必可赛前公益众筹赛

第五试

时间: 2024 **年** 10 **月** 10 **日** 08:00 ~ 12:00

题目名称	卡门	商人	自行车	记忆
题目类型	传统型	传统型	传统型	传统型
目录	kamen	merchant	bike	memory
可执行文件名	kamen	merchant	bike	memory
输入文件名	kamen.in	merchant.in	bike.in	memory.in
输出文件名	kamen.out	merchant.out	bike.out	memory.in
每个测试点时限	1.0 秒	1.0 秒	2.0 秒	2.0 秒
内存限制	512 MB	512 MB	512 MB	512 MB
测试点数目	10	10	6	6
测试点是否等分	是	是	否	否

提交源程序文件名

对于 C++ 语言	kamen.cpp	merchant.cpp	bike.cpp	memory.cpp

编译选项

对于 C++ 语言	-lm -O2 -std=c++14

注意事项与提醒（请选手务必仔细阅读）

1. 选手请直接提交源程序至 becoder.com.cn 上的对应比赛。
2. 输入输出文件名必须使用英文小写。
3. 选手提交的源程序必须存放在**已建立**好的，且**带有样例文件和下发文件**的文件夹中，文件夹名称与对应试题英文名一致。
4. 文件名 (包括程序名和输入输出文件名) 必须使用英文小写。
5. C++ 中函数 main() 的返回值类型必须是 int，值必须为 0。
6. 若无特殊说明，结果比较方式为**忽略行末空格、文末回车后的全文比较**。
7. 程序可使用的栈空间大小与该题内存空间限制一致。
8. 在终端中执行命令 ulimit -s unlimited 可将当前终端下的栈空间限制放大，但你使用的栈空间大小不应超过题目限制。
9. 每道题目所提交的**代码文件大小限制为 100KB**。
10. 若无特殊说明，输入文件与输出文件中同一行的相邻整数均使用一个空格分隔。

11. 输入文件中可能存在行末空格，请选手使用更完善的读入方式 (例如 scanf 函数) 避免出错。

12. 直接复制 PDF 题面中的多行样例，数据将带有行号，建议选手直接使用对应目录下的样例文件进行测试。

13. 使用 std::deque 等 STL 容器时，请注意其内存空间消耗。

14. 请务必使用题面中规定的的编译参数，保证你的程序在本机能够通过编译。此外**不允许在程序中手动开启其他编译选项**，一经发现，本题成绩以 0 分处理。

15. **对于因未遵守以上规则对成绩造成的影响，相关申诉不予受理**。

卡门（kamen）

【题目描述】

在一个 $r \times c$ 的方阵里，有些点是‘.’，表示为空；有些点是‘X’，表示这里是一堵墙。我们可以认为这个方阵在竖直方向放置。

有一个人在这 c 列的第一行会抛下 n 块石头，用‘O’来表示。这一个石头由于重力作用会向下滚动。具体来说，就是会从第一行向最后一行滚动，规则如下：

1. 如果下一个格子是空格，那么向下运动一格。
2. 如果下一个格子是墙或者已经到了第 r 行，则停止滚动并停在原处。
3. 如果下一个格子是一块停止的石头，则如果在左侧和左下方都为空格时首选滚动到左侧的那一行，否则如果右侧和右下方都为空格，则滚动到右侧的那一行。如果两侧都不为空，则石头静止不再移动。

只有前一块石头永久静止后，下一块石头才会被抛下。

请你输出最终方阵的状态。

【输入格式】

从文件 ***kamen.in*** 中读入数据。

输入第一行为两个整数 r, c。

接下来的 r 行，每行 c 个字符，每个字符可能为‘.’或‘X’。

下一行为一个整数 n，表示石头的数量。

接下来的 n 行，每行一个整数 $1 \le x \le c$，表示这个石头位于第 x 列被抛下，按照输入顺序依次抛下。

【输出格式】

输出到文件 ***kamen.out*** 中。

输出一个 $r \times c$ 的方阵，包含‘.’‘X’‘O’，为最终石头都抛完后的状态。

【样例 1 输入】

```
5 4
....
....
X...
....
....
4
1
```

```
1
1
1
```

【样例 1 输出】

```
....
O...
X...
....
OOO.
```

【样例 1 解释】

4 块石头依次在第一列被抛下。第一块石头被唯一一堵墙堵住。这样剩下的石头都可以向右滚动一列。第二块石头毫无障碍地下落，第三四块分别落在了它的左边和右边。

【样例 2 输入】

```
7 6
......
......
...XX.
......
......
.XX...
......
6
1
4
4
6
4
4
```

