

A+B Problem

题目描述

输入2个整数，输出它们的和。

输入格式

一行，包括两个整数。

输出格式

一行，包括一个整数

语言及编译选项信息

#	名称	编译器	额外参数	代码长度限制 (B)
0	g++ with std11	g++	-O2 -std=c++11 -DONLINE_JUDGE	65536
1	g++	g++	-O2 -DONLINE_JUDGE	65536
2	gcc with std11	gcc	-O2 -std=c11 -DONLINE_JUDGE	65536
3	gcc	gcc	-O2 -DONLINE_JUDGE	65536
4	java	javac		65536
5	python	python		65536
6	python3	python3		65536

递交历史

#	状态	时间
---	----	----

#	状态	时间
31832	Accepted (/#!/contest/64/detail/31832)	06:24:17 PM 有效递交

1

递交答案 (剩余次数: 31)

语言和编译选项

g++ with std11

▼

1

递交评测

文件请拖入编辑器中，或 上传文件

k叉树

时间限制：1.0 秒

空间限制：512 MB



题目描述

有一个图，一共有 m 条边，节点编号从 0 到 m 。现在只知道 m 条边和一个整数 k 。

如果该图构成一棵树，则尝试找出一个节点满足：以该节点为根，所有节点的儿子数都不超过 k 。

如果给定的 m 条边无法构成一棵树，则输出 It's not a tree!

如果不存在这样的节点作为根，则输出 No such a node!

否则输出一个节点的编号，表示以该节点为根，所有节点的儿子数都不超过 k ，如果同时存在多个节点满足条件，则输出编号最小的节点编号。

输入格式

从标准输入读入数据。

输入的第一行包含两个正整数 m 和 k ，保证 $m \leq 10^5$ ， $k \leq 10^5$ 。

输入的接下来 m 行包含两个整数 a, b ，表示 a 和 b 相连，保证 $0 \leq a, b \leq m$ 。

输出格式

输出到标准输出。

输出一个字符串或者一个整数表示对应的答案。

样例1输入

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

样例1输出

```
2
```

样例2输入

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

样例2输出

```
It's not a tree!
```

子任务

- 子任务一(20分): 保证有解, 且答案为 0 或 1。
- 子任务二(20分): 保证图构成一棵树。
- 子任务三(15分): 保证 $k = 1$ 。
- 子任务四(20分): 保证有解。
- 子任务五(25分): 无特殊情况。

语言及编译选项信息

#	名称	编译器	额外参数	代码长度限制 (B)
0	g++ with std14	g++	-O2 -std=c++14 -DONLINE_JUDGE	65536
1	g++ with std98	g++	-O2 -std=c++98 -DONLINE_JUDGE	65536
2	gcc with std11	gcc	-O2 -std=c11 -DONLINE_JUDGE	65536
3	gcc with std99	gcc	-O2 -std=c99 -DONLINE_JUDGE	65536
4	java	javac		65536
5	python	python		65536
6	python3	python3		65536

递交历史

#	状态	时间
32138	Accepted (/#!/contest/64/detail/32138)	07:22:12 PM 有效递交
32055	Wrong Answer (/#!/contest/64/detail/32055)	07:06:51 PM
32046	Wrong Answer (/#!/contest/64/detail/32046)	07:05:20 PM
31986	Time Limit Exceeded (/#!/contest/64/detail/31986)	06:53:44 PM
31925	Time Limit Exceeded (/#!/contest/64/detail/31925)	06:43:47 PM
		1

递交答案 (剩余次数: 27)

语言和编译选项

g++ with std14

1

递交评测

文件请拖入编辑器中, 或

上传文件

租约机制

时间限制：2.0 秒

空间限制：512 MB

相关文件：题目目录



题目描述

鸽鸽用一台土豆机器搭建了自己的博客，一开始只有几百人访问，一台土豆机器的单机系统工作非常流畅。现在有几百万人访问，一台土豆机器无法满足性能需求，给用户的体验很差。鸽鸽准备用多台土豆机器搭建一个分布式土豆系统来提高性能。

