另一个简单模拟赛

2025 年 2 月 8 日

题目名称	抽卡	分裂	假币问题
题目类型	传统	传统	交互
目录	`card`	`split`	`coin`
可执行文件名	`card`	`split`	`coin`
输入文件名	`card.in`	`split.in`	
输出文件名	`card.out`	`split.out`	
每个测试点时限	1 秒	1 秒	1 秒
内存限制	1024 MB	1024 MB	1024 MB

编译选项：`-O2 -lm -std=c++17`

Hint: 不保证题目按难度顺序排列.

Hint: 请不要使用网络. 骗哥们可以, 别把你自己也骗到了就行.

1 抽卡

现在有一个神秘的抽卡机器, 其有 n 行 m 列, 每一个格子里有一张卡. 每当你抽一次卡, 你将会获得一对相邻 (四联通) 格子中的卡 (卡可以重复获得), 获得每一对相邻格子卡的概率都相等. 具体的, 在 $2nm-n-m$ 个相邻的格子对中, 随机一个格子对获得这两个格子中的卡. 每次抽卡之间是独立的.

在 $n \times m$ 张卡中, 有一些卡是你想要的. 你想要知道, 期望多少次抽卡后你可以得到所有你想要的卡.

1.1 输入格式

第一行两个整数 $n, m(1 \leq n \leq 7, 2 \leq m \leq 200)$.

接下来 n 行, 每行一个长为 m 的 01 串, 1 表示你想要的卡, 0 表示你不关心的卡.

1.2 输出格式

一行一个整数, 表示答案, 你只需要输出答案对 998244353 取模后得到的结果.

1.3 样例 1 输入

```
1 3
111
```

1.4 样例 1 输出

```
3
```

1.5 样例 2 输入

```
3 3
001
101
010
```

1.6 样例 2 输出

```
404051295
```

1.7 样例 3

见 `card/card3.in` 和 `card/card3.ans`.

1.8 数据范围与限制

对于全部数据, $1 \leq n \leq 7, 2 \leq m \leq 200$.

记 cnt 为输入 01 串中 1 的个数

子任务编号	输入规模范围	得分
1	$n = 1$	20
2	$cnt \leq 6$	10
3	$cnt \leq 20$	20
4	$n \leq 6, m \leq 100$	35
5	无特殊限制	15

2 分裂

考虑一个 4 行无穷列的二维网格, 假设 (x, y) 表示第 $x(0 \le x < 4)$ 行第 $y(0 \le y)$ 列的格子.

现有一种神秘生物, 假如有一只位于 (x, y), 其可以分裂为两只, 分别位于 $(x, y+1)$ 和 $((x+1) \bmod 4, y+1)$, 注意, 这需要满足这两个格子都**没有**神秘生物.

一开始, 有一只神秘生物位于 $(0, 0)$, 其可能分裂了若干次, 你需要对 $i = 1, \ldots, n$ 分别求出其分裂了恰好 i 次后有多少种可能的位置分布. 两种位置分布不同当且仅当存在一个格子仅在其中一种分布中有神秘生物.

由于答案可能很大, 你只需要输出答案对 m 取模得到的结果.

2.1 输入格式

一行两个整数 $n, m(1 \le n \le 10^6, 10^8 \le m \le 10^9)$.

2.2 输出格式

n 行, 每行一个整数, 第 i 行的表示恰好 i 次分裂后可能的位置分布数量对 m 取模后得到的结果.

2.3 输入输出样例

见 `split/split1.in` 和 `split/split1.ans`.

2.4 数据范围

对于所有数据, $1 \le n \le 10^6, 10^8 \le m \le 10^9$.

子任务编号	输入规模范围	分数
1	$n \le 20$	1
2	$n \le 30$	14
3	$n \le 5 \times 10^2$	15
4	$n \le 5 \times 10^3$	15
5	$n \le 5 \times 10^4$	15
6	$n \le 5 \times 10^5$	20
7	$n \le 10^6$	20

3 假币问题

这是一个交互式问题.

小 W 是一名硬币收藏家. 在完成了他的欧洲长途旅行后, 他带回了 $n(2 \le n \le 100)$ 枚漂亮的硬币. 不幸的是, 他知道这 n 枚硬币中正好有一枚是假币. 所有 $n-1$ 枚真币的重量均为 x, 而假币更轻, 其重量为 $y(1 \le y < x \le 10^7, x, y \in \mathbb{Z})$. 注意, x 和 y 的具体值是未知的.

为了找到假币, 小 W 首先将所有硬币从 1 到 n 编号. 之后, 他多次使用天平：在一次操作中, 小 W 取出 n 枚硬币中的某一子集, 将它们放到天平上, 测量它们的总重量. 当然, 作为一名经验丰富的硬币收藏家, 小 W 使用了最少的操作次数以保证找到假币.

在本题中, 你需要做同样的事情：对于给定的 n, 在保证找到假币的情况下使操作次数尽量小. 注意, 假设 A 是对 n 枚硬币, 任意 x, y 和假币编号的情形保证找到假币的操作次数最大值, 你只需要保证操作次数 $\le A$ 即可.

3.1 实现模式

你不需要, 也不应该实现主函数. 你只需要实现函数：`int Ask(std::vector<int> a)` 与 `int Solve(int n)`. 它们的含义分别是询问 a 中的所有硬币的重量和, 和开始解决一个规模为 n 的数据.

每个测试点有 $T = 3000$ 组数据. 每个测试点中, 交互库会调用 T 次 `Solve` 函数. 当测试完以后, 整个程序结束, 判断分数. 为了避免不必要的麻烦, 每个测试点中调用的 n 都相同.

下发文件中提供了 `sample_coin.cpp`, `grader.cpp`, 以及 `coin.h`, 推荐的实现方式是将 `sample_coin.cpp` 备份并重命名为 `coin.cpp`, 并在此基础上作答与调试. 若如此, 你只需要修改 `coin.cpp`, 并使用 `g++ coin.cpp grader.cpp -O2 -std=c++17 -o coin` 指令进行编译, 或运行 `compilerun.bat` 进行编译和运行.

3.2 交互示例

样例交互库并不会评分, 但会判断你的答案是否正确. 你需要依次输入 n, T, 分别表示硬币个数和数据组数. 每组数据读入三个整数 x, y, c, 其中 c 是假币的编号, 以开始交互.

输入：

```
4 1
2 1 3
```

输出：

```
? 1
2
? 1 4
4
? 1 4 3
5
! 3
Your answer is correct! :D
```

输出中, `?` 表示你调用了一次询问函数, 而 `!` 表示一次 `Solve` 函数的返回值.

注意样例的 $T = 1$, 而非 3000.

3.3 数据范围、限制与评分标准

对于全部数据, $2 \leq n \leq 100, 1 \leq y < x \leq 10^7, x, y \in \mathbb{Z}$. 交互库用时不超过 100 毫秒, 空间不超过 10MB.

每一个数据的评判标准如下：

- 若你猜测的答案错误、时间超限、运行时错误、或者攻击交互库, 你获得 0 % 的分数.
- 否则, 若你猜测的次数达到或小于最差情况下猜测次数最小值, 你获得 100 % 的分数.
- 否则, 若你猜测的次数仅比最差情况下猜测次数最小值大 1, 你获得 40 % 的分数.
- 否则, 你获得 20 % 的分数.

一个测试点的得分是其中所有数据得分百分比的最小值 × 该测试点得分.

测试点编号	输入规模范围	得分
1	$n \leq 8$	20
2	$n > 8$ 且 $n = 2^k$	30
3	$n > 8$ 且 $n = 2^k + 1$	35
4	无特殊限制	15