NOI2024 联合省选

第二试

时间：2024 年 3 月 3 日 08:30 ～ 13:00

题目名称	迷宫守卫	重塑时光	最长待机
题目类型	传统型	传统型	传统型
目录	maze	timeline	sleep
可执行文件名	maze	timeline	sleep
输入文件名	maze.in	timeline.in	sleep.in
输出文件名	maze.out	timeline.out	sleep.out
每个测试点时限	0.5 秒	1.5 秒	2.0 秒
内存限制	512 MiB	1024 MiB	1024 MiB
测试点数目	20	20	25
测试点是否等分	是	是	是

提交源程序文件名

对于 C++ 语言	maze.cpp	timeline.cpp	sleep.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. 全国统一评测时采用的机器配置为：Intel(R) Core(TM) i7-4790 CPU @ 3.60GHz，内存 16GB。上述时限以此配置为准。
9. 只提供 Linux 格式附加样例文件。
10. 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。

迷宫守卫（maze）

【题目描述】

Alice 拥有一座迷宫，这座迷宫可以抽象成一棵拥有 2^n 个叶节点的满二叉树，总节点数目为 $(2^{n+1}-1)$，依次编号为 $1 \sim (2^{n+1}-1)$。其中编号为 $2^n \sim (2^{n+1}-1)$ 的是叶节点，编号为 $1 \sim (2^n-1)$ 的是非叶节点，且非叶节点 $1 \le u \le (2^n-1)$ 的左儿子编号为 $2u$，右儿子编号为 $(2u+1)$。

每个非叶节点都有一个石像守卫，初始时，所有石像守卫均在沉睡。唤醒 u 点的石像守卫需要 w_u 的魔力值。

每个叶节点都有一个符文，v 点的符文记作 q_v。**保证** $q_{2^n}, q_{2^n+1}, \cdots, q_{2^{n+1}-1}$ **构成** $1 \sim 2^n$ **的排列**。

探险者初始时持有空序列 Q，从节点 1 出发，按照如下规则行动：

- 到达叶节点 v 时，将 v 点的符文 q_v 添加到序列 Q 的末尾，然后返回父节点。
- 到达非叶节点 u 时：
  - 若该点的石像守卫已被唤醒，则只能先前往左儿子，（从左儿子返回后）再前往右儿子，（从右儿子返回后）最后返回父节点。
  - 若该点的石像守卫在沉睡，可以在以下二者中任选其一：
    - 先前往左儿子，再前往右儿子，最后返回父节点。
    - 先前往右儿子，再前往左儿子，最后返回父节点。

返回节点 1 时，探险结束。可以证明，探险者一定访问每个叶节点各一次，故此时 Q 的长度为 2^n。

探险者 Bob 准备进入迷宫，他希望探险结束时的 Q 的字典序越小越好，与之相对，Alice 希望 Q 的字典序越大越好。

在 Bob 出发之前，Alice 可以选择一些魔力值花费之和不超过 K 的石像守卫，并唤醒它们。Bob 出发时，他能够知道 Alice 唤醒了哪些神像。若**双方都采取最优策略**，求序列 Q 的最终取值。

对于两个长度为 2^n 的序列 Q_1, Q_2，称 Q_1 字典序小于 Q_2 当且仅当以下条件成立：

- $\exists i \in [1, 2^n]$ 满足以下两个条件：
  - $\forall 1 \le j < i$，$Q_{1,j} = Q_{2,j}$；
  - $Q_{1,i} < Q_{2,i}$。

【输入格式】

从文件 ***maze.in*** 中读入数据。

本题有多组测试数据。输入的第一行包含一个正整数 T，表示测试数据组数。

接下来依次 T 组测试数据。对于每组测试数据：

- 第一行两个整数 n, K 表示迷宫规模和 Alice 可用于唤醒石像守卫的魔力值上限。
- 第二行 $(2^n - 1)$ 个整数 $w_1, w_2, \cdots, w_{2^n-1}$ 表示唤醒各个石像守卫耗费的魔力值。
- 第三行 2^n 个整数 $q_{2^n}, q_{2^n+1}, \cdots, q_{2^{n+1}-1}$ 表示各个叶节点上的符文。

