

题目名称	遍历	排序	网格	排列
题目类型	传统型	传统型	传统型	传统型
英文名	traverse	sort	grid	perm
输入文件名	traverse.in	sort.in	grid.in	perm.in
输出文件名	traverse.out	sort.out	grid.out	perm.out
每个测试点时限	1s	1s	1s	2s
内存限制	512MB	512MB	512MB	512MB
提交文件名	traverse.cpp	sort.cpp	grid.cpp	perm.cpp
测试点个数	25	10	10	10
单个测试点分数	4	10	10	10

# 遍历

## 题目描述

给定一棵  $n$  个结点以 1 为根的有根树，结点编号为 1 到  $n$ ，每个结点  $i$  关联一个非负整数  $a_i$ ，表示该结点的处理耗时。定义一条从根出发的路径合法当且仅当其满足以下条件：

- 必须遍历整棵树，且最终回到结点 1；
- 每条边最多经过 2 次；

设路径长度为  $L$ ，设结点  $i$  第一次被访问的时间为  $r_i$ ，则定义其完成时间为：

- 若  $i \neq 1$ ，则  $t_i = r_i + a_i$ ；
- 若  $i = 1$ ，则  $t_1 = L + a_1$ 。

定义一条合法路径的耗时：

$$T = \max_{1 \leq i \leq n} t_i$$

请对于所有合法路径，求出最短的耗时。

## 输入格式

第一行一个正整数  $n$  ( $1 \leq n \leq 3 \times 10^5$ )；

第二行  $n$  个非负整数  $a_1, a_2, \dots, a_n$  ( $0 \leq a_i \leq 10^9$ )；

接下来  $n - 1$  行，每行两个正整数  $u_i, v_i$ ，表示树中的一条无向边。

## 输出格式

输出一个整数  $T$ ，表示所有合法路径中最短的耗时。

## 样例输入

```
5
1 4 1 3 9
1 2
2 3
2 4
1 5
```

## 样例输出

```
10
```

## 数据范围

对于 40% 的数据,  $n \leq 10$ ;

对于 60% 的数据,  $n \leq 3000$ ;

另有 8% 的数据满足  $\forall i \in [1, n-1], v_i = u_i + 1$ ;

另有 8% 的数据满足  $\forall i \in [1, n-1], u_i = 1, v_i = i + 1$ 。

对于 100% 的数据,  $1 \leq n \leq 3 \times 10^5, 0 \leq a_i \leq 10^9$ , 且输入构成一棵树。

## 排序

### 题目描述

设有一个从 1 到  $n$  的排列  $P = [p_1, p_2, \dots, p_n]$ , 初始时为任意打乱的排列。

定义如下过程反复作用于  $P$ , 直至  $P$  恢复为升序排列:

1. 若  $P = [1, 2, \dots, n]$ , 则结束操作。
2. 扫描当前排列, 若存在一个连续子区间  $[i, i+1, \dots, j]$ , 满足对于区间内任意相邻元素都有  $p_{k+1} - p_k = 1$ , 则将该区间视为一块粘连体。在后续操作中, 该粘连体视作一个整体, 不能被拆开。
3. 将当前的所有粘连体 (包括未粘连的单元素块和已经粘连的多元素块) 打乱顺序, 并将它们首尾连接成新的排列  $P$ , 回到步骤 1。

*注: 步骤 2 中可以存在多个不相交的粘连区间, 且粘连不可逆, 即一旦粘连, 不会拆开。*

定义该过程的操作步骤数为执行“步骤 3”的次数, 记为  $X$ 。求  $E(n) = \mathbb{E}[X]$ , 即初始为任意排列时, 该过程结束所需的期望步骤次数。由于答案可能为有理数, 输出其模  $10^9 + 7$  意义下的值:

$$E(n) \bmod (10^9 + 7)$$

其中模运算应使用有理数模意义下的乘法逆元, 即若  $E(n) = \frac{a}{b}$ , 应输出  $a \cdot b^{-1} \bmod (10^9 + 7)$ , 其中  $b^{-1}$  是  $b$  在模  $10^9 + 7$  下的乘法逆元。

### 输入格式

一行一个整数  $n$  ( $1 \leq n \leq 2000$ ), 表示排列的大小。

## 输出格式

一行一个整数，表示  $E(n) \bmod (10^9 + 7)$ 。

## 样例输入

3

## 样例输出

333333338

## 样例说明

当  $n = 3$  时，期望操作次数  $E(3) = \frac{7}{3}$ 。

## 数据范围

测试点编号	$n$
1	1
2	5
3	8
4	10
5	13
6	$\leq 1000$
7	$\leq 1000$
8	$\leq 2000$
9	$\leq 2000$
10	$\leq 2000$

