

2026 深圳市高级中学模拟赛

CSP-J/S 2026 第二轮模拟

入门级

时间：2026 年 8 月 25 日 08:30 ~ 11:30

题目名称	史莱姆	饼干	艺术品	怕老师的小 T
题目类型	传统型	传统型	交互型	传统型
目录	slime	cookie	table	class
可执行文件名	slime	cookie	table	class
输入文件名	slime.in	cookie.in	table.in	class.in
输出文件名	slime.out	cookie.out	table.out	class.out
每个测试点时限	1.0 秒	1.0 秒	1.0 秒	4.0 秒
内存限制	128 MiB	128 MiB	256 MiB	512 MiB
测试点数目	10	10	20	10
测试点是否等分	是	是	是	是

提交源程序文件名

对于 C++ 语言	slime.cpp	cookie.cpp	table.cpp	class.cpp
-----------	-----------	------------	-----------	-----------

编译选项

对于 C++ 语言	- O2 - std=c++14 - static
-----------	---------------------------

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. `main` 函数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. 全国统一评测时采用的机器配置为：Intel(R) Core(TM) i7-8700K CPU @3.70GHz，内存 32GB。上述时限以此配置为准。
9. 只提供 Linux 格式附加样例文件。
10. 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。

史莱姆 (slime)

【题目背景】

Peter.H.N 是一名保安，他身高五尺，膘肥体重。

【题目描述】

最近，Peter.H.N 迷上了史莱姆，非常想要一个史莱姆农场。

但他并不满足于只有一个史莱姆，也不想要低级的史莱姆。

于是他找到了樱桃炸弹，从他那里买来了更高级的史莱姆。这种史莱姆每秒可以炸死周围的僵尸向他周围八个方向分别分裂一个史莱姆，直到分裂到农场边缘。

但由于 Peter.H.N 的农场非常大，这使 Peter.H.N 在农场内放完所有史莱姆后迷失在农场，只有任意史莱姆或分裂出来的史莱姆到达他的位置他才能获救。

樱桃炸弹为了让 Peter.H.N 尽快出来给他史莱姆的钱，于是他想问你 Peter.H.N 会在什么时候获救。

【输入格式】

第一行输入三个正整数 n, Px, Py ，表示史莱姆的数量，与 Peter.H.N 的位置的坐标。

接下来 n 行，每行输入两个正整数 x_i, y_i ，表示每只史莱姆的位置的坐标。

【输出格式】

输出 Peter.H.N 获救的时间（1 秒时史莱姆开始分裂）

【样例 1】

输入

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

输出

```
1 3
```

样例 1 解释

P 表示 Peter.H.N。0 表示空地。1 表示有史莱姆。

1 秒时：

```
1 P 0 0 1 1
```

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

2 秒时:

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

3 秒时:

```
1 P 1 1 1 1
2 1 1 1 1 1
3 1 1 1 1 1
4 1 1 1 1 1
5 1 1 1 1 1
```

(此时 Peter.H.N 所在的格子上也有史莱姆)

【样例 2】

见选手目录下的 slime/slime2.in 与 slime/slime2.ans。
该样例满足测试点 1 ~ 5 的约束条件。

【样例 3】

见选手目录下的 slime/slime3.in 与 slime/slime3.ans。
该样例满足测试点 6 ~ 10 的约束条件。

【数据范围】

测试点	$n \leq$	a, b, x_i, y_i
1 ~ 5	3×10^3	$1 \leq a, b, x_i, y_i \leq 3 \times 10^3$
6 ~ 10	2×10^5	$-10^{18} \leq a, b, x_i, y_i \leq 10^{18}$

饼干 (cookie)

【题目描述】

Eikooc 是一个吃货，他非常喜欢吃饼干。

但是他并不喜欢吃太大的饼干，所以对于每一个大小大于 1 的饼干，他都会将其分成大小为 $\lfloor \frac{x}{2} \rfloor$ 和 $\lceil \frac{x}{2} \rceil$ 的部分，其中 x 为这块饼干的大小。但是这也意味着他需要消耗 x 点体力去分开它。