【输出格式】

输出到文件 ***maze.out*** 中。

对于每组数据，输出一行 2^n 个整数 $Q_1, Q_2, \cdots, Q_{2^n}$，表示双方都采取最优策略的情况下，序列 Q 的最终取值。

【样例 1 输入】

```
3
1 0
1
2 1
1 1
1
2 1
3 3
3 2 1 2 1 2 1
4 2 6 3 7 1 5 8
```

【样例 1 输出】

```
1 2
2 1
2 4 6 3 5 8 7 1
```

【样例 1 解释】

- 第一组数据中，Alice 无法唤醒石像守卫，Bob 可以选择先访问叶节点 3，再访问叶节点 2，得 $Q = \{1, 2\}$。
- 第二组数据中，Alice 可以唤醒节点 1 的石像守卫，Bob 只能先访问叶节点 2，再访问叶节点 3，得 $Q = \{2, 1\}$。
- 第三组数据中，Alice 的最优策略是唤醒节点 $5, 6$ 的石像守卫。

【样例 2】

见选手目录下的 ***maze/maze2.in*** 与 ***maze/maze2.ans***。
该组数据满足特殊性质 A。

【样例 3】

见选手目录下的 ***maze/maze3.in*** 与 ***maze/maze3.ans***。
该组数据满足特殊性质 B。

【样例 4】

见选手目录下的 ***maze/maze4.in*** 与 ***maze/maze4.ans***。

【样例 5】

见选手目录下的 ***maze/maze5.in*** 与 ***maze/maze5.ans***。

【子任务】

设 $\sum 2^n$ 表示单个测试点中所有测试数据的 2^n 的和。对于所有测试数据，保证

- $1 \le T \le 100$；
- $1 \le n \le 16$，$1 \le \sum 2^n \le 10^5$；
- $0 \le K \le 10^{12}$；
- $\forall 1 \le u \le (2^n - 1)$，$0 \le w_u \le 10^{12}$；
- $q_{2^n}, q_{2^n+1}, \cdots, q_{2^{n+1}-1}$ 构成 $1 \sim 2^n$ 的排列。

测试点编号	$n \le$	$\sum 2^n \le$	特殊性质
$1 \sim 5$	4	80	无
6	6	200	A
$7 \sim 8$			B
$9 \sim 10$			无
11	11	4000	A
$12 \sim 13$			B
$14 \sim 15$			无
16	16	10^5	A
$17 \sim 18$			B
$19 \sim 20$			无

特殊性质 A：$\forall 2^n \le v \le (2^{n+1} - 1)$，$q_v = (2^{n+1} - v)$。
特殊性质 B：$\forall 1 \le u \le (2^n - 1)$，$w_u = 1$。

重塑时光（timeline）

【题目描述】

小 T 正在研究某段时间中所发生的事件。经观测，有 n 个编号为 $1 \sim n$ 的事件在这段时间内按顺序依次发生，第 i 个发生的是事件 p_i。这个描述事件发生顺序的排列 p 可称为这段时间的**时间线**。

突然，邪恶生物小 S 攻击了这条时间线，将这 n 个事件的发生顺序 p 变为了在所有长为 n 的排列中等概率随机选取的一个排列。不仅如此，小 S 还用剪刀把时间线剪断，通过进行 k 次操作，将排列 p 分割成了 $(k+1)$ 段。

具体而言，在小 S 进行第 i 次操作时，排列 p 和之前所有插入的剪断点构成了一个长度为 $(n+i-1)$ 的序列。该序列包括所有相邻元素之间和序列开头、末尾处共有 $(n+i)$ 个插入位置。小 S 将从这些插入位置中等概率随机选取一个位置，插入一个新的剪断点。最后，小 S 从最终被插入的 k 个剪断点处把序列剪开，将排列 p 分割成了 $(k+1)$ 段序列。这 $(k+1)$ 段序列中可能有空序列。

为了拯救这条即将毁灭的时间线，小 T 决定把这 $(k+1)$ 段序列按某种顺序重新拼接成一个长度为 n 的排列，形成一条新的时间线。不过，由于事件之间存在一定的逻辑关系，事件的发生时间之间也存在一些先后顺序要求。经研究，共存在 m 条先后顺序要求 (u, v)，要求事件 u 的发生时间必须在事件 v 之前。也就是说，u 在时间线中的出现位置必须在 v 之前。

请你设计程序，计算有多大的概率，存在至少一种重新排列这 $(k+1)$ 段序列，并将其重新拼接为一条新的时间线的方案，能够使所有的 m 条事件发生时间之间的先后顺序要求都得到满足。