【样例 2 输出】

```
......
```

```
...O..
...XX.
.......
.OO...
.XX...
O..O.O
```

【样例 3】

见附加文件中的 kamen/ex_kamen3.in 与 kamen/ex_kamen3.out。

【数据范围】

对于 60% 的数据，保证 $r \leq 30$。

对于 100% 的数据，保证 $1 \leq r \leq 3 \times 10^4$，$1 \leq c \leq 30$，$1 \leq n \leq 10^5$。保证每次抛石子头（假设在第 i 列抛）之前，第一行第 i 列的位置上没有石头。

商人（merchant）

【题目描述】

给定一张 n 个点 m 条边的有向图（点的下标从 1 开始）。

对于每条边 (u, v, r, p) 表示有一条从 u 到 v 的有向边，走过前必须至少持有 r 元钱，走过后会增加 p 元钱。分别求出从每个点出发能够无限地游走下去所需要的最少的初始钱数。特别地，若从这个点以任意初始钱数出发都无法无限游走下去则输出 -1 。

【输入格式】

从文件 ***merchant.in*** 中读入数据。

输入 $1+m$ 行。

第一行，两个整数 n m，分别表示有向图的点数和边数。

接下来 m 行，每行四个非负整数 u_i, v_i, r_i, p_i ，描述一条有向边。

【输出格式】

输出到文件 ***merchant.out*** 中。

输出一行，n 个数，分别表示每个点的答案。

【样例 1 输入】

```
5 5
3 1 4 0
2 1 3 0
1 3 1 1
3 2 3 1
4 2 0 2
```

【样例 1 输出】

```
2 3 3 1 -1
```

【样例 2 输入】

```
5 7
1 2 0 1
2 1 10 2
2 3 1 3
3 4 5 2
4 2 2 4
```

```
3 5 1 5
4 5 0 3
```

【样例 2 输出】

```
1 2 5 2 −1
```

【样例 3】

见附加文件中的 merchant/ex_merchant3.in 与 merchant/ex_merchant3.ans。
该样例满足 $n \le 2 \times 10^3, m \le 2 \times 10^3$。

【样例 4】

见附加文件中的 merchant/ex_merchant4.in 与 merchant/ex_merchant4.ans。
该样例满足 $n \le 2 \times 10^5, m \le 2 \times 10^5$ 且 $r_i = 0$。

【样例 5】

见附加文件中的 merchant/ex_merchant5.in 与 merchant/ex_merchant5.ans。
该样例满足 $n \le 2 \times 10^5, m \le 2 \times 10^5$。

【数据范围】

对于全部数据，$1 \le n, m \le 2 \times 10^5, 0 \le r_i, p_i \le 10^9$。
对于 20% 的数据，保证 $n \le 5, m \le 15$。
对于另外 20% 的数据，保证 $n \le 2 \times 10^3, m \le 2 \times 10^3$。
对于另外 20% 的数据，保证 $n \le 2 \times 10^5, m \le 2 \times 10^5$。满足性质 A。
对于另外 40% 的数据，保证 $n \le 2 \times 10^5, m \le 2 \times 10^5$。
特殊性质 A：$r_i = 0$。

自行车（bike）

【题目描述】

在隆德，骑自行车是一种常见的交通方式，但有时狭窄的街道难以容纳骑车和自行车通行。为了改善这一状况，当地政府希望完全重新设计当地的街道网络。

隆德共有 N 个关键位置（编号从 0 到 $N-1$），人们常常在其间通行。每个新修的街道连接两个不同的关键位置，宽度都为 W。人们通过一条路径在两地之间通行，其中路径是一系列的街道。街道可以被任意分割为自行车道和机动车道，例如，如果自行车道的宽度为 b，那么机动车道的宽度为 $W-b$。一种交通工具（车或自行车）可以在一条路径上通行，当且仅当经过的所有车道的宽度都大于等于交通工具自身的宽度。在隆德，一些工程师最近发明了宽度为 0 的车和自行车（它们可以在宽度为 0 的车道上通行）。

这些工程师测量了城市中车和自行车的宽度。对于每一对关键位置，他们知道其间通行的最宽的车和最宽的自行车的宽度，但政府还要求更宽的车和自行车不能在其间通行。

形式化地，对于任意 i,j（$0 \le i < j \le N-1$），你都知道两个整数 $C_{i,j}$ 和 $B_{i,j}$。你的任务是构造一个街道网络连接这 N 个位置。街道的宽度均为 W，但对于任意街道 s，你可以选择它的自行车道宽度 b_s 和机动车道宽度 $W-b_s$。这个网络必须满足以下要求：

