Python算法之旅（第17期）

上期回顾：

描述：“快速幂取模”问题，即快速求x^n % k

函数名：fmod(x, n, k)

参数表：x -- 底数； n -- 指数； k – 模。

返回值：返回x^n % k。

算法分析：

使用迭代法解决“快速幂取模”问题。

#迭代法求x^n % k

def fmod2(x, n, k):

b = []

while n > 0:

b.append(n % 2)

n //= 2

r = 1

for i in b[::-1]:

r = r \* r % k

if i == 1:

r = (x % k) \* r % k

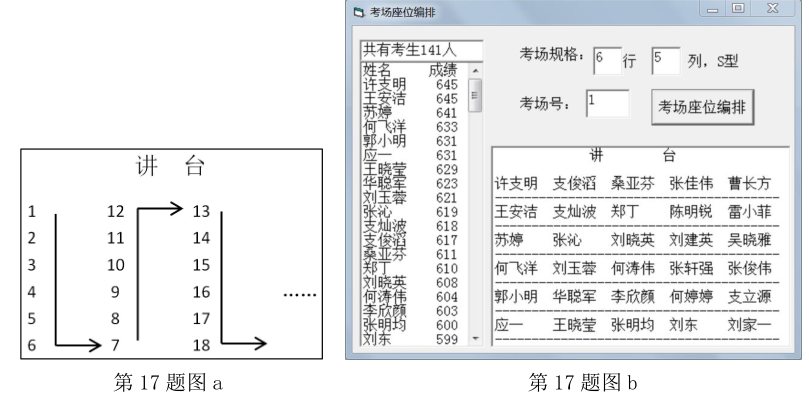
return r

题目：考场座位编排（湖丽衢2018年9月三地市高三教学质量检测试卷17题）

难度：4星 有趣：3星 有用：4星

分类：模拟， 二维数组， 蛇形矩阵

描述：小王用VB编写了一个考场座位编排程序，实现从考生数据库里读出所有学生的姓名和前一次考试成绩，分别存入数组xm和数组cj,按成绩从高到低排序后显示在Listl中，输入考场规格和考场号，单击“考场座位编排”按钮后，从第一位考生开始按S型(如第17题图a所示)依次安排座位，并在List2中输出该考场的座位表，每个姓名输出占4个汉字字符宽度，不足4个字符的在姓名后面补充空格(如第17题图b所示)，尾试场多余位置显示“空”字。



程序的部分代码如下，请回答下列问题:

(1)当考生总人数为141人，考场规格为6行5列时，第3考场的第6位考生是Listl中第▲ 位考生。

(2)从下面代码可以看出，姓名和成绩数据取自数据库中名为 ▲ 的数据表。

(3)实现上述功能的VB程序如下，请补充完善代码。

Dim conn As New ADODB.Connection

Dim rs As New ADODB.Recordset

Dim n As Integer

Dim xm(1 To 1000) As String

Dim cj(1 To 1000) As Integer

Function fm(s As String, k As Integer) As String

x = Len(s)

fm = s + Mid(" ", 1, (k - x) \* 2) '此处空格字符串长度足够

End Function

Private Sub Form\_Load()

Dim i As Integer, j As Integer, t1 As Integer, t2 As String

conn.ConnectionString = "Provider-Microsoft.ACE.OLEDB.12.0;DATA Source=" & App.Path & "\cjk.accdb"

conn.Open

Set rs.ActiveConnection = conn

rs.Open "select \* from 总分表"

n = 0

Do While Not rs.EOF

n = n + 1

xm(n) = rs.Fields(1): cj(n) = rs.Fields(2)

rs.MoveNext

Loop

rs.Close

conn.Close

Text4.Text = "共有考生" & n & "人"

'此处代码实现将xm和cj数组按成绩降序排序并从Listl输出，代码略。

End Sub

Private Sub Command1\_Click()

Dim kc As Integer 'kc为考场号

Dim p As Integer, q As Integer 'p为行数，q为列数

Dim i As Integer, num As Integer, ks As Integer

List2.Clear

List2.AddItem " 讲 台 "

