金华十校2018-2019学年高三技术卷第17题分析

17. [加试题]学校筹办社团节,每个社团先到A场地做“准备”,然后到B场地“风采展示汇报"。同一场地，同一时间只允许一个社团使用。每个社团使用A、B场地时间都有所不同。已知学校共n个社团，第i个社团使用A场地时长为a[i]分钟,使用B场地时长为b[i]分钟。为了更高效地组织这次活动，某同学编写了如下VB程序计算此次活动的最小总时长和社团参会的顺序。

算法思路:

1.统计m[i]表示第i个社团中在A和B两个场地中用时的较小值。

2.按m[i]值从小到大排序，然后按m[i]值的顺序，逐个社团安排参会顺序，策略如下：

为了使得总时长最短，让A场地用时最少的最先开始；B场地用时最少的最后开始。

对于每个社团，若m[i]与该社团在A场地使用时间相同，则将它排在剩余的可排位置的最前面；若m[i]与该社团B场地使用的时间相等，则将它安排在剩余可排位置的最后面。

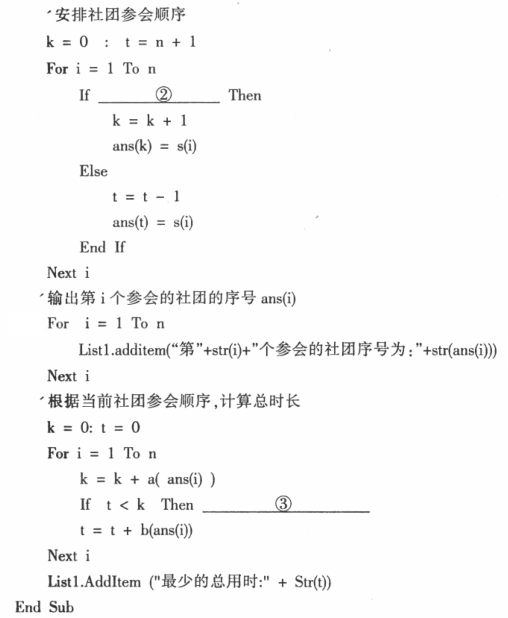
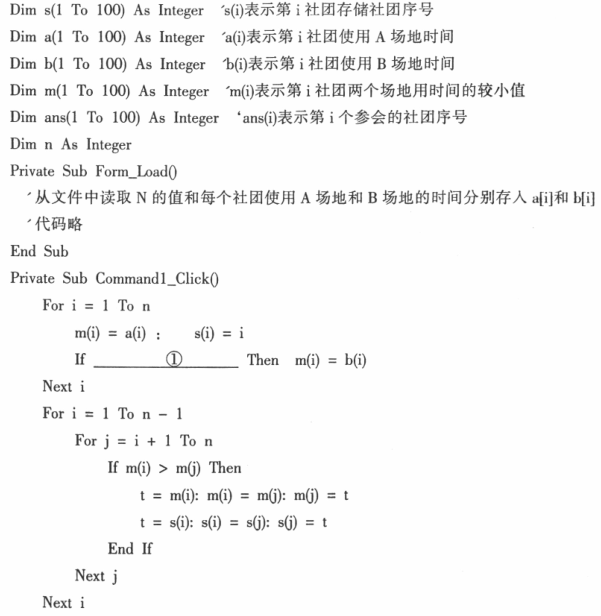
例如: N=5，社团序号分别是(1, 2, 3, 4, 5)

1至5号社团使用A场地的时间依次为：(3, 5, 8, 7, 10)

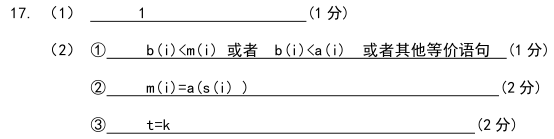
1至5号社团使用B场地的时间依次为：(6, 2, 1, 4, 9)

按上述算法可求得5个社团m[i]的值依次为：(3, 2, 1, 4, 9)

(1)按上述算法策略，5个社团中最先进人A场地的社团的序号是\_\_\_\_\_\_\_\_ ( 填数字)

