IOI 2018 中国国家队选拔赛暨精英赛

CTSC 2018

第二试

时间：2018 年 5 月 9 日 08:30 ~ 13:30

题目名称	混合果汁	字典树	组合数问题
题目类型	传统型	传统型	提交答案型
目录	juice	trie	placement
可执行文件名	juice	trie	N/A
输入文件名	juice.in	trie.in	placement*.in
输出文件名	juice.out	trie.out	placement*.out
每个测试点时限	2.0 秒	5.0 秒	N/A
内存限制	512 MB	512 MB	N/A
测试点数目	20	20	10
每个测试点分值	5	5	10

提交源程序文件名

对于 C++ 语言	juice.cpp	trie.cpp	N/A
对于 C 语言	juice.c	trie.c	N/A
对于 Pascal 语言	juice.pas	trie.pas	N/A

编译选项

对于 C++ 语言	-O2 -lm	-O2 -lm	N/A
对于 C 语言	-O2 -lm	-O2 -lm	N/A
对于 Pascal 语言	-O2	-O2	N/A

混合果汁（juice）

【题目描述】

小 R 热衷于做黑暗料理，尤其是混合果汁。

商店里有 n 种果汁，编号为 $0,1,2,\ldots,n-1$。i 号果汁的美味度是 d_i，每升价格为 p_i。小 R 在制作混合果汁时，还有一些特殊的规定，即在一瓶混合果汁中，i 号果汁最多只能添加 l_i 升。

现在有 m 个小朋友过来找小 R 要混合果汁喝，他们都希望小 R 用商店里的果汁制作成一瓶混合果汁。其中，第 j 个小朋友希望他得到的混合果汁总价格不大于 g_j，体积不小于 L_j。在上述这些限制条件下，小朋友们还希望混合果汁的美味度尽可能地高，一瓶混合果汁的美味度等于所有参与混合的果汁的美味度的最小值。请你计算每个小朋友能喝到的最美味的混合果汁的美味度。

【输入格式】

从文件 ***juice.in*** 中读入数据。

输入第一行包含两个正整数 n,m，表示果汁的种数和小朋友的数量。

接下来 n 行，每行三个正整数 d_i,p_i,l_i，表示 i 号果汁的美味度为 d_i，每升价格为 p_i，在一瓶果汁中的添加上限为 l_i。

接下来 m 行依次描述所有小朋友：每行两个数正整数 g_j,L_j 描述一个小朋友，表示他最多能支付 g_j 元钱，他想要至少 L_j 升果汁。

【输出格式】

输出到文件 ***juice.out*** 中。

对于所有小朋友依次输出：对于每个小朋友，输出一行，包含一个整数，表示他能喝到的最美味的混合果汁的美味度。如果无法满足他的需求，则输出 -1。

【样例 1 输入】

```
3 4
1 3 5
2 1 3
3 2 5
6 3
5 3
10 10
20 10
```

【样例 1 输出】

```
3
2
-1
1
```

【样例 2】

见选手目录下的 ***juice/juice2.in*** 与 ***juice/juice2.ans***。

【样例 3】

见选手目录下的 ***juice/juice3.in*** 与 ***juice/juice3.ans***。

【子任务】

对于所有的测试数据，保证 $n, m \le 100000$，$1 \le d_i, p_i, l_i \le 10^5$，$1 \le g_j, L_j \le 10^{18}$。

测试点编号	$n=$	$m=$	其他限制
1-3	10	10	无
4-6	500	500	
7-9	5000	5000	
10-12	100000	100000	$p_i = 1$
13-15			$l_i = 1$
16-20			无

字典树（trie）

【题目描述】

Access Globe 有若干个递增的正整数序列。他把这些正整数序列中的每个正整数的十进制表示（无前导零）依次写了下来，相邻两个整数之间用逗号`,`隔开。Access Globe 把这个序列视为一个由 0 – 9 的数字和逗号 `,` 组成的字符串，然后用一棵 Trie 树存储这些字符串。你并不需要知道 Trie 树究竟是什么，你只需要知道，Access Globe 得到的 Trie 是一棵以 0 号节点为根的有根树，每条边上都有一个字符，并且从根到每个叶节点的路径上的边上的字符顺次拼接构成的字符串是一个他写下的一个递增的正整数序列。

