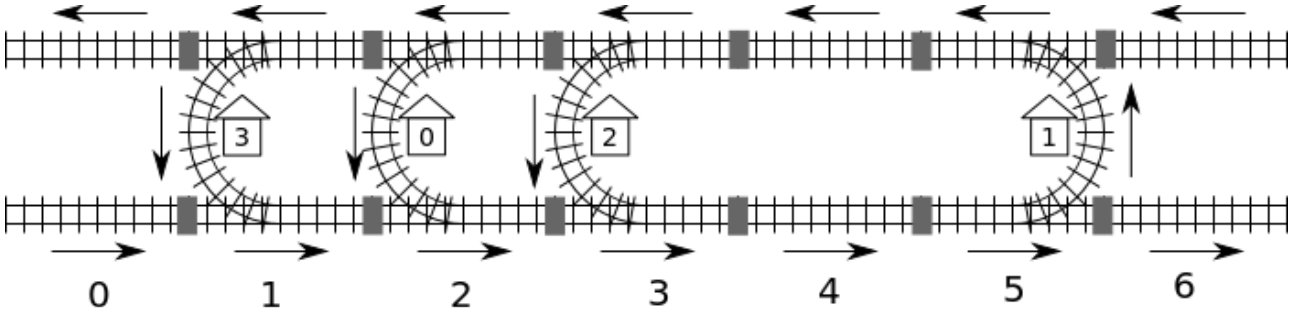
铁路（Rail）

台湾有一个连接着岛的东、西两岸的庞大的铁路线。这个铁路线包含有 m 个区段（block）。这些相连的区段以数字 $0, \ldots, m-1$ 为编号，且编号由西端开始。每一个区段都包含一段在北面向西单行的路轨，以及一段在南面向东单行的路轨。部分区段还可能会有一个位于这两段路轨之间的车站。

区段可以分成三种类型。其中 *C* 类型的区段会有一个车站，而且只能从北面的路轨进入车站并且从南面的路轨离开。 *D* 类型的区段也会有一个车站，而且只能从南面的路轨进入车站并且从北面的路轨离开。 *空 (empty)* 类型的区段则没有车站。举例来说，如下图所示，区段 0、4和6是空类型区段，区段 1、2和3是C类型，而区段 5 则是D类型。这些区段沿水平方向连接在一起。相邻区段的路轨之间以 *连接器（connectors）* 彼此相连，在图中所示的灰色方块就是连接器。



在这个铁路系统中共有 n 个车站，它们分别以数字 0 到 $n-1$ 为编号。这里假设 *我们可以由任何一个车站出发、沿着路轨的方向到达任何其他车站*。例如我们可以由车站 0 到达车站 2，所经路线是由区段 2 开始，经区段 3 及区段 4 的南面路轨，然后经由区段5通过车站 1，再经区段 4 的北面路轨，最后到达区段 3 并由北面路轨进入车站 2。

由于从一个车站到另一个车站有多个不同的可行路线，我们定义由一个车站到另一个车站的距离为所有可行路线中所经的连接器的最少数量。例如由车站 0 到车站 2 的最短路线所经的区段是 2-3-4-5-4-3 ，所经的连接器数目为 5，因此距离为 5。

该铁路系统是由计算机系统管理的。不幸的是，在一次电力事故后，计算机系统丢失了各车站所在的区段编号及其相应的类型。目前仅知道车站 0 所在的区段编号，且该区段总是 C 类型。幸运的是计算机能够查询出任意两个车站之间的距离。例如计算机可以查询'车站 0 与车站 2 的距离'，并且得到答案为 5。

任务

你需要编写一个函数 `findLocation`，以确定每一个车站所在的区段编号及其相应的类型。

- `findLocation(n, first, location, stype)`
  - `n`: 车站的数目。

- first: 车站 0 所在区段的编号。
- location: 大小为 n的数组; 你需要将车站 i 所在的区段编号存放到 location[i]中。
- stype: 大小为 n的数组; 你需要将车站 i 的类型存放到 stype[i]中：其中 1表示 C 类型，而 2 则表示 D 类型。

你可以调用一个函数 getDistance 以帮助你找出各车站的所在区段及其类型。

- getDistance(i, j) 将返回车站 i 到车站 j 的距离。如果调用 getDistance(i, i) 将返回 0。如果i 或 j 在 $0 \leq i, j \leq n-1$ 范围之外，getDistance(i, j) 将返回 -1 。

子任务

在所有的子任务中, 区段的数目 m 不会超过 1,000,000 。在部分子任务中，调用getDistance 的次数是受限的，且该限制将因子任务而异。如果调用次数超过相应限制，你的程序将被判为'wrong answer'。

子任务	分值	n	getDistance 的调用次数	备注
1	8	$1 \leq n \leq 100$	无限制	除车站 0 外，其他车站均在 D 类型区段中。
2	22	$1 \leq n \leq 100$	无限制	所有在车站 0 东侧的车站均在 D 类型区段中，而所有在车站 0 西侧的车站均在 C 类型区段中。
3	26	$1 \leq n \leq 5,000$	$n(n-1)/2$	无其他限定
4	44	$1 \leq n \leq 5,000$	$3(n-1)$	无其他限定

实现细节

你只能提交一个文件, 文件名为 rail.c, rail.cpp 或 rail.pas。该文件中需要实现如上所述的 findLocation 函数，并且遵循下面的命名和接口。对于C/C++ 程序，你还需要包含头文件 rail.h。

C/C++程序

```
void findLocation(int n, int first, int location[], int stype[]);
```

Pascal程序

```
procedure findLocation(n, first : longint; var location,
stype : array of longint);
```

函数getDistance的接口信息如下。

C/C++程序

```
int getDistance(int i, int j);
```

Pascal程序

```
function getDistance(i, j: longint): longint;
```

评测方式

评测系统将读入如下格式的输入数据:

- 第1行： 子程序编号
- 第2行： n
- 第 $3+i$行，($0 \le i \le n-1$)：stype[i] (1 表示 C 类型，2 表示 D 类型), location[i].

在 findLocation返回后，如果你的程序计算出的location[0] ... location[n-1] 和stype[0] ... stype[n-1]与输入数据一致，评测系统将输出 Correct，否则输出 Incorrect。