There is a heavy rain in NUS (a 2-D grid of size **N x M**). You are currently in a building with border 'A' and want to reach another building with border 'B'. You can move (that is, run) from one place (a cell in the 2-D grid) to another adjacent place in 1 second. We say that cell A is adjacent to cell B if cell A is on the North/East/South/West of cell B. What should be the path taken by you to minimize the amount of water accumulating in your body?

Sample Map of NUS (a 2-D grid of size 7 x 25):

```
....................CCCC.
AAAAA...............C**C.
A***A.AAA...........C**C.
A***AAA*A....###....CCCC.
A*******A................
AAAAAAAAA................
.......DD..............BB
```

Legend:

- '.' denotes open space. You will get wet by one unit of water if you stand or run in this cell for 1 second.
- '#' denotes obstacles in the open space. You **cannot go through these obstacles**.
- You can assume that **there will be at least 1 valid path from building A to building B** even though we have obstacles.
- '*' denotes the interior of a building. **You are safe from rain water if you are in this cell**. Some buildings are small and thus they do not have interior cells.
- 'A'/'B'/.../'Z' denotes the border of a building. Each building is either a convex or concave polygon with the same border. You are safe from rain water if you are in this cell.
- Note that since we are using uppercase alphabets, there will only be up to 26 different buildings in this task. You can assume the given inputs are always valid.

The solution for the sample map shown above is by following this path:

- From inside building A, you move to cell (2, 8) -- 0-based indexing. This is one of the border of building A and you are still dry at the moment.
- Then move (run) straight to the adjacent building B following the path shown as 1->2->3->...->9->0 (actually 10)->1(actually 11).
- Upon reaching the border of building C at cell (2, 20), you have accumulated 11 units of water.
- Then walk inside building C (you are safe from rain water when you are inside a building) and move to cell (3, 23).
- Run to the target building B by following the path shown as 2 (actually 12)->3 (actually 13).
- Now you have reached building B with 13 units of water in your body.
- This is the best possible way, so output 13 as your answer.

Note: You can try going via building A -> go to adjacent building D without having to go through open space, then run straight from building D cell (6, 8) to building B cell (6, 23).

This path causes you to arrive at building B with 14 units of water in your body, 1 more than the optimal path, as such:

```
....................CCCC.
AAAAA...............C**C.
A***A.AAA12345678901C**C.
A***AAA*A....###....CCCC.
A*******A..............2.
AAAAAAAAA..............3.
.......DD..............BB
```

TASK

Given N and M, and the map of NUS, output an integer representing the minimum amount of wetness from building A to building B. Note that in order to score credits for any of the subtasks, you have to solve **all** test cases that constitute the subtask.

Subtask 1 [20 points]

We have several test cases with 1 <= N, M <= 30 and where each building is just a single character, i.e. each building has size 1 x 1 and has no interior cells. Moreover, there will only be two buildings in the map, building A (where you are) and building B (target building).

Subtask 2 [20 points]

We have several test cases with 1 <= N, M <= 30 and where each building is just a single character, that is, each building has size 1 x 1 and has no interior cells. There can be up to 26 different 1 x 1 buildings in this subtask.

Subtask 3 [20 points]

We have several test cases with 1 <= N, M <= 700 and where each building is just a single character, i.e. each building has size 1 x 1 and has no interior cells, but the NUS map is rather large, so your code has to be efficient.

Subtask 4 [20 points]

We have several test cases with 1 <= N, M <= 30 and each building can be a convex or concave polygon as defined in the problem description. In fact, the sample input shown above fits this category.

Subtask 5 [20 points]

We have several test cases with 1 <= N, M <= 700 and each building can be a convex or concave polygon as defined in the problem description. This is the hardest form of this task and your code has to be very efficient. NUS is a large campus with lots of building, after all.

FEEDBACK

You have **10** tokens that you can use to release the **complete** feedback to the question.

INPUT

The first line of the input is TC, the number of test cases in a particular subtask.

For each test case, you will read two integers N and M in one line separated by a space. These two values are the size of the 2-D map. Then, the input is followed by N rows by M columns, as defined above.

OUTPUT

Output TC number of integers separated by a newline character. Each integer represents the least amount of wetness to get from building ‘A’ to building ‘B’ for each test case.

EXAMPLES

Sample Input	Sample Output
2 7 25 ....CCCC. AAAAA...............C**C. A***A.AAA...........C**C. A***AAA*A....###....CCCC. A*******A................ AAAAAAAAA................ ....DD..............BB 7 25 ....CCCC. AAAAA...............C**C. A***A.AAA...........C**C. A***AAA*A....###....CCCC. A*******A................ AAAAAAAAA................ ....DD..............BB	13 13

IMPLEMENTATION DETAILS

You can use the following skeleton code that will use the correct I/O format:

```
#include < cstdio >
using namespace std;

#define MAX_NM 710 // to cater for subtask 3 and 5

int N, M;
char grid[MAX_NM][MAX_NM];

int compute() {
  int ans = 0;

  // implement your answer here

  return ans;
}

int main() {
  int TC;

  sscanf(gets(grid[0]), "%d", &TC);
  while (TC--) {
    sscanf(gets(grid[0]), "%d %d", &N, &M);
    for (int i = 0; i < N; i++)
      gets(grid[i]);
    printf("%d\n", compute());
  }

  return 0;
}
```