The Cop and the Robber

Crime level in the city of Bytemore is hitting an all–time high. Among other misdemeanours, robberies are happening every day. And when the crime is comitted, it is always up to a lone patrolling police officer to chase down the robber through the narrow alleys that connect street corners (commonly referred to simply as *corners*). Unfortunately, more often than not, robbers escape their pursuers, because they know the city much better than the police.

The Bytemore City Police Department (BCPD) is organising a summit to reduce crime. One of the initiatives is to use computer aid when pursuing the robbers. For this purpose, the BCPD has made a precise map of the city. Now they need computer software to find the best chasing strategy.

The pursuit problem for two agents: one officer chasing one robber, is modelled as follows:

1. The police officer chooses a corner on which to patrol.
2. The robber then chooses a corner for the robbery (he knows where the officer is). From this moment on it is always assumed that both the officer and the robber know where each other is.
3. The police officer's move consists of him moving to a neighbouring corner (i.e. one that is connected to the current one by an alley) or waiting (i.e. not moving).
4. The robber's move consists of him moving to a neighbouring corner. Note that, unlike the police, robbers cannot wait. It is in their instinct to keep running.
5. The police officer and the robber keep making moves one after another (starting with the officer) until one of the following happens:
   (a) situation repeats itself (situation is defined by the positions of both agents and the side whose turn it is to move next). This corresponds to the robber being able to avoid the police officer indefinitely, so the robber escapes;
   (b) both agents meet on the same corner after a move of either of them. In this case the police officer catches the robber.

Task

You have to write a program which, given the map of the city, would determine whether catching the robber is possible, and if it is, would catch him by making moves on behalf of the police officer.

Your program must assume that the robber moves optimally.

Implementation

You need to implement two functions:

- `start(N, A)` which takes the following parameters:
  - N — the number of corners (corners are labelled from 0 to $N-1$);

- A — a two–dimensional array that describes the alleys: for $0 \leq i, j \leq N-1$,

$$A[i][j] \text{ is } \begin{cases} 0 & \text{if } i \text{ and } j \text{ are not joined by any alley} \\ 1 & \text{if } i \text{ and } j \text{ are joined by an alley} \\ -1 & \text{if } i = j \end{cases}$$

If it is possible to catch the robber on the map described by the parameters, function `start` has to return the label of the corner on which the police officer chooses to patrol. Otherwise, it has to return -1.

- `nextMove(R)` which takes as a parameter the label R of the current corner of the robber and must return the label of the corner where the officer will be after his move.

Function `start` will be called exactly once before any calls to `nextMove` are made. If `start` returns -1, then `nextMove` will not be called. Otherwise, `nextMove` will be called repeatedly until the pursuit ends. More precisely, the program will terminate as soon as one of the following happens:

- `nextMove` returns an invalid move;
- the situation repeats itself;
- the robber is caught.

Example

Provide an example.

Scoring

Explain scoring.

Prepare subtasks.

Subtask 1 (25 points):

Subtask 2 (25 points):

Subtask 3 (25 points):

Subtask 4 (25 points):

Constraints

Check constraints.

Time limit: 1 s.

Memory limit: 256 MB.

Need to add info about graders.