# HARRY POTTER AND THE FORBIDDEN FOREST

Voldemort is back and this time even more stronger. To stay immortal this time he has divided his soul into more than 7 parts (horcrux) and has hidden them in the Forbidden Forest.

Forbidden Forest is represented by a rectangular field of n x m cells. Each cell is either empty or has a tower. Empty cells are marked with '.' and cells with tower are marked with '*'. Every pair of adjacent cells of different type (one empty and one having a tower) has a hole in the wall of the tower with one horcrux hidden.

As Harry Potter is the chosen one he has to weaken Voldemort as much as he can by finding maximum possible number of horcrux. Harry starts from an empty cell and can only move to those empty cells which share a common side with the current one. Harry can find those horcrux which is in a hole of the wall of the tower adjacent to the cell which Harry will visit.

Note: One tower can have more than one holes (and thus more than one horcrux), each in different walls having an adjacent empty cell.

For different starting positions you have to calculate how many Horcrux Harry Potter can find.

## Input

First line contains an integer t - number of test cases.
Each test case contains three integers n, m and k - dimension of the Forbidden Forest and the number of starting positions to process.
$3 \leq n,m \leq 500$
$1 \leq k \leq \min(n*m, 100000)$
Each of the next n lines contains m symbols '.' or '*' - description of the Forbidden Forest. All the border cell are guaranteed to be '*' so that Harry don't go out of the Forest.
Each of the last k lines contains two integers x and y ($1 \leq x \leq n$, $1 \leq y \leq m$) - the row and column of one of the starting position of Harry Potter respectively. Rows are numbered from top to bottom and columns from left to right. All starting positions are guaranteed to be empty cells.

## Output

Print k integers (each in new line) - the maximum number of Horcrux Harry Potter can find if he starts in the corresponding position.

## Example

**Input:**

2

5 5 2

*****

*.*.*

***.*

*.*.*

*****

2 2

2 4

4 5 1

*****

*..**

**..*

*****

3 4

**Output:**

4

8

10