CSP-J 模拟

题目名称	走格子	多米诺	同构连通块	舞蹈
输入/输出文件名	walk.in/out	domino.in/out	same.in/out	dance.in/out
测试点时限	$1\ \mathrm{s}$	$1\ \mathrm{s}$	$1\ \mathrm{s}$	$1\ \mathrm{s}$
内存限制	$256\mathrm{MB}$	$256\mathrm{MB}$	$256\mathrm{MB}$	$256\mathrm{MB}$
分值	100	100	100	100
测试点/子任务个数	10	10	10	10
题目类型	传统型	传统型	传统型	传统型

注意事项

1. **需要建立子文件夹。**
2. 文件名（包括程序名和输入输出文件名）必须使用英文小写。
3. 结果比较方式为忽略行末空格、文末回车后的全文比较。
4. 程序可使用的栈空间大小与该题内存空间限制一致。
5. 编译选项：`-O2 -std=c++14 -lm`。

1. 走格子（walk.cpp）

描述

在一个 n 行 n 列的矩阵里，有一个小人在走格子。

小人的初始位置在第一行第一列。

- 如果小人所在的位置不是最后一列，那么一个单位时间后，小人所在位置的列数就会加一。
- 如果小人走到了最后一列，且当前所在位置不是最后一行，那么一个单位时间后，小人会走到下一行的第一列的位置。
- 如果小人走到了最后一行的最后一列，那么一个单位时间后，小人会走到第一行第一列的位置。

那么，t 个单位时间后，小人在第几行第几列？

格式

输入格式

一行两个正整数 n, t 表示矩阵的大小和小人移动的时间。

输出格式

一行两个整数 x, y 表示小人最后的位置在第 x 行第 y 列。

样例

输入样例1

```
3 5
```

输出样例1

```
2 3
```

输入样例2

```
4 2
```

输出样例2

```
1 3
```

输入样例3

```
2 7
```

输出样例3

```
2 2
```

数据范围

对于 50% 的数据：$t \leq n^2 - 1$。

对于 80% 的数据：$t \leq n^2$。

对于 100% 的数据：$1 \leq n \leq 10$，$0 \leq t \leq 500$。

2. 多米诺（domino.cpp）

描述

小 A 在桌上用积木搭建了 n 个圆柱排成一排，第 i 个圆柱的高度为 h_i。

小 B 想把小 A 的积木全部推倒，他决定从没有被推倒的积木的最左侧或最右侧开始推，并且他希望花费的力气最少。

对于一个高度为 h_i 的圆柱，小 B 需要花费 h_i 的力气。若第 i 个圆柱是当前最左侧的圆柱，那么小 B 可以将其向右推，并产生相应的连锁反应；若第 i 个圆柱是当前最右侧的圆柱，那么小 B 可以将其向左推，并产生相应的连锁反应。

设初始的势能为当前手动推倒的圆柱 k 的高度 $p = h_k$，则此轮连锁反应中第 i 个圆柱倒向第 j 个圆柱时：

- 若 $p \geq h_j$，则第 j 个圆柱也会被推倒，并令 $p = h_j$，继续连锁反应。
- 若 $p < h_j$，则第 j 个圆柱的高度变为 $h_j - p$，并停止连锁反应。

注意：如果是从最左侧推倒圆柱，则圆柱会向右倒，相应的连锁反应也是如此；如果是从最右侧推倒圆柱，则圆柱会向左倒，相应的连锁反应也是如此。

请你求出推倒所有的圆柱所需的最小力气。

格式

输入格式

第一行一个整数 T，表示数据组数。

对于每一组数据，第一行一个整数 n，表示圆柱的数量，第二行 n 个整数，分别表示每个圆柱的高度。

输出格式

对于每组数据，输出一行一个整数表示答案。

样例

输入样例1

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

输出样例1

```
7
6
8
```

样例解释

对于第一组数据：第一次从左边推，耗费 2 点力气，使得 5 个圆柱的高度分别变为：$0,1,1,4,5$。 第二次从右边推，耗费 5 点力气，使得所有圆柱都被击倒。 共耗费 7 点力气。

对于第二组数据：从左边或者右边都可以一次性推完，共耗费 6 点力气。

对于第三组数据：第一次从右边推，耗费 4 点力气，使得 6 个圆柱的高度分别变为：$1,1,4,4,0,0$。第二次从右边推，耗费 4 点力气，使得所有圆柱都被击倒。共耗费 8 点力气。