为了避免精度误差，请你输出答案对 10^9+7 取模的结果。形式化地，可以证明答案可被表示为一最简分数 $\frac{p}{q}$，请你输出一个 x 满足 $0 \le x < 10^9+7$ 且 $qx \equiv p \bmod (10^9+7)$。可以证明在题目条件下这样的 x 总是存在。

【输入格式】

从文件 ***timeline.in*** 中读入数据。

第一行三个整数 n, m, k，分别描述事件的个数，事件之间先后顺序的条数以及小 S 进行的剪断操作次数。

接下来 m 行，每行两个整数 u, v，表示一条事件发生时间的先后顺序要求。

【输出格式】

输出到文件 ***timeline.out*** 中。

输出一行一个整数，表示所求答案。

【样例 1 输入】

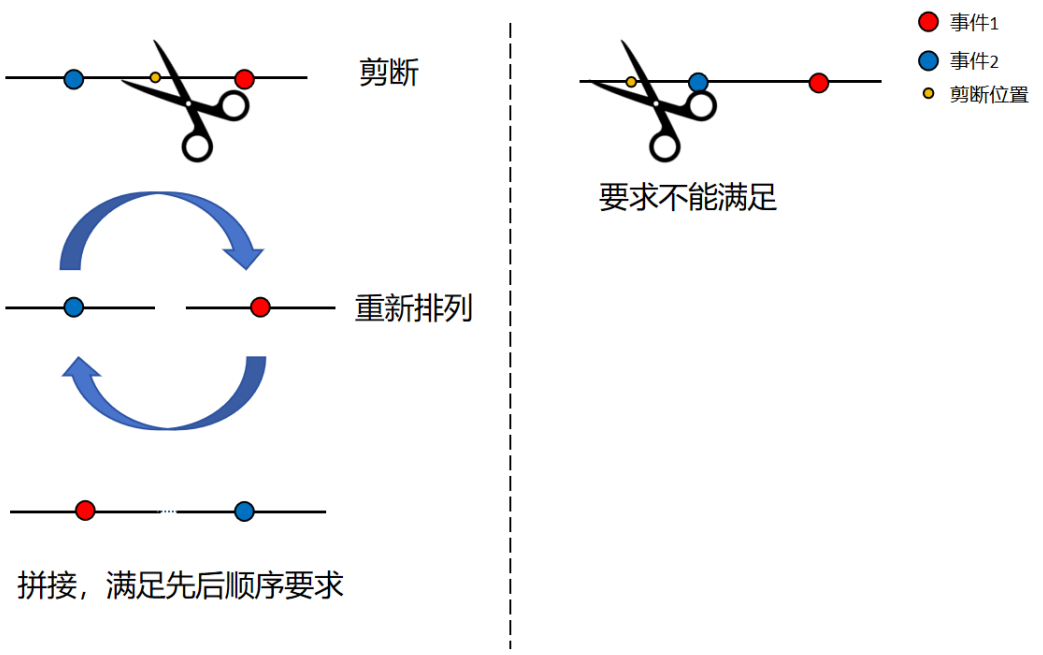
```
2 1 1
1 2
```

【样例 1 输出】

```
666666672
```

【样例 1 解释】

假如事件 1 的发生时间早于事件 2，那么无论怎样拼接都是可行方案，一定可以满足要求。否则，只有剪断时间线的位置位于事件 1 和事件 2 的发生时间之间，才能满足要求。答案为 $\frac{1}{2}+\frac{1}{2}\times\frac{1}{3}=\frac{2}{3}$。



【样例 2 输入】

```
3 0 2
```

【样例 2 输出】

```
1
```

【样例 2 解释】

没有任何事件发生时间之间的先后顺序要求，因此无论怎样拼接都是可行的方案，答案为 1。

【样例 3 输入】

```
4 4 4
1 2
1 3
1 4
2 4
```

【样例 3 输出】

```
937500007
```

【样例 4】

见选手目录下的 ***timeline/timeline4.in*** 与 ***timeline/timeline4.ans***。

【样例 5】

见选手目录下的 ***timeline/timeline5.in*** 与 ***timeline/timeline5.ans***。
该组样例满足数据范围中的特殊性质 B。

【样例 6】

见选手目录下的 ***timeline/timeline6.in*** 与 ***timeline/timeline6.ans***。
该组样例满足数据范围中的特殊性质 A。

【样例 7】

见选手目录下的 ***timeline/timeline7.in*** 与 ***timeline/timeline7.ans***。

【子任务】

对于所有测试数据，

- $1 \le n \le 15$，
- $0 \le m \le \frac{n(n-1)}{2}$，$0 \le k \le n$，
- $1 \le u < v \le n$，保证不存在两对 (u, v) 完全相同。