(2)请在划线处填人合适的代码  


答案：



解析：

要理解本题的算法（代码），首先要搞清楚如何才能使活动总时长最少。基本原则是先准备的先表演，若上一个社团表演完了，接下来要表演的社团仍未准备好，则等待。假设已经按策略把各个社团排好队（即按照社团1到社团n的顺序依次出场），因为要求每个社团都要先到场地A准备，再去场地B表演，所以总时间 = 社团1准备时间 + 社团活动时间（场地A和B同时进行）+ 社团n表演时间。

社团活动时间是指从社团1开始表演到社团n准备结束所用的时间，在这段时间里，A,B两个场地同时进行，其中场地A的活动可以连续不断的进行，但场地B的活动可能会有停顿，例如当社团2的准备时间比社团1的表演时间长时，当社团1表演完毕后，场地B就处于停顿状态，需要等待社团2准备完毕才能再进行表演。

为了使活动总时长最少，我们需要确保社团1的准备时间和社团n的表演时间都是最短的。一个比较直接的思路是把社团分成两类，一类是准备时间小于表演时间，另一类则相反。为了使活动总时长最少，我们需要让准备时间小于表演时间的社团先出场，而且是准备时间越短的越早出场；同理让表演时间小于准备时间的社团后出场，而且是表演时间越短的越晚出场。基于这个考虑，可以找出最先和最后出场的社团，再依此类推，这是一个贪心策略。

题目中的算法很巧妙的实现了此策略，它通过比较每个社团的准备时间和表演时间，将较小值存储到数组m中，再将数组m排序，尽可能让准备时间短的社团先出场，而表演时间短的后出场。然后再比较a[i]和m[i]的值，判断社团i的准备时间是否小于表演时间，从而确定出场顺序。

我们通过两个示例来计算活动总时长。

例1：N=5，社团序号(1, 2, 3, 4, 5)，a = (3, 5, 8, 7, 10)，b = (6, 2, 1, 4, 9)。

根据算法策略，我们得到m = (3, 2, 1, 4, 9)，出场顺序ans = (1, 5, 4, 2, 3)，

总时长 = 社团1准备时间 +（社团1表演时间 + 停顿时间（可为0））+ （社团2表演时间 + 停顿时间（可为0））+ ….. + 社团n表演时间

即3 +（6 + 4）+ 9 + 4 + (2 + 5) + 1 = 34。

出场顺序表：

社团1准备时间表：0--3，表演时间表：3—9 （之后停顿4分钟）

社团5准备时间表：3--13，表演时间表：13—22

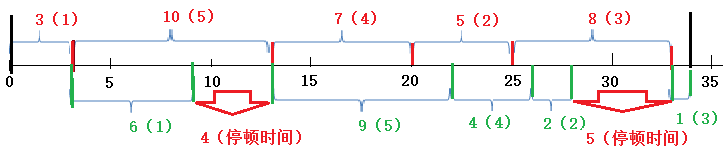
社团4准备时间表：13--20，表演时间表：22—26

社团2准备时间表：20--25，表演时间表：26—28 （之后停顿5分钟）

社团3准备时间表：25--33，表演时间表：33—34

活动总时长：34

出场时刻表如图示：



说明：因为社团1表演结束时，社团5仍未准备好，所以场地B停顿了4分钟（相当于将社团1的表演结束时间延迟到第13分钟）；同样后面因为社团3没有准备好，也停顿了5分钟（相当于将社团2的表演结束时间延迟到第33分钟）。