可爱的小 Tommy 决定篡改这棵 Trie 树。他先将 Trie 上的一些边上的字符删去，然后填上另一些字符。为了不被发现，Tommy 必须保证修改后的 Trie 仍然满足上述性质，即从根到每个叶节点的路径上的边上的字符顺次拼接构成的字符串是一个递增的正整数序列，且每个正整数无前导零。

现在 Tommy 已经删去了一些边上的字符，请你帮他完成“填上字符”的操作。如果有多解，请输出字典序最小的解。

【输入格式】

从文件 ***trie.in*** 中读入数据。

输入文件包含多组数据，整个文件的第一行是一个整数 T，表示数据组数。对于每一组数据：

第一行包含一个长度为 n 的、仅包含 `0` 到 `9`、`,` 和 `?` 的字符串，第 i 个整数表示连接节点 i 的父亲和节点 i 的边上的字符，`?` 表示这条边上的字符已经被删去；

第二行包含 n 个整数，第 i 个整数表示节点 i 的父亲节点 f_i，保证 $0 \le f_i < i$。

【输出格式】

输出到文件 ***trie.out*** 中。

输出 T 行。对于每组数据，输出一个长度为 n 的字符串，表示字典序最小的填写问号的方式中每个点的字符，第 i 个整数表示节点 i 的字符。

如果不存在任何合法的填写方式，请输出 `failed`。

【样例 1 输入】

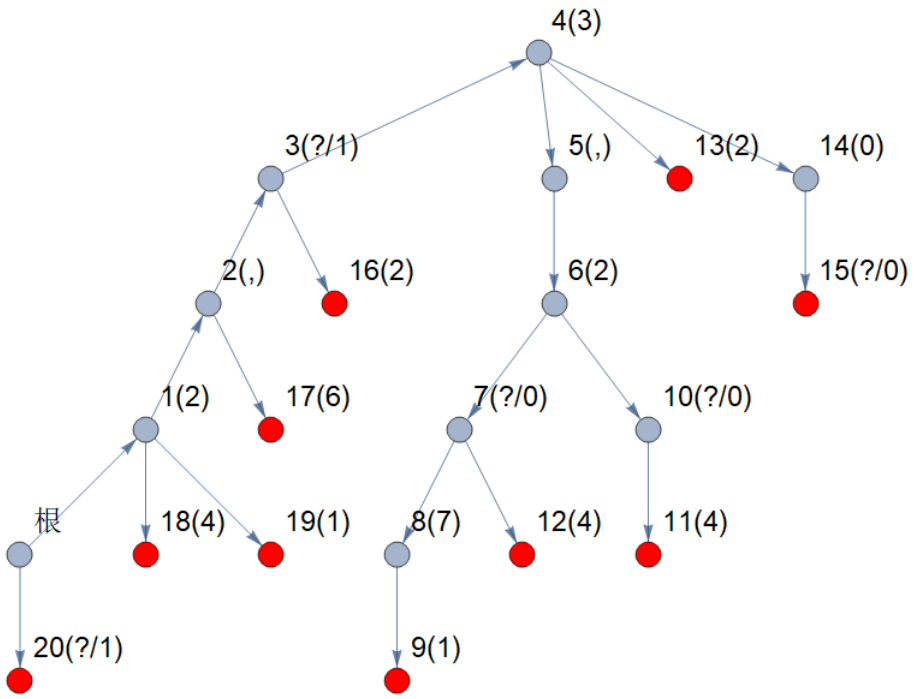
```
1
2,?3,2?71?4420?2641?
0 1 2 3 4 5 6 7 8 6 10 7 4 4 14 3 2 1 1 0
```

【样例 1 输出】

```
2,13,207104420026411
```

【样例 1 解释】

Tommy 填写的 Trie 如下图，红色的节点为所有叶子节点，注意，根节点在左下方。



【样例 2】

见选手目录下的 ***trie/trie2.in*** 与 ***trie/trie2.ans***。

【子任务】

对于 20% 的数据，$T \le 10$，对于每组数据 $n \le 80$ ，? 个数不超过 8；
对于另外 10% 的数据，$T \le 20$ ，对于每组数据 $n \le 80, f_i = i - 1$；
对于另外 20% 的数据，$f_i = i - 1$；
对于另外 10% 的数据，$n \le 80$；
对于所有数据，$T \le 100$ 对于每组数据 $n \le 200$。

【提示】

, 的 ascii 码为 44 ，比所有的数字都要小。

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）。