11.20 模拟赛

StudyingFather

2020 年 11 月 20 日

题目名称	领带	准高速电车	关灯	排序
题目类型	传统	传统	传统	传统
目录	`neckties`	`train`	`light`	`sort`
输入文件名	`neckties.in`	`train.in`	`light.in`	`sort.in`
输出文件名	`neckties.out`	`train.out`	`light.out`	`sort.out`
时间限制	1.0 秒	1.0 秒	1.0 秒	6.0 秒
内存限制	256 MiB	256 MiB	512 MiB	256 MiB
子任务数目	3	3	20	10
子任务是否等分	否	否	是	是

注意事项：

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() 的返回值类型必须是 int，程序正常结束时的返回值必须是 0。
3. 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）。
4. 题目栈空间限制和内存限制一致。
5. 评测在 NovaOJ 上进行，比赛采用 OI 赛制，即每道题取最后一次提交计分。
6. 对于采用子任务捆绑测试的题目，你在该题上的得分等于各子任务的得分之和，而各子任务的得分，等于该子任务下每个测试点的最低得分。
7. 赛后可以在组题人的洛谷博客[1]上找到题解。

[1] `https://studyingfather.blog.luogu.org/simulation-contests-log`

1 领带

1.1 题目描述

W 公司的最新发明是「只不过是长的领带」。共有 $N+1$ 条领带，并以 $1,\ldots,N+1$ 编号。第 i 种领带的长度为 A_i，其中 $1 \le i \le N+1$。

公司聚集了他们的员工，并准备举办一场试戴派对。参加该聚会的员工共有 N 个，且第 j 个员工一开始戴着长度为 B_j 的领带，其中 $1 \le j \le N$。派对的流程如下：

1. W 公司的 CEO 首先选出一条领带，它将不会在接下来的派对中使用。
2. 然后，每个员工从其余领带中选择一条，且需保证没有两个员工选择了同一条领带。
3. 最终，每个员工取下一开始戴着的领带，并试戴他 / 她选择的领带。

若某个员工一开始戴着的领带长度为 b 而最后试戴的领带长度为 a，则他 / 她会产生 $\max\{a-b,0\}$ 个单位的**奇怪感**。整场派对的**奇怪度**定义为所有员工中最大的奇怪感。

由此，我们定义 C_k 为当 CEO 选择第 k 条领带时，整场派对最后可能的最小奇怪度。

请你对于给定的 $A_1,A_2,\ldots,A_{N+1}$ 和 $B_1,B_2,\ldots,B_N$ 求出 $C_1,C_2,\ldots,C_{N+1}$。

1.2 输入格式

第一行，一个正整数 N，表示员工总数。

第二行，$N+1$ 个正整数 $A_1,A_2,\ldots,A_{N+1}$，表示每条领带的长度。

第三行，N 个正整数，$B_1,B_2,\ldots,B_N$，表示每个员工初始穿戴的领带的长度。

1.3 输出格式

一行，$N+1$ 个整数 $C_1,C_2,\ldots,C_{N+1}$。

1.4 样例

1.4.1 样例输入 1

```
3
4 3 7 6
2 6 4
```

1.4.2 样例输出 1

```
2 2 1 1
```

1.4.3 样例解释 1

以下为一场试戴派对的例子：

- CEO 选择第 4 条领带。
- 员工 1 选择第 1 条领带，员工 2 选择第 2 条领带，员工 3 选择第 3 条领带。

- 每个员工试戴其选择的领带。

此时，所有员工的奇怪感分别为 $2, 0, 3$，故整场派对的奇怪度为 3。
实际上，我们可以通过改变员工的决策将整场派对的奇怪度减少到 1。例如：

- CEO 选择第 4 条领带。
- 员工 1 选择第 2 条领带，员工 2 选择第 3 条领带，员工 3 选择第 1 条领带。
- 每个员工试戴其选择的领带。

此时，所有员工的奇怪感分别为 $1, 1, 0$，故整场派对的奇怪度为 1。
若 CEO 选择第 4 条领带，这便是可能的最小的奇怪度，因此 $C_4 = 1$。