将停顿时间算到上一场的表演时间里面以后，活动总时长就等于首个出场的社团的准备时间加上所有社团的表演时间。

例2：N=5，社团序号(1, 2, 3, 4, 5)，a = (3 , 5 , 2 , 4 , 10)，b = (6 , 8 , 1 , 7 , 9)。

根据算法策略，我们得到m = (3, 5, 1, 4, 9)，出场顺序ans = (1, 4, 2, 5, 3)，

总时长 = 3 + 6 + 7 + 8 + 9 + 1 = 34。

出场顺序表：

社团1准备时间表：0--3，表演时间表：3--9

社团4准备时间表：3--7，表演时间表：9--16

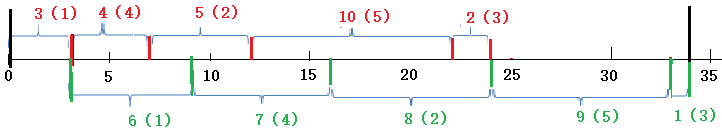
社团2准备时间表：7--12，表演时间表：16--24

社团5准备时间表：12--22，表演时间表：24--33

社团3准备时间表：22--24，表演时间表：33--34

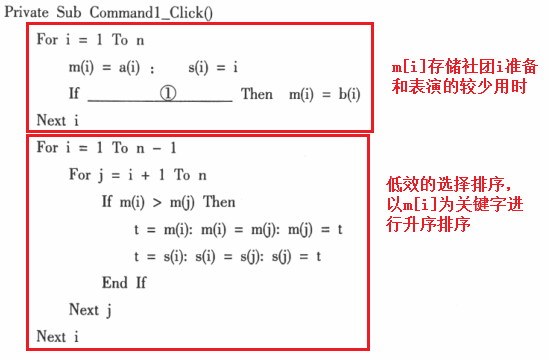
活动总时长：34

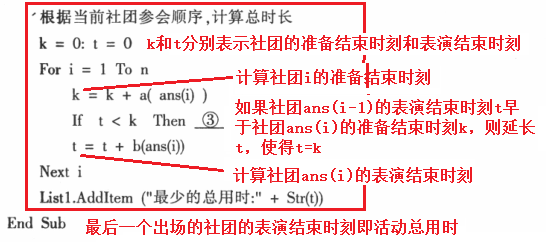
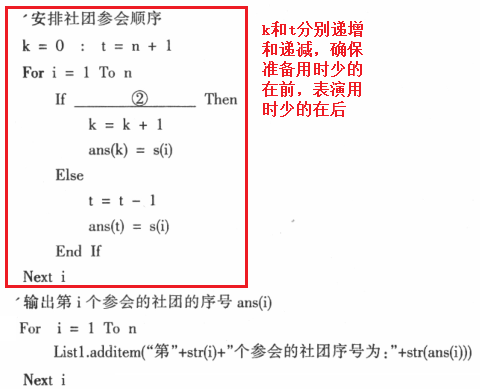
出场时刻表如图示：



说明：例2比较简单，A,B场地都利用充分，没有停顿时间，故只需将所有社团的表演时间加起来就是总的表演用时，再加上社团1的准备用时，就等于活动总用时。

搞清楚了算法原理，代码就比较好理解了。我们结合代码说明如下：





还有一个问题：当社团1和社团2的准备时间均小于表扬时间，且a[1]=a[2]，b[1]<b[2]时，谁先出场？

题目所给的策略中似乎没有考虑这个问题，当m[i]=a[i]时，它只对a[i]进行排序，不考虑b[i]。很明显，如果考虑了b[i]，那出场顺序会不一样，那此时不一样的出场顺序对总时长是否有影响呢？

答案是没有影响！因为当对于m[i]=a[i]这些社团，他们的活动总用时是首个出场团队的准备用时加上各个团队的表演用时，与出场顺序无关。

同理，当社团1和社团2的准备时间均大于表扬时间，且b[1]=b[2]，a[1]<a[2]时，他们俩谁先出场不影响活动总时长。

考虑两个极端的例子：

例1：N=:4，社团序号(1, 2, 3, 4)，a = (1 , 1 , 6 , 8)，b = (3 , 4 , 4 , 4)。

根据算法策略，我们得到m = (1, 1, 4, 4)，此时出场顺可以是ans = (1, 2, 3, 4)，或ans = (1, 2, 4, 3)，或ans = (2, 1, 3, 4)，或ans = (2, 1, 4, 3)。

