

CSP2023模拟赛

题目名称	十一之争	环球旅行	封印	双倍经验
题目类型	传统型	传统型	传统型	传统型
可执行文件名	yo	journey	seal	double
输入文件名	yo.in	journey.in	seal.in	double.in
输出文件名	yo.out	journey.out	seal.out	double.out
每个测试点时限	1.0 秒	2.0 秒	1.0 秒	1.0 秒
内存限	512 MiB	512 MiB	512 MiB	512 MiB
子任务/测试点数目	5	6	3	10
是否等分	否	否	否	是

提交源文件程序名

对于C++语言	yo.cpp	journey.cpp	seal.cpp	double.cpp
---------	--------	-------------	----------	------------

编译选项

对于C++语言	-lm -O2 -std=c++17
---------	--------------------

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()的返回类型必须是int，程序正常结束时返回值必须是0。
3. 选手提交的程序代码文件请在**个人目录下以及子文件夹内各放一份**。
4. 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）。
5. 选手提交的程序源文件必须不大于100KB。
6. 程序可使用的栈空间内存限制于题目的内存限制一直。
7. 使用std::deque等STL容器时，请注意其内存空间消耗。
8. 评测时采用的机器配置为 AMD Ryzen 7 5800H with Radeon Graphics，内存16GiB。上述时限以此配置为准。
9. 评测在Windows 10下进行，使用LemonLine进行评测。

十一之争 (yo)

【题目描述】

yoimiya 和 oimiya 在争夺 11!

给定一个长度为 n 的数字串 s 和只包含 `y``o` 的字符串 t , yoimiya 会和 oimiya 玩 n 轮游戏, 初始有一个数字串 x 为 0, 每次:

- 如果 t_i 是 `y` 则是 yoimiya 操作, 如果是 `o` 则是 oimiya 操作。
- 每次操作: 将 s_i 或者 0 加入 x 的末尾。

如果最后 x 是 11 的倍数, 那么 yoimiya 获胜, 否则 oimiya 获胜。

假设两人都是绝顶聪明的, 那么最后谁会获胜?

【输入格式】

第一行, 一个数 n 。

第二行, 一个长度为 n 的数字串 s 。

第三行, 一个长度为 n 的字符串 t 。

【输出格式】

输出胜者: `yoimiya` 或者 `oimiya`。

【输入输出样例1】

yo.in	yo.out
5 19755 yoyyy	oimiya

【数据规模与约定】

本题采用**捆绑测试**。

对于所有测试数据, 保证 $1 \leq n \leq 10^6$ 。

子任务编号	$n \leq$	特殊限制	分数
1	10		15
2	10^3		15
3	10^5	t 全是 <code>y</code> 或者 <code>o</code>	15
4	10^5		25
5	10^6		30

环球旅行 (journey)

【题目描述】

小 M 在上一次环游世界后，又攒够了足够的钱，即将开启新的一次环球旅行。

现在，小 M 所在的星球变成了一张 n 个点 m 条边的无向图，无向图上的每条边都有为士兵看守。

具体来说：

- 如果小 M 在 $\leq i$ 的时刻进入了第 i 条边，那么他可以在 i 时刻走出这条边。
- 如果小 M 在 $> i$ 的时刻进入了第 i 条边，那么他将被第 i 条边的士兵抓住，从此不能再走出这条边。

现在，小 M 有 q 次环球旅行的计划。对于一次计划，小 M 将用一个形如 (l, r, s, t) 的四元组来描述，表示他将在 l 时刻从 s 出发，并需要在 r 时刻及以前到达 t 节点进行游玩。

小 M 想知道，他的哪些环球旅行计划是可以实现的。但是他时间有限，于是请你来解决这个问题。

【输入格式】

第一行三个整数 n, m, q ，分别表示点数，边数与小 M 的计划数量。

接下来 m 行，第 i 行给出两个整数 u, v ，表示第 i 条边连接了 u 与 v 两个节点。

接下来 q 行，每行四个整数 l, r, s, t ，表示一次旅行计划，含义见题目描述。

【输出格式】

输出共 q 行，每行一个字符串 `Yes` 或 `No`，表示一次旅行计划是否能够实现，能够实现则输出 `Yes`，否则输出 `No`。

【输入输出样例1】

journey.in	journey.out
5 4 6	
1 2	
2 3	Yes
3 4	Yes
3 5	Yes
1 3 1 4	No
1 3 2 4	No
1 4 4 5	Yes
1 4 4 1	
2 3 1 4	
2 2 2 3	

【数据规模与约定】

本题采用捆绑测试。

对于所有测试数据，保证
 $2 \leq n \leq 1000, 1 \leq m \leq 2 \times 10^5, 1 \leq q \leq 10^6, 1 \leq l \leq r \leq m, 1 \leq s, t \leq n$ ，给出的无向图无重边无自环。

子任务编号	特殊性质	分值
1	$n, m, q \leq 10$	14
2	$n, m, q \leq 1000$	16
3	$l = 1$	11
4	$l = r$	3
5	$n \leq 50$	25
6		31

封印 (seal)

【题目描述】

小 M 最近在研究一座神秘的古堡。

古堡内居住着 n 位法师。古堡的大门被施加了封印，只有**大法师**才可以在**位于大门前时**随意施加和破解封印。古堡内被圣伊娜庇护，因此在古堡大门内的法师均可以随意施加和破解封印，无论这位法师是否是大法师。

