NOI2024 模拟赛

NOI2024 Simulation

时间：2024 年 2 月 21 日

题目名称	决斗	炸飞机	肿胃数
题目类型	传统题	传统题	传统题
目录	`duel`	`aircraft`	`nediam`
可执行文件名	`duel`	`aircraft`	`nediam`
输入文件名	`duel.in`	`aircraft.in`	`nediam.in`
输出文件名	`duel.out`	`aircraft.out`	`nediam.out`
每个测试点时限	1 秒	2 秒	3 秒
内存限制	512 MB	1024 MB	512 MB
子任务数目	10	7	10
测试点是否等分	是	否	是

提交源程序文件名

对于 C++ 语言	`duel.cpp`	`aircraft.cpp`	`nediam.cpp`

编译选项

对于 C++ 语言	`-O2 -std=c++14`

注意事项（请仔细阅读）

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. 提交的程序代码文件的放置位置请参考各省的具体要求。
4. 因违反以上三点而出现的错误或问题，申诉时一律不予受理。
5. 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）。
6. 选手提交的程序源文件必须不大于 100KB。
7. 程序可使用的栈空间内存限制与题目的内存限制一致。
8. 只提供 Linux 格式附加样例文件。
9. 评测在当前最新公布的 NOI Linux 下进行，各语言的编译器版本以此为准。

决斗（duel）

【题目描述】

n 个人进行一场 CF Duel 比赛，比赛流程如下：

1. n 个人排成一列，第 i 个人的初始能力值为 a_i。
2. 裁判每次选出**相邻**的两个人进行决斗，设两个人的能力值为 x 和 y，那么：
   - 如果 $x \neq y$，那么能力值大的人获胜。
   - 如果 $x = y$，那么裁判指定一个人获胜。
   - 败者从队列中移除，胜者的能力值增加 1。
3. 进行 $n-1$ 次决斗后，仅剩下的一个人为最终胜者。

对于一个长度为 n 序列 a，我们定义 $f(a)$ 是一个长度为 n 的序列，满足：

$$\forall 1 \leq i \leq n : f(a)_i = \begin{cases} 1 & ,i \text{ 可能为最终胜者} \\ 0 & ,i \text{ 不可能为最终胜者} \end{cases}$$

其中，“i 可能为最终胜者”表示：存在一种安排选手决斗顺序以及谁获胜（对于 $x = y$ 的情形）的方式，使得 i 为最终胜者。

给定 n 以及一个长度为 n 的序列 c，以及 n 个集合 $S_1, \ldots, S_n$，计数有多少个满足如下条件的长度为 n 的序列 a：

$$\forall 1 \leq i \leq n : a_i \in S_i \wedge (c_i = 2 \vee f(a)_i = c_i)$$

答案对 998244353 取模。

【输入格式】

从文件 ***duel.in*** 中读入数据。

第一行两个整数 n, m。

第二行 n 个整数表示 $c_1, c_2, \ldots, c_n$。

接下来 n 行，第 i 行一个长度为 m 的 01 串 s_i，满足 $\forall 1 \leq j \leq m : s_{i,j} = [j \in S_i]$。

【输出格式】

输出到文件 ***duel.out*** 中。

输出一行一个整数表示答案。

【样例 1 输入】

```
5 5
0 1 2 2 2
10100
01000
01000
00010
00001
```

【样例 1 输出】

```
1
```

【样例 1 解释】

可能的序列 a 有两种：$[1,2,2,4,5]$ 或 $[3,2,2,4,5]$。
可以计算得到：$f([1,2,2,4,5])=[0,1,1,1,1]$，$f([3,2,2,4,5])=[1,1,1,1,1]$。

【样例 2】

见选手目录下的 ***duel/duel2.in*** 与 ***duel/duel2.ans***。
该样例满足测试点 1 的限制。

【样例 3】

见选手目录下的 ***duel/duel3.in*** 与 ***duel/duel3.ans***。
该样例满足测试点 5, 6 的限制。

【样例 4】

见选手目录下的 ***duel/duel4.in*** 与 ***duel/duel4.ans***。
该样例满足测试点 8, 9, 10 的限制。

【数据范围】

对于所有数据，$1\le n\le 30$，$1\le m\le 40$，$0\le c_i\le 2$。

测试点编号	$n \leq$	$m \leq$	特殊限制
$1, 2$	5	5	无
$3, 4$	30	40	$\prod_{i=1}^{n} \lvert S_i \rvert \leq 10^7$
$5, 6$			$c_1 = 1, \forall\, 2 \leq i \leq n : c_i = 2$
7			$\exists 1 \leq i \leq n : c_i = 1 \wedge \forall\, j \neq i : c_j = 2$
$8, 9, 10$			无

