NOIP 模拟题

第 13 套

题目名称	飞行时间	二阶和	合人问题
题目类型	传统型	传统型	传统型
目录	timezone	sumsum	merge
可执行文件名	timezone	sumsum	merge
输入文件名	timezone.in	sumsum.in	merge.in
输出文件名	timezone.out	sumsum.out	merge.out
每个测试点时限	2.0 秒	6.0 秒	1.0 秒
内存限制	512 MB	512 MB	512 MB
测试点/包数目	1	10	10
测试点是否等分	否	是	否

提交源程序文件名

对于 C++ 语言	timezone.cpp	sumsum.cpp	merge.cpp
对于 C 语言	timezone.c	sumsum.c	merge.c
对于 Pascal 语言	timezone.pas	sumsum.pas	merge.pas

编译选项

对于 C++ 语言	-O2 -std=c++14
对于 C 语言	-O2 -std=c14
对于 Pascal 语言	-O2

飞行时间（timezone）

【问题背景】

小 h 前往美国参加比赛。小 h 的女朋友发现小 h 上午十点出发，上午十二点到达美国，于是感叹到“现在飞机飞得真快，两小时就能到美国了”。

小 h 对超音速飞行感到十分恐惧。仔细观察后发现飞机的起降时间都是当地时间。由于北京和美国东部有 12 小时时差，故飞机总共需要 14 小时的飞行时间。

不久后小 h 的女朋友去中东交换。小 h 并不知道中东与北京的时差。但是小 h 得到了女朋友来回航班的起降时间。小 h 想知道女朋友的航班飞行时间是多少。

【问题描述】

对于一个可能跨时区的航班，给定来回程的起降时间。假设飞机来回飞行时间相同，求飞机的飞行时间。

【输入格式】

从文件 ***timezone.in*** 中读入数据。

一个输入文件包含多组数据。

输入第一行为一个正整数 T，表示输入数据组数。

每组数据包含两行，第一行为去程的 起降 时间，第二行为回程的 起降 时间。

起降时间的格式如下

```
h1:m1:s1 h2:m2:s2
```

或

```
h1:m1:s1 h3:m3:s3 (+1)
```

或

```
h1:m1:s1 h4:m4:s4 (+2)
```

表示该航班在当地时间 h1 时 m1 分 s1 秒起飞；

第一种格式表示在当地时间 当日 h2 时 m2 分 s2 秒降落；

第二种格式表示在当地时间 次日 h3 时 m3 分 s3 秒降落；

第二种格式表示在当地时间 第三天 h4 时 m4 分 s4 秒降落。

对于此题目中的所有以 h:m:s 形式给出的时间，保证 $0 \le h \le 23, 0 \le m, s \le 59$；对于不足两位数的情况，保证添加了前导零补齐两位。

【输出格式】

输出到文件 ***timezone.out*** 中。

对于每一组数据输出一行一个时间 `hh:mm:ss`，表示飞行时间为 `hh` 小时 `mm` 分 `ss` 秒。

注意，当时间为一位数时，要补齐前导零。如三点四分五秒应写为 `03:04:05`。

【样例输入】

```
3
17:48:19 21:57:24
11:05:18 15:14:23
17:21:07 00:31:46 (+1)
23:02:41 16:13:20 (+1)
10:19:19 20:41:24
22:19:04 16:41:09 (+1)
```

【样例输出】

```
04:09:05
12:10:39
14:22:05
```

【限制与约定】

保证输入时间合法，飞行时间不超过 `24` 小时。

二阶和（sumsum）

【题目描述】

数列的“二阶和”在数值分析、物理模拟等领域非常常见。现在我们要构造一个数据结构来维护“二阶和”。

给出一个长度为 n 的数列 $\{a_1, a_2, \ldots, a_n\}$ 并对其进行 m 次操作。操作分为两类：

1. 区间加。给出三个参数 l, r, d，对于 $l \le i \le r$，将数列中的 a_i 替换成 $a_i + d$。
2. 区间询问二阶和。给出两个参数 l, r，输出子数列 $\{a_l, a_{l+1}, \ldots, a_r\}$ 的二阶和除以 $1,000,000,007$ 的余数。

