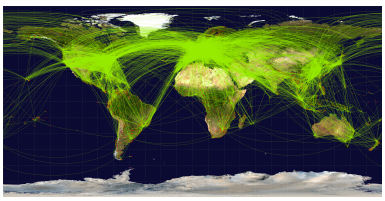
2 Problem: Assigning Airline Crews to Flights

Problem Introduction

In this problem, you will apply an algorithm for finding maximum matching in a bipartite graph to assign airline crews to flights in the most efficient way.



Problem Description

Task. The airline offers a bunch of flights and has a set of crews that can work on those flights. However, the flights are starting in different cities and at different times, so only some of the crews are able to work on a particular flight. You are given the pairs of flights and associated crews that can work on those flights. You need to assign crews to as many flights as possible and output all the assignments.

Input Format. The first line of the input contains integers n and m — the number of flights and the number of crews respectively. Each of the next n lines contains m binary integers (0 or 1). If the j-th integer in the i-th line is 1, then the crew number j can work on the flight number i, and if it is 0, then it cannot.

Constraints. $1 \le n, m \le 100$.

Output Format. Output n integers — for each flight, output the 1-based index of the crew assigned to this flight. If no crew is assigned to a flight, output -1 as the index of the crew corresponding to it. All the positive indices in the output must be between 1 and m, and they must be pairwise different, but you can output any number of -1's. If there are several assignments with the maximum possible number of flights having a crew assigned, output any of them.

Time Limits.

language	C	C++	Java	Python	C#	Haskell	JavaScript	Ruby	Scala
time in seconds	1	1	1.5	5	1.5	2	5	5	3

Memory Limit. 512Mb.

Sample 1.

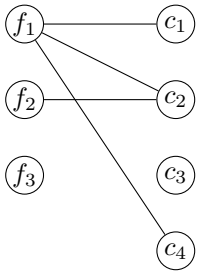
Input:

```
3 4
1 1 0 1
0 1 0 0
0 0 0 0
```

Output:

```
1 2 -1
```

Explanation:
In this sample, the bipartite graph of flights (on the left) and crews (on the right) looks like this:



We can assign first crew to the first flight and second crew to the second flight, and no crews can work on the third flight, so this is an optimal assignment.

Sample 2.

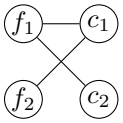
Input:

```
2 2
1 1
1 0
```

Output:

```
2 1
```

Explanation:
In this sample, the bipartite graph of flights (on the left) and crews (on the right) looks like this:



If we assign the first crew to the first flight, we won't be able to assign any crew to the second flight. It is optimal to assign the second crew to the first flight and the first crew to the second flight, because this way we have a crew assigned to each flight.

Starter Files

The starter solutions for this problem read the data from the input, pass it to a blank procedure that implements an incorrect greedy algorithm for finding the maximum matching, and output the result. You need to change this procedure to implement a correct algorithm for finding the maximum matching if you are using `C++`, `Java`, or `Python3`. For other programming languages, you need to implement a solution from scratch. Filename: `airline_crews`

What To Do

Implement an algorithm for finding the maximum matching in the bipartite graph that was described in the lectures.

Need Help?

Ask a question or see the questions asked by other learners at this forum thread.