List2.AddItem " "

p = Val(Text1.Text): q = Val(Text2.Text): kc = Val(Text3.Text)

ks = p \* q \* (kc - 1)

For i = 1 To p

ks = ks + 1

s = fm(xm(ks), 8) '输出的信息占4个汉字字符宽度

For j = 1 To q - 1

If j Mod 2 = 1 Then

num = ks + p \* 2 \* (j + 1) \ 2 - 1 - (i - 1) \* 2

Else

num = ①

End If

If num > n Then

s = s + fm("空", 4)

Else

s = s + ②

End If

Next j

List2.AddItem s

List2.AddItem "------------------------"

Next i

End Sub

答案：

17.（1）66 1分

（2）总分表 1分

（3） ①ks + p \* j 或num+1+(i-1)\*2

或 num+2\*i-1 2分

②fm(xm(num), 4) 2分

算法分析：

本题的算法思想是模拟，有点类似“输出蛇形矩阵”，但是多了一个输出空位的变化。算法难度本身不大，只要搞清楚“蛇形矩阵”的规律就能填出各个位置的序号与行列坐标的关系。但是题目提供的代码设置了一些陷阱，诱导学生入坑。

陷阱1：题目代码中的行号i取值范围是[1,p]，但列号j取值范围是[0,q-1]，即当j=1时表示第2列。

陷阱2：题目代码通过ks = ks + 1和s = fm(xm(ks), 8) 这两条语句通过递增ks和初始化s，来实现换行并输出每行第1列的功能，所以后面输出第2到q列时，变量j是从1开始的。

陷阱3：题目代码num = ks + p \* 2 \* (j + 1) \ 2 - 1 - (i - 1) \* 2故弄玄虚，先乘以2再整除2，增加理解难度，其实写成num = ks + p \* (j + 1) - i \* 2 + 1就好。

那么这条语句是怎么得来的呢？仔细阅读代码，发现这是在生成偶数列（j为奇数）的座位号，我们假设ks0 = p \* q \* (kc - 1)，则ks = ks0 + i，又num = ks0 + p \* (j + 1) - i + 1，故num = ks + p \* (j + 1) - i \* 2 + 1。

如果搞清楚了题目的算法思想，我们可以写出更简明的代码：

Private Sub Command3\_Click()

Dim kc As Integer 'kc为考场号

Dim p As Integer, q As Integer 'p为行数，q为列数

Dim i As Integer, num As Integer, ks As Integer

List2.Clear

List2.AddItem " 讲 台 "

List2.AddItem " "

p = Val(Text1.Text): q = Val(Text2.Text): kc = Val(Text3.Text)

ks = p \* q \* (kc - 1)

For i = 1 To p

For j = 1 To q

If j Mod 2 = 0 Then

num = ks + p \* j - (i - 1)

Else

num = ks + p \* (j - 1) + i

End If

If j = 1 Then s = "" '换行

If num > n Then

s = s + fm("空", 4)

Else

s = s + fm(Str(num), 4)

End If

Next j

List2.AddItem s

List2.AddItem "------------------------"

Next i

End Sub

使用Python语言可以实现相同功能，描述如下：

描述：根据考生总数n，输出p行q列的考场kc的座位表，从第一位考生开始按S型依次安排座位，

每个座位号占4个字符宽度，不足4个字符的在数字前面补充空格，尾试场多余位置显示数字0。

函数名：seat\_numbe(n, p, q, kc)

