Problem 10 | **72 Points**

Amazing Mouse

Program Name: mouse.java **Input File: mouse.in**

Write a program that simulates the behavior of a mouse in a maze.

As viewed from overhead, the maze is made of discrete square units. Each unit is either a wall, an open space, an open space with a piece of cheese, or an open space containing the mouse (exactly one). The mouse can see across open space in the four cardinal directions but cannot see through walls.

If the mouse can see at least one piece of cheese, he will move one unit toward the piece closest to him. When the mouse moves and occupies the same square as a piece of cheese, he eats it, removing the cheese from the maze. If there is no closest piece of cheese (i.e., there are two or more pieces that tie for the closest or he can see no cheese) then the mouse can't decide what to do next and the simulation ends.

Input

The first line of input will contain a single integer *n* indicating the number of simulations to run.

Each simulation consists of the following input:

1. A single line containing a single integer *m* from 1 to 10 (inclusive) indicating the size of the maze (*m*x*m*)
2. The next *m* lines will each contain *m* characters and represents the maze. Possible characters are:
   'm' – The mouse. There will be exactly one mouse in each maze.
   '#' – A wall.
   '.' – An open space.
   'C' – A piece of cheese.

Note that the only assumption you can make about the maze is that there is exactly one mouse. It is possible to have a maze without walls, without cheese, or without open space.

Output

For each simulation, output a picture of the maze at the end of the simulation. Use the same symbols as used in the input. Do not print the dimensions of the maze.

Example Input File

```
3
5
#####
#m.C#
#.#.#
#..C#
#####
4
####
#mC#
#C.#
####
7
.......
.m....C
.#.#...
...C.C.
.......
.C....C
...C...
```

Example Output To Screen

```
#####
#...#
#.#.#
#..m#
#####
####
#mC#
#C.#
####
.......
.......
.#.#...
.......
.......
.C....C
...m...
```