C   Cake Promise

Time limit: 2s

Today is the big day: professor P. Wright is organizing a programming contest. All of his students are participating, because he has promised to bake a cake for all teams that can beat him in the contest.[1] After the contest has ended, professor Wright receives a raw copy of the scoreboard, in which the teams are out of order. He is short on time: besides baking, he also has some exams to grade. Therefore, he needs a program to calculate how many teams beat him in the contest.

A Fresh Pineapple Cake, the prize for beating professor Wright.
CC-BY-NC 2.0 By Megan Duong on Flickr

The contest consists of a given number of problems, which the teams will need to solve as fast as possible. One team beats another team if the number of problems solved by the first team is strictly greater. If the number of problems solved is equal between two teams, the team that required a lower sum of minutes to solve the problems beats the other. Professor Wright is also willing to bake a cake for teams that solved the same number of problems *and* required the same sum of minutes as himself.

Input

The input consists of:

- One line containing two integers t and p ($2 \leq t, p \leq 1000$), the number of teams participating and the number of programming problems, respectively. Professor Wright counts as one of the teams.

- t lines, each containing p values, the results for each team. The first line represents the results of professor Wright. Each result is either "X", indicating that the team did not solve this problem, or an integer r ($1 \leq r \leq 10^6$), representing the number of minutes that this team needed to solve the problem.

Output

Output the number of teams for which professor Wright needs to bake a cake.

Sample Input 1	Sample Output 1
5 5 40 X 10 30 20 X X 20 50 40 50 40 20 30 10 X X 30 X 50 X 30 5 40 20	2

---

[1]All characters and events in this problem description are fictional. Any resemblance to actual lecturers is purely coincidental.

Sample Input 2

Sample Output 2

4 4 4 5 6 X 4 5 X 5 4 X 5 6 X 4 5 7	2