测试点	n	m	k	特殊性质
1	≤ 3	$= n-1$	$= 0$	B
2	≤ 5	$\le \frac{n(n-1)}{2}$	$\le n$	无
3, 4	≤ 14	$= n-1$		B
5			$= 0$	A
6			$\le n$	
7		$= 0$		无
8		$= \frac{n(n-1)}{2}$		
9, 10	≤ 9	≤ 15		
11	≤ 13	$\le \frac{n(n-1)}{2}$	$= 0$	
12			$\le n$	
$13 \sim 17$	≤ 14			
$18 \sim 20$	≤ 15			

特殊性质 A：对于每个事件 x，至多存在一条先后顺序 (u, v) 使得 $v = x$。
特殊性质 B：对于所有先后顺序 (u, v)，均满足 $u = 1$。

最长待机（sleep）

【题目描述】

精灵程序员小 ω 和小 $\aleph$ 拥有无限的寿命，因此在写代码之余，它们经常玩一些对抗游戏来打发时间。尽管如此，时间还是太多，于是它们发明了一款专用于消磨时间的游戏：最长待机。

为了了解最长待机的规则，首先要了解精灵们使用的编程语言 Sleep++ 的规则：

- 程序由 n 个函数组成，第 $i(1 \le i \le n)$ 个函数具有种类 e_i 和子函数编号序列 $Q_i = (Q_{i,1}, Q_{i,2}, \cdots, Q_{i,l_i})$。$Q_i$ 可以为空，此时 l_i 为 0。
- n 以及所有的 e_i 和 Q_i 可以由程序员任意给出，但它们需要满足以下所有条件：
  - $n \ge 1$；
  - $\forall 1 \le i \le n$，$e_i \in \{0, 1\}$；
  - $\forall 1 \le i \le n$，Q_i 中元素两两不同且均为 $[i+1, n]$ 中的整数；
  - $\forall 2 \le j \le n$，恰好有一个 $Q_i(1 \le i \le n)$ 包含了 j。
- 调用函数 $i(1 \le i \le n)$ 时，按顺序执行如下操作：
  - 若 $e_i = 0$，令变量 r_i 为 1；否则程序员需要立即为 r_i 输入一个正整数值。
  - 若 Q_i 为空，程序等待 r_i 秒；否则重复以下操作 r_i 次：
    - 按顺序调用编号为 $Q_{i,1}, Q_{i,2}, \cdots, Q_{i,l_i}$ 的函数。
- 若一个种类为 1 的函数 j 被调用多次，则其每次调用都需要输入 r_j。
- 我们认为，在函数调用中，除了“等待 r 秒”之外的操作不消耗任何时间，即函数调用、运行和输入都在瞬间完成。因此，一个时刻内程序员可能输入多个数。

可以证明，调用任意一个 Sleep++ 程序的任意一个函数，无论如何设定输入，消耗的时间总是有限的。

“最长待机”的游戏规则如下：

- 小 ω 和小 $\aleph$ 准备好各自的 Sleep++ 程序并选择各自程序中的一个函数。它们互相知晓对方程序的结构以及选择的函数。
- 在时刻 0，小 ω 和小 $\aleph$ 同时调用自己选择的函数，游戏开始。
- 在时刻 t（$t \ge 0$），双方可以看到对方在时刻 0 至 $(t-1)$ 输入的所有数字，并相应调整自己在时刻 t 输入的数字，但双方无法得知对方在时刻 t 输入的数字。
- 函数调用先结束的一方输掉游戏，另一方胜利。两个调用同时结束算作平局。

小 ω 和小 $\aleph$ 都是绝顶聪明的，在它们眼中，如果有一方存在必胜策略，那么这局游戏是不公平的。换言之，双方都不存在必胜策略的游戏是公平的。

小 ω 写了一个 n 个函数的 Sleep++ 程序并进行了 m 次操作，操作有以下两种：

- 操作一：给出 k，将 e_k 修改为 $(1 - e_k)$；
- 操作二：给出 k，与小 $\aleph$ 玩一局“最长待机”，开始时小 ω 会调用自己的函数 k。