不同的用户实际访问的机器是不同的，但是效果必须得是相同。鸽鸽搭建系统时，面临的第一个问题就是，某台机器修改数据后，如何让所有的机器也获取到最新数据。一个朴素的想法，每台机器都存储一份完整的数据，任意一台机器更新数据时都将最新的数据广播给其他机器，让所有机器同时更新。但是这个办法很难保证所有机器都同时更新数据。可能第一台机器更新了数据，第二台机器还没来得及更新数据，在这期间有两个用户分别通过第一台和第二台机器来访问该数据。在这种场景中，两个用户得到的数据就会不一样，一个是新数据，一个是旧数据，这就造成了数据不一致。同时这种方法需要广播数据，性能也较低。鸽鸽想了想，决定只让一台机器作为中心节点可以更新数据，其他机器作为辅助节点可以访问但是无法更新数据。

鸽鸽的这个分布式系统中，有一个中心节点，存储、维护数据。系统中的其他节点通过访问中心节点读取、修改其上的数据。除了中心节点，其他节点初始化是没有数据的。当用户访问到某个节点，请求读取数据时，如果它没有数据就会从中心节点读取数据，然后返回给用户。如果每次用户访问数据，都要从中心节点读取数据，性能比单机系统还差。所以从中心节点获取数据的同时，会获取一个租约时间，**如果当前时间不超过（不大于）租约，就将租约和数据都保存下来**。下次用户访问时，如果**当前时间没有超过租约（租约没到期）**，就**直接返回数据**；否则，先清空现有数据，再**从中心节点读取数据**。聪明的你，肯定想问，如果租约没到期直接返回数据，这个时候中心节点已经更新了数据，不也会出现数据不一致吗？你能想到这个问题，鸽鸽也能想到。所以中心节点在所有节点租约到期前，不会更新数据。中心节点会阻塞所有更新请求，将这些更新请求都按时间顺序保存下来，等**所有租约全部到期后（大于最晚租约的最小时间）**，再一一更新。聪明的你，肯定又想问，那如果一直有其他节点从中心节点获取更晚的租约，那不是会一直无法更新数据？你能想到这个问题，鸽鸽也能想到。所以中心节点在处理完所有更新数据请求前，发送给任何节点的租约只会是**当前已有的最晚租约**，不会更晚。当用户访问某个节点，请求更新数据时，它会把这个请求转发给中心节点。因为**只有中心节点才能更新数据，其他节点只是缓存中心节点的数据**。鸽鸽给自己的这个方法取了个名字，叫租约协议。

再叙述一遍租约协议的读写流程：

- 用户读数据：判断数据是否已经存在本地且租约未到期。是，直接返回本地的数据；否，向中心节点发送读取数据和租约请求，读取成功后返回数据，如果租约没到期，则在本地缓存数据。
- 用户写数据：被访问节点向中心节点发出写数据请求。中心节点收到写数据请求后，阻塞缓存下来，同时所有新的读数据请求只发放已有最晚的租约。中心节点等待所有租约到期后，一一处理缓存的写请求。

鸽鸽的分布式土豆系统非常神奇，保证时间都是一样的，网络也是永远健康良好的，其他的因素也都是正常可靠的。

给定 n 台机器，编号从 1 到 n 。其中，1 号机器是中心节点。时间从 0 开始，租约时长为 k 。假设在 a 时间，向中心节点发出读请求，如果没有需要处理的写请求时，中心节点返回的租约一定是 $a + k$ ，否则返回已有的最晚租约。中心节点处理一个写请求的时间为 d 。假设在 a 时间一共有 D 个写请求需要处理，在 $a + Dd$ 时间处理完成。**读操作会立即返回结果，写操作有执行延迟。**

现在有 m 个用户请求， $R\ t\ x$ 表示在 t 时间向 x 节点发出读数据请求， $W\ t\ x$ 表示在 t 时间向 x 节点发出写数据请求。对于每个 $R\ t\ x$ 请求，输出 x 对应的操作： RWB 表示从中心节点读取数据和租约，再保存在本地，然后返回给用户； RB 表示从中心节点读取数据和租约，发现租约到期，不保存在本地，然后返回给用户； B 表示直接从本地读取数据返回给用户。**保证对于同一时间，可能会有多个读请求，但是只会有一个写请求。如果同一时间，同时有读写请求，先处理写请求。保证用户只会直接访问非中心节点。**