炸飞机（aircraft）

【题目背景】

《炸飞机》是一种推理游戏，主要通过炸取区格得到的信息“空，伤，沉”来判断对方飞机的摆放位置，最先将对方飞机击沉的人获胜。

——百度百科

【题目描述】

比起如何炸飞机，小 ♠ 更加关心如何把飞机铺满整个棋盘。

总共有 n 个棋盘，第 i 个棋盘是一个长为 a_i，宽为 k 的矩形，被均匀分成了 $a_i \times k$ 个格子。有四种飞机，分别为 A,B,C,D 型，如图：

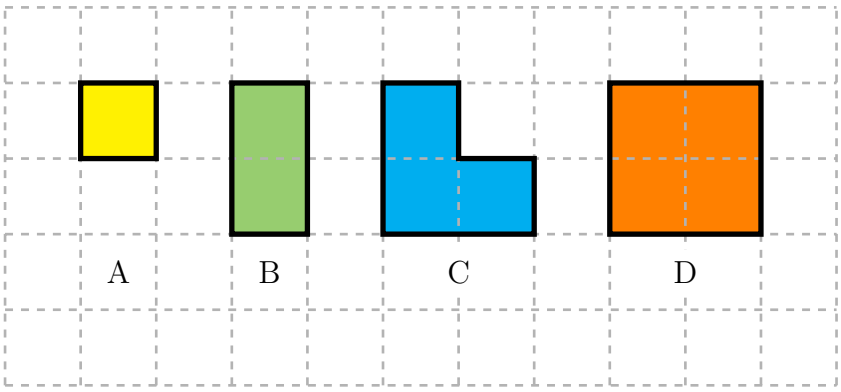

图 1: 四种类型的飞机

组成飞机的每一个格子与棋盘上一个格子大小相同。飞机可以进行旋转或翻转。将飞机放进棋盘时，要求飞机的每一个格子都与棋盘上的某个格子完全重合，并且不同的飞机不能相交。

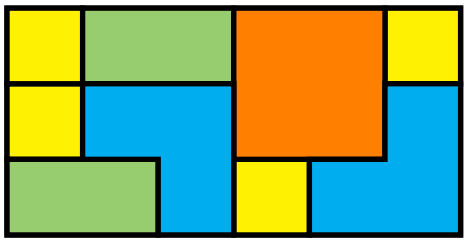

图 2: 一种 6×3 的棋盘的放置方案

然而，实际游戏时，**小 ♠ 只能使用四种飞机的一个子集**。

现在小 ♠ 关心每一个棋盘有多少种放飞机的方式，满足棋盘的每一个格子都被一个飞机覆盖。两种放飞机方式不同当且仅当存在两个相邻格子，在一种方式中两个格子属于同一个飞机，而在另一种方式中不属于（棋盘不能旋转，即旋转后相同不算相同）。

游戏是多变的，所以你需要顺次处理 q 个事件，每个事件形如：

1. 给定 l, r, x，对于 $\forall l \le i \le r$，令 $a_i \leftarrow a_i + x$。
2. 给定 l, r，记 $f(i)$ 表示第 i 个棋盘有多少种放飞机的方式，求 $\sum_{i=l}^{r} f(i) \bmod (10^9+7)$。

【输入格式】

从文件 ***aircraft.in*** 中读入数据。

第一行一个正整数 k。

第二行一个字符串 s，保证其为 ABCD 的子序列，字母 α 在 s 内就代表 α 型飞机可以使用。

第三行两个正整数，n, q。

第四行 n 个正整数表示初始的 $a_1, a_2, \ldots, a_n$。

接下来 q 行，首先一个整数 o。

- 若 $o = 1$，接下来三个正整数 l, r, x，代表一个第一类事件。
- 若 $o = 2$，接下来两个正整数 l, r，代表一个第二类事件。

【输出格式】

输出到文件 ***aircraft.out*** 中。

对于每个第二类事件，输出一行一个整数表示答案。

【样例 1 输入】

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

【样例 2 输出】

```
12
28
```

【样例 2 解释】

对于第一个事件，三个棋盘长度分别为 $1,1,2$，答案为 $f(1)+f(1)+f(2)=2+2+8=12$。

对于第三个事件，两个棋盘长度分别为 $1,3$，答案为 $f(1)+f(3)=2+26=28$。

【样例 2】

见选手目录下的 ***aircraft/aircraft2.in*** 与 ***aircraft/aircraft2.ans***。

【样例 3】

见选手目录下的 ***aircraft/aircraft3.in*** 与 ***aircraft/aircraft3.ans***。

【数据范围】

对于所有数据，s 为 ABCD 的非空子序列，$1\le k\le 6,\ 1\le n\le 2\times 10^4,\ 1\le q\le 10^4,\ 1\le a_i,x\le 10^{18},\ 1\le l\le r\le n,\ o\in\{1,2\}$。

