全国青少年信息学奥林匹克竞赛

CCF NOI 2018

第一试

时间：2018 年 7 月 18 日 08:00 ~ 13:00

题目名称	归程	冒泡排序	你的名字
题目类型	传统型	传统型	传统型
目录	return	inverse	name
可执行文件名	return	inverse	name
输入文件名	return.in	inverse.in	name.in
输出文件名	return.out	inverse.out	name.out
每个测试点时限	4.0 秒	1.0 秒	4.0 秒
内存限制	512 MB	512 MB	1 GB
测试点/包数目	20	25	25
测试点是否等分	是	是	是

提交源程序文件名

对于 C++ 语言	return.cpp	inverse.cpp	name.cpp
对于 C 语言	return.c	inverse.c	name.c
对于 Pascal 语言	return.pas	inverse.pas	name.pas

编译选项

对于 C++ 语言	-O2 -lm
对于 C 语言	-O2 -lm
对于 Pascal 语言	-O2

注意事项：

1、提交的源文件必须存放在已建立好的下发样例的文件夹中（该文件夹与试题同名）。

2、文件名（包括程序名和输入输出文件名）必须使用英文小写。

3、结果比较方式为忽略行末空格、文末回车后的全文比较。

4、C/C++ 中函数 main() 的返回值类型必须是 int，值为 0。

5、对于因未遵守以上规则对成绩造成的影响，相关申诉不予受理。

归程（return）

【题目背景】

本题的故事发生在魔力之都，在这里我们将为你介绍一些必要的设定。

魔力之都可以抽象成一个 n 个节点、m 条边的**无向连通图**（节点的编号从 1 至 n）。我们依次用 l, a 描述一条边的**长度**、**海拔**。

作为季风气候的代表城市，魔力之都时常有雨水相伴，因此道路积水总是不可避免的。由于整个城市的排水系统连通，因此**有积水的边一定是海拔相对最低的一些边**。

我们用**水位线**来描述降雨的程度，它的意义是：所有海拔**不超过**水位线的边都是**有积水**的。

【题目描述】

Yazid 是一名来自魔力之都的 OIer，刚参加完 IONI2018 的他将踏上归程，回到他温暖的家。

Yazid 的家恰好在魔力之都的 1 号节点。对于接下来 Q 天，每一天 Yazid 都会告诉你他的出发点 v ，以及当天的水位线 p。

每一天，Yazid 在出发点都拥有一辆**车**。这辆车由于一些故障不能经过有积水的边。Yazid 可以在任意节点下车，这样接下来他就可以步行经过有积水的边。但车会被留在他下车的节点并不会再被使用。

- 需要特殊说明的是，第二天车会被重置，这意味着：
  - 车会在新的出发点被准备好。
  - Yazid 不能利用之前在某处停放的车。

Yazid 非常讨厌在雨天步行，因此他希望在完成回家这一目标的同时，最小化他**步行经过的**边的总长度。请你帮助 Yazid 进行计算。

本题的部分测试点将强制在线，具体细节请见【输入格式】和【子任务】。

【输入格式】

从文件 ***return.in*** 中读入数据。

单个测试点中包含多组数据。输入的第一行为一个非负整数 T，表示数据的组数。接下来依次描述每组数据，对于每组数据：

- 第一行 2 个非负整数 n, m，分别表示节点数、边数。
- 接下来 m 行，每行 4 个正整数 u, v, l, a，描述一条连接节点 u, v 的、长度为 l、海拔为 a 的边。
  - 在这里，我们保证 $1 \le u, v \le n$。

- 接下来一行 3 个非负数 Q, K, S，其中 Q 表示总天数，$K \in \{0, 1\}$ 是一个会在下面被用到的系数，S 表示的是可能的最高水位线。
- 接下来 Q 行依次描述每天的状况。每行 2 个整数 v_0, p_0 描述一天：
  - 这一天的出发节点为 $v = (v_0 + K \times lastans - 1) \bmod n + 1$。
  - 这一天的水位线为 $p = (p_0 + K \times lastans) \bmod (S + 1)$。
  - 其中 $lastans$ 表示上一天的答案（最小步行总路程）。特别地，我们规定第 1 天时 $lastans = 0$。
  - 在这里，我们保证 $1 \le v_0 \le n$，$0 \le p_0 \le S$。