输入格式

从标准输入读入数据。

输入的第一行包含四个正整数 n, m, k, d ，保证 $n \leq 10^5$ ， $m \leq 10^6$ ， $k \leq 10^2$ ， $d \leq 10^3$ 。

输入的接下来 m 行，每行一个字符、两个整数，用空格隔开，一定是 $R\ t\ x$ 或者 $W\ t\ x$ 。保证操作是按时间顺序，前面任意操作的时间一定不超过后面任意操作的时间。保证所有时间 t 不超过 10^9 。

输出格式

输出到标准输出。

输出若干行，每行一个字符串表示对应读操作的响应。

样例1输入

```
3 22 10 3
R 0 2
R 0 3
R 1 2
R 1 3
R 10 2
R 10 3
R 11 2
W 12 2
R 20 3
R 21 3
R 24 2
R 24 3
R 25 2
R 25 3
W 40 2
R 43 2
R 43 3
W 43 2
R 46 2
R 46 3
R 56 2
R 56 3
```

样例1输出

RWB
 RWB
 B
 B
 B
 B
 RWB
 RWB
 B
 RB
 RB
 RWB
 RWB
 RB
 RB
 RWB
 RWB
 B
 B

样例1解释

有 3 个节点，22 个操作，租约时长为 10，处理一个写操作时间 d 为 3。

0 时间，2 和 3 节点数据为空，用户读数据，需要从中心节点获取数据和租约，租约为 $0 + 10 = 10$ ，当前时间未超过租约，所以需要执行 RWB 操作。

1 时间，2 和 3 节点缓存了数据，租约为 10，没有超过租约，用户读数据，直接返回，所以需要执行 B 操作。

10 时间，2 和 3 节点缓存了数据，租约为 10，没有超过租约，用户读数据，直接返回，所以需要执行 B 操作。

11 时间，2 节点缓存了数据，租约为 10，超过租约，用户读数据，需要重新从中心节点获取数据和租约，租约为 $11 + 10 = 21$ ，当前时间未超过租约，所以需要执行 RWB 操作。

12 时间，2 节点向中心节点发送写请求，因为还有租约未超时，所以被阻塞保存下来，将在时间 22 进行。

20 时间，3 节点缓存了数据，租约为 10，超过租约，用户读数据，需要重新从中心节点获取数据和租约，因为有写请求未处理完，所以租约为已有最晚租约 21，当前时间未超过租约，所以需要执行 RWB 操作。

21 时间，3 节点缓存了数据，租约为 21，没有超过租约，用户读数据，直接返回，所以需要执行 B 操作。

22 时间，最晚租约为 21，所有租约过期，中心节点开始处理写请求。

24 时间，2 和 3 节点缓存了数据，租约为 21，超过租约，用户读数据，需要重新从中心节点获取数据和租约，因为有写请求未处理完，所以租约为已有最晚租约 21，当前时间超过租约，所以需要执行 RB 操作。

25 时间，2 和 3 节点缓存了数据，租约为 21，超过租约，用户读数据，需要重新从中心节点获取数据和租约，因为所有写请求已处理完，所以租约为 $25 + 10 = 35$ ，当前时间超过租约，所以需要执行 RWB 操作。

40 时间，2 节点向中心节点发送写请求，最晚租约为 35，所有租约过期，中心节点开始处理写请求。

43 时间，2 节点向中心节点发送写请求，最晚租约为 35，所有租约过期，中心节点开始处理写请求。

43 时间，2 和 3 节点缓存了数据，租约为 35，超过租约，用户读数据，需要重新从中心节点获取数据和租约，因为有写请求未处理完，所以租约为已有最晚租约 35，当前时间超过租约，所以需要执行 RB 操作。

46 时间，2 和 3 节点缓存了数据，租约为 35，超过租约，用户读数据，需要重新从中心节点获取数据和租约，因为所有写请求已处理完，所以租约为 $46 + 10 = 56$ ，当前时间超过租约，所以需要执行 RWB 操作。

