Python算法之旅（第5期）

上期回顾：

题目描述：学校的小礼堂每天都会有许多活动，有时候这些活动的计划时间会发生冲突，

只能选择举办其中的部分活动。小刘的工作就是安排学校小礼堂的活动，每个时间最多安排一个活动。

现在小刘有一些活动计划的时间表，他想尽可能的安排更多的活动，请问他该如何安排。

注意：如果上一个活动在t时间结束，下一个活动最早应该在t+1时间开始。

函数名：schedule(n, mt)

参数表：n,正整数，表示总共n个活动

mt,列表，mt[i][0]和mt[i][1]分别表示活动i的起始和结束时间，确保mt[i][0]<mt[i][1]

返回值：返回最多能够安排的活动数量。

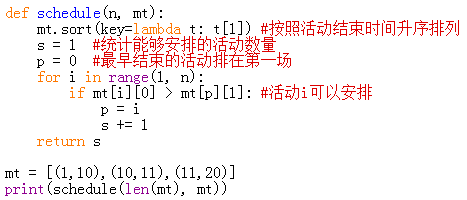
示例1：n=2,mt=[(1,10),(10,11)]时，返回1

示例2：n=3,mt=[(1,10),(10,11),(11,20)]时，返回2

算法分析：因为只有一个会场，所以同一时间内只有一个活动可以安排。要想安排尽可能多的活动，则需要优先安排结束时间靠前的活动，这是典型的贪心算法。

我们可以按照活动结束时间升序排列，变量p代表当前被安排活动的序号，然后从第二个活动开始，遍历列表，查看活动i的开始时间是否大于活动p的结束时间，若大于，则说明活动i可以被安排，将p指向i。继续查找下一个可以被安排的活动。

代码如下：



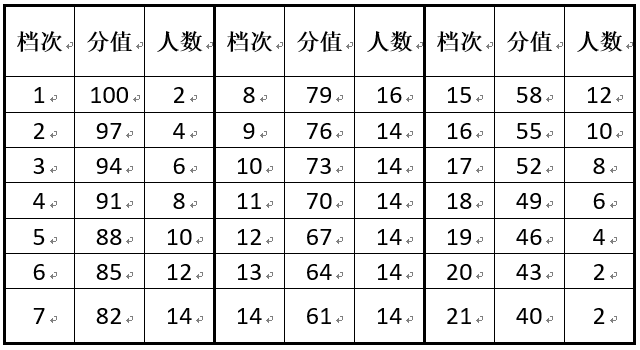
为准备信息技术高考，浙江各地举行了各种联考，模拟考，出题老师绞尽脑汁，殚精竭虑，为广大学子准备了大量优秀的VB算法练习题。下面我为大家隆重推荐一道好题。

题目：选考成绩赋分(2018年12月浙江省重点中学期末热身联考17题)

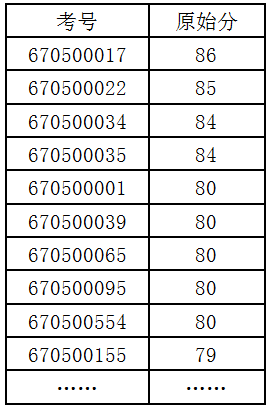
难度：3星 有趣：3星 有用：4星

分类：模拟，贪心

描述：成绩赋分。浙江省的选考成绩采用等级赋分制，共分为21档。赋分方法是先将该科目的成绩从高到低排序，再依次根据不同档次的人数进行赋分。当成绩相同但人数超出该等级人数时，则减少下一档的人数。当原始成绩为0分，则赋分成绩也为0分。某学校某门科目的选考人数为200人，部分成绩如第17题图a所示，其相应等级的人数如第17题图b所示。请回答下列问题：



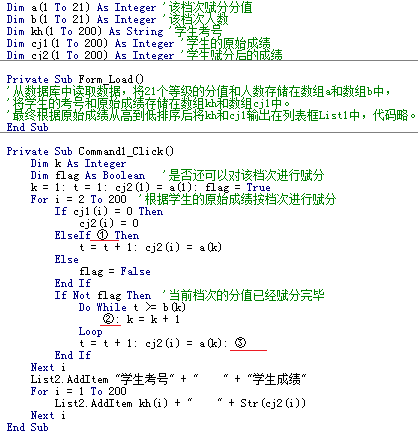
第17题图b



第17题图a

（1）根据题意，观察赋分表，考号670500554的学生的赋分成绩是 ▲ 。

（2）实现上述功能的程序如下，请在划线处填上合适的代码。



答案：（1）97 （1分）

（2）①cj1(i) = cj1(i - 1) Or t < b(k) （2分）

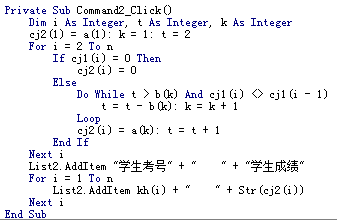
②t = t - b(k) 或 b(k + 1) = b(k + 1) + b(k) （2分）

③flag = True （1分）

算法分析：

题目提供的代码模拟了手工赋分的过程，为了不改变数组b的值，他用变量t累计当前档次已赋分人数，并引入变量flag标记当前档次赋分是否结束。总体来说代码思路还是很清晰的，能很好的完成既定任务。