对于输入中的每一行，如果该行包含多个数，则用单个空格将它们隔开。

【输出格式】

输出到文件 ***return.out*** 中。

依次输出各组数据的答案。对于每组数据：

- 输出 Q 行每行一个整数，依次表示每天的最小步行总路程。

【样例 1 输入】

```
1
4 3
1 2 50 1
2 3 100 2
3 4 50 1
5 0 2
3 0
2 1
4 1
3 1
3 2
```

【样例 1 输出】

```
0
50
200
50
150
```

【样例 1 解释】

第一天没有降水，Yazid 可以坐车直接回到家中。

第二天、第三天、第四天的积水情况相同，均为连接 1, 2 号节点的边、连接 3, 4 号点的边有积水。

对于第二天，Yazid 从 2 号点出发坐车只能去往 3 号节点，对回家没有帮助。因此 Yazid 只能纯靠徒步回家。

对于第三天，从 4 号节点出发的唯一一条边是有积水的，车也就变得无用了。Yazid 只能纯靠徒步回家。

对于第四天，Yazid 可以坐车先到达 2 号节点，再步行回家。

第五天所有的边都积水了，因此 Yazid 只能纯靠徒步回家。

【样例 2 输入】

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

【样例 2 输出】

```
0
2
3
1
```

【样例 2 解释】

本组数据强制在线。

第一天的答案是 0，因此第二天的 $v = (5 + 0 - 1) \bmod 5 + 1 = 5$，$p = (2 + 0) \bmod (3 + 1) = 2$。

第二天的答案是 2，因此第三天的 $v = (2 + 2 - 1) \bmod 5 + 1 = 4$，$p = (0 + 2) \bmod (3 + 1) = 2$。

第三天的答案是 3，因此第四天的 $v = (4 + 3 - 1) \bmod 5 + 1 = 2$，$p = (0 + 3) \bmod (3 + 1) = 3$。

【样例 3】

见选手目录下的 ***return/return3.in*** 与 ***return/return3.ans***。

【样例 4】

见选手目录下的 ***return/return4.in*** 与 ***return/return4.ans***。

【样例 5】

见选手目录下的 ***return/return5.in*** 与 ***return/return5.ans***。

【子任务】

所有测试点均保证 $T \le 3$，所有测试点中的所有数据均满足如下限制：

- $n \le 2 \times 10^5$，$m \le 4 \times 10^5$，$Q \le 4 \times 10^5$，$K \in \{0, 1\}$，$1 \le S \le 10^9$。
- 对于所有边：$l \le 10^4$，$a \le 10^9$。
- 任意两点之间都直接或间接通过边相连。

为了方便你快速理解，我们在表格中使用了一些简单易懂的表述。在此，我们对这些内容作形式化的说明：

- 图形态：对于表格中该项为“一棵树”或“一条链”的测试点，保证 $m = n - 1$。除此之外，这两类测试点分别满足如下限制：
  - 一棵树：保证输入的图是一棵树，即保证边不会构成回路。
  - 一条链：保证所有边满足 $u + 1 = v$。
- 海拔：对于表格中该项为“一种”的测试点，保证对于所有边有 $a = 1$。
- 强制在线：对于表格中该项为“是”的测试点，保证 $K = 1$；如果该项为“否”，则有 $K = 0$。
- 对于所有测试点，如果上述对应项为“不保证”，则对该项内容不作任何保证。

<table>
<tr><th>n</th><th>m</th><th>$Q=$</th><th>测试点</th><th>图形态</th><th>海拔</th><th>强制在线</th></tr>
<tr><td>≤ 1</td><td>≤ 0</td><td>0</td><td>1</td><td rowspan="6">不保证</td><td rowspan="6">一种</td><td rowspan="8">否</td></tr>
<tr><td>≤ 6</td><td>≤ 10</td><td>10</td><td>2</td></tr>
<tr><td>≤ 50</td><td>≤ 150</td><td>100</td><td>3</td></tr>
<tr><td>≤ 100</td><td>≤ 300</td><td>200</td><td>4</td></tr>
<tr><td>≤ 1500</td><td>≤ 4000</td><td>2000</td><td>5</td></tr>
<tr><td>≤ 200000</td><td>≤ 400000</td><td>100000</td><td>6</td></tr>
<tr><td rowspan="3">≤ 1500</td><td rowspan="5">$=n-1$</td><td rowspan="3">2000</td><td>7</td><td rowspan="3">一条链</td><td rowspan="11">不保证</td></tr>
<tr><td>8</td></tr>
<tr><td>9</td></tr>
<tr><td rowspan="3">≤ 200000</td><td rowspan="3">100000</td><td>10</td><td rowspan="2">一棵树</td></tr>
<tr><td>11</td><td>是</td></tr>
<tr><td rowspan="3">≤ 400000</td><td>12</td><td rowspan="5">不保证</td><td rowspan="3">否</td></tr>
<tr><td>13</td></tr>
<tr><td>14</td></tr>
<tr><td rowspan="2">≤ 1500</td><td rowspan="2">≤ 4000</td><td rowspan="2">2000</td><td>15</td><td rowspan="5">是</td></tr>
<tr><td>16</td></tr>
<tr><td rowspan="4">≤ 200000</td><td rowspan="4">≤ 400000</td><td rowspan="2">100000</td><td>17</td></tr>
<tr><td>18</td></tr>
<tr><td rowspan="2">400000</td><td>19</td></tr>
<tr><td>20</td></tr>
</table>

