通信（message）

这是一道通信题。

【题目描述】

有若干个节点，它们一开始分别存储有一个数字 $a_i \in \{0,1\}$，它们想要通过 K 轮通信知道其它每个节点存储的数字。

每一轮通信开始的时候，每个节点 i 对每个节点 j，都会选择一个数字 $c_{i,j} \in \{0,1\}$，表示它将会向节点 j 发送数字 $c_{i,j}$，而在这轮通信结束的时候，节点 j 会收到所有节点向它发送的数字的和，具体而言节点 j 会收到一个数字 $s_j = \sum_i c_{i,j}$。

现在给定 K，你需要找到一个尽量大的 N，满足在通过 K 轮通信之后每个节点都可以知道所有节点存储的数字。

【实现细节】

你不需要，也不应该实现 `main` 函数。

你应确保提交的程序包含头文件 `message.h`，可在程序开头加入以下代码实现：

```
#include "message.h"
```

你需要实现以下函数：

```
int init(int K);
```

- 该函数传入通信总轮数 K 的值。**保证** $1 \le K \le 7$。
- 该函数需要返回你选择的节点数量 N，**你需要保证** $1 \le N \le 31$。
- 对于每次代码运行，**保证在任意 `send` 函数调用前，该函数会被交互库调用恰好一次**。

```
unsigned int send(int K, int N, int round, int number, const
  std::vector<int>&received);
```

- 该函数传入通信总轮数 K，你实现的 `init` 函数返回的节点数量 N，当前通信的轮数 $round$，当前你需要实现的节点的编号 $number$，当前节点之前通信的轮数中收到的数字 $received$，其中 $received[0]$ 表示这个节点一开始存储的数字，而 $received[i]$ $(1 \le i < round)$ 表示这个节点第 i 轮通信结束的时候收到的数字。**保证** $1 \le K \le 7$，$1 \le round \le K+1$，$0 \le number < N$，**且** $received$ **的长度为** $round$。
- 若 $1 \le round \le K$，你需要返回一个无符号三十二位整数 x 表示在这轮通信中节点 $number$ 发送给所有节点的数字，其中 x 的第 i 位为节点 $number$ 发送给节点 i 的数字，高位用 0 补齐。

- 若 $round = K + 1$，你需要返回一个无符号三十二位整数 x 表示节点 $number$ 经过 K 轮通信后确定的每个节点存储的数字，其中 x 的第 i 位为编号为 i 的节点存储的数字，高位用 0 补齐。
- 对于每次代码运行，**保证该函数会被交互库调用不超过** 3×10^4 **次**。

注意：你需要保证，对于任意两次传入参数相同的函数调用（包括 `init` 和 `send`），返回值也应当相同，否则你的程序将会直接被判定为错误。

题目保证在规定的限制下，交互库在每次代码运行中的运行时间不会超过 100 ms；交互库使用的内存大小固定，且不超过 64 MiB。这意味着在每次代码运行中你的代码可以使用至少 900 ms 的时间和 448 MiB 的空间。

【测试程序方式】

下发文件中的 `implementer.cpp`，`communicator.cpp` 是提供的交互库参考实现，最终测试时所用的交互库实现与该参考实现有所不同，因此你的解法不应该依赖交互库实现。

将你的程序命名为 `message.cpp` 并放置于下发文件目录下后，你可以在下发文件目录下使用如下命令进行测试：

```
bash run.sh
```

- 上述脚本将从**标准输入**读入以下格式的数据：
  - 输入的第一行一个整数 0。
  - 输入的第二行两个正整数 T, K，其中 T 表示进行通信的**次数**，K 表示**每次通信**的轮数。你需要保证 $1 \leq T \leq 101$，$1 \leq K \leq 7$。
  - 输入的第 $i + 2 (0 \leq i < T)$ 行一个无符号三十二位整数 x_i，表示第 i 次通信时每个节点初始存储的数字，其中 x_i 的第 j 位表示 j 号节点初始存储的数字。你需要保证 $\forall 0 \leq i < T, 0 \leq x_i < 2^{32}$。
- 上述脚本将输出以下格式的数据到**标准输出**：
  - 若通信结果正确，则输出一行一个正整数 N，表示调用函数 `init()` 得到的结果；
  - 若通信结果错误，则输出 `Wrong answer!`。

【下发文件说明】

在下发文件中：

1. `implementer.cpp`，`communicator.cpp` 是提供的交互库参考实现。
2. `message.h` 是头文件，你不需要关心其具体内容。
3. `template_message.cpp` 是提供的示例代码，你可以在此代码的基础上实现。

【子任务】

对于所有测试数据，保证 $1 \leq K \leq 7$，且对于每次代码运行，`send` 会被交互库调用不超过 3×10^4 次。

测试点编号	分值	$k=$
1	5	1
2		2
3	10	3
4	15	4
5		5
6	20	6
7	30	7

【评分方式】

注意：

- 你不应当通过非法方式获取交互库的内部信息，如试图直接读取交互库中存储的值，或直接与标准输入、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有所不同，因此你的解法不应该依赖交互库实现。**

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。你只能在程序中访问自己定义的和交互库给出的变量及其对应的内存空间，尝试访问其他空间将可能导致编译错误或运行错误。

在上述条件基础上，在每个测试点中，程序得到的分数将按照以下方式计算：

- 若对于任意两次传入参数相同的该函数调用，返回值不同，则获得 0 分；
- 若 K 轮通信后确定的每个节点存储的数字与每个节点初始存储的数字不同，则获得 0 分；
- 否则设 N 为调用函数 `init()` 得到的结果，则该测试点的得分为 $s \times 0.7^{\max(C(K)-N,0)}$，其中 s 为该测试点的分值，$C(K)$ 的计算方式如下表所示：

$K=$	1	2	3	4	5	6	7
$C(K)=$	2	4	6	11	14	21	25