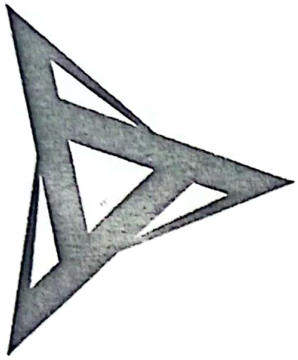
Welcome to PUCon'25

"First, solve the problem. Then, write the code."

– John Johnson



PF-PUCon'25

Round 2

Programming Competition

FCIT

University of the Punjab

Problem B

Word Search

You are given an **m x n** grid of characters, called board, and a string word. Return true if the word can be found in the grid, and false otherwise. The word must be formed by sequentially adjacent letters in the grid. Adjacent cells are those that are horizontally or vertically neighbouring.

Important: You may not reuse the same cell more than once while constructing the word.

Constraints

- m = = board.length
- n = = board[i].length
- 1 <= m, n <= 6
- 1 <= word.length <= 15
- board and word consist only of lowercase and uppercase English letters.

Input

The first line of the input takes the **m** and **n**. It shows the size of the matrix **mxn**. Then, there will be m lines each line containing **n** space separated values that will fill up the matrix. In the end you will get a string , that you need to find in the matrix.

Output

You need to check if the string is present in the matrix or not. If it is present you need to return "true" otherwise "false".

Sample Input

```
3 4
A B C E
s F o S
A D E E
SEE
```

Sample Output

true

Explanation

A	B	C	E
s	F	o	S
A	D	E	E

Problem C

Weaponized Warfare

In a futuristic arena, N elite warriors, each armed with a unique weapon, are lined up to battle. In this deathmatch, a daring commander orchestrates the order of attacks to maximize the total "damage points" earned during the battle.

The rules are as follows:

1. **Battle Moves:**
   a. In each move, the commander chooses two distinct warriors currently in the arena.
   b. One warrior is designated as the attacker and the other as the target.
   c. The attacker strikes the target, inflicting damage points specified by a scoring matrix.
2. **Elimination:**
   a. When attacked, the target is immediately eliminated from the arena.
   b. The remaining warriors close ranks (the order is maintained only by identity, not position).
3. **Game End:**
   a. The battle continues until only one warrior remains.
   b. The commander's goal is to plan the attack order so that the total damage points accumulated are maximized.

Scoring Matrix

1. Let **S[i][j]** denote the damage points earned when warrior **i** attacks warrior **j**.
2. For all warriors, **S[i][i] = 0** (a warrior can never attack itself).
3. All other entries are positive integers.

Input Format

The first line contains an integer **T** ($1 \leq T \leq 512$), the number of test cases. For each test case:

- The first line contains an integer **N** ($2 \leq N \leq 10$), the number of warriors.
- The next **N** lines each contain **N** space–separated integers; the j–th integer on the i–th line is **S[i][j]**.

Output Format

For each test case, output a single line containing the maximum total damage points achievable by optimally ordering the attacks.

Sample Input

```
2
2
0 9
12 0
3
0 125 1
68 0 1
1 59 0
```

Sample Output

```
12
127
```

Explanation

Test Case 1 (N=2):
With warriors 0 and 1, the optimal move is for warrior 1 to attack warrior 0 earning 12 damage points.
Test Case 2: (N=3)
The optimal strategy is: warriors 2 attacks warriors 1 (68 points), then warriors 3 attacks warriors 2 (59 points) => 127 points.

Problem D

Partition Labels

At FCIT, things are getting wild! You're in charge of organizing a week full of awesome student events, workshops, and hackathons. Imagine a list of unique resources "s': it could be a professor, a classroom, a projector, or even the campus cat (yes, she's very busy).

To avoid chaos and double-booking, you need to split the schedule into as many parts as possible without using the same resource in more than one part. Once all parts are combined in order, the full schedule should still match the original string exactly.

Your mission: Return a space-separated list of numbers showing how long each of these chaos-free, resource-friendly blocks is.

Input Format:
The first line contains a integer **T**, the number of test cases. The next T line contains a string s (the resource)

Output Format:
For each test case output the space separated integers representing the size of each partition for string s.

Sample Input:

```
2
ababcbacadefegdehijhklij
eccbbbbdec
```

Sample Output:

```
9 7 8
10
```

Explanation:
The partition is "ababcbaca", "defegde", "hijhklij".

This is a valid partition so that each letter appears in at most one part.

A partition like "ababcbacadefegde", "hijhklij" is incorrect because it splits s into less parts

Constraints:

- 1 <= s.length <= 500
- s consists of lowercase English letters ('a' to 'z').

Problem A

Battle of a Lifetime

In a desperate space battle, your ship has been deployed to secure a critical enemy outpost within contested territory. The battlefield is represented as an N x M grid of tiles, each corresponding to a strategic location. Your ship's movement is limited by fuel constraints, and certain tiles are inaccessible due to enemy fortifications or environmental hazards.

The journey's total cost is determined by accumulating the fuel costs of the traversed tiles. Your goal is to find the path to the enemy outpost that minimises fuel consumption. You can move only vertically or horizontally to adjacent tiles. Landing directly on the outpost tile incurs an additional 10 units of fuel.

Input

The first line of the input consists of T ($1 \leq T \leq 25$) representing the total number of test cases given. The first line of each test case has exactly two numbers, the number of rows N ($2 \leq N \leq 30$), followed by the number of columns M ($2 \leq M \leq 30$). Next N lines of each test case have M space separated elements of the map. The letter S on the map represents the ship's position, letter O represents the outpost's position and letter X.represents blocked tile. Numeric values represent the fuel cost to traverse that particular tile.

Output

For each test case, output one line with Case #x: y, where x is the test case number (starting from 1) and y is the minimal amount of fuel required to reach the outpost. If there is no possible path to the outpost, the minimal amount of fuel is -1.

Sample Test Case:

Sample Input	Sample Output
2 5 5 1 1 X 1 1 X 1 1 X 1 1 X 1 1 1 1 S 1 1 O 1 1 X 1 1 5 5 1 1 X 1 1 X 1 1 X 1 1 X 1 1 1 1 S 1 5 O 1 1 X 1 1	Case#1: 12 Case#2: 14