NOI2024 联合省选 Day1

时间：2024 年 2 月 22 日

题目名称	chef	draw	message
题目类型	传统题	传统题	传统题
目录	chef	draw	message
可执行文件名	chef	draw	message
输入文件名	chef.in	draw.in	message.in
输出文件名	chef.out	draw.out	message.out
每个测试点时限	3 秒	2 秒	2 秒
内存限制	1024 MB	512 MB	512 MB
子任务数目	5	3	4
测试点是否等分	否	否	否

提交源程序文件名

对于 C++ 语言	chef.cpp	draw.cpp	message.cpp

编译选项

对于 C++ 语言	-O2 -std=c++14 -static

注意事项（请仔细阅读）

1. 文件名（程序名和输入输出文件名）必须使用英文小写。
2. C/C++ 中函数 `main()` 的返回值类型必须是 `int`，程序正常结束时的返回值必须是 0。
3. 提交的程序代码文件的放置位置请参考各省的具体要求。
4. 因违反以上三点而出现的错误或问题，申诉时一律不予受理。
5. 若无特殊说明，结果的比较方式为全文比较（过滤行末空格及文末回车）。
6. 选手提交的程序源文件必须不大于 100KB。
7. 程序可使用的栈空间内存限制与题目的内存限制一致。
8. 只提供 Linux 格式附加样例文件。
9. 评测在当前最新公布的 NOI Linux 下进行，各语言的编译器版本以此为准。

chef（chef）

【题目描述】

YYX 转职成了厨师！

YYX 刚刚转职，才学会了两道菜的制作，第一道菜制作步骤有 N 步，第 i 步耗时 A_i 个单位时间，另一道菜制作步骤有 M 步，第 i 步耗时 B_i 个单位时间。

做菜是一个连续的过程，当 YYX 开始某个制作步骤时，她必须把这个步骤做完，但是把一个步骤完整做完之后可以先去做另一道菜的步骤，但她在任意时刻都不能休息。同时，在开始某个菜的第 $i(i \geq 2)$ 个制作步骤前，她必须把那道菜的第 $i-1$ 步做完。

现在，YYX 同时开始为 Dr. Wu 制作这两道菜。Dr. Wu 是个挑剔的人，他不仅追求最终菜的美味程度，还追求菜的制作速度以及厨师对厨房时间的分配策略。

具体的：

- $\forall 1 \leq i \leq n$，如果 YYX 在制作开始后 S_i 个单位时间内完成了第一道菜的第 i 步，则 Dr. Wu 的满足度上升 P_i。
- $\forall 1 \leq i \leq m$，如果 YYX 在制作开始后 T_i 个单位时间内完成了第二道菜的第 i 步，则 Dr. Wu 的满足度上升 Q_i。

初始时 Dr. Wu 的满意度为 0，请注意，P_i, Q_i 可能是负数。

YYX 当然希望 Dr. Wu 的满意度尽可能大。但是她太笨了，所以她找到了你，希望你帮她解决问题。

【输入格式】

从文件 ***chef.in*** 中读入数据。

第一行两个整数 N, M。

接下来 N 行，每行三个整数表示 A_i, S_i, P_i。

接下来 M 行，每行三个整数表示 B_i, T_i, Q_i。

【输出格式】

输出到文件 ***chef.out*** 中。

输出一个整数表示 Dr. Wu 满意度的最大值。

【样例 1 输入】

```
4 3
2 1 1
3 8 1
2 13 1
1 13 1
3 6 1
2 11 1
2 15 1
```

【样例 1 输出】

```
6
```

【样例 1 解释】

YYX 依次花费时间 $B_1, A_1, A_2, B_2, A_3, A_4, B_3$，可以证明这是最优方案。

【样例 2 输入】

```
5 7
16 73 16
17 73 10
20 73 1
14 73 16
18 73 10
3 73 2
10 73 7
16 73 19
12 73 4
15 73 15
20 73 14
15 73 8
```

