

# NOIP 模拟赛

时间：

题目名称	强连通分量	原神	叶子	石子合并
题目类型	传统	传统	传统	传统
目录	scc	o	leaf	stone
可执行文件名	scc	o	leaf	stone
输入文件名	scc.in	o.in	leaf.in	stone.in
输出文件名	scc.out	o.out	leaf.out	stone.out
每个测试点时限	1.0 秒	2.0 秒	3.0 秒	3.0 秒
内存限制	1024 MiB	1024 MiB	1024 MiB	1024 MiB
是否打包	否	否	否	否
子任务数目	10	20	20	20
测试点是否等分	是	是	是	是

提交源程序文件名

对于 C++ 语言	scc.cpp	o.cpp	leaf.cpp	stone.cpp
-----------	---------	-------	----------	-----------

编译选项

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

## 【注意事项（请仔细阅读）】

1. 在 NOI Linux 2.0 下使用 LemonLime 测评，编译器版本为 G++ 9.3.0，评测机 CPU 配置和内存大小与选手用机相同。
2. 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。
3. 程序可使用的栈空间大小与该题内存空间限制一致。
4. 每道题目所提交的代码文件大小限制为 100KB。
5. 若无特殊说明，输入文件与输出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进行分隔。
6. 输入文件中可能存在行末空格与文末回车，请选手使用更完善的读入方式（例如 scanf 函数）避免出错。
7. 请务必使用题面中规定的编译参数，保证你的程序在本机能够通过编译。此外不允许在程序中手动开启其他编译选项，一经发现，本题成绩以 0 分处理。

## 强连通分量 (scc)

### 【题目描述】

小 O 有一张  $n$  个点的有向图，每个点  $i$  只有一条到  $a_i$  的出边。

但是由于一些原因，小 O 会从原图中删除若干个点及其相邻的边，导致这张图变得不完整。

现在，小 O 想要知道，对于所有  $2^n$  种删除点的方案，这张有向图的强连通分量的个数的和对  $10^9 + 7$  取模后的值。

**强连通分量：**如果有向图的一张导出子图中任意两点可以相互到达，且加入任意若干个子图外的点都不满足这个条件，那么就称这个子图是有向图的强连通分量。

**导出子图：**由一些点和原图中所有连接这些点的边组成的子图。

### 【输入格式】

从文件 `scc.in` 中读入数据。

输入第一行包含一个整数  $n$ ，表示有向图的点数。

第二行包含  $n$  个整数  $a_1, a_2, \dots, a_n$ ，表示有向边。

### 【输出格式】

输出到文件 `scc.out` 中。

输出一行一个整数表示答案。

### 【样例 1 输入】

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

### 【样例 1 输出】

```
1 10
```

### 【样例 1 解释】

如果一个点都不删，则强连通分量数为 2。

如果删除点 1，则强连通分量数为 1。

如果删除点 2 或点 3，强连通分量数均为 2。

如果删除任意两个点，强连通分量数均为 1。

如果全删除，则强连通分量数为 0。

所有方案强连通分量数的和即为  $2 + 1 + 2 \times 2 + 1 \times 3 + 0 = 10$ 。

**【样例 2】**

见选手目录下的 *scc/scc2.in* 和 *scc/scc2.ans*。

该样例满足  $a_1, a_2, \dots, a_n$  互不相同。

**【样例 3】**

见选手目录下的 *scc/scc3.in* 和 *scc/scc3.ans*。

**【数据规模与约定】**

对于 100% 的数据:  $1 \leq n \leq 10^6$ ,  $1 \leq a_i \leq n$ 。

对于 30% 的数据:  $n \leq 18$ 。

对于 40% 的数据:  $n \leq 300$ 。

对于 50% 的数据:  $n \leq 2000$ 。

对于另外 10% 的数据:  $a_i = i$ 。

对于另外 20% 的数据:  $a_1, a_2, \dots, a_n$  互不相同。

## 原神 (o)

### 【题目描述】

小 O 喜欢原神。

为了更好的管理角色，小 O 给每个角色赋了一个权值，构成一个权值序列  $a_i$ 。

小 O 给他们的权值分别是 1, 3, 4, 6, 8, 9，也就是  $a = \{1, 3, 4, 6, 8, 9\}$ 。

不幸的是，小 O 的记性并不好，他甚至忘记了序列  $a_i$  具体长啥样，只记得序列  $a_i$  是一个长度为  $n$  的严格单调递增的正整数序列，且满足  $1 \leq a_i \leq m$ 。

