前往何方（wheretoreach）

这是一道交互题。

【题目描述】

有一棵 n 个点的树，你不知道其形态。你有一个初始为空的点集 S。你可以进行以下两种操作：

1. 向 S 中加入一个结点。
2. 从 S 中删除一个结点。

每次操作结束后，交互库将会返回树对 S 的导出子图（即仅保留 S 中的点和两端都在 S 中的边得到的图）的最大连通块大小。

你需要使用尽可能少的操作还原树的结构。

【实现细节】

请确保你的程序开头有 `#include "wheretoreach.h"`。

你不需要也不应该实现主函数。你需要实现以下函数：

```
void solve (int n);
```

- 其中 n 表示树的结点个数，结点从 1 至 n 编号。

你可以调用下列由交互库提供的函数：

```
int add (int x);
int remove (int x);
void report (int x,int y);
```

- `add(x)` 用于向 S 中加入一个结点 x，若 x 本身就在 S 中则什么也不做。
  - 你需要保证 $1 \le x \le n$。
  - 该函数返回值为操作后 S 的导出子图最大连通块大小。
- `remove(x)` 用于从 S 中删除一个结点 x，若 x 本身就不在 S 中则什么也不做。
  - 你需要保证 $1 \le x \le n$。
  - 该函数返回值为操作后 S 的导出子图最大连通块大小；
- `report(x,y)` 表示你找到了树的一条边 (x, y)。
  - 你需要恰好调用该函数 $n-1$ 次以提交树的所有边。边的顺序和方向任意。

你调用 `add` 与 `remove` 的总次数不得超过 2×10^6。

保证在满足题目条件和数据范围的情况下，最终测试时交互库的运行时间不会超过 2000 ms，运行空间不会超过 128 MiB。

交互库不是自适应的，即树形态固定，不会随着交互过程改变。

【测试程序方式】

试题目录下的 implementer.cpp 是我们提供的交互库参考实现。最终测试的交互库与样例交互库有一定不同，故你的实现不应该依赖样例交互库实现。

你需要在本题目录下使用如下命令编译得到可执行程序：

```
g++ implementer.cpp sample.cpp -o sample -O2 --std=c++14 -lm
```

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：
  - 第一行一个整数 n 表示树的点数。你需要保证 $1 \leq n \leq 10^4$。
  - 接下来 $n-1$ 行，每行两个整数 x, y 描述树上的一条边。边的顺序和方向是任意的。
- 读入完成之后，交互库将调用恰好一次函数 `solve`。
- 当你的实现不符合【实现细节】中的要求时，交互库只会在标准输出流输出一行一个整数 `-1`。具体地，以下情况发生时交互库会输出 `-1`：
  - 对 `add` 和 `remove` 的函数调用中传入了不满足 $1 \leq x \leq n$ 的 x；
  - 调用 `add` 和 `remove` 的次数和超过了 2×10^6；
  - `solve` 函数结束时 `report` 函数的调用次数不为 $n-1$。
- 否则，交互库会在标准输出流按照以下格式输出：
  - 第一行两个整数 Q, c，Q 表示你调用 `add` 和 `remove` 的总次数，$c = 1$ 表示你的答案正确，$c = 0$ 表示不正确。
  - 接下来若干行，为你调用 `add`，`remove` 和 `report` 的记录，其格式可以参考样例 1。

【样例 1 输入】

```
3
1 2
2 3
```

【样例 1 输出】

```
4 1
add(2);
add(3);
add(1);
remove(2);
```

```
report(3,2);
report(1,2);
```

【样例 1 解释】

该样例输出为下发的样例程序在该组样例下的输出。

【子任务】

对于所有测试数据，$1 \le n \le 10^4$。

Subtask 1 (10%)：$n = 500$。

Subtask 2 (20%)：$n = 2500$。

Subtask 3 (70%)：$n = 10^4$。

【评分方式】

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、超过空间限制都会导致相应测试点得 0 分。选手只能在程序中访问自己定义的和交互库给出的变量或数据，及其相应的内存空间。尝试访问其他位置空间将可能导致编译错误或运行错误。

对于 Subtask 1,2 中的测试点，只要你的交互符合【实现细节】中的规则，且给出了正确答案，则可以获得满分。

对于 Subtask 3 中的测试点，在你的交互符合【实现细节】中的规则，且给出了正确答案的基础上，设你调用 `add` 和 `remove` 的总次数为 Q，你的得分为 $f(Q)$，其中

$$f(Q) = \begin{cases} 70 & (0.0 \times 10^6 \le Q \le 0.6 \times 10^6) \\ 70 - \frac{Q - 0.6 \times 10^6}{5000} & (0.6 \times 10^6 < Q \le 0.7 \times 10^6) \\ 50 - \frac{Q - 0.7 \times 10^6}{15000} & (0.7 \times 10^6 < Q \le 1.0 \times 10^6) \\ 30 - \frac{Q - 1.0 \times 10^6}{25000} & (1.0 \times 10^6 < Q \le 1.5 \times 10^6) \\ 10 - \frac{Q - 1.5 \times 10^6}{50000} & (1.5 \times 10^6 < Q \le 2.0 \times 10^6) \end{cases}$$

是一个连续的分段线性函数。你可以理解为：你的前 0.6×10^6 次调用是免费的；此后的 0.1×10^6 次调用中每 5000 次扣 1 分；随后的 0.3×10^6 次调用中每 15000 次扣 1 分；随后的 0.5×10^6 次调用中每 25000 次扣 1 分；随后的 0.5×10^6 次调用中每 50000 次扣 1 分。

你在每个子任务上获得的分数为在其中所有测试点上分数的最小值。

选手不应通过非法方式获取交互库的内部信息，如试图与标准输入、输出流进行交互。此类行为将被视为作弊。