至于剩下的大小为 1 的饼干，Eikooc 当然是愉快地吃掉啦！

现在有一个不幸的事情，Farmer Nhoj 一次性丢给了 Eikooc 足足有 T 块饼干，这使得 Eikooc 不知道自己将消耗多少体力来分割饼干。不过吃饼干这件事对于他来说还是非常幸运的，至于不幸的计算过程，就交给你吧。

【输入格式】

第一行两个正整数 C, T ，表示测试点编号和询问次数（样例测试点编号为 0）。

接下来 T 行，每行一个正整数 N ，表示初始的饼干大小。

【输出格式】

输出 T 行，每行一个整数，表示对应询问的答案。

【样例 1】

输入	
1	0 2
2	3
3	340
输出	
1	5
2	2888

样例 1 解释

对于第一个询问 $N = 3$ ：

- 初始时，有一个大小为 3 的饼干。
- Eikooc 选择大小为 3 的饼干，消耗 3 点体力，分开 3，留下大小为 $\lfloor \frac{3}{2} \rfloor = 1$ 和 $\lceil \frac{3}{2} \rceil = 2$ 的饼干。
- 此时有一个大小为 2 的饼干和一个大小为 1 的饼干。

- Eikooc 选择大小为 2 的饼干，消耗 2 点体力，分开 2，留下大小为 $\left\lfloor \frac{2}{2} \right\rfloor = 1$ 和 $\left\lceil \frac{2}{2} \right\rceil = 1$ 的饼干。
 - 此时有三个大小为 1 的饼干。
 - 此时已没有大于等于 2 的饼干，操作结束。
- 整个过程中 Eikooc 共消耗 $3 + 2 = 5$ 点体力，因此输出 5。

【样例 2】

见选手目录下的 `cookie/cookie2.in` 与 `cookie/cookie2.ans`。
该样例满足测试点 2 的约束条件。

【样例 3】

见选手目录下的 `cookie/cookie3.in` 与 `cookie/cookie3.ans`。
该样例满足测试点 5 的约束条件。

【样例 4】

见选手目录下的 `cookie/cookie4.in` 与 `cookie/cookie4.ans`。
该样例满足测试点 6 ~ 10 的约束条件。

【数据范围】

对于所有数据，保证 $1 \leq T \leq 10^4$ ， $2 \leq N \leq 10^{17}$ 。

测试点编号	T	N	特殊性质
1	≤ 10	$\leq 10^3$	无
2	$\leq 10^2$	$\leq 10^6$	无
3 ~ 4	$\leq 10^3$	$\leq 10^{12}$	无
5	≤ 40	$\leq 10^{17}$	N 为 2 的幂
6 ~ 10	$\leq 10^4$	$\leq 10^{17}$	无

特殊性质约定：若 N 为 2 的幂，则存在正整数 k 使得 $N = 2^k$ 。

艺术品 (table)

【题目描述】

这是一道交互题 (吗?)。

Kevin 正在完成 T 幅艺术品。

事实上, 这其实就是一个 N 行 M 列的表格 (或许我们可以将它看成一个围棋棋盘?), Kevin 的手上有非常非常多的黑色小卡片与白色小卡片, 他可以随意的将黑色小卡片和白色小卡片放到表格内。

但是他完成这个艺术品后需要将它交给 scy 检查。scy 是一个对艺术品有着很高要求的艺术家, 他觉得一幅合格的艺术品需要满足这些条件, 不然 Kevin 就要遭到痛斥了:

- 对于这个艺术品的每一行, 黑色卡片与白色卡片中数量较少的那种的数量应该为 A 。
- 对于这个艺术品的每一列, 黑色卡片与白色卡片中数量较少的那种的数量应该为 B 。

Kevin 可不想被骂, 但他是一个艺术造诣为 0 的手残党, 所以他把这个艰巨的任务交给了你, 那么就由你来设计这个艺术品吧!