同时，为了方便管理，小 O 实现了一个奇怪的算法：

---

#### 算法 1 查找算法

---

```
function CHECK( $p, x$ )  
    return  $p \leq n$  and  $a_p \leq x$   
end function
```

```
function SEARCH( $x$ )  
     $p \leftarrow 0$   
     $l \leftarrow 1$   
    while CHECK( $p + l, x$ ) do  
         $p \leftarrow p + l$   
         $l \leftarrow l \times 2$   
    end while  
    while  $l > 0$  do  
        if CHECK( $p + l, x$ ) then  
             $p \leftarrow p + l$   
        end if  
         $l \leftarrow \lfloor \frac{l}{2} \rfloor$   
    end while  
    return  $p + 1$   
end function
```

---

容易看出，这个算法的作用是找到序列  $a_i$  中第一个大于  $x$  的位置。

现在，小 O 给出了  $q$  次询问  $x_i$ ，他想要知道在所有的序列  $a_i$  上运行 **Search**( $x_i$ ) 调用 **Check** 函数的次数的和。

由于答案和输出量很大，令第  $i$  次询问的答案为  $w_i$ ，你只需要输出 ( $\oplus$  表示异或)：

$$\bigoplus_{i=1}^q i(w_i \bmod 1\,000\,000\,007)$$

【输入格式】

从文件 *o.in* 中读入数据。  
输入第一行三个整数  $n, m, q$ ，分别表示序列长度、序列中元素的范围和询问次数。

【输出格式】

输出到文件 *o.out* 中。  
输出一行一个整数表示压缩后的答案。

【样例 1 输入】

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

【样例 1 输出】

```
1 20
```

【样例 1 解释】

$w_i = \{8, 14, 16, 18, 24\}$ 。  
具体的数据如下表：

$a_i \backslash x$	0	1	2	3	4
1, 2, 3	2	4	4	6	6
1, 2, 4	2	4	4	4	6
1, 3, 4	2	4	4	4	6
2, 3, 4	2	2	4	4	6

【样例 2】

见选手目录下的 *o/o2.in* 和 *o/o2.ans*。  
该样例满足  $m - n \leq 20$ 。

【样例 3】

见选手目录下的 *o/o3.in* 和 *o/o3.ans*。

【数据规模与约定】

对于 100% 的数据： $1 \leq n, m, q \leq 10^6$ ， $0 \leq x_i \leq m$ 。

子任务编号	$n, m \leq$	特殊限制
1 ~ 2	10	无
3 ~ 4	100	
5 ~ 9	5000	
10	$2 \times 10^5$	$n > m$
11		$n = m$
12 ~ 14		$m - n \leq 20$
15 ~ 17		无
18 ~ 20	$10^6$	

## 叶子 (leaf)

### 【题目描述】

小 O 参加了赛车比赛，赛车比赛的场地是一棵  $n$  个点的带边权无根树，树上的每个点是比赛的计分点，边则是场地的道路，边权代表道路的长度。

赛车比赛的规则是：每名选手驾驶自己的赛车从自己的起点开往终点（起点和终点都是计分点，也就是树上的点），当最早经过一个计分点时得到一分，如果有两名选手同时经过一个计分点，则认为起点编号较小的选手比较早经过了 this 计分点。

这场赛车比赛为了增加趣味性，每名选手不能在赛前知道包括自己在内的所有选手的起点以及终点，只知道选手数量等于树上的叶子（即度数为 1 的点）数量，每名选手的起点是树上的其中一片叶子，并且任意两名选手的起点不同，以及所有选手的终点相同。

据小 O 了解，参加这场赛车比赛的选手都是身经百战的老选手，大家的赛车水平都差不多，所以小 O 认为所有选手都保持 1 单位的速度开车。

现在小 O 有  $q$  次询问，每次询问如果他的起点是  $s$  号点（保证是叶子），所有选手的终点都是  $t$  号点，那么他能够拿到多少分？

### 【输入格式】

从文件 *leaf.in* 中读入数据。

第一行一个整数  $n$ ，表示赛场的计分点数量。

第 2 ~  $n$  行，每行三个整数  $u, v, w$ ，表示赛场有一条长度为  $w$  的道路连接  $u$  号点和  $v$  号点。

第  $n+1$  行一个整数  $q$ ，表示小 O 的询问数量。

接下来  $q$  行，每行两个整数  $s, t$ ，表示一次询问中小 O 的起点编号和终点编号。

### 【输出格式】

输出到文件 *leaf.out* 中。

输出  $q$  行，每行一个整数表示询问的结果。

### 【样例 1 输入】

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

【样例 1 输出】

### 【样例 1 解释】

第 8 页 共 11 页



【样例 3】

见选手目录下的 *leaf/leaf3.in* 和 *leaf/leaf3.ans*。

【数据规模与约定】

对于 100% 的数据： $1 \leq n, q \leq 3 \times 10^5, 1 \leq u, v \leq n, u \neq v, 1 \leq w \leq 10^9, 1 \leq s, t \leq n$ ,  $s$  是叶子。