【样例 2 输出】

```
63
```

【样例 3 输入】

```
9 11
86 565 58
41 469 -95
73 679 28
91 585 -78
17 513 -63
48 878 -66
66 901 59
72 983 -70
68 1432 11
42 386 -87
36 895 57
100 164 10
96 812 -6
23 961 -66
54 193 51
37 709 82
62 148 -36
28 853 22
15 44 53
77 660 -19
```

【样例 3 输出】

```
99
```

【样例 4】

见选手目录下的 ***chef/chef4.in*** 与 ***chef/chef4.ans***。

【数据范围】

对于全部的数据，$1 \le N, M \le 10^6, 1 \le A_i, B_j \le 10^9, 1 \le S_i, T_j \le 2 \times 10^{15}, |P_i|, |Q_j| \le 10^9$。

subtask1 5pts，$1 \le N, M \le 2 \times 10^5, S_1 = S_2 = \cdots = S_N = T_1 = T_2 = \cdots = T_M$。

subtask2 40pts，$1 \le N, M \le 2 \times 10^5, P_i = Q_i = 1$。

subtask3 20pts，$1 \le P_i, Q_i$。

subtask4 20pts，$1 \le N, M \le 2 \times 10^5$。

subtask5 15pts，无特殊限制。

draw（draw）

【题目描述】

YYX 转职成了画家！

作为画家，YYX 创作了一幅她生平最喜爱的作品，这个作品由 $n \times m$ 个像素组成，每个像素要么是黑的要么是白的。

有一天，YYX 看见 Dr. Wu 房间里有一面洁白的墙，她决定把自己引以为傲的作品重绘在这面墙上，于是她拿来了白油漆和黑油漆准备作画。YYX 可以采用如下方式作画：

- 用某个颜色的油漆，水平或垂直地画出 x 个像素，由于起笔处和收尾处需要一些特殊技巧，这需要消耗 $ax + b$ 单位的油漆。x 可以是任意正整数。
- 用某个颜色的油漆，画一个像素，这需要消耗 c 单位的油漆。

油漆干了之后，可以用其他颜色的油漆在上面覆盖，但是白油漆材质特殊特别难干。所以，YYX 可以在已经涂了黑油漆的位置用白油漆覆盖，但不能在已经涂了白油漆的位置用黑油漆覆盖。

同时，YYX 知道 Dr. Wu 发现她的恶作剧之后会生气，所以她必须保证涂油漆的层数不能太厚以至于太难去除。所以，任何一个位置**至多**被涂上油漆两次。

YYX 重绘时，需要满足，作品中黑像素对应的位置必须最终被黑油漆涂上，白像素对应的位置要么不涂油漆，要么最终被白油漆涂上。

YYX 当然希望消耗的油漆最少。但是她太笨了，所以她找到了你，希望你帮她解决问题。

【输入格式】

从文件 ***draw.in*** 中读入数据。

第一行五个整数 n, m, a, b, c。

接下来 n 行，每行一个长度为 m 的只含 `.` 和 `#` 的字符串，其中第 i 行第 j 个字符为 `#` 表示 YYX 作品中第 i 行第 j 列的像素为黑像素，否则为白像素。

【输出格式】

输出到文件 ***draw.out*** 中。

输出一个整数表示最小消耗油漆量。

【样例 1 输入】

```
3 3 1 2 3
.#.
###
.#.
```

【样例 1 输出】

```
10
```

【样例 1 解释】

YYX 用黑油漆从 $(1,2)$ 画到 $(3,2)$，又从 $(2,1)$ 画到 $(2,3)$，总花费 $(1\times3+2)+(1\times3+2)=10$，可以证明没有更优方案。

【样例 2 输入】

```
5 5 1 4 4
..#..
..#..
##.##
..#..
..#..
```

【样例 2 输出】

```
24
```

【样例 2 解释】

YYX 用黑油漆从 $(1,3)$ 画到 $(2,3)$，从 $(4,3)$ 画到 $(5,3)$，从 $(3,1)$ 画到 $(3,2)$，从 $(3,4)$ 画到 $(3,5)$，总花费 $(1\times2+4)\times4=24$，可以证明没有更优方案。