数据范围

对于全部数据，保证：$1 \leq T \leq 10$，$1 \leq n \leq 10^5$，$1 \leq h_i \leq 10^9$。

对于 20% 的数据：$n \leq 10$。

对于 50% 的数据：$n \leq 1000$。

对于另外的 20% 的数据：存在一个 $p(1 < p < n)$，使得圆柱的高度满足
$h_1 \leq h_2 \leq \cdots \leq h_p \geq \cdots \geq h_{n-1} \geq h_n$。

对于 100% 的数据：$n \leq 10^5$。

3. 同构连通块（same.cpp）

描述

在一个 $n \times m$ 的网格图上，放置着 $n \times m$ 个有颜色的方块。

如果两个方块是四连通的（上下左右四个相邻位置），且他们的颜色相同，则这两个方块是直接连通的。如果两个方块同时与另一个方块直接连通，那么这两个方块间接连通。我们称一个方块与其直接连通和间接连通的方块共同构成一个连通块。

对于两个不同的连通块，如果可以通过平移让他们完全重合，那么这两个连通块是同构的。

现在需要知道，在网格图上，最多能找到多少个连通块，他们是**互不同构**的。

格式

输入格式

第一行两个正整数 n, m 表示网格图的大小。

接下来 n 行每行 m 个正整数，表示网格图中每一个方块的颜色。

输出格式

输出一个整数表示答案。

样例

输入样例1

```
5 6
1 2 3 4 4 5
1 2 3 3 4 5
1 2 2 3 4 5
1 6 6 7 7 8
6 6 7 7 8 8
```

输出样例1

```
7
```

样例解释

共有 8 个连通块，其中颜色为 $6, 7$ 的连通块是同构的，所以答案为 7。

数据范围

对于 30% 的数据：$1 \leq n, m \leq 20$，连通块的大小均不超过 5×5。

对于另外 30% 的数据：所有连通块均为 $1 \times x$ 或 $x \times 1$ 形态，其中 x 是任意正整数。

对于 100% 的数据：$1 \leq n, m \leq 500$，$1 \leq a_{i,j} \leq n \times m$。

4. 舞蹈（dance.cpp）

描述

在一场晚会上，n 个人围成一圈表演舞蹈。

他们围成的圈上一共有 n 个位置，分别编号为 $0 \sim n-1$，其中编号为 $n-1$ 位置的下一个位置是编号为 0 的位置。舞蹈表演开始时，演出的人分别编号为 $0 \sim n-1$ 并站在相同编号的位置上。

舞蹈表演时，有 k 个活跃位置 $0 = a_1 < a_2 < \cdots < a_k < n$，在活跃位置上的表演者会进行移动。具体来说，舞蹈表演的每一分钟，都会发生两件事情：

1. 在活跃位置的表演者发生轮换：在 a_1 位置的表演者会移动到 a_2 位置，在 a_2 位置的表演者会移动到 a_3 位置，……，在 a_k 位置的表演者会移动到 a_1 位置。位置的改变将同时完成。
2. 活跃位置发生变化：a_1 变为 a_1+1，a_2 变为 a_2+1，以此类推。特别的，若 a_i 变化前的值为 $n-1$，则变化后的值为 0。

表演进行了 t 分钟，请求出表演结束后，第 $0 \sim n-1$ 个位置上的表演者的编号分别是多少。

格式

输入格式

第一行三个正整数 n, k, t，含义如题所述。

第二行 k 个整数，描述初始的活跃位置。

输出格式

一行 n 个整数，表示表演结束后第 $0 \sim n-1$ 个位置上的表演者的编号。

样例

输入样例1

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

输出样例1

```
1 2 3 4 0
```

样例解释

$t=0$ 时：表演者编号依次为 `0 1 2 3 4`，活跃位置编号为 `0 2 3`。

$t=1$ 时：表演者编号依次为 `3 1 0 2 4`，活跃位置编号为 `1 3 4`。

$t=2$ 时：表演者编号依次为 `3 4 0 1 2`，活跃位置编号为 `2 4 0`。

$t=3$ 时：表演者编号依次为 `2 4 3 1 0`，活跃位置编号为 `3 0 1`。

$t=4$ 时：表演者编号依次为 `1 2 3 4 0`，活跃位置编号为 `4 1 2`。

数据范围

对于 40% 的数据：$n \le 10^3$，$t \le 10^4$。

对于 100% 的数据：$1 \le k \le n \le 2 \times 10^5$，$1 \le t \le 10^9$。