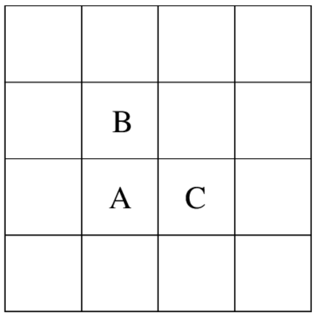
The die is cast

InterGames is a high-tech startup company that specializes in developing technology that allows users to play games over the Internet. A market analysis has alerted them to the fact that games of chance are pretty popular among their potential customers. Be it Monopoly, Ludo or Backgammon, most of these games involve throwing dice at some stage of the game.

Of course, it would be unreasonable if players were allowed to throw their dice and then enter the result into the computer, since cheating would be way too easy. So instead, InterGames has decided to supply their users with a camera that takes a picture of the thrown dice, analyses the picture and then transmits the outcome of the throw automatically.

For this, they desperately need a program that, given an image containing several dice, determines the numbers of dots on the dice.

We make the following assumptions about the input images. The images contain only three different pixel values: for the background, the dice and the dots on the dice. We consider two pixels *connected* if they share an edge - meeting at a corner is not enough. In the figure, pixels `A` and `B` are connected, but `B` and `C` are not.



A set S of pixels is connected if for every pair (a, b) of pixels in S, there is a sequence $a_1, a_2, \ldots, a_k$ in S such that $a = a_1$, $b = a_k$ and a_i is connected to a_{i+1} for $1 \leq i < k$.

We consider all maximally connected sets consisting solely of non-background pixels to be dice. "Maximally connected" means that you cannot add any other non-background pixels to the set without making it disconnected. Likewise we consider every maximal connected set of dot pixels to form a dot.

Input

The first line of the input contains an integer t. t test cases follow, each of them separated by a blank line.

Each testcase starts with a line containing two numbers w h, the width and height of the picture, respectively. h lines follow, each containing w characters. The characters can be "`.`" for a background pixel, "$\star$" for a pixel of a die and "`X`" for a pixel of a die's dot.

Dice may have different sizes and not be entirely square due to optical distortion.

Output

For each test case, print a line containing "Case #i: s" where i is its number, starting at 1, and s is a space-separated list containing the number of dots on the dice in the picture, sorted in increasing order. Each line of the output should end with a line break.

Constraints

- $1 \leq t \leq 20$
- $3 \leq w, h \leq 50$

- Each testcase will contain at least one die.
- Each die will have between 1 and 6 dots, inclusively.

Sample Input 1

```
1
30 15
..............................
..............................
...............*..............
...*****......****............
...*X***.....**X***...........
...*****....***X**............
...***X*.....****.............
...*****.......*..............
..............................
........***........******.....
.......**X****.....*X**X*.....
......*******......******.....
.....****X**.......*X**X*.....
........***........******.....
..............................
```

Sample Output 1

```
Case #1: 1 2 2 4
```