但我在阅读代码时总感觉它某些部分有重复，如果调整一下代码结构，应该可以简化代码。经过思考，决定取消变量flag，并把内层循环整合到else语句内。代码如下：



如果跳出题目提供代码的影响，重新考虑实现选考成绩赋分功能，其实最直接的思路是先根据已经按降序排列的原始成绩，计算出对应的排名（成绩相同的排名也相同），存储到数组mc；然后根据各档学生人数计算到第i档为止的人数，存储到数组b2。这样mc和b2就一一对应起来了，可以直接根据它们的关系对学生进行赋分。

本题的VB代码没有采用上述分步走的思路，而是把计算排名，更新各档次人数和赋分放到一起来做了。

我们现在使用Python语言来实现相关功能，各项数据我们直接从Excel表格中读取。

描述：浙江省的选考成绩采用等级赋分制，共分为21档。赋分方法是先将该科目的成绩从高到低排序，再依次根据不同档次的人数进行赋分。当成绩相同但人数超出该等级人数时，则减少下一档的人数。当原始成绩为0分，则赋分成绩也为0分。

函数名：grade(a, b, c)

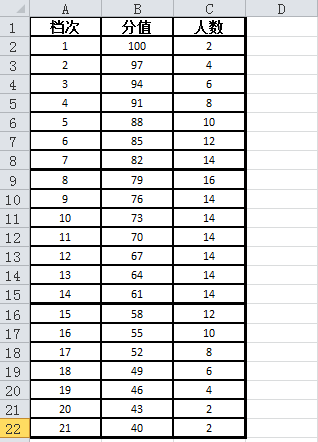
参数表：a -- 元组，a[1:]存储各档次赋分分值。

b -- 元组，b[1:]存储各档次人数。

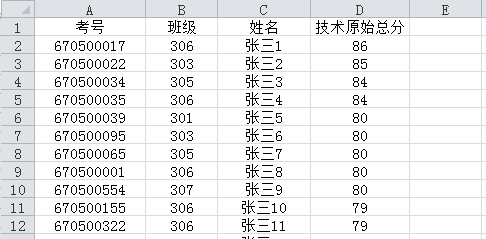
c -- 元组，c[1:]存储按降序排列原始成绩。

返回值：列表c2，其中c2[0]="赋分",c2[1:]存储学生赋分成绩。

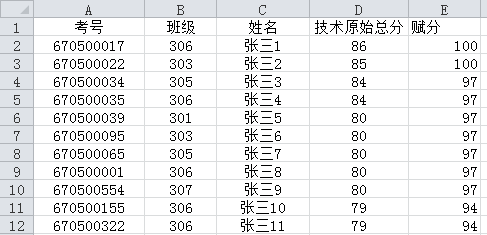
赋分标准表：

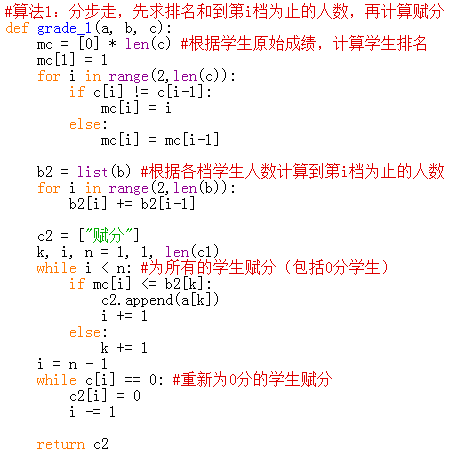


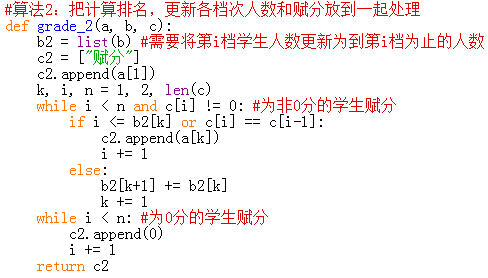
计算赋分前学生成绩表格：



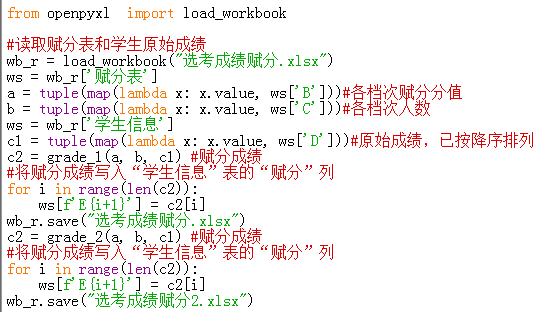
计算赋分后学生成绩表格：







主函数部分，我们调用了第三方类openpyxl来操作Excel表格。



上述两种算法，grade\_1()是直接模拟手工计算赋分的过程，分成3步走，先根据已排序的原始成绩计算排名，再根据根据各档学生人数计算到第i档为止的人数，最后在生成赋分成绩，思路简洁明了，条理清晰，满足Python语言“简洁而不是简短”的训诫；grade\_2()把3步并成1步，代码更简洁，但是理解上稍有难度，需要认真分析才能体会它的精妙之处。

课后思考：

在函数grade\_1()中，我们根据已排序的学生原始成绩，计算学生排名，只需一重循环就搞定了。现在的问题是：如果原始成绩是未排序的，计算学生排名又该怎么做呢？

描述：已知元组t存储了若干个正整数，请编写函数返回各元素在元组中的排名。

排名规则：数值越大排名越靠前，数值相同则排名相同。

函数名：rank(t)

参数表：t -- 存储了正整数的元组。

返回值：返回一个列表，其元素值为相同下标的元素在元组中的排名。

示例：对于元组a=(3,2,2,4,3,5)，返回[3,5,5,2,3,1]