子任务编号	子任务分值	$n\le$	$q\le$	$k\le$	特殊限制
1	6	2×10^4	10^4	6	$s=$ A
2	10				$a_i\le 10$，没有第一类事件
3	18	100	100		无
4	13	2×10^4	10^4	1	
5	11			4	
6	19			5	
7	23			6	

肿胃数（nediam）

【题目背景】

多年以后，面对残缺的序列 a，小 ♣ 将会回想起他写下 Nediam 函数的那个遥远的下午。

【题目描述】

对于长度为 k 的序列 A，定义 A 的中位数为 A 的第 $\left\lfloor \frac{k+1}{2} \right\rfloor$ 小值。

小 ♣ 曾写了一份求中位数的程序（序列下标从 0 开始）：

Algorithm 1: A fake algorithm to get the median of a sequence.

Input: An sequence A

Output: The fake median of sequence A

```
function Sort(A):
    return The sequence of A sorted in non-decreasing order.
end
function Nediam(A):
    n ← |A|;
    if n ≤ 2 then
        return min{x | x ∈ A};
    end
    tl ← ⌊(n − 1)/3⌋ + 1;
    tr ← ⌊(2n − 1)/3⌋ + 1;
    c ← [Nediam(A[0 : tl − 1]), Nediam(A[tl : tr − 1]), Nediam(A[tr : n − 1])];
    c′ ← Sort(c);
    return c′[1];
end
```

显然这份程序求的不是中位数，我们把 Nediam(A) 称为序列 A 的“肿胃数”。

某天，小 ♣ 拿到了一个序列 a，他惊讶地发现序列 a 的肿胃数和中位数是相等的！

然而，时过境迁，小 ♣ 早已将序列某些位置的值忘却，但是他还记得序列 a 的每个元素都是 $[0, m)$ 中的整数。

现在给定 n, m，求有多少个可能的序列 a，使得 a 的肿胃数与中位数相等。答案对 $10^9 + 7$ 取模。

【输入格式】

从文件 ***nediam.in*** 中读入数据。

本题有多组测试数据。

第一行一个整数 T 表示测试数据组数。

对于每一组数据：

第一行两个整数 n, m。

第二行 n 个整数 $a_0, a_1, \ldots, a_{n-1}$。若 $a_i = -1$ 则表示小 ♣ 忘了 i 位置的值。

【输出格式】

输出到文件 ***nediam.out*** 中。

对于每组测试数据，输出一行一个整数表示答案。

【样例 1 输入】

```
3
6 5
-1 1 4 5 1 4
7 9
1 9 -1 9 8 -1 0
12 100000
1 99834 56275 43172 -1 -1 756 -1 2345 1078 99 -1
```

【样例 1 输出】

```
2
46
851501245
```

【样例 2】

见选手目录下的 ***nediam/nediam2.in*** 与 ***nediam/nediam2.ans***。

【样例 3】

见选手目录下的 ***nediam/nediam3.in*** 与 ***nediam/nediam3.ans***。

【样例 4】

见选手目录下的 ***nediam/nediam4.in*** 与 ***nediam/nediam4.ans***。

【样例 5】

见选手目录下的 ***nediam/nediam5.in*** 与 ***nediam/nediam5.ans***。

【样例 6】

见选手目录下的 ***nediam/nediam6.in*** 与 ***nediam/nediam6.ans***。

【样例 7】

见选手目录下的 ***nediam/nediam7.in*** 与 ***nediam/nediam7.ans***。

【数据范围】

对于所有数据，$1 \le T \le 100,\ 1 \le n \le 3^8 = 6561,\ 1 \le m \le 10^9,\ a_i \in \{-1\} \cup [0, m)$。

记 $k = \sum_{i=0}^{n-1} [a_i = -1]$。

测试点编号	$T \le$	$n \le$	$k \le$	$m \le$
1	100	81	5	20
2			8	100
3	10		12	
4, 5		$3^5 = 243$	20	5×10^4
6, 7		$3^6 = 729$		10^9
8, 9, 10	5	$3^8 = 6561$	30	