

NOIP模拟赛

题目名称	制胡窜	路径	三元组	抽牌
题目类型	传统型	传统型	传统型	传统型
输入文件名	string.in	path.in	triple.in	draw.in
输出文件名	string.out	path.out	triple.out	draw.out
每个测试点时限	1.0 秒	1.0 秒	3.0 秒	6.0 秒
内存限制	512 MB	512 MB	512 MB	512 MB
测试点数目	10	10	10	10
测试点是否等分	是	是	是	是

提交源程序文件名

对于 C++ 语言	string.cpp	path.cpp	triple.cpp	draw.cpp
-----------	------------	----------	------------	----------

编译选项

对于 C++ 语言	-lm -O2 -std=c++14
-----------	--------------------

注意事项

- 1. 文件名（包括程序名和输入输出文件名）必须使用英文小写。
- 2. C++ 中函数 `main()` 的返回值类型必须是 `int`，值必须为 `0`。
- 3. 若无特殊说明，输入文件中同一行内的多个整数、浮点数、字符串等均使用一个空格分隔。
- 4. 若无特殊说明，结果比较方式为忽略行末空格、文末回车后的全文比较。
- 5. 程序可使用的栈空间内存限制与题目的内存限制一致。
- 6. 题目不一定按照难度顺序排序，请注意掌握时间。

制胡窜 (string)

【题目描述】

小 T 不喜欢子序列。

定义正整数 a 与长为 L 的字符串 s 的运算 $a \cdot s$ 的结果为 t ，其中 t 由 a 个 s 顺次拼接而成。换句话说， t 是一个长为 aL 的循环节为 s 的字符串。

小 T 有两个字符串 S, T 。 q 次询问，每次询问给定一个整数 k ，你要求出最小的正整数 p ，使得 $p \cdot S$ 不是 $k \cdot T$ 的子序列。

【输入格式】

从文件 *string.in* 中读入数据。

第一行一个整数 q ，表示询问次数。

第二行一个字符串 S 。

第三行一个字符串 T 。

接下来 q 行，每行一个整数 k ，表示这次询问的参数。

【输出格式】

输出到文件 *string.out* 中。

共 q 行，每行一个正整数 p ，表示答案。

【样例 1 输入】

```
3
abaab
abaabacaba
1
3
4
```

【样例 1 输出】

```
2
5
7
```

【样例 2】

见选手目录下的 *string/string2.in* 与 *string/string2.ans*。

【样例 3】

见选手目录下的 *string/string3.in* 与 *string/string3.ans*。

【数据范围与提示】

记 n 为 S 的长度, m 为 T 的长度。

对于 20% 的数据, $n, m, q, k \leq 300$ 。

对于 50% 的数据, $k \leq 5000$ 。

对于另外 20% 的数据, 保证 $S_i = \text{a}$, $T_i = \text{a}$ 。

对于 100% 的数据, $1 \leq n, m \leq 5000$, $1 \leq q \leq 3 \times 10^5$, $1 \leq k \leq 10^{14}$ 。字符串 S, T 中仅包含小写字母。

路径 (path)

【题目描述】

小 T 刚刚学会给定两条树上的路径求出它们的交集，所以他在想这个问题能不能反过来问。

小 T 有一棵 n 个结点的树。 q 次询问，每次询问给定两个点 x, y ，保证 $x \neq y$ ，你需要求出有多少有序四元组 (a, b, c, d) 满足点 a, b 之间的简单路径与点 c, d 之间的简单路径的交集恰好为点 x, y 之间的路径。

你只需要输出答案对 998244353 取模的结果。

【输入格式】

从文件 *path.in* 中读入数据。

第一行两个整数 n, q ，表示树的大小和询问次数。

接下来 $n - 1$ 行，第 i 行有两个整数 u_i, v_i ，表示一条树边。

接下来 q 行，每行两个整数 x, y ，表示一次询问。

【输出格式】

输出到文件 *path.out* 中。

输出一行一个整数，表示答案对 998244353 取模的结果。

【样例 1 输入】

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

【样例 1 输出】

```
44
84
4
```

【样例 1 解释】

例如，对于第一个询问， $(3, 7, 7, 1)$, $(5, 7, 3, 7)$, $(7, 4, 7, 1)$ 都是合法的四元组。

对于第三个询问，共有四个合法的四元组，分别为 $(3, 6, 3, 6)$, $(3, 6, 6, 3)$, $(6, 3, 3, 6)$, $(6, 3, 6, 3)$ 。

【样例 2】

见选手目录下的 *path/path2.in* 与 *path/path2.ans*。

【样例 3】

见选手目录下的 *path/path3.in* 与 *path/path3.ans*。

【数据范围与提示】

对于 10% 的数据， $n, q \leq 20$ 。

对于 30% 的数据， $n, q \leq 5000$ 。

对于另外 10% 的数据，保证树是一条链（但不保证标号的顺序）。

对于另外 20% 的数据，保证 $u_i = i + 1$ ， v_i 从 $[1, i]$ 中独立均匀随机生成。

对于 100% 的数据， $1 \leq n, q \leq 3 \times 10^5$ ， $1 \leq u_i, v_i, x, y \leq n$ ，保证 $x \neq y$ ，保证给出的边构成了一棵树。

三元组 (triple)

【题目描述】

小 T 有一个 n 个结点的树，第 i 条边有边权 w_i ，两点 x, y 之间的距离 $d(x, y)$ 定义为它们简单路径上边权的和。保证树中每个点的度数 ≤ 3 。