【实现细节】

选手不需要, 也不应该实现 `main` 函数。

选手需要确保提交的程序包含头文件 `table.h`, 即在程序开头加入以下代码:

```
1 #include "table.h"
```

选手需要在提交的程序源文件 `table.cpp` 中实现以下两个函数:

```
1 void init(int c, int T);
```

- c, t 分别表示测试点编号与测试数据组数。 $c = 0$ 表示该测试点为样例。
- 对于每个测试点, 该函数会在程序开始运行时被交互库调用恰好一次。

```
1 std::vector<std::vector<int>> solve(int N, int M, int A, int B);
```

- N, M 表示矩阵的行数和列数, A, B 为题面所述的条件参数。
- 该函数需要返回一个 $n \times m$ 的二维数组 (元素仅为 0 或 1), 表示满足条件的填写方案。
- 如果无法满足条件, 请返回一个空的 `std::vector<vector<int>>`。
- 对于每个测试点, 该函数会被交互库调用恰好 t 次。

注意: 在任何情况下, 最终测试时所用的交互库运行所需时间均不会超过 0.1 秒, 所用内存为固定大小, 且均不超过 64 MiB。

【测试程序方式】

试题目录下的 **grader.cpp** 是交互库参考实现，最终测试时所用的交互库实现与该参考实现有所不同，因此选手的解法不应该依赖交互库实现。

选手可以在本题目录下使用如下命令编译得到可执行程序：

```
1 g++ grader.cpp table.cpp -o table -std=gnu++14 -O2 -static
```

对于编译得到的可执行程序：

- 可执行文件将从**标准输入**读入以下格式的数据：
 - 输入的第一行包含两个非负整数 c, t ，分别表示测试点编号和测试数据组数。
 - 接下来依次为每组测试数据，对于每组测试数据，包含一行四个非负整数 n, m, A, B 。
- 可执行文件将输出以下格式的数据至**标准输出**：
 - 对于每组测试数据，输出一行字符串表示测试结果：
 - * **Correct** 表示选手返回的结果正确；
 - * **Wrong answer** 表示选手返回的结果错误或格式不合法。

【样例 1】

【输入】

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

【输出】

```
1 Correct
2 Correct
```

【样例 1 解释】

该样例共包含两组测试数据。

对于第一组测试数据， $n = 3, m = 3, A = 1, B = 1$ 。

一种可能的交互过程为：

- 调用 `solve(3, 3, 1, 1)`，返回矩阵：

```
1 1 0 0
2 0 1 0
3 0 0 1
```

交互库验证每行和每列的 0 与 1 的较小值均为 1，判定为正确。

对于第二组测试数据, $n = 1, m = 5, A = 2, B = 0$ 。
一种可能的交互过程为:

- 调用 `solve(1, 5, 2, 0)`, 返回矩阵:

```
1 0 1 0 1 0
```

交互库验证通过, 判定为正确。

【样例 2】

见选手目录下的 `table/table2.in` 与 `table/table2.ans`。
该样例满足测试点 1 ~ 4 的约束条件。

【样例 3】

见选手目录下的 `table/table3.in` 与 `table/table3.ans`。
该样例满足测试点 5 ~ 8 的约束条件。

【样例 4】

见选手目录下的 `table/table4.in` 与 `table/table4.ans`。
该样例满足测试点 11 ~ 12 的约束条件。

【样例 5】

见选手目录下的 `table/table5.in` 与 `table/table5.ans`。
该样例满足测试点 15 ~ 20 的约束条件。

【下发文件说明】

在本试题目录下:

1. `grader.cpp` 是提供的交互库参考实现。
2. `table.h` 是头文件, 选手不用关心具体内容。
3. `template_table.cpp` 是提供的示例代码, 选手可参考并实现自己的代码。

选手注意对所有下发文件做好备份。最终评测时只测试本试题目录下的 `table.cpp`, 对该程序以外文件的修改不会影响评测结果。

【数据范围】

对于所有测试数据, 均有:

