# 三道笔试题解读

作者：kanadeblisst
原文链接：https://ld246.com/article/1566553919169
来源网站：链滴

# 第一题

给定一组数字， 一组有9个数字，将这9个数字填写到3*3的九宫格内;使得横，竖，斜对角一条线上的个数字之和相等;如果无解则打印无解。

**思路**

思考：给定的这组数字如果有相等的数字，是否满足要求

解释：除非9个都相等，否则不满足

1、判断9个数字相加的和是否是3的倍数，不是则无解

解释：因为三行数据相加是同一个数a，那么总和就是3 * a

2、判断a1 + a9 == a2 + a8 == a3 + a7 == a4 + a6（这里的a1并未指下标，而是第几个元素）

解释：假设a数据是已经排好序的而且能够组成，那么a5一定是第二行第二列的位置（也就是最中间位置），怎么排列我们先不管，我们先看包含a5的所有组合，一共有四个分别是中间行、中间列和两对角线，这样找的原因是这些数据在四个组合中都只出现了一次。那么为什么一定是最小的a1加最大a9呢？因为a1加其他的数绝对不可能四组数和相等。

3、我们把中间这个元素a5先去掉，然后计算其他的三个数字和等于a的组合。也就是外围的四个边长满足的所有组合中出现两次的数字为四个顶点，出现一次的为边长中间的数字

4、如果以上条件成立，那么我们可以认为这九个数可以满足题目要求，然后放置位置

代码如下(文章编辑器的原因，缩进有点乱)：

```
L = list(range(1, 10)) # 假设给定的一组数字为1-9
sum_ = int(sum(L)/3) # 每行每列的和
# 复制一份数组，并去掉中间这个元素
L1 = L.copy()
L1.remove(L[4])
# 计算新数组中所有满足3个元素和等于sum_的组合
d = {x:0 for x in L} # 存储
result = [0] * 9
for i in range(8):
    for j in range(i+1, 8):
    for k in range(j+1, 8):
        n1 = L1[i]
        n2 = L1[j]
            n3 = L1[k]
        if (n1 + n2 + n3) == sum_:
         d[n1] += 1
         d[n2] += 1
         d[n3] += 1

result[0], result[2], *_ = [x for x in d if d[x]==2] # 第一行第一列和第一行第三列的值先固定。
result[4] = L[4] # 中间的值
# 然后借助上面三个数字求得其他值
result[1] = sum_ - result[0] - result[2]
result[6] = sum_ - result[4] - result[2]
result[3] = sum_ - result[0] - result[6]
result[5] = sum_ - result[3] - result[4]
result[7] = sum_ - result[1] - result[4]
result[8] = sum_ - result[2] - result[5]
```

```
# 如果后面求得的数组和给定的数组元素相等的话则满足条件
if L == sorted(result):
    for i in range(9):
        if i%3 == 2:
        print(result[i], end='\n')
        else:
         print(result[i], end=' ')
```

那么什么样的9个数字可以满足题目要求呢？等差数列。不知道有没有遗漏，如果没有，那么我们直判断是不是等差数列就行了。

# 第二题

给定形如下的矩阵：

1 1 1 1 1 1

1 1 0 0 0 1

1 0 0 0 1 0

1 1 0 1 1 1

0 1 0 1 0 0

1 1 1 1 1 1

上面矩阵的中的1代表海岸线，0代表小岛。求第二岛的面积(即被1中包围的0的个数，如果只有一个岛，输出最大岛的面积)。注意:1. 仅求这样的0，该0所在行中被两个1包围，该0所在列中被两个1包围;2. 输入矩阵中包含的小岛K >= 1;

样例输入：

1 1 1 1 1 1

1 1 0 0 0 1

1 0 0 0 1 0

1 1 0 1 1 1

0 1 0 1 0 0

1 1 1 1 1 1

样例输出: 8

**思路**

不想做这种题目，下一道

# 第三题

设计一个股票模拟交易系统。假设我们有一个很牛叉的AI系统，已经预测到未来一段时间内给定股票价格，以数组来表示，它的第i个元素是一支给定的股票在第i天的价格。假设:1. 如果你最多只允许完两次交易(一次交易是指:买入和卖出);2. 你有本金K单位(K >= 1)，一单位本金可以购买1股票;这意味你寻找的是K单位本金条件下最大利润。提示:K = 1的时候最简单，可以先考虑。设计一个算法来找最大利润。

**思路**

题目意思大概是，有一个数组包含n天股票的价格，因为中国的股票只能允许当天买入，至少第二天出，也就是所谓的T+1。我们先考虑一次交易一单位本金的情况，那么就是后面的某个值减去前面的个值达到最大即是最优解，用程序我们可以这样实现：我们从第一个元素开始迭代，用后一个元素减前一个元素得到一个值，如果这个值是负数则归零，如果是正数则判断是否比前一个的差值大，最后大值就出来了（语言表述能力不咋地，直接看代码吧）。

```
def maxPro(L):
    m = 0
    b = 0
    for i in range(len(L)-1):
        a = L[i+1] - L[i]
        b += a
        if b<0:
            b=0
        if b>m:
            m = b
    return m
```

现在考虑两次交易的情况，其实两次交易不过是将数组分成两段，分别计算一次交易得到的和。因为只有一单位本金，所以两次交易并不会重叠。

```
L = [7,1,5,3,2,4]
# 将数组L分割成两个数组，每个数组至少包含两个元素
m = 0
for i in range(2,len(L)-1):
    m1 = maxPro(L[:i])
    m2 = maxPro( L[i:])
    if m1+m2 > m:
        m = m1 + m2
```

那么当你有K单位本金呢，这毫无疑问，既然一单位本金这样操作可以达到最大收益，那么我有K单位金，也这样操作不是也是最大收益吗。

**以上思路如果有错误还请指出**