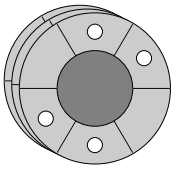
Lucky Wheels of Doom

It's time for your weekly trip to the fortune teller. Last week she read your palm; this week she is bringing out the Lucky Wheels of Doom.

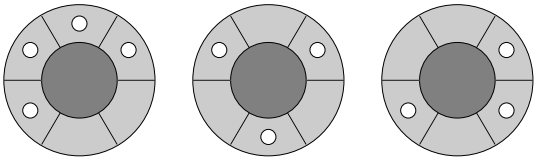
The Lucky Wheels consist of a set of w circular wheels, placed one behind another as illustrated below. The rim of each wheel is divided into s equal segments, some of which have holes cut out of them.



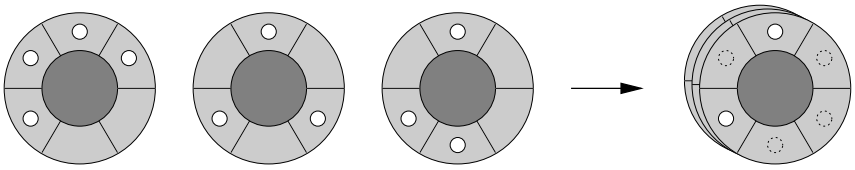
Your fortune teller will ask you to spin each wheel in turn. Once all of the wheels have been spun, she will look for holes that are lined up all the way from the front to the back (so that you can see right through the entire set of wheels). For each set of holes that are lined up, you are guaranteed a year of prosperity and bug-free code.

Fortunately you have done this many times before, and so you are able to spin each wheel precisely as far as you would like. Your task is to decide how far to spin each wheel so that many sets of holes can be lined up.

As an example, consider the set of $w = 3$ wheels illustrated below, each of which is divided into $s = 6$ segments.



By spinning the second wheel one segment clockwise and spinning the third wheel two segments clockwise, you can arrange the three wheels as shown below. When the wheels are placed one behind another, there are two sets of holes lined up all the way from front to back. This is illustrated in the final diagram on the right hand side.



You are not required to give the best possible solution. Instead you must line up as many sets of holes as you can. Your program will then compete with other programs, with the better solutions scoring more points.

Constraints

- $1 \leq w \leq 50$, where w is the total number of wheels;
- $1 \leq s \leq 50$, where s is the number of segments into which each wheel is divided.

Furthermore, for 30% of the available marks, the number of segments will satisfy $s \leq 20$.

Input

Your program must read from standard input. The first line will contain the two integers w and s separated by a single space, as described above.

Following this will be w lines, each describing a single wheel. Each of these lines will contain s integers separated by spaces, describing the segments of the wheel in clockwise order from the top. For each segment, the integer 0 represents a hole in the wheel and the integer 1 represents no hole.

Output

Your program must write $w+1$ lines to standard output. Of the first w lines, the ith line corresponds to the ith wheel from the input data, and should contain a single integer describing the number of segments the wheel should be rotated clockwise. Each output integer must be between 0 and $s-1$ inclusive.

After these w lines have been written, your program must write one additional line of output. This line must contain a single integer giving the total number of sets of holes that are aligned all the way from front to back.

Sample Input

```
3 6
0 0 1 1 0 0
1 0 1 0 1 0
1 0 0 1 0 1
```

Sample Output

```
0
1
2
2
```

Scoring

There is no particular "best solution" that you are required to achieve. Instead, your score will be determined relative to the other contestants whom you are competing against (as well as the judges' solution).

One simple solution involves examining each wheel in turn from top to bottom, and at each stage rotating the wheel to line up as many holes as possible with the wheels above it. Such a solution will be defined to score 30%.

For each input scenario, the contestant who lines up the most sets of holes will be identified. Suppose that this contestant lines up h sets of holes in total. Your score for this input scenario will then be:

- 100% if your program finds a solution also with h sets of holes lined up;
- 30% if your program finds a solution equal to the simple solution outlined above;
- 0% if your program generates an incorrect solution (e.g., you incorrectly calculate how many sets of holes are aligned, or you do not follow the output format exactly);
- otherwise determined by a linear scale according to how many sets of holes your program aligns, with the 100% and 30% marks on the scale corresponding to the solutions described above.

For example, consider an input scenario for which the simple solution aligns 26 sets of holes. If the best solution found by any contestant (or the judges) aligns 40 sets of holes, then the scoring scale for a *correct* solution would be as follows:

Holes aligned	40	36	32	28	26	24	20	16
Score	100%	80%	60%	40%	30%	20%	0%	0%