小 T 想选出一个三元组 (x, y, z) ，它表示了树上的三个互不相同的点 x, y, z 。三元组的权值定义为这三个点的两两距离和。

小 T 想让权值尽量大。但是小 T 这两天做取模题做魔怔了，于是他要求这个三元组的权值对 L 取模的结果最大。

对所有点 x ，请求出在包含 x 的三元组中，三元组的权值对 L 取模的结果的最大值。

注意：三元组中的三个点必须两两不同。

【输入格式】

从文件 *triple.in* 中读入数据。

第一行两个整数 n, L ，表示树的大小和模数。

接下来 $n - 1$ 行，每行三个整数 u_i, v_i, w_i ，表示一条边权为 w_i 的树边 (u_i, v_i) 。

【输出格式】

输出到文件 *triple.out* 中。

共 n 行，第 i 行一个整数 ans_i ，表示三元组包含点 i 时的权值最大值。

【样例 1 输入】

```
6 9
1 2 3
2 3 2
3 4 5
3 6 1
2 5 7
```

【样例 1 输出】

```
8
7
6
7
8
8
```

【样例 1 解释】

对于点 1 来说, 最优的三元组是 $(1, 5, 6)$, 权值为 $(d(1, 5) + d(5, 6) + d(1, 6)) \bmod 9 = 8$ 。

对于点 2 来说, 最优的三元组是 $(2, 4, 6)$, 权值为 $(d(2, 4) + d(4, 6) + d(2, 6)) \bmod 9 = 7$ 。

其余点可以自行检验。

【样例 2】

见选手目录下的 *triple/triple2.in* 与 *triple/triple2.ans*。

【样例 3】

见选手目录下的 *triple/triple3.in* 与 *triple/triple3.ans*。

【数据范围与提示】

对于 30% 的数据, $n \leq 300$ 。

对于另外 20% 的数据, 保证树是一条链。

对于另外 20% 的数据, $u_i = i + 1, v_i = \lfloor (i + 1) / 2 \rfloor$ 。

对于 100% 的数据, $3 \leq n \leq 3000, 1 \leq u_i, v_i \leq n, 0 \leq w_i < L, 2 \leq L \leq 10^9$, 保证给出的边形成了一棵树, 保证树中每个点的度数 ≤ 3 。

抽牌 (draw)

【题目描述】

小 T 在玩卡牌游戏。

众所周知，在卡牌游戏里，过牌是很关键的，所以目前小 T 的牌库中，只可能有数字牌 0, 1, 2, 3, 4。数字牌 x 的含义是当你打出它的时候，会从牌库的顶端抽 x 张牌到自己手里，若牌库中不足 x 张牌则将牌库抽空为止。打出的数字牌 x 会放入弃牌堆中，在题目中你可以认为这张牌不会再被用到了。

目前，牌库里有 n 张牌，从牌堆顶到牌堆底数第 i 张牌为数字牌 a_i 。

在开始回合时，发牌员会进行一次切牌，切牌的结果是从牌堆顶到牌堆底的牌的顺序变为了 $a_s, a_{s+1}, \dots, a_n, a_1, \dots, a_{s-1}$ 。

接着，小 T 会抽 k 张牌堆顶的牌到自己手上。每次小 T 可以打出一张牌，但这一回合中小 T 至多打出 p 张牌。小 T 可以在任意时刻结束回合。

请问，这一回合中若小 T 使用最优策略，那么牌库里最少还剩多少牌。

进一步地，有 q 次这样的询问，每次询问给定三个整数 s, k, p ，你需要输出牌库里最少还剩多少牌。

注意：每次询问是独立的，也就是说每次询问并不会以任何方式影响到之后的询问。

【输入格式】

从文件 `draw.in` 中读入数据。

第一行两个整数 n, q ，表示牌的总数和询问次数。

第二行 n 个整数 a_i ，表示从牌堆顶到牌堆底数第 i 张牌为数字牌 a_i 。

接下来 q 行，每行三个整数 s, k, p ，表示一次询问。

【输出格式】

输出到文件 `draw.out` 中。

共 q 行，每行一个整数，表示答案。

【样例 1 输入】

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

【样例 1 输出】

```
0
```

2
1
0

【样例 1 解释】

在第一次询问中，回合开始时手上的牌为 $[2, 0, 1]$ 。先打出 2，手上的牌为 $[0, 1, 1, 0]$ ，再打出 1，手上的牌为 $[0, 1, 0, 1]$ ，再打出 1，手上的牌为 $[0, 0, 1, 1]$ ，此时结束回合，牌库中没有牌了，所以答案为 0。

在第三次询问中，回合开始时手上的牌为 $[0, 1, 1, 0]$ ，先打出 1，手上的牌为 $[0, 1, 0, 1]$ ，再打出 1，手上的牌为 $[0, 0, 1, 0]$ ，此时结束回合，牌库中有一张牌，所以答案为 1。

【样例 2】

见选手目录下的 *draw/draw2.in* 与 *draw/draw2.ans*。

【样例 3】

见选手目录下的 *draw/draw3.in* 与 *draw/draw3.ans*。

【样例 4】

见选手目录下的 *draw/draw4.in* 与 *draw/draw4.ans*。

【数据范围与提示】

对于 20% 的数据， $n, q \leq 2000$ 。

对于另外 20% 的数据， $a_i \leq 2$ 。

对于另外 20% 的数据，保证 a_i 在 $[0, 4]$ 中独立均匀随机生成。

对于 100% 的数据， $1 \leq n, q \leq 3 \times 10^5$ ， $0 \leq a_i \leq 4$ ， $1 \leq s, k, p \leq n$ 。