一位法师从外面回到古堡大门前，或想要从古堡大门内离开古堡时，如果大门被施加封印，那么这位法师必须解除封印，才能通过大门。法师通过大门后，如果 TA 现在具有施加封印的能力，那么 TA 可以选择是否给大门施加封印。封印被施加和被破解的速度都非常快，**消耗的时间可以看做0**。

新的一天开始了，法师要离开古堡进行战斗。初始 n 名法师都在古堡中，且大门被施加了封印。第 i 名法师将会在时刻 a_i 从古堡大门内离开古堡，并且在时刻 b_i 从外面回到古堡大门前。一天共有 t 个单位时间，因为法师们的作息非常规律，于是法师们一定在时刻 t 之前回到古堡。当法师不打算通过大门时，TA **不可以**给大门施加或破解封印，也就是说，第 i 名法师只可能在**时刻 a_i 和 b_i** 给大门施加或破解封印。

根据神秘的羊皮纸记载，古堡内只会有**恰好 m** 名大法师，并且所有法师都在回到古堡大门前之后顺利地进入了古堡，**没有在大门前等待任何时间**。并且，同一个时刻只有至多一名法师出现在大门前，也就是说，没有两名法师在**同一个时刻**离开或回到古堡。

小 M 想请您求出，在所有可能的满足上述要求的情况中，这一天内大门被封印的最长时间。

【输入格式】

第一行三个整数 n, t, m 。

接下来 n 行，第 i 行两个正整数 a_i, b_i 。

【输出格式】

一行一个非负整数，表示大门被封印的最长时间。

【输入输出样例1】

travel.in	travel.out
4 20 2 12 17 4 11 6 10 5 15	14

【输入输出样例1说明】

我们把时刻 $x - 1$ 到时刻 x 形成的这段时间称作第 x 个单位时间。

一种可能的情况是，第 2 名法师和第 3 名法师是大法师。

- 时刻 4，第 2 名法师解除封印，离开古堡时施加封印；
- 时刻 5，第 4 名法师解除封印并离开古堡；

- 时刻 6，第 3 名法师解除封印，离开古堡时施加封印；
- 时刻 10，第 3 名法师解除封印，回到古堡内，并施加封印；
- 时刻 11，第 2 名法师解除封印，回到古堡内，并施加封印；
- 时刻 12，第 1 名法师解除封印并离开古堡；
- 时刻 15，第 4 名法师直接回到古堡；
- 时刻 17，第 1 名法师直接回到古堡，并施加封印。

容易发现，大门仅在第 6 个单位时间，第 13 ~ 17 个单位时间没有被施加封印，因此被封印了 14 个单位时间。容易证明没有比这更长的封印时间。

注意到第 4 名法师回到古堡后不能给古堡大门施加封印，这会使第 1 名法师无法进入古堡，因为第 1 名法师无法在古堡大门前解除封印。

【数据规模与约定】

本题采用捆绑测试。

对于所有数据，保证 $2 \leq n \leq 2000, 5 \leq t \leq 10^9, 1 \leq m < n, 1 \leq a_i < b_i < t$, 对于任意 $i \neq j$, 满足 $a_i \neq a_j, b_i \neq b_j, a_i \neq b_j$ 。

子任务编号	特殊限制	分值
1	$n \leq 20, t \leq 10^6$	20
2	$n \leq 200$	30
3	无	50

双倍经验 (double)

【题目描述】

你有一棵 n 个点的树，每条边的边权 $\in [0, 10^{12}]$ ，多次询问任意两个点的距离。

显然这样还不够，现在有了两棵树，一棵是芒树，一棵是荒树，这两个树形态一模一样，但是边权不一样，并且两棵树对应节点间有边，具体地：

- 有边 $(2i - 1, 2i, w_i)$
- 以及 $(2x_j - 1, 2y_j - 1, wt_{j,1}), (2x_j, 2y_j, wt_{j,2})$

Q 次询问，每次询问任意两个节点之间的距离。

【输入格式】

第一行，一个数 n 。

第二行， n 个数，表示 $(2i - 1, 2i)$ 之间的边权。

接下来 $n - 1$ 行，每行 (x, y, w_1, w_2) ，表示原树上有边 (x, y) ，芒树上是 $(2x - 1, 2y - 1, w_1)$ ，荒树上是 $(2x, 2y, w_2)$ 。

下一行一个数 Q 。

接下来 Q 行，每行两个数 $u, v (1 \leq u, v \leq 2n, u \neq v)$ 表示询问。

【输出格式】


Q 行，表示答案。

【输入输出样例1】

double.in	double.out
5 3 6 15 4 8 1 2 5 4 2 3 5 7 1 4 1 5 1 5 2 1 3 1 2 5 6 1 10	3 15 4

【数据规模与约定】

对于所有测试数据， $1 \leq n, Q \leq 10^5, 1 \leq u, v \leq 2n, u \neq v$, 边权 $\in [0, 10^{12}]$ 。

测试点编号	\$n \backslash e\$		特殊性质
1 ~ 3	1000	1000	

测试点编号	$n \leq$	$Q \leq$	特殊性质
4 ~ 5	10^5	10^5	保证原树是一条链
6 ~ 7	10^5	10^5	保证原树是一朵菊花
8 ~ 10	10^5	10^5	