对于长度为 m 的数列 $\{b_1, b_2, \ldots, b_m\}$，它的**一阶前缀和数列**是一个长度为 m 的数列 $\{s_1, s_2, \ldots, s_m\}$，其中

$$s_i = \sum_{j=1}^{i} b_j$$

它的**二阶和**为

$$\sum_{i=1}^{m} s_i$$

也就是说，一个数列一阶前缀和的第 i 项等于数列前 i 项之和，二阶和等于所有一阶前缀和之和。

现在请你处理上述操作。

【输入格式】

从文件 ***sumsum.in*** 中读入数据。

输入的第一行包含两个正整数 n, m，第二行包含 n 个非负整数 a_i，保证 $0 \le a_i < 1,000,000,007$。

接下来 m 行，每行描述一个操作。操作一定以下列之一的形式给出：

- 1 l r d，表示区间加，保证 $1 \le l \le r \le n$，$0 \le d < 1,000,000,007$。
- 2 l r，表示区间询问二阶和，保证 $1 \le l \le r \le n$。

【输出格式】

输出到文件 ***sumsum.out*** 中。

对于每个区间询问二阶和，输出一行一个整数，表示所询问的二阶和除以 $1,000,000,007$ 的余数。

【样例 1 输入】

```
10 10
62 95 16 57 28 17 90 76 1 18
2 3 5
1 3 8 59
2 4 7
1 3 6 0
1 1 10 62
2 1 8
1 1 8 91
1 1 6 33
2 3 10
1 4 9 46
```

【样例 1 输出】

```
190
1026
5432
9428
```

【样例 1 解释】

每次操作的中间结果如下：

1. $\{a_3, \ldots, a_5\} = \{16, 57, 28\}$，
   $\{s_l, \ldots, s_r\} = \{16, 73, 101\}$。
2. $\{a_i\} = \{62, 95, 75, 116, 87, 76, 149, 135, 1, 18\}$。
3. $\{a_4, \ldots, a_7\} = \{116, 87, 76, 149\}$，
   $\{s_l, \ldots, s_r\} = \{116, 203, 279, 428\}$。
4. $\{a_i\} = \{62, 95, 75, 116, 87, 76, 149, 135, 1, 18\}$。
5. $\{a_i\} = \{124, 157, 137, 178, 149, 138, 211, 197, 63, 80\}$。
6. $\{a_1, \ldots, a_8\} = \{124, 157, 137, 178, 149, 138, 211, 197\}$，
   $\{s_l, \ldots, s_r\} = \{124, 281, 418, 596, 745, 883, 1094, 1291\}$。
7. $\{a_i\} = \{215, 248, 228, 269, 240, 229, 302, 288, 63, 80\}$。
8. $\{a_i\} = \{248, 281, 261, 302, 273, 262, 302, 288, 63, 80\}$。
9. $\{a_3, \ldots, a_{10}\} = \{261, 302, 273, 262, 302, 288, 63, 80\}$，
   $\{s_l, \ldots, s_r\} = \{261, 563, 836, 1098, 1400, 1688, 1751, 1831\}$。

10. $\{a_i\} = \{248, 281, 261, 348, 319, 308, 348, 334, 109, 80\}$。

【样例 2】

见选手目录下的 ***sumsum/sumsum2.in*** 与 ***sumsum/sumsum2.ans***。

【子任务】

各组测试数据的规模如下

测试点	$n = m =$
1	30
2	300
3	10^3
4	$3,000$
5	$5,000$
6	10^5
7	2×10^5
8	3×10^5
9	4×10^5
10	5×10^5

合人问题（merge）

【题目描述】

众所周知，小葱同学擅长合并，尤其把两个人合并成一个人。但小葱只擅长合并两个人的情况，当有很多个人之后就会比较苦恼。现在小葱给了你 n 个人排成一个环（即第一个人和最后一个人相邻），并且每个人左手和右手上各有一个数。每次你可以合并相邻的两个人，合并这两个人的代价为他们右手的数的差的绝对值，合并之后会变成一个新的人，新的人左手的数是原来左边的人左手的数，右手的数是原来右边的人右手的数。现在小葱希望你通过 $n-1$ 次合并把所有人合并成一个人，并且使得代价最小。求最小代价。

【输入格式】

从文件 ***merge.in*** 中读入数据。
第一行包括 1 个正整数 n。
接下来 n 行每行 2 个正整数代表每个人左手和右手的数。

【输出格式】

输出到文件 ***merge.out*** 中。
输出一行一个整数代表最小代价。

【样例 1 输入】

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

【样例 1 输出】

```
2
```

【样例 1 解释】

先合并前两个人。

【子任务】

对于 50% 的数据，$n \le 10$。

对于 100% 的数据，$1 \le n \le 100$，手上的数均为不超过 10^3 的正整数。