无论哪种出场顺序，总时长都是1 + 1 + 6 + 8 + 4 = 20。

例2：N=:4，社团序号(1, 2, 3, 4)，a = (3 , 3 , 3 , 4)，b = (6 , 8 , 2 , 2)。

根据算法策略，我们得到m = (3, 3, 2, 2)，此时出场顺可以是ans = (1, 2, 3, 4)，或ans = (1, 2, 4, 3)，或ans = (2, 1, 3, 4)，或ans = (2, 1, 4, 3)。

无论哪种出场顺序，总时长都是3 + 6 + 8 + 2 + 2 = 21。

使用Python语言可以得到更简洁的代码。描述如下：

描述：学校筹办社团节,每个社团先到A场地做“准备”,然后到B场地“风采展示汇报"。同一场地，同一时间只允许一个社团使用。每个社团使用A、B场地时间都有所不同。已知学校共n个社团，第i个社团使用A场地时长为a[i]分钟,使用B场地时长为b[i]分钟。为了更高效地组织这次活动，需要计算此次活动的最小总时长并输出社团出场的顺序。

函数名：min\_time(a, b)

参数表：a--数组，存储社团i在场地A的用时

b--数组，存储社团i在场地B的用时

返回值：活动的最小总时长，并输出社团出场的顺序和出场时刻表。

例如：社团序号为(1, 2, 3, 4, 5)，当a = (3, 5, 8, 7, 10)，b = (6, 2, 1, 4, 9)时，返回34，并输出：

出场顺序： 1 5 4 2 3

社团1准备时间表：0--3，表演时间表：3--9

社团5准备时间表：3--13，表演时间表：13--22

社团4准备时间表：13--20，表演时间表：22--26

社团2准备时间表：20--25，表演时间表：26--28

社团3准备时间表：25--33，表演时间表：33—34

算法分析：

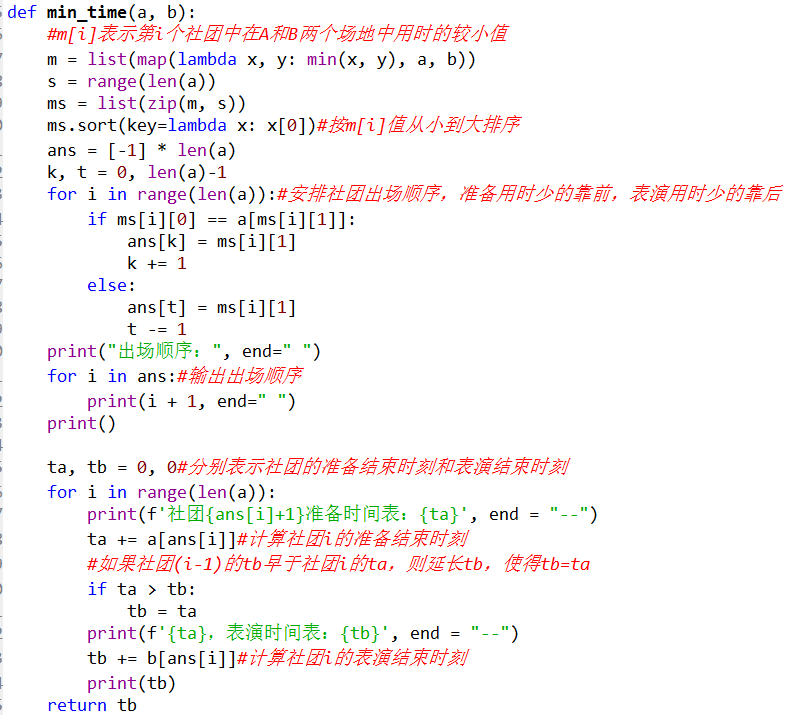
1.统计m[i]表示第i个社团中在A和B两个场地中用时的较小值。

2.按m[i]值从小到大排序，然后按m[i]值的顺序，逐个社团安排参会顺序，策略如下：

为了使得总时长最短，让A场地用时最少的最先开始；B场地用时最少的最后开始。

对于每个社团，若m[i]与该社团在A场地使用时间相同，则将它排在剩余的可排位置的最前面；若m[i]与该社团B场地使用的时间相等，则将它安排在剩余可排位置的最后面。