【样例 3 输入】

```
10 25 3 30 10
##########################
#.#####.##.#####.##.###.#
#..###..##..###..##..#..#
##.###.####.###.####...##
##..#..####..#..#####.###
###...#######...######...##
####.########.#####..#..#
####.########.#####.###.#
####.########.#####.###.#
##########################
```

【样例 3 输出】

```
1315
```

【数据范围】

对于全部的数据，$1 \le n, m \le 40, 0 \le a, b, c \le 40, c \le a + b$。

subtask1 30pts，$a = b$。

subtask2 30pts，$m \le 8$。

subtask3 40pts，无特殊限制。

message（message）

【题目描述】

YYX 转职成了间谍!

YYX 秘密潜入了 Dr. Wu 的实验室，获得了 n 条情报并对其分别进行了加密，第 i 条情报的密文长度为 a_i 。

现在 YYX 需要利用秘密信道把这些密文传输出去，为了保证友军可以破译密文，她传输一条密文时必须持续地把它传输完，而不能分成小段分别传输。但好消息是信道的容量很大，她可以同时传输若干条密文 (这些密文的开始时间可以不同)。一条长度为 l 的密文传输需要花费连续的 l 个单位时间。

具体的，YYX 可以任意地给每条密文分配一个非负整数 s_i 表示其开始传输的时间，那么第 i 条密文的传输时间为 $[s_i, s_i + a_i]$ 。这些区间可以任意地重叠、相交或不交。

但是 Dr. Wu 其实已经发现了秘密信道，他偶尔会来监听这条信道 x 个连续的单位时间。YYX 知道，如果 Dr. Wu 获得了至少三条完整的密文，那么他就可以把密码系统破解出来，所以，YYX 必须保证对于任意一个长度为 x 的时间区间，其完整包含的密文数量不能超过两个。具体的，

$\forall p, \sum [s_i \geq p \wedge s_i + a_i \leq p + x] \leq 2$

同时，YYX 如果进行传输的时间太长，也会引起 Dr. Wu 的疑心，故 YYX 希望密文传输结束的时间（即 $\max\{s_i + a_i\}$ 尽可能早。但是她太笨了，所以她找到了你，希望你帮她解决问题。

【输入格式】

从文件 ***message.in*** 中读入数据。

第一行两个整数 n, x 。

第二行 n 个整数表示 a_i 。

【输出格式】

输出到文件 ***message.out*** 中。

输出一个整数表示密文传输结束的最小时间。

【样例 1 输入】

```
6 10
2 3 4 5 6 7
```

【样例 1 输出】

```
16
```

【样例 1 解释】

考虑 $\{s_i\} = [0, 13, 11, 0, 5, 4]$ ，容易验证是一组解。可以证明没有更优的解，比如当 $\{s_i\} = [0, 12, 11, 0, 5, 4]$ $[5, 15]$ 这个区间就包含了第 $2, 3, 5$ 条密文。

【样例 2 输入】

```
7 6
9 3 2 3 8 3 3
```

【样例 2 输出】

```
11
```

【样例 2 解释】

考虑到第 1,5 条密文永远不会被监听到，$\{s_i\} = [0, 0, 9, 4, 0, 0, 4]$ 即为一组解，可以证明没有更优的解。

【样例 3】

见选手目录下的 ***message/message3.in*** 与 ***message/message3.ans***。

【数据范围】

对于全部的数据，$1 \le n \le 2 \times 10^4, 1 \le a_i, x \le 10^4$ 。

subtask1 10pts, $n, x \le 100$ 。

subtask2 20pts, $x \le 100$ 。

subtask3 20pts, $n \le 100$ 。

subtask4 50pts，无特殊限制。