1.4.4 样例输入 2

```
5
4 7 9 10 11 12
3 5 7 9 11
```

1.4.5 样例输出 2

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

1.5 子任务

对于所有测试数据，$1 \le N \le 2 \times 10^5, 1 \le A_i \le 10^9, 1 \le B_j \le 10^9$ $(1 \le i \le N+1, 1 \le j \le N)$。

子任务编号	分值	$N \le$
1	1	10
2	8	2000
3	91	2×10^5

2 准高速电车

2.1 题目描述

X 铁路公司是 X 国唯一的铁路公司。

在某条铁路沿线共有 N 座车站，依次编号为 $1 \ldots N$。目前，正在服役的车次按照运行速度可分为两类：**高速电车**（简称快车）与**普通电车**（简称慢车）。

- 慢车每站都停。乘慢车时，对于任意一座车站 $i(1 \leq i < N)$，车站 i 到车站 $i+1$ 用时均为 A。
- 快车只在车站 $S_1, S_2, \ldots, S_M$ 停车 $(1 = S_1 < S_2 < \cdots < S_M = N)$。乘快车时，对于任意一座车站 $i(1 \leq i < N)$，车站 i 到车站 $i+1$ 用时均为 B。

X 铁路公司现拟开设第三类车次：**准高速电车**（简称准快车）。乘准快车时，对于任意一座车站 $i(1 \leq i < N)$，车站 i 到车站 $i+1$ 用时均为 C。准快车的停站点尚未确定，但满足以下条件：

- 快车在哪些站停车，准快车就得在哪些站停车。
- 准快车必须恰好有 K 个停站点。

X 铁路公司希望，在 T 分钟内（**不含换乘时间**），车站 1 可以抵达的车站（不含车站 1）的数量尽可能多。但是，**「后经过的车站的编号」必须比「先经过的车站的编号」大**。

求出在 T 分钟内，可抵达车站的最大数目。

2.2 输入格式

第一行有三个整数 N, M, K，用空格分隔。

第二行有三个整数 A, B, C，用空格分隔。

第三行有一个整数 T。

在接下来的 M 行中，第 i 行有一个整数 S_i。

输入的所有数的含义见题目描述。

2.3 输出格式

一行，一个整数，表示在 T 分钟内，可抵达车站的最大数目。

2.4 样例

2.4.1 样例输入 1

```
10 3 5
10 3 5
30
1
6
10
```

2.4.2 样例输出 1

```
8
```

2.4.3 样例解释 1

在这组样例中，这条铁路上有 10 个车站，快车在车站 1, 6, 10 停车。如果准快车在车站 1, 5, 6, 8, 10 停车，除车站 9 外的其它所有车站都可在 30 分钟内到达。

以下是从地点 1 到达某些站点的最快方案：

- 到达车站 3：乘坐慢车，耗时 20 分钟。
- 到达车站 7：先乘坐快车，在车站 6 转慢车，耗时 25 分钟。
- 到达车站 8：先乘坐快车，在车站 6 转准快车，耗时 25 分钟。
- 到达车站 9：先乘坐快车，在车站 6 转准快车，在车站 8 再转慢车，耗时 35 分钟。

2.4.4 样例输入 2

```
10 3 5
10 3 5
25
1
6
10
```

2.4.5 样例输出 2

```
7
```

2.4.6 样例输入 3

```
90 10 12
100000 1000 10000
10000
1
10
20
30
40
50
60
70
80
90
```

2.4.7 样例输出 3

```
2
```

2.4.8 样例输入 4

```
12 3 4
10 1 2
30
1
11
12
```

2.4.9 样例输出 4

```
8
```

2.4.10 样例输入 5

```
300 8 16
345678901 123456789 234567890
12345678901
1
10
77
82
137
210
297
300
```

2.4.11 样例输出 5

```
72
```

2.4.12 样例输入 6

```
1000000000 2 3000
1000000000 1 2
1000000000
1
1000000000
```

2.4.13 样例输出 6

```
3000
```

2.5 子任务

对于所有数据，$1 \le N \le 10^9, 2 \le M \le K \le 3000, K \le N, 1 \le B < C < A \le 10^9, 1 \le T \le 10^{18}, 1 = S_1 < S_2 < \cdots < S_M = N$。

