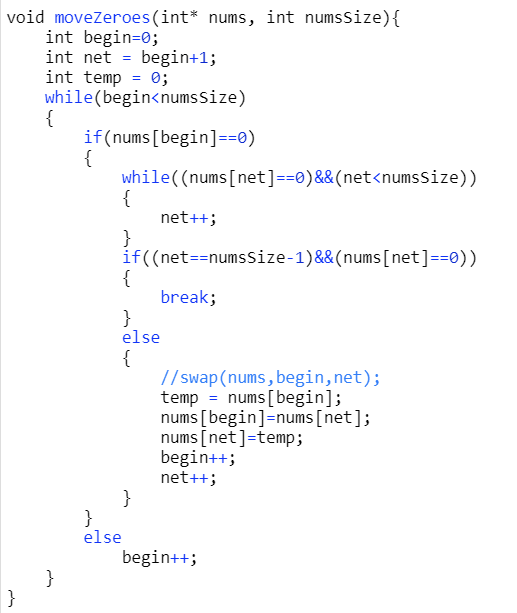
### 移动零

给定一个数组 nums，编写一个函数将所有 0 移动到数组的末尾，同时保持非零元素的相对顺序。

**示例:**

**输入:** [0,1,0,3,12]

**输出:** [1,3,12,0,0]



## 字符串

## 链表

## 树

## 排序和搜索

## 动态规划

## 设计问题

## 数学

## 其他