56 时间，2 和 3 节点缓存了数据，租约为 56，没有超过租约，用户读数据，直接返回，所以需要执行 B 操作。

子任务

测试点编号	$n \leq$	$m \leq$	$k \leq$	$d \leq$	操作说明	分数
1	2	10^2	10^2	10^3	无	10
2	2	10^2	10^2	10^3	无	10
3	10^2	10^3	10^2	10^3	中心节点修改数据期间不会有用户访问	10
4	10^2	10^3	10^2	10^3	中心节点修改数据期间不会有用户访问	10
5	10^3	10^4	10^2	10^3	没有修改请求	10
6	10^3	10^4	10^2	10^3	没有修改请求	10
7	10^4	10^5	10^2	10^3	无	10
8	10^4	10^5	10^2	10^3	无	10
9	10^5	10^6	10^2	10^3	无	10
10	10^5	10^6	10^2	10^3	无	10

语言及编译选项信息

#	名称	编译器	额外参数	代码长度限制 (B)
0	g++ with std14	g++	-O2 -std=c++14 -DONLINE_JUDGE	65536
1	g++ with std98	g++	-O2 -std=c++98 -DONLINE_JUDGE	65536
2	gcc with std11	gcc	-O2 -std=c11 -DONLINE_JUDGE	65536
3	gcc with std99	gcc	-O2 -std=c99 -DONLINE_JUDGE	65536

#	名称	编译器	额外参数	代码长度限制 (B)
4	java	javac		65536
5	python	python		65536
6	python3	python3		65536

递交历史

#	状态	时间
32191	<div>Wrong Answer</div> (/#!/contest/64/detail/32191)	07:32:20 PM <div>有效递交</div>

1

递交答案 (剩余次数: 31)

语言和编译选项

g++ with std14



1

递交评测

文件请拖入编辑器中，或

上传文件

初等数论

时间限制: 2.0 秒

空间限制: 512 MiB



题目描述

定义 $f(i, R, P)$ 为满足 $a^R \times (b^2 + b)^i = b^i \pmod{P}$ 且 $0 \leq a, b < P$ 的 (a, b) 数对的个数。

一共有 Q 组询问, 每组询问中输入 R, P, N, K , 要求计算出

$\sum_{i=1}^N i^K \times f(i, R, P) \pmod{998,244,353}$ 的值。

输入格式

从标准输入读入数据。

输入的第一行包含一个正整数 Q , 表示数据组数, 保证 $1 \leq Q \leq 10$ 。

输入的接下来 Q 行每行包含四个非负整数 R, P, N, K , 保证 $1 \leq R \leq 3, 2 \leq P \leq 998,244,353, 1 \leq N < P, 0 \leq K < 10^5$, 且保证 P 为质数。

输出格式

输出到标准输出。

输出一共有 Q 行, 每行对应一个询问的答案。

样例1输入

```
3
2 7 5 3
1 17 9 4
3 29 23 9
```

样例1输出

```
2907
490656
480228738
```

子任务

保证所有数据满足 $1 \leq Q \leq 10, 1 \leq R \leq 3, 2 \leq P \leq 998, 244, 353, N < P, 0 \leq K < 10^5$, 且保证 P 为质数。

- 子任务一(11分): 保证 $P \leq 300$
- 子任务二(14分): 保证 $R \leq 2, P \leq 5000$
- 子任务三(15分): 保证 $K = 0$ 且如果 $R = 3$, 那么 P 满足 $P \bmod 3 \neq 1$ 。
- 子任务四(17分): 保证 $R \leq 1$
- 子任务五(19分): 保证 $K = 0$
- 子任务六(24分): 无特殊情况。

语言及编译选项信息

#	名称	编译器	额外参数	代码长度限制 (B)
0	g++ with std14	g++	-O2 -std=c++14 -DONLINE_JUDGE	65536
1	g++ with std98	g++	-O2 -std=c++98 -DONLINE_JUDGE	65536
2	gcc with std11	gcc	-O2 -std=c11 -DONLINE_JUDGE	65536
3	gcc with std11	gcc	-O2 -std=c99 -DONLINE_JUDGE	65536
4	java	javac		65536
5	python	python		65536
6	python3	python3		65536

递交历史

#	状态	时间
32294	<div>Time Limit Exceeded</div> (/#!/contest/64/detail/32294)	07:52:05 PM <div>有效递交</div>
32027	<div>Time Limit Exceeded</div> (/#!/contest/64/detail/32027)	07:01:44 PM

递交答案 (剩余次数: 30)

语言和编译选项

g++ with std14

▼

1

提交评测

文件请拖入编辑器中，或

上传文件