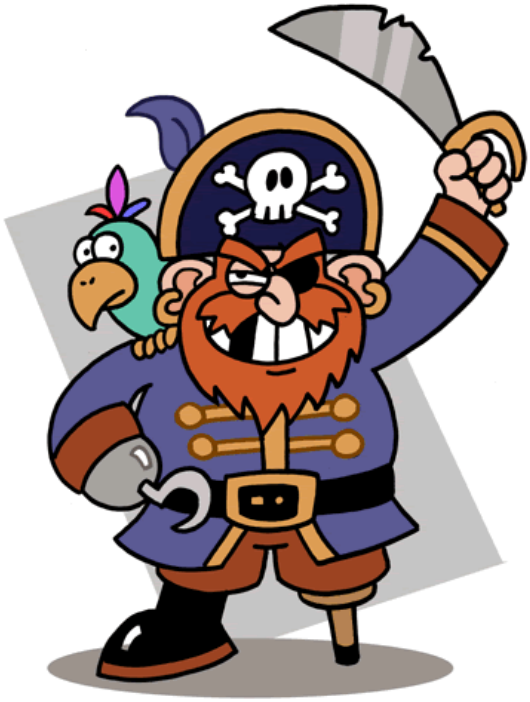
INOI, Iranian National Olympiad in Informatics
Online Contest, April 2008
Second Exam

Treasure Island

Time limit for each test: 2500 miliseconds[1]
Memory limit: 120 megabytes[2]



Long John Silver eventually set foot on the Treasure Island. It was well-known that, after coming to this island, Captain Flint had hidden his golden coins and chests full of jewelry and pearls. Therefore, Long John Silver started his treasure hunt. He was looking for the island maps and treasure traces when he ran into Ben Gunn. He has been left there alone after Captain Flint. Silver asked Ben Gunn to show him to the treasures, and in turn, Silver promised to take him away from this deserted island. In reply to this offer, Ben Gunn only uttered "The map of the island is like an $m \times n$ rectangle, and the treasure pieces could be in any of the $m \times n$ squares."

Since Silver didn't want to search in all the squares, he said to Ben Gunn "Then, tell me about the location of the hidden treaures as well!" Ben Gunn, not having had toasted cheese for so long, said "You gotta give me some toasted cheese!" Long John, tired of Ben's weird fads, said "Go to hell!"

Ben Gunn: I will never tell you their locations.

Long John: Come on! I'll give you some toasted cheese.

Ben Gunn: `:D` In each 2×2 subsquare of the island, the number of cells having some treasure is even. I will let you ask me question like this: "Is there some treasure in this cell of the island?" For each such question, you have to give me one piece of toasted cheese. And to punish you further, I will deprive you of asking about certain cells. But, I will tell you about those cells beforehand.

You are to help Long John Silver find all the cells containing some treasure. This should be achieved by the least costly method, i.e. by asking the minimum number of questions. Besides, let Silver know, as soon as you find out the treasure cells cannot be uniquely determined by Ben Gunn's help. Assume that Ben always tells the truth. Also, beware that Ben's stomach has room for only 30,000 toasted cheese pieces. He wouldn't bother to answer any more questions after that!

Problem

Write a program that

- Gets the size of the island (m and n) and the places of forbidden cells from the *library*.
- Asks appropriate questions from Ben Gunn using the *library*;
- Using minimum number of questions, reports the exact set of cells having treasure or reports that this set could not be specified uniquely.

[1] Time limit is in fact 3000 miliseconds, and the library will use at most 500 miliseconds.
[2] Memory limit is in fact 130 megabytes, and the library will use at most 10 megabytes.

Library

The list of the functions and definitions in the library is as follows:

```
define MAX_M 1000
define MAX_N 1000
define MAX_Q 30000
typedef int Table[MAX_M][MAX_N];
```

These constants define the maximum size and the structure of the table. Cell `[i][j]` in a table, refers to the cell in the i^{th} row ($0 \leq i < m$) and the j^{th} column ($0 \leq j < n$).

`void init(int *m, int *n, Table *forbidden);`

This function sets the two integer variables `m` and `n` to the number of rows and the number of columns. Furthermore, the table pointed by `forbidden` is filled like this: If the cell `[i][j]` ($0 \leq i < m$, $0 \leq j < n$) is forbidden, the value `(*forbidden)[i][j]` equals 1, otherwise it equals 0. You must define and allocate the forbiden table in your own program before calling this finction; the memory is not allocated in the library.