参数表：n--考生总数;

p，q--考场排成p行q列;

kc--考场序号（从1开始数起）。

返回值:无返回值，但是会按要求输出蛇形矩阵。

示例1：n, p, q, kc = 82, 6, 5, 1时，输出：

1 12 13 24 25

2 11 14 23 26

3 10 15 22 27

4 9 16 21 28

5 8 17 20 29

6 7 18 19 30

示例1：n, p, q, kc = 82, 6, 5, 3时，输出：

61 72 73 0 0

62 71 74 0 0

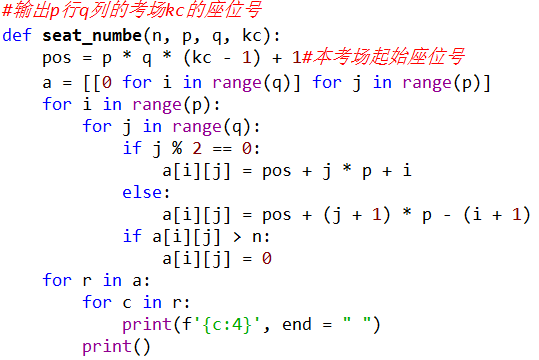
63 70 75 82 0

64 69 76 81 0

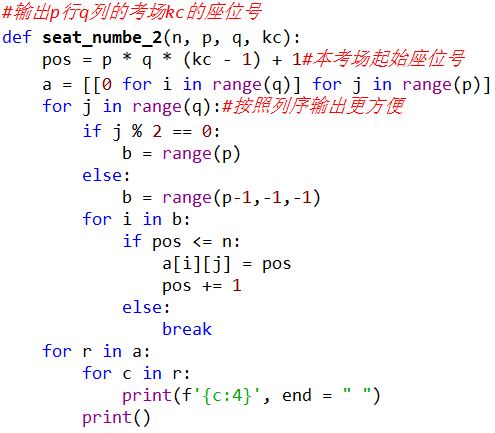
65 68 77 80 0

66 67 78 79 0

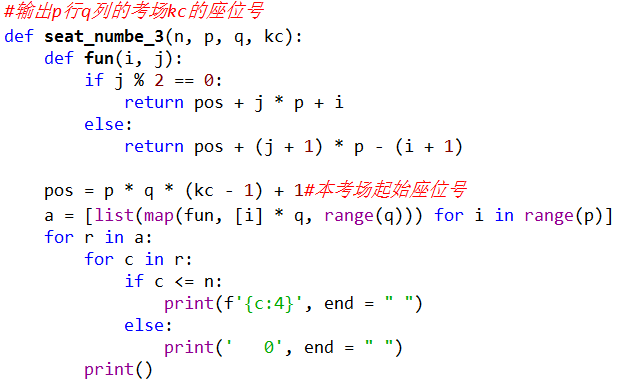
首先采用与上述VB程序相同的算法，代码如下：



我们注意到，由于考场是按照列序来输出蛇形矩阵的，所以采用列序优先的算法更好理解，代码如下：



上述代码虽然很好的表达了蛇形矩阵算法的基本思想，但是还没有体现Python语言简洁明了的特征，我们可以充分利用Python语言的map函数和列表生成式，写出更简洁的代码：



课后思考：

前面我们说考场座位编排算法的基本思想是“输出蛇形矩阵”，但是由于涉及到考场起始座位号和尾试场多余位置的问题，实际的代码比纯“输出蛇形矩阵”问题要复杂一些。为了让大家更好的掌握“蛇形矩阵”问题，我们的课后思考题就是输出如下的蛇形矩阵：

描述：输出p行q列的蛇形矩阵。

函数名：serpentine\_matrix(p, q)

参数表：p--总行数;

q--总列数

返回值:无返回值，按要求输出蛇形矩阵。

示例1：p, q = 6, 5时，输出：

1 12 13 24 25

2 11 14 23 26

3 10 15 22 27

4 9 16 21 28

5 8 17 20 29

6 7 18 19 30

示例1：p, q = 5, 6时，输出：

1 2 3 4 5 6

12 11 10 9 8 7

13 14 15 16 17 18

24 23 22 21 20 19

25 26 27 28 29 30

相信聪明的你通过学习上述算法，应该已经掌握“蛇形矩阵”问题的本质了吧，那就赶快试试吧！

另外，如果你有更 Pythonic（优雅的、地道的、整洁的）代码，或者与本文不同的算法思路和代码实现，请你一定留言或联系我，让我们一起讨论，共同进步。