1. 必须可以在任意两个位置间通行。这可能需要一辆宽度为 0 的自行车或车。
2. 对于每一对位置 i,j（其中 $i<j$），可以只借助宽度至少为 $C_{i,j}$ 的机动车道，在 i,j 间通行。同时，$C_{i,j}$ 是最大的有这一性质的数。这意味着，对于所有 i,j 间的路径，必须满足至少一条街道的机动车道宽度不超过 $C_{i,j}$。
3. 对于每一对位置 i,j（其中 $i<j$），可以只借助宽度至少为 $B_{i,j}$ 的自行车道，在 i,j 间通行。同时，$B_{i,j}$ 是最大的有这一性质的数。

你能帮隆德政府设计一个这样的街道网络吗？因为预算有限，你可以修建至多 2023 条街道。你可以在同一对关键位置间修建多条街道，但你不能修建一条连接自己和自己的街道。所有街道都是双向的。如果有多种设计方案，你只需给出任意一种即可。

【输入格式】

从文件 ***bike.in*** 中读入数据。

第一行两个整数 N,W，表示关键位置数量和街道宽度。

接下来 $N-1$ 行包含 $C_{i,j}$。其中的第 j 行包含所有满足 $i<j$ 的 $C_{i,j}$。因此第一行只会包含 $C_{0,1}$，第二行包含 $C_{0,2}$ 和 $C_{1,2}$，第三行包含 $C_{0,3},C_{1,3},C_{2,3}$，以此类推。

接下来 $N-1$ 行包含 $B_{i,j}$，格式同 $C_{i,j}$。

【输出格式】

输出到文件 ***bike.out*** 中。

如果不存在符合要求的街道网络，输出 ‘NO‘。

否则，第一行一个整数 M，表示你修建的街道数。

接下来 M 行，每行三个整数 u, v, b，表示一条自行车道宽度为 b（机动车道宽度为 $W - b$）的连接 u, v 的街道。

你可以使用至多 2023 条街道。你输出的街道必须满足 $0 \leq b \leq W$，$0 \leq u, v \leq N-1$ 且 $u \neq v$。你可以在同一对关键位置间使用多条街道（可以有不同的自行车道宽度）。

如果有多解，你可以输出任意一个。

【样例 1 输入】

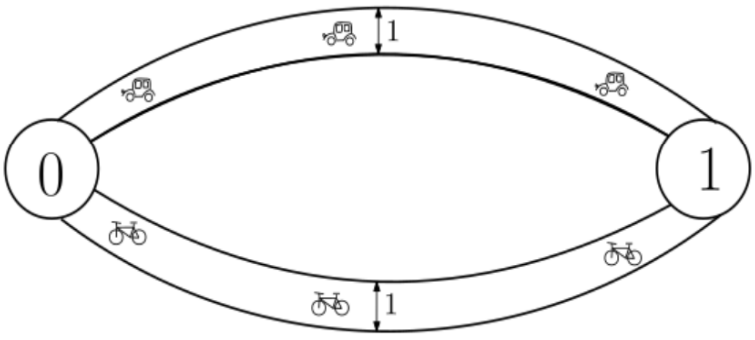
```
2 1
1
1
```

【样例 1 输出】

```
2
0 1 0
0 1 1
```

【样例 1 解释】

在样例 1 中，街道的宽度为 1，我们需要宽度至少为 1 的自行车道和机动车道各一条连接 $0, 1$。解决方案是用两条分开的街道连接，一条自行车道宽度为 1，另一条机动车道宽度为 1。



(a)

【样例 2 输入】

```
4 1
0
```

```
0 1
0 0 1
1
1 1
1 1 1
```

【样例 2 输出】

```
NO
```

【样例 2 解释】

在样例 2 中，街道的宽度依然是 1，需要有一条宽度为 1 的自行车道连接任意两个关键位置，且有宽度为 1 的机动车道连接 1, 2 和 2, 3。这与 $C_{1,3} = 0$ 矛盾，不应该有宽度为 1 的从 1 到 3 的机动车道，而我们可以合并两条先前提到的路径组成这样一条路径。因此不可能构造出一个符合要求的街道网络。

【样例 3 输入】

```
6 6
5
4 4
1 1 1
1 1 1 3
1 1 1 5 3
2
3 2
6 2 3
3 2 5 3
3 2 4 3 4
```

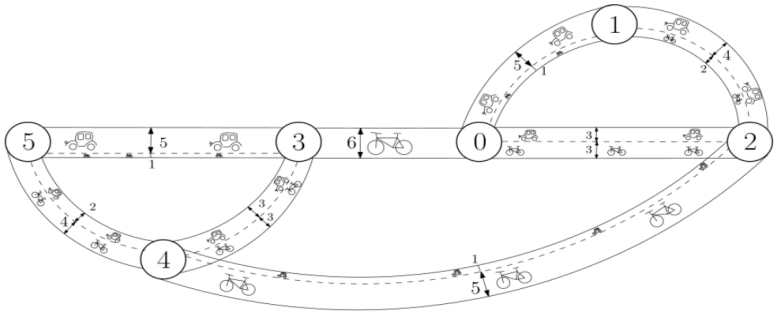
【样例 3 输出】

```
8
0 1 1
0 2 3
1 2 2
0 3 6
2 4 5
```

```
3 4 3
3 5 1
4 5 4
```