特殊限制 A：树上只有最多一个点的度数大于 1。

特殊限制 B：树上所有点的度数不超过 2。

特殊限制 C：所有询问的  $s$  相同。

特殊限制 D：所有询问的  $t$  相同。

子任务编号	$n, q \leq$	特殊限制
1 ~ 2	10	无
3 ~ 6	$10^3$	
7 ~ 10	$10^5$	
11	$3 \times 10^5$	A
12		B
13 ~ 15		C
16 ~ 18		D
19 ~ 20		无

## 石子合并 (stone)

### 【题目描述】

小 O 有一个长度为  $n$  的由 A,B,C,D 四种字符组成的字符串  $t_1t_2\cdots t_n$ 。

小 O 想对这个字符串进行若干次操作（可以不操作），每次操作要求：

1. 必须保证  $n > 1$  才能操作， $n = 1$  时不能操作，即只剩一个字符时不能操作。
2. 一次操作中，需要选择一个整数  $i$  满足  $1 \leq i \leq n$  并删除  $t_i$ ，即字符串变为：

$$t_1, t_2 \cdots t_{i-1} t_{i+1} t_{i+2} \cdots t_n$$

3. 操作后，字符串中会多次发生合并直至不能发生合并。具体来说，如果当前存在整数  $j$  满足  $1 < j < n$ ， $t_{j-1} = t_j = t_{j+1}$  且  $t_{j-1}, t_j, t_{j+1}$  经历的合并次数相同，则它们会合并成为  $t_j$ ，即字符串变为：

$$t_1, t_2 \cdots t_{j-2} t_j t_{j+2} t_{j+3} \cdots t_n$$

并且这个  $t_j$  经历的合并次数为原来  $t_{j-1}, t_j, t_{j+1}$  经历的合并次数加 1。

例如原来有三个在字符串中连续的字符 BBB，它们经历的合并次数分别为 3, 3, 3，则合并后会保留一个字符 B，其经历的合并次数为 4。

4. 若可以发生合并的  $j$  不唯一，则选择最小的  $j$ 。
5. 初始时  $t_1, t_2, \cdots, t_n$  经历的合并次数均为 0。
6. 注意每次操作或合并过程结束后， $n$  会更新为当前的字符串长度。

现在小 O 想要知道，经过若干次操作后，可以得到的本质不同的字符串数量有多少个。其中两个字符串  $t_{1,1}t_{1,2}\cdots t_{1,n}$  和  $t_{2,1}t_{2,2}\cdots t_{2,m}$  本质不同当且仅当  $n \neq m$  或者存在整数  $i$  满足  $1 \leq i \leq \min\{n, m\}$ ，使得  $t_{1,i}$  与  $t_{2,i}$  的字符或者经历的合并次数不同。

由于答案很大，你只需要输出答案对 998244353 取模后的结果即可。

### 【输入格式】

从文件 `stone.in` 中读入数据。

第一行一个整数  $n$ ，表示字符串的长度。

第二行  $n$  个连续的字符  $t_1t_2\cdots t_n$ ，表示字符串。

### 【输出格式】

输出到文件 `stone.out` 中。

一行一个整数表示答案。

### 【样例 1 输入】

1

3

2

ABA

【样例 1 输出】

1

6

【样例 1 解释】

最终可以得到的本质不同的字符串有：ABA, AA, AB, BA, A, B。

【样例 2】

见选手目录下的 *stone/stone2.in* 和 *stone/stone2.ans*。

【样例 3】

见选手目录下的 *stone/stone3.in* 和 *stone/stone3.ans*, 该样例满足特殊性质 A。

【样例 4】

见选手目录下的 *stone/stone4.in* 和 *stone/stone4.ans*。

【数据规模与约定】

对于 100% 的数据： $1 \leq n \leq 10^6$ ， $t_i \in \{A, B, C, D\}$ ，保证初始时不会发生合并，即不存在整数  $j$  满足  $1 < j < n$  使得  $t_{j-1} = t_j = t_{j+1}$ 。

子任务编号	$n \leq$	特殊性质
1 ~ 2	8	无
3 ~ 4	16	
5 ~ 7	$10^3$	A
8 ~ 10		无
11 ~ 13	$10^5$	A
14 ~ 16		无
17 ~ 18	$10^6$	A
19 ~ 20		无

特殊性质 A：初始时对于任意  $j$  满足  $1 \leq j < n$ ，有  $t_j \neq t_{j+1}$ 。