- 子任务 1（18 分）：$N \le 300, K - M = 2, A \le 10^6, T \le 10^9$。
- 子任务 2（30 分）：$N \le 300$。
- 子任务 3（52 分）：无特殊约束。

3 关灯

3.1 题目描述

Y 君在玩一个游戏，这个游戏由 n 个灯和 n 个开关组成，给定这 n 个灯的初始状态，下标为从 1 到 n 的正整数。

每个灯有两个状态亮和灭，我们用 1 来表示这个灯是亮的，用 0 表示这个灯是灭的，游戏的目标是使所有灯都灭掉。

但是当操作第 i 个开关时，所有编号为 i 的约数（包括 1 和 i）的灯的状态都会被改变，即从亮变成灭，或者是从灭变成亮。

Y 君发现这个游戏很难，于是想到了这样的一个策略，每次等概率随机操作一个开关，直到所有灯都灭掉。

这个策略需要的操作次数很多，Y 君想到这样的一个优化。如果当前局面，可以通过操作小于等于 k 个开关使所有灯都灭掉，那么他将不再随机，直接选择操作次数最小的操作方法（这个策略显然小于等于 k 步）操作这些开关。

Y 君想知道按照这个策略（也就是先随机操作，最后小于等于 k 步，使用操作次数最小的操作方法）的操作次数的期望。

这个期望可能很大，但是 Y 君发现这个期望乘以 n 的阶乘一定是整数，所以他只需要知道这个整数对 100003 取模之后的结果。

3.2 输入格式

第一行两个整数 n, k。

接下来一行 n 个整数，每个整数是 0 或者 1，其中第 i 个整数表示第 i 个灯的初始情况。

3.3 输出格式

输出一行，为操作次数的期望乘以 n 的阶乘对 100003 取模之后的结果。

3.4 样例

3.4.1 样例输入 1

```
4 0
0 0 1 1
```

3.4.2 样例输出 1

```
512
```

3.4.3 样例输入 2

```
5 0
1 0 1 1 1
```

3.4.4 样例输出 2

```
5120
```

3.5 子任务

对于 0% 的测试点，和样例一模一样；

对于另外 30% 的测试点，$n \le 10$；

对于另外 20% 的测试点，$n \le 100$；

对于另外 30% 的测试点，$n \le 1000$；

对于 100% 的测试点，$1 \le n \le 100000, 0 \le k \le n$；

对于以上每部分测试点，均有一半的数据满足 $k = n$。

3.6 提示

对于一个离散型随机变量 X，若其可能的取值为 $X_1, X_2, \ldots, X_k$，各取值对应的概率分别是 $P_1, P_2, \ldots, P_k$（满足 $\sum_{i=1}^{k} P_i = 1$），则 X 的期望 $E(X)$ 满足如下等式：

$$E(X) = \sum_{i=1}^{k} X_i P_i$$

4 排序

4.1 题目描述

小 Z 喜欢上了数字序列。因而他经常研究关于序列的一些奇奇怪怪的问题，现在他在研究一个难题，需要你来帮助他。

这个难题是这样子的：给出一个 1 到 n 的全排列，现在对这个全排列序列进行 m 次局部排序，排序分为两种：

1. $(0, l, r)$ 表示将区间 $[l, r]$ 的数字升序排序；
2. $(1, l, r)$ 表示将区间 $[l, r]$ 的数字降序排序。

排序后询问第 q 位置上的数字。

4.2 输入格式

输入数据的第一行为两个整数 n 和 m。n 表示序列的长度，m 表示局部排序的次数。

第二行为 n 个整数，表示 1 到 n 的一个全排列。

接下来输入 m 行，每一行有三个整数 op, l, r，op 为 0 代表升序排序，op 为 1 代表降序排序, l, r 表示排序的区间。最后输入一个整数 q，q 表示排序完之后询问的位置。

4.3 输出格式

输出数据仅有一行，一个整数，表示按照顺序将全部的部分排序结束后第 q 位置上的数字。

4.4 样例

4.4.1 样例输入

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

4.4.2 样例输出

```
5
```

4.5 子任务

对于 30% 的数据，$n, m \leq 100$。

对于所有数据，$1 \leq n, m \leq 10^5$，$1 \leq q \leq n$。