【样例 3 解释】

在样例 3 中，下图的街道网络满足所有要求。例如，应该有一条宽度为 $1 = C_{0,5}$ 的机动车道在 $0, 5$ 之间（路径 $0 \to 2 \to 4 \to 5$），一条宽度为 $3 = B_{0,5}$ 的自行车道在 $0, 5$ 之间（路径 $0 \to 3 \to 4 \to 5$）。同时，可以验证，不存在路径有着更大的宽度下限。注意样例 3 可能有很多其他解法。



(b)

【数据范围】

本题采用捆绑测试。

对于全部数据，$2 \le N \le 500$，$1 \le W \le 10^6$，$0 \le C_{i,j}, B_{i,j} \le W$。

子任务一（10 分）：所有 $C_{i,j}$ 都相同，所有 $B_{i,j}$ 都相同，$N \le 40$。

子任务二（5 分）：所有 $C_{i,j}$ 都相同，所有 $B_{i,j}$ 都相同。

子任务三（17 分）：$N \le 40$。

子任务四（18 分）：$W = 1$。

子任务五（19 分）：所有 $B_{i,j}$ 都相同。

子任务六（31 分）：无特殊限制。

记忆（memory）

【题目描述】

有 n 株草，其中第 i 株在每一天早晨会长高 i 厘米。

现在是第 0 天的黄昏，你刚刚将所有的草都割到了 0 厘米高。为了更好地维护你的草坪，你制定了一个长度为 m 天的除草计划，并且准备循环执行它，除草计划长成这样：

在计划的第 i 天里，你准备在这一天的黄昏去视察你的草坪，并且将所有高度严格高于 h_i 的草割到高度 h_i。

割草很累，具体而言，每割一株草，你需要消耗 1 点体力。

你想要知道，要是你坚持执行这个计划，你将会在前 k 天中消耗多少体力。

【输入格式】

从文件 ***memory.in*** 中读入数据。

第一行为三个整数 n, m, q ，分别表示草的数目，除草计划的天数和询问的个数。

第二行一行 m 个整数，第 i 个表示除草计划第 i 天的 h_i。

接下来 q 行，一行一个整数 k ，表示询问前 k 天你会消耗多少体力。

【输出格式】

输出到文件 ***memory.out*** 中。

q 行 q 个整数，第 i 个表示第 i 组询问的答案。

【样例 1 输入】

```
3 2 5
3 2
2
3
4
5
6
```

【样例 1 输出】

```
2
4
7
9
12
```

【样例 1 解释】

这里给出除草计划前 6 天的情况:

第一天黄昏，草的高度为 1 2 3 ，不需要割草;

第二天黄昏，草的高度为 2 4 6 ，需要割掉后两株草，消耗体力 2;

第三天黄昏，草的高度为 3 4 5 ，需要割掉后两株草，消耗体力 2;

第四天黄昏，草的高度为 4 5 6 ，需要割掉全部三株草，消耗体力 3;

第五天黄昏，草的高度为 3 4 5 ，需要割掉后两株草，消耗体力 2;

第六天黄昏，草的高度为 4 5 6 ，需要割掉全部三株草，消耗体力 3;

【样例 2 输入】

```
5 5 3
4 7 8 7 4
3
1000000007
1000000000000
```

【样例 2 输出】

```
6
3800000022
3799999999996
```

【样例 3】

见选手下发文件中 memory/ex_memory3.in 与 memory/ex_memory3.ans。

该组数据满足子任务 2 的限制。

【样例 4】

见选手下发文件中 memory/ex_memory4.in 与 memory/ex_memory4.ans。

该组数据满足子任务 5 的限制。

【数据范围】

本题采用捆绑测试。

对于全部数据，$1 \le n, m, q \le 3 \times 10^5, 1 \le k \le 3 \times 10^{12}, 0 \le h_i \le 10^{18}$，保证询问给出的 k 递增。

子任务 1(10 分)： $n, m, q, k \le 5000$。

子任务 2(20 分)： $k \le 10^5$。

子任务 3(10 分)：$n = 1$。
子任务 4(10 分)：$n \leq 10$。
子任务 5(20 分)：$n, m, q \leq 5 \times 10^4$。
子任务 6(30 分)：无特殊约定。