- $1 \leq T \leq 10^4$;
- $1 \leq N, M \leq 1000$;
- $0 \leq A, 2 \times A \leq m$;
- $0 \leq B, 2 \times B \leq n$;
- 保证所有测试数据的 $\sum(N \times M) \leq 10^6$;

- 输入的所有数值均为整数。

测试点编号	$T \leq$	$N, M \leq$	特殊性质
1 ~ 4	10	10	无
5 ~ 8	10^4	10	无
9 ~ 10	100	100	无
11 ~ 12	100	100	A
13 ~ 14	1	1000	A
15 ~ 20	1	1000	无

- 特殊性质 A: $A = 0$ 。

【评分方式】

注意：

- 选手不应当通过非法方式获取交互库的内部信息，如直接与标准输入、输出流进行交互。此类行为将被视为作弊；
- 最终的评测交互库与样例交互库的实现不同。

本题首先会受到和传统题相同的限制，例如编译错误会导致整道题目得 0 分，运行时错误、超过时间限制、超过空间限制等会导致相应测试点得 0 分等。选手只能在程序中访问自己定义的变量以及交互库给出的变量，尝试访问其他地址空间将可能导致编译错误或运行错误。

每次调用 `solve` 函数时，若返回的矩阵不被认为是正确的，或返回的矩阵中包含非 0/1 的非法元素，或返回的矩阵维度不匹配，则相应测试点得 0 分。

在上述条件基础上，若选手通过了该测试点的所有 T 组数据验证，则获得该测试点的满分；否则得 0 分。

怕老师的小 T (class)

【题目背景】

小 T 十分害怕老师。

【题目描述】

现在有 n 个教室，编号为 $1 \sim n$ ，教室之间有 m 条边，每条边连接 x_i, y_i 号教室，上面有 v_i 个老师。

小 T 一开始处于 1 号教室，他想要去到 n 号教室，并且碰到尽量少的老师。

为此，他斥巨资购买了 k 个老师消失器，每个老师消失器可以让一条长度不超过 a_i 的简单路径上的老师全部消失。

如果小 T 聪明绝顶，请问他最少碰到几个老师？

题目保证不存在重边，自环。

【输入格式】

第一行三个整数 n, m, k ，表示教室数量，路径数量与老师消失器数量。

接下来 m 行，每行三个整数 x_i, y_i, v_i ，意思见题目描述。

接下来一行 k 个整数 a_i ，意思见题目描述。

【输出格式】

输出一个整数，表示小 T 碰到的老师的最小数量。

【样例 1】

【输入】

```
1 13 14 1
2 1 2 1
3 3 4 1
4 5 6 1
5 7 13 1
6 1 9 4
7 11 12 1
8 10 11 100
9 8 9 100
10 8 13 100
11 13 12 1
12 10 9 1
```

```
13 2 3 1
14 4 5 1
15 6 7 1
16 2
```

【输出】

```
1 4
```

【样例 2】

见选手目录下的 class/class2.in 与 class/class2.ans。
该样例满足测试点 1 – 2 的约束条件。

【样例 3】

见选手目录下的 class/class3.in 与 class/class3.ans。
该样例满足测试点 5 的约束条件。

【样例 4】

见选手目录下的 class/class4.in 与 class/class4.ans。
该样例满足测试点 6 – 10 的约束条件。

【数据范围】

共 10 组测试数据。

数据点	$n \leq$	$m \leq$	特殊性质
1 – 2	100	1000	无
3 – 4	10^4	$10^4 - 1$	$m = n - 1$
5	10^4	10^5	$k = 0$
6 – 10	10^4	3×10^5	无

对于所有数据， $1 \leq n \leq 10^4, 1 \leq m \leq \min(3 \cdot 10^5, \frac{n(n-1)}{2}), 1 \leq x_i, y_i \leq n, 0 \leq k \leq 5, 1 \leq a_i \leq 10, 1 \leq v_i \leq 10^9$