## 网格

### 题目描述

给定一个  $h$  行  $w$  列的网格，每个单元格需填入  $[1, m]$  范围内的整数。存在  $n$  条约束，每条约束指定一个子矩阵（左上角  $(x_1, y_1)$ ，右下角  $(x_2, y_2)$ ），要求该子矩阵内所有单元格的\*\*最大值恰好等于  $v$ 。形式化地，设网格矩阵为  $A$ ， $A_{i,j}$  表示第  $i$  行第  $j$  列的值，每条约束要求：

$$\max\{A_{i,j} \mid x_1 \leq i \leq x_2, y_1 \leq j \leq y_2\} = v$$

计算满足所有约束的网格填数方案数量，结果对  $10^9 + 7$  取模。

## 输入格式

第一行输入四个整数  $h, w, m, n$  ( $1 \leq h, w, m \leq 10000, 1 \leq n \leq 10$ )，分别表示网格行数、列数、数值上限、约束数。

接下来  $n$  行，每行五个整数  $x_1, y_1, x_2, y_2, v$  ( $1 \leq x_1, x_2 \leq h, 1 \leq y_1, y_2 \leq w, 1 \leq v \leq m$ )，描述一条约束。

## 输出格式

一行一个整数，表示方案数模  $10^9 + 7$  的结果。

## 样例输入

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

## 样例输出

```
60
```

## 数据范围

对于 20% 的数据， $h, w, n, m \leq 3$ ;

另有 10% 的数据满足  $n = 0$ ;

另有 20% 的数据满足  $n = 1$ ;

对于 100% 的数据， $h, w, n, m \leq 10000, 1 \leq v \leq m, 0 \leq n \leq 10$ 。

## 排列

### 题目描述

大家都知道  $n$  阶排列有  $n!$  个，神奇的 zzy 认为这很没意思。

他施展魔法把所有的  $n$  阶排列传送到了异世界，然后创死了所有最长上升子序列长度  $\geq 3$  的排列。

但 zzy 依旧觉得很无趣，他又创造了  $m$  条世界线，第  $i$  条世界线的法则是  $x$  必须位于  $pos_i$  位置。

对于每个世界线，你需要告诉 zzy 符合条件的排列有多少个，对  $10^9 + 7$  取模。

形式化地，对于第  $i$  个问题，符合他条件的排列  $P$  满足：

- $P$  是  $n$  阶排列。
- $\forall 1 \leq i < j < k \leq n, \neg(P_i < P_j < P_k)$ 。
- $P_{pos_i} = x$ 。

## 输入格式

第一行三个整数  $n, m, x$  ( $1 \leq n, m \leq 500, 1 \leq x \leq n$ )。

第二行  $m$  个整数  $pos_1, pos_2, \dots, pos_n$  ( $1 \leq pos_i \leq n$ )。

## 输出格式

一行,  $m$  个整数, 第  $i$  个整数表示第  $i$  个问题的答案。

## 样例 1 输入

```
10 10 5
1 2 3 4 5 6 7 8 9 10
```

## 样例 1 输出

```
429 1375 2310 2520 1764 588 1848 2673 2288 1001
```

## 样例 2 输入

```
50 5 12
2 12 22 32 42
```

## 样例 2 输出

```
649624491 426442934 805738681 404720023 80426482
```

## 样例 3 输入

```
500 5 212
12 112 212 312 412
```

## 样例 3 输出

```
38822761 799706133 331782079 959426566 240547714
```

## 数据范围

对于 100% 的数据,  $1 \leq n, m \leq 500, 1 \leq x, pos_i \leq n$ 。

	$n$	$m$	$x$	$pos_i$
1, 2, 3	$\leq 9$	$\leq n$	$\leq n$	
4	$\leq 500$	$= 1$	$= 1$	$= 1$
5, 6	$\leq 500$	$= 1$	$= n$	$= 1$
7, 8	$\leq 500$	$= 1$	$= 1$	$= n$
9, 10	$\leq 500$	$= 1$	$\leq n$	
11 – 14	$\leq 50$	$\leq n$	$\leq n$	
15 – 20	$\leq 500$	$\leq n$	$\leq n$	