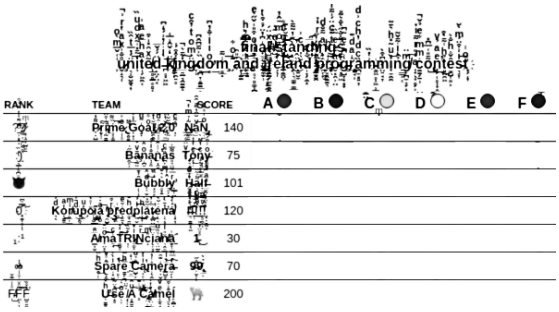
Problem C
Corrupt Judge



You are organising a programming competition in which the rank of a team is first determined by how many problems they have solved. In case of a tie, the team with the lowest time penalty is ranked above the other. However, contrary to the UKIEPC, the time penalty is equal to t if the *latest* accepted submission was submitted in the tth minute, or 0 if no problem was solved.

For example, if team A solved their first problem in the 5th minute, their second problem in the 10th minute and their third problem in the 60th minute, then their time penalty is 60. If team B also solved three problems, in the 30th, 40th and 50th minute, their time penalty is 50 and they would rank above team A.

The contest has finished and you would like to enter the final standings. However, due to a corrupted file you have lost part of the scoreboard. In particular, the column indicating how many problems each team has solved is gone. You do still have the time penalties of all the teams and know that they are in the right order. You also remember how many problems the contest had. You wonder whether, given this information, it is possible to uniquely reconstruct the number of problems that each team has solved.

Input

- One line containing two integers: n ($1 \leq n \leq 10^4$), the number of teams participating, and p ($1 \leq p \leq 10^4$), the number of contest problems.
- n lines with on line i the time score t_i in minutes ($0 \leq t_i \leq 10^6$) of the team that is ranked in the ith place.

A positive time score of t indicates that a team has submitted their last accepted submission in the tth minute. A time score of 0 indicates that a team hasn't solved any problem.

The input always originates from a valid scoreboard.

Output

If it is possible to uniquely reconstruct the scores of all the teams, output n lines containing the number of problems that the ith team has solved on the ith line. Otherwise, output "`ambiguous`".

Sample Input 1

```
9 3
140
75
101
120
30
70
200
0
0
```

Sample Output 1

```
3
2
2
2
1
1
1
0
0
```

Sample Input 2

```
6 3
100
40
40
50
0
0
```

Sample Output 2

```
ambiguous
```