小 ℵ 信奉极简主义，它希望对于每一局游戏设计出函数个数最少的程序，使得选择其中某个函数能让这局游戏是公平的。你能帮它求出最少所需的函数个数吗？

可以证明，小 ℵ 总是能设计一个程序并选择其中一个函数，使得游戏是公平的。

【输入格式】

从文件 ***sleep.in*** 中读入数据。

输入的第一行包含两个正整数 n, m，表示小 ω 的程序中函数的个数以及操作次数。

接下来 n 行，第 i 行若干个整数，描述小 ω 程序中的函数 i：

- 前两个整数 e_i, l_i 表示函数种类和子函数编号序列长度；
- 接下来 l_i 个整数 $Q_{i,1}, Q_{i,2}, \cdots, Q_{i,l_i}$ 描述子函数编号序列。

接下来 m 行，第 j 行两个整数 o_j, k_j 描述一次操作，其中 $o_j = 1$ 表示操作一，$o_j = 2$ 表示操作二。

【输出格式】

输出到文件 ***sleep.out*** 中。

对于每个操作二输出一行一个整数，表示小 ℵ 的程序中最少所需的函数个数。

【样例 1 输入】

```
3 6
0 2 2 3
0 0
0 0
2 1
1 3
2 1
1 3
1 2
2 1
```

【样例 1 输出】

```
3
3
1
```

【样例 1 解释】

- 对于前两次游戏，小 ℵ 可以给出与小 ω 完全一致的程序并在游戏开始时调用函数 1。可以证明不存在函数个数更少的方案。
- 对于第三次游戏，小 ℵ 可以给出一个仅包含一个种类为 1 的函数的程序，并在游戏开始时调用函数 1。
  - 在时刻 0，小 ω 输入其程序中的 r_2，小 ℵ 输入其程序中的 r_1。
    - 注意：r 变量在小 ω 和小 ℵ 的程序之间是独立的，不会互相影响。
  - 输入完成后，小 ω 的程序在时刻 (r_2+1) 结束，小 ℵ 的程序在时刻 r_1 结束。
  - 由于两人在时刻 0 互不知道对方的决策，不能保证 (r_2+1) 和 r_1 的大小关系，故双方均不存在必胜策略，这局游戏是公平的。

【样例 2】

见选手目录下的 ***sleep/sleep2.in*** 与 ***sleep/sleep2.ans***。
该组数据满足特殊性质 AD。

【样例 3】

见选手目录下的 ***sleep/sleep3.in*** 与 ***sleep/sleep3.ans***。
该组数据满足特殊性质 BD。

【样例 4】

见选手目录下的 ***sleep/sleep4.in*** 与 ***sleep/sleep4.ans***。
该组数据满足特殊性质 D。

【样例 5】

见选手目录下的 ***sleep/sleep5.in*** 与 ***sleep/sleep5.ans***。
该组数据满足特殊性质 C。

【子任务】

对于所有测试数据，

- $1 \le n \le 5 \times 10^5$，$1 \le m \le 2 \times 10^5$；
- $\forall 1 \le i \le n$，$e_i \in \{0,1\}$，$0 \le l_i < n$；
- $\forall 1 \le i \le n, 1 \le j \le l_i$，$i < Q_{i,j} \le n$；
- $\forall 1 \le i \le n, 1 \le p < q \le l_i$，$Q_{i,p} \ne Q_{i,q}$；
- $\forall 2 \le j \le n$，恰好有一个 $Q_i (1 \le i \le n)$ 包含了 j；

- $\forall 1 \le j \le m$，$1 \le o_j \le 2$，$1 \le k_j \le n$。

测试点编号	$n \le$	$m \le$	特殊性质
$1 \sim 2$	3	24	无
3	80	400	AD
4			BD
$5 \sim 6$			D
7	3×10^5	10^5	AD
8			BD
$9 \sim 10$			D
11			A
12			BC
13			B
$14 \sim 15$			C
$16 \sim 17$			无
$18 \sim 19$	5×10^5	2×10^5	A
20			BC
21			B
$22 \sim 23$			C
$24 \sim 25$			无

特殊性质 A：保证

- 任意时刻 e_1 均为 0；
- $\forall 2 \le i \le n$，$l_i \le 1$；
- 操作二的 k 均为 1。

特殊性质 B：保证

- 操作二的 k 满足当时的 e_k 为 1。

特殊性质 C：保证

- $\forall 2 \le i \le n$，$i \in Q_{\lfloor \frac{i}{2} \rfloor}$；
- $\forall 1 \le i \le n$，序列 Q_i 单调递增。

特殊性质 D：保证

- 操作二不超过 10 个；
- 操作二的 k 均为 1。