Fool the Police

Dhamaka Singh (a crook) has just robbed a bank and would like to get out of the country as soon as possible. But there is a slight problem, the police! On his way out of the country he has to pass through some police stations. Each police station has a certain risk (for Dhamaka Singh) associated with it. He wants to get to the airport within a certain time **T** or else he'll miss his flight. He also wants to take a path that minimizes the total risk associated with it. Help Dhamaka Singh get out of the country.

Input

The first line of the input contains an integer **t**, the number of test cases. **t** test cases follow.

The first line of each test case contains 2 integers **N** (3 <= **N** <= 100) and **T** (1 <= **T** <= 250), denoting the number of police stations and the total time he has to reach the airport, respectively.

Dhamaka Singh has to start from the first police station and reach the **N**[th] one (the airport is just after the **N**[th] police station). You can consider the time taken between the **N**[th] police station and the airport to be negligible.

Next there are **N** lines with **N** numbers in each line, separated by single spaces. All numbers are separated by a single space. The jth integer in the ith line represents the time taken to reach the jth police station from the ith police station.

Next there are another **N** lines with **N** numbers in each line. All numbers are separated by a single space. The jth integer in the ith line represents the risk involved in travelling to the jth police station from the ith police station.

Output

For each test case output one line containing 2 integers separated by a single space.

The first integer denotes the minimum total risk to reach the airport. The second integer denotes the minimum time required to reach the airport at the minimum total risk.

If it is impossible to reach the airport within time **T** (inclusive), just print "-1" (quotes for clarity).

Example

Input:
```
1
4 10
0 6 2 3
6 0 2 3
3 1 0 2
3 3 2 0
0 2 2 7
2 0 1 2
2 2 0 5
7 2 5 0
```

Output:
4 9