为了优化你的阅读体验，我们在表格中把测试点的编号放在了中间，请注意这一点。

【提示】

- 样例 3 满足海拔为一种，且不强制在线。
- 样例 4 满足图形态为一条链，且强制在线。
- 样例 5 满足不强制在线。

冒泡排序（inverse）

【题目背景】

最近，小 S 对冒泡排序产生了浓厚的兴趣。为了问题简单，小 S 只研究对 1 到 n 的排列的冒泡排序。

下面是对冒泡排序的算法描述。

输入：一个长度为 n 的排列 p[1...n]
输出：p 排序后的结果。

```
for i = 1 to n do
    for j = 1 to n - 1 do
        if(p[i] > p[i + 1])
            交换 p[i] 与 p[i + 1] 的值
```

冒泡排序的交换次数被定义为交换过程的执行次数。可以证明交换次数的一个下界是 $\frac{1}{2}\sum_{i=1}^{n}|i-p_i|$，其中 p_i 是排列 p 中第 i 个位置的数字。如果你对证明感兴趣，可以看提示。

【题目描述】

小 S 开始专注于研究长度为 n 的排列中，满足交换次数 $=\frac{1}{2}\sum_{i=1}^{n}|i-p_i|$ 的排列（在后文中，为了方便，我们把所有这样的排列叫“好”的排列）。他进一步想，这样的排列到底多不多？它们分布的密不密集？

小 S 想要对于一个给定的长度为 n 的排列 q，计算字典序严格大于 q 的“好”的排列个数。但是他不会做，于是求助于你，希望你帮他解决这个问题，考虑到答案可能会很大，因此只需输出答案对 998244353 取模的结果。

【输入格式】

从文件 ***inverse.in*** 中读入数据。

输入第一行包含一个正整数 T，表示数据组数。

对于每组数据，第一行有一个正整数 n, 保证 $n \le 6\times10^5$。

接下来一行会输入 n 个正整数，对应于题目描述中的 q_i，保证输入的是一个 1 到 n 的排列。

【输出格式】

输出到文件 ***inverse.out*** 中。

输出共 T 行，每行一个整数。

对于每组数据，输出一个整数，表示字典序严格大于 q 的“好”的排列个数对 998244353 取模的结果。

【样例 1 输入】

```
1
3
1 3 2
```

【样例 1 输出】

```
3
```

【样例 1 解释】

字典序比 1 3 2 大的排列中，除了 3 2 1 以外都是“好”的排列，故答案为 3。

【样例 2 输入】

```
1
4
1 4 2 3
```

【样例 2 输出】

```
9
```

【样例 3】

见选手目录下的 ***inverse/inverse3.in*** 与 ***inverse/inverse3.ans***。

【子任务】

下面是对本题每个测试点的输入规模的说明。
对于所有数据，均满足 $T = 5$ (样例可能不满足).
记 n_{max} 表示每组数据中 n 的最大值，$\sum n$ 表示所有数据的 n 的和。

测试点	$n_{max}=$	$\sum n \le$	特殊性质
1	8	$5n_{max}$	无
2	9		
3	10		
4	12		
5	13		
6	14		
7	16		
8			
9	17		
10	18		
11			
12	122	700	$\forall i \ \ p_i = i$
13	144		无
14	166		
15	200		
16	233		
17	777	4000	$\forall i \ \ p_i = i$
18	888		无
19	933		
20	1000		
21	266666	2000000	$\forall i \ \ p_i = i$
22	333333		无
23	444444		
24	555555		
25	600000		

【提示】

下面是对交换次数下界是 $\frac{1}{2}\sum_{i=1}^{n}|i-p_i|$ 的证明。

排序本质上就是数字的移动，因此排序的交换次数应当可以用数字移动的总距离来描述。对于第 i 个位置，假设在初始排列中，这个位置上的数字是 p_i，那么我们需要将这个数字移动到第 p_i 个位置上，移动的距离是 $|i-p_i|$。从而移动的总距离就是 $\sum_{i=1}^{n}|i-p_i|$，而冒泡排序每次会交换两个相邻的数字，每次交换可以使移动的总距离至多减少 2。因此 $\frac{1}{2}\sum_{i=1}^{n}|i-p_i|$ 是冒泡排序的交换次数的下界。