This function must be called once and before any other calls to the other library functions.

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

This function, asks a question from Ben Gunn and returns his answer. This question is related to cell `[i][j]` ($0 \leq i < m$, $0 \leq j < n$). If there is treasure in that cell, the return value equals 1 and otherwise it equals 0. Notice that you must not ask a question about a forbidden cell. Otherwise your program will stop as malfunction.

`void reportNo();`

`void reportYes(Table *value);`

You can report the final answer using one of these two functions. These functions do not return and your program ends immediately after calling one of them. The function `reportNo` must be called if the set of treasure cells could not be determined uniquely by asking questions. Otherwise, the function `reportYes` should be called by passing a pointer `value` to a `Table` data structure, such that `(*value)[i][j]` equals 1 for the treasure cells and 0 for non-treasure cells ($0 \leq i < m$, $0 \leq j < n$).

Your program must not work with any file or standard input/output. Furthermore, your program must not try to access to memory outside its allocated area. Breaking these rules might result to your disqualification.

You are provided with two files named `tisland.h` and `tislandlib.cpp` for testing purposes. You should add this line above your code: `#include "tisland.h"`
For compiling your program, put these files in the same folder as your program and run this command:

```
g++ -O2 -static tisland.cpp tislandlib.cpp -lm
```

A file named `stisland.cpp` is given to you to serve as an example of using the provided library. Please, note that:

1. The provided `stisland.cpp` does not solve the problem. It is only provided as an example to show how to use the library.
2. When we are testing your programs, they will be linked to a *different* library which implements the functions declared in `tisland.h`. As long as you use the library in the prescribed manner, this wouldn't cause any problems for your program.

Testing

You can download the test library from the contest interface. The provided test library reads the required data from the standard input. The first line of the input should contain two integers m (the number of rows) and n (the number of columns) seperate by a single space. In each of the following m lines, you must enter n space separated 0s and 1s. the jth integer in the ith row, is equal to 1 if and only if there is treasure in cell `[i][j]` ($0 \leq i < m$, $0 \leq j < n$).

When using the contest's "Test Run" feature, your program interacts with this same test library, so the file you provide as input should have the above format.

You are free to change `tislandlib.cpp`, but it is advised not to change `tisland.h`.

Grading

On any test case, if your program exceeds its time or memory limits, or if it does not properly interact with the test library, it will lose the grade of that test case.

Also, if your program makes more than 30000 calls to the `ask` function, it will be stopped and it will not get any marks for the test case. If you report the places of all the treasure cells using `reportYes`, while it is actually impossible to determine the unique output using your questions, you will not gain any point in that test. Your program will get the points of each test, only if it asks the minimum number of questions required. (Notice that when it is possible to report the correct answer without any questions, asking even a single question, would cause you to loose the points of that case.)

In all of the test cases $1 \leq m, n \leq 1000$, but in 60% of the test cases $1 \leq m, n \leq 100$.

Interaction

The following table shows a simple interaction with the library.

Function call	Description
`int m,n;`	Definition of integers `m` and `n`
`Table forbidden, value;`	Define `forbidden` and `value` as Table.
`init(&m, &n, &forbidden);`	Start interaction by passing `m`, `n` and `forbidden`.
`int a=-1, b=-1;`	Define two temporary variables.
`if(!forbidden[0][0]) a=ask(0,0);`	Ask a question about the upper leftmost cell.
`if(!forbidden[m-1][n-1]) b=ask(m-1,n-1);`	Ask a question about the lower rightmost cell.
`if(a<0 || b<0){`	
`  reportNo();`	Report the unique solution doesn't exist.
`}else{`	
`  value[0][0]=a;`	Set the upper leftmost cell.
`  value[m-1][n-1]=b;`	Set the lower rightmost cell.
`  reportYes(&value);`	Report the answer using the table `value`.
`}`	

The following is an example input file for the testing library.

```
4 5
0 0 1 0 0
0 0 0 1 0
1 0 0 0 0
1 1 1 0 1
1 1 0 0 1
0 0 1 1 0
1 1 0 0 1
1 1 0 0 1
```