组合数问题（placement）

这是一道提交答案题。

【题目背景】

众所周知，小葱同学擅长计算，尤其擅长计算组合数，但这题不仅和组合数没什么关系，甚至和小葱也没有关系。

【题目描述】

这次我们的主人公是小何同学。众所周知，小何同学非常有钱，他买了 K 台 TPU 来解决 A+B 问题。现在小何同学有 N 个 A+B 问题，第 i 个 A+B 问题在第 j 台 TPU 上计算需要 t_{ij} 的时间。与传统的 A+B 问题不一样的是，这 N 个 A+B 问题之间有 M 个依赖关系，如果问题 i 依赖于问题 j，那么问题 i 必须等待问题 j 计算完毕，并将计算的结果传输到问题 j 所在的 TPU 之后，问题 j 才能开始进行计算。如果问题 i 在第 p 台 TPU 上进行计算，问题 j 在第 q 台机器上进行计算，那么问题 i 的结果传输到问题 j 所需要的时间为 r_{pq}。数据之间的传输是并行的，即同时有多台机器向一台机器传输数据，或一台机器向多台机器发送数据，都并不会影响到数据传输的速度；但我们规定数据的传输是不能转发的，即如果要从第 i 台机器向第 j 台机器传输某个数据，不能先传输到第 k 台机器再传输到第 j 台机器。

虽然小何同学特别有钱，但是小何同学没有多少的时间毕竟他还要去陪妹子，所以现在小何同学希望你来帮他决定每个 A+B 问题分配到哪台机器上计算，使得所有 TPU 的计算时间总和最小或者所有任务完成的时间最小。所谓 TPU 的计算时间，其定义为 TPU 用来计算问题的时间加上所有数据传输的时间之和。所有任务完成时间定义为从第一个计算开始到最后一个计算结束的时间。

虽然小何同学特别有钱，并且小何同学知道你也没有多少时间来做这个题，所以小何同学为了简化问题，对上述任务作出了如下规定：

1、问题的依赖关系之间没有环。

2、任何一个时刻，一台 TPU 只能计算一个问题，且一旦开始这个问题的计算，就不会被打断，会一直计算到这个问题计算完成。

3、如果一台 TPU 此时没有计算任何一个问题，并且存在一个或多个问题已经准备好数据可以计算，那么 TPU 会选择其中编号最小的问题开始计算。

4、一台 TPU 同时进行多个数据传输，且彼此之间互相不会影响速度，计算问题的同时也可以进行数据传输，且彼此之间的速度都不会受到影响。

5、数据不能进行转发，只能直接在相应的机器之间传输。

6、保证 $r_{ii} = 0$。

7、如果一个任务不依赖于其他任务，则该任务所需要的数据已经直接在对应机器上准备好了不需要传输。

在上面的这些条件下，小何同学认为这个问题已经足够简单了，于是他愉快地去找妹子玩耍，并把这个问题交给了你。

【输入格式】

这是一道提交答案题，共有 10 组输入数据，这些数据命名为 ***placement1.in*** ~ ***placement10.in***。

第一行四个整数 N, M, K, op。如果 $op = 1$，则代表你要最小化 TPU 的计算时间总和，否则你需要最小化最后一个完成的任务的完成时间。

接下来 M 行每行两个数 i, j，代表问题 j 依赖于问题 i。

接下来 N 行每行 K 个数，代表 t_{ij}。

接下来 K 行每行 K 个数，代表 r_{ij}。

【输出格式】

对于每组输入数据，你需要提交相应的输出文件 ***placement1.out*** ~ ***placement10.out***。

一行 N 个整数，代表每个任务被分配给哪台机器。

【样例 0 输入】

```
3 3 3 1
1 2
2 3
1 3
1 2 3
2 3 1
3 1 2
0 2 1
2 0 3
1 3 0
```

【样例 0 输出】

```
1 3 2
```

【评分方式】

本题的每个测试点有十个评分标准，如果你的答案小于等于其中 k 个评分标准，那么该测试点你会得到 k 分。

这些标准保存在选手目录的 ***placement*.ans*** 中。

【提示】

为了方便你测试自己的答案，小何同学把妹子甩了然后给你写了一个模拟器。你可以在你的目录下找到一个可执行文件 `simulator`。它的用法为在终端中执行`./simulator <input_file> <output_file>`。其中 `<input_file>` 为输入文件，`<output_file>` 为你的答案。它会告诉你你的分配方案的总计算时间和所有任务的完成时间。

注意：由于陪妹子玩耍降低了小何同学的编程技巧，小何同学不能保证这个模拟器能对不是他提供的输入文件给出正确的结果。

每个答案文件的大小不能超过 `10kB`（事实上，合法的解的大小一定小于这个限制）。