并不是所有的排列都达到了下界，比如在 $n=3$ 的时候，考虑排列 3 2 1, 这个排列进行冒泡排序以后的交换次数是 3，但是 $\frac{1}{2}\sum_{i=1}^{n}|i-p_i|$ 只有 2。

你的名字（name）

【题目背景】

实力强大的小 A 被选为了 ION2018 的出题人，现在他需要解决题目的命名问题。

【题目描述】

小 A 被选为了 ION2018 的出题人，他精心准备了一道质量十分高的题目，且已经把除了题目命名以外的工作都做好了。

由于 ION 已经举办了很多届，所以在题目命名上也是有规定的，ION 命题手册规定：每年由命题委员会规定一个小写字母字符串，我们称之为那一年的命名串，**要求每道题的名字必须是那一年的命名串的一个非空连续子串，且不能和前一年的任何一道题目的名字相同**。

由于一些特殊的原因，小 A 不知道 ION2017 每道题的名字，但是他通过一些特殊手段得到了 ION2017 的命名串，现在小 A 有 Q 次询问：每次给定 ION2017 的命名串和 ION2018 的命名串，求有几种题目的命名，使得这个名字一定满足命题委员会的规定，即是 ION2018 的命名串的一个非空连续子串且一定不会和 ION2017 的任何一道题目的名字相同。

由于一些特殊原因，所有询问给出的 ION2017 的命名串都是某个串的连续子串，详细可见输入格式。

【输入格式】

第一行一个字符串 S ，之后询问给出的 ION2017 的命名串都是 S 的连续子串。

第二行一个正整数 Q，表示询问次数。

接下来 Q 行，每行有一个字符串 T 和两个正整数 l, r，表示询问如果 ION2017 的命名串是 $S[l..r]$，ION2018 的命名串是 T 的话，有几种命名方式一定满足规定。

保证输入中给出的字符串都是由小写字母构成的。

【输出格式】

输出 Q 行，第 i 行一个非负整数表示第 i 个询问的答案。

【样例 1 输入】

```
scbamgepe
3
smape 2 7
```

```
sbape 3 8
sgepe 1 9
```

【样例 1 输出】

```
12
10
4
```

【样例 2】

见选手目录下的 ***name/name2.in*** 与 ***name/name2.ans***。

【子任务】

<table>
<tr><th>测试点</th><th>$|S| \le$</th><th>$Q \le$</th><th>$\sum |T| \le$</th><th>询问限制</th><th>其他限制</th></tr>
<tr><td>1</td><td>200</td><td rowspan="5">200</td><td rowspan="3">40000</td><td rowspan="17">对于所有询问有 $l=1, r=|S|$</td><td rowspan="3">$|T| \le 200$</td></tr>
<tr><td>2</td><td rowspan="4">1000</td></tr>
<tr><td>3</td></tr>
<tr><td>4</td><td rowspan="4">$5*10^5$</td><td rowspan="4">无</td></tr>
<tr><td>5</td></tr>
<tr><td>6</td><td rowspan="2">$5*10^5$</td><td rowspan="2">1</td></tr>
<tr><td>7</td></tr>
<tr><td>8</td><td rowspan="2">10^5</td><td rowspan="15">10^5</td><td rowspan="2">$2*10^5$</td></tr>
<tr><td>9</td><td>字符串随机</td></tr>
<tr><td>10</td><td rowspan="2">$2*10^5$</td><td rowspan="2">$4*10^5$</td><td>无</td></tr>
<tr><td>11</td><td>字符串随机</td></tr>
<tr><td>12</td><td rowspan="2">$3*10^5$</td><td rowspan="2">$6*10^5$</td><td>无</td></tr>
<tr><td>13</td><td>字符串随机</td></tr>
<tr><td>14</td><td rowspan="2">$4*10^5$</td><td rowspan="2">$8*10^5$</td><td>无</td></tr>
<tr><td>15</td><td>字符串随机</td></tr>
<tr><td>16</td><td rowspan="2">$5*10^5$</td><td rowspan="10">10^6</td><td>无</td></tr>
<tr><td>17</td><td>字符串随机</td></tr>
<tr><td>18</td><td>$2*10^5$</td><td rowspan="8">无</td><td rowspan="8">无</td></tr>
<tr><td>19</td><td>$3*10^5$</td></tr>
<tr><td>20</td><td>$4*10^5$</td></tr>
<tr><td>21</td><td rowspan="5">$5*10^5$</td></tr>
<tr><td>22</td></tr>
<tr><td>23</td></tr>
<tr><td>24</td></tr>
<tr><td>25</td></tr>
</table>

对于所有数据，保证 $1 \le l \le r \le |S|, 1 \le |T| \le 5*10^5$