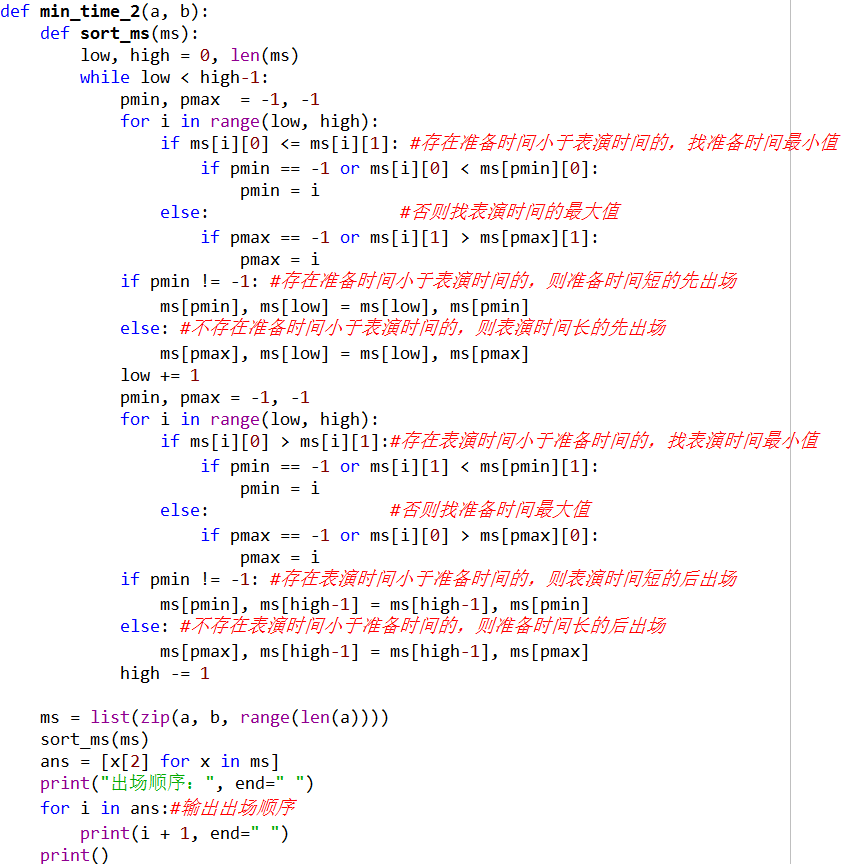
代码如下：



上述算法引入了一个数组m，通过对m赋值和排序，巧妙的获得了出场顺序。另外，我们也可以采用比较直接的想法（有些笨拙）。

参考双向选择排序的算法（即每趟排序同时找出最小值和最大值，分别交换到左端和右端，然后左右边界各向中间移动一步），我们建立一个二维数组ms = list(zip(a, b, range(len(a))))，扫描待排序区间，若存在准备时间小于表演时间的，找到准备时间的最小值，否则找到表演时间的最大值，与左端元素交换，并将左边界下标右移一位。然后扫描待排序区间，若存在表演时间小于准备时间的，找到表演时间的最小值，否则找到准备时间的最大值，与右端元素交换，并将右边界下标左移一位。这样就完成了一趟排序，按照相同规律把二维数组ms排好序即得到了出场顺序。

代码如下：



后面利用数组ans和数组a,b计算活动总时长的代码与算法1是一模一样的。

课后思考：

上述题目是对现实生活的简化，事实上应该考虑社团从场地A走到场地B的时间，才更符合实际情况。我们现在增加一个条件：各社团从场地A走到场地B的时间均为1分钟。

加了这个条件以后，各社团的出场顺序是否有变化？活动总时长又该如何计算呢？

聪明的你不妨想一想，修改哪些代码就可以实现目的呢？

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。

算法分析：只需在原来的基础上，加上社团从场地A走到场地B的时间即可，若社团ans[i]到达场地B后，社团ans[i-1]仍未结束表演，则延长tb = ta + walk\_time。代码如下：

