Python算法之旅（第23期）

上期回顾：

描述：原列表a = [('aa','bb'),('cc','ee'),('bb','cc'),('aa','cc'),('bb','dd')]，现在要求对a进行排序，主要关键字是a[i][0]，次要关键字是a[i][1]，其中对主要关键字按升序排序，次要关键字按降序排序。

例如对于上述的列表a，排序后为a = [('aa', 'cc'), ('aa', 'bb'), ('bb', 'dd'), ('bb', 'cc'), ('cc', 'ee')]

算法分析：

因为列表中的元素是字符串，不能在元素前添加负号来实现逆序排序功能，需要用到一些第三方模块来实现复杂的排序，例如可以使用functools模块中的cmp()函数来设置比较的条件，代码如下：

import functools

def cmp(a, b):

if a[0] < b[0]: #主要关键字按升序排序

return -1

elif a[0] > b[0]:

return 1

elif a[1] > b[1]:#次要关键字按降序排序

return -1

elif a[1] < b[1]:

return 1

else:

return 0

a = [('aa','bb'),('cc','ee'),('bb','cc'),('aa','cc'),('bb','dd')]

b = sorted(a, key = functools.cmp\_to\_key(cmp))

print(b)

题目：装球问题

难度：2星 有趣：3星 有用：2星

分类：穷举，整数拆分

描述：设有n个盒子(n足够大，可装入任何数量的球)，分别编号1,2，.....同时有k个小球(k>0)。今将k个小球装入到盒子中去。装入规则如下：

  (1)第一个盒子不能为空。

(2)装入必须严格按递增(盒子中球的数量不能相同)顺序进行。

  例如,当k=8，n=6时，装入方法有1，2，5或1，3，4。

(3)在满足上面的两个条件下要求有球的盒子尽可能多。

(4)装完后,相邻盒子中球个数差的绝对值之和最小(未装的盒子不计)。

如上例中：装入方法1，2，5，则差的绝对值之和为(2-1)+(5-2)=4。装入方法1，3，4，则差的绝对值之和为(3-1)+(4-3)=3。因此装入方法最终选1，3，4。

 给出k(k表示小球的个数)之后，求出满足上述四个条件的装入方法。

(1)依据上述描述设计如下VB程序,请在划线处填人合适的代码：

Const n = 20

Private Sub Command1\_Click()

Dim a(1 To n) As Integer 'a(i)表示第 i个盒子中装人球的数量

Dim i As Integer, j As Integer, k As Integer 'k是球的数量

k = Val(Text1.Text)

j = 1

Do While ①

a(j) = j

k = k - j

j = j + 1

Loop

②

Do While k > 0

a(i) = a(i) + 1

③

i = i – 1

Loop

List1.Clear

For i = 1 To j - 1

List1.AddItem a(i)

Next i

End Sub

 (2)如果有12个小球,6个盒子.则满足上述四个条件的装人方法是

答案：

（１） ①j <= k ②i = j - 1 ③k = k - 1

（２）1,2,4,5

算法分析：

本题的算法思想是先依次给ａ(1)，ａ(2)，ａ(3)， ……ａ(j)装入1，2，3，……ｊ个球， 直到剩下小球的数量k小于j+1为止。 剩下的k个球则依次给ａ(j)，ａ(j-1)，ａ(j-2)……， 即逆序给每个盒子再装入1个，直到k=0 为止。 这样可以保证装入数严格递增，并且盒子数最多，而且相邻两个盒子小球数相差1或2，保证了相邻盒子中球个数差的绝对值之和最小。

这种算法思想与贪心算法类似，先顺序递增装入小球，确保满足条件的局部最优解，如果有多出的球，再逆序加到盒子中，保证相邻盒子中球个数差最小。

但是使用此方法的一个前提条件是“n足够大”，否则算法就会出错。

进一步思考，假设盒子足够大，给定n和k值，如何修改代码实现题目要求呢？

若n值不是足够大，则顺序递增装入小球时，需要保证j<=n；若剩下的k>=n，则先平分小球，再逆序给每个盒子再装入1个，直到k=0 为止。即可满足条件。

使用Python代码描述如下：

函数名：box(n, k)

参数表：n--总的盒子数量

k--小球的数量

返回值：返回一个列表b，其中b[i]的值表示第i+1个盒子装入的小球数量。

def box(n, k):

b = []

j = 1

while j <= n and j <= k:

b.append(j)#先顺序递增装入小球

k -= j

j += 1

if k >= n: #平分小球

b = [i+k//n for i in b]

k %= n

i = len(b) - 1

while k > 0: #将多出的球逆序加到盒子中

b[i] += 1

k -= 1

i -= 1

return b

n, k = 6, 8

a = box(n, k)

print(a)

课后思考：

在本题中我们采用贪心策略，模拟了整个装球的过程，得到了最优解，应该说算法还是比较简单明了的。

装球问题是组合数学中的一道经典问题，除了模拟装球过程，我们还可以从其他角度来分析，下面就来看一道与本题类似的问题：把m个同样的小球放在n个同样的盒子里，允许有的盒子空着不放，问共有多少种不同的分法？其中5，1，1和1，5，1 是同一种分法。

聪明的你不妨思考一下这个问题，看看能采用哪些方法来解决它？

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。