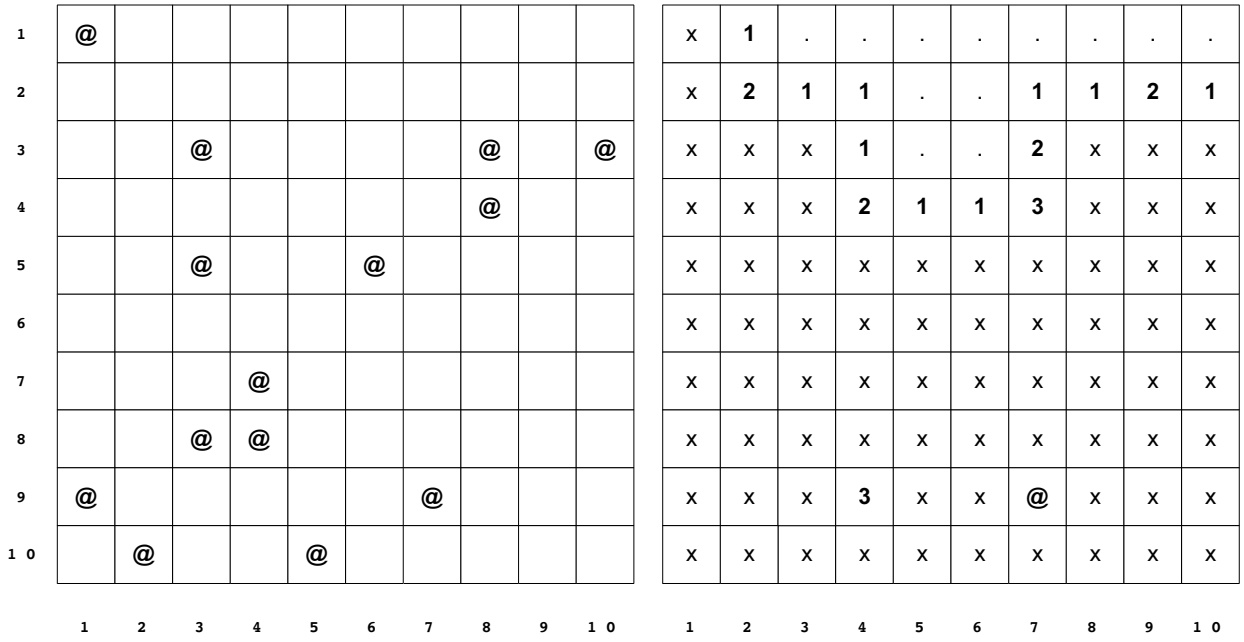
Problem 1 | **Mine Hunt** | **6 Points**

Program Name: minehunt.cpp **Input File: minehunt.dat**

In the game Mine Hunt, you are trying to determine the position of some number of mines on a mine field (an NxN grid). Each position on the field can be represented in 1 of four ways:

Cell Type	Condition of Cell	Neighbors	Notation on Cell	System response
A	Unrevealed position – All positions start as unrevealed and are "revealed" when the player touches them.	Does not matter	x	N/A
B	Revealed position with a mine on it.	Does not matter	@	System displays the mine.
C	Revealed position with no mine on it.	1-8 neighbors have mines on them	The number (1-8) of neighboring positions that have mines.	System displays the number of neighbors with a mine.
D	Revealed position with no mine on it.	0 neighbors have mines on them.	.	System displays a . and processes a user touch on all unrevealed neighbors.

In this program, you will be given a definition of a mine field followed by a series of "touches". Your program is to display the mine field as it would appear after the touches. For example, given the mine field on the left, you can see that the series of touches on the initial screen of (9,4) – (2,5) – (9,7) would result in the screen on the right.



Input

Input to your program is a single mine field followed by a series of "touches". The first 20 lines contain the mine field each with a row of 20 mine field positions on it in columns 1-20. Each position contains either a "." period or an "@" with a period representing an empty position and the "@" representing a mine at that position. Following the mine field definition will be a series of "touches". Each touch ($1 \le row,column \le 20$) will be on a line by itself with the (row,column) combination representing the position found by counting from top to bottom starting with 1 for the row and from left to right starting with 1 for the column. The row value will start in column 1 of the input file and will be followed by exactly 1 blank space followed by the column.

Output

Using the mine field definition and the series of touches, your program should simulate the mine search starting from the initial state (all fields display "x" whether they contain a mine or not) by applying the touches 1 at a time per the rules in the above table. Your program should then print the completed table with the notations from the above table on a series of 20 lines of output with the columns of the mine field in columns 1-20.

Your program should print only the completed simulated mine field. Any other output will be considered extraneous output and will be judged incorrect.

Example: Input File

```
@.........@.........
....................
..@....@.@..@....@.@
.......@.........@..
..@..@......@..@....
....................
...@.........@......
..@@........@@......
@.....@...@.....@...
.@..@......@..@.....
@.........@.........
....................
..@....@.@..@....@.@
.......@.........@..
..@..@......@..@....
....................
...@.........@......
..@@........@@......
@.....@...@.....@...
.@..@......@..@.....
9 4
2 5
9 7
17 8
```

Output to screen

```
x1.......1xxxxxxxxxx
x211..1122xxxxxxxxxx
xxx1..2xxxxxxxxxxxx
xxx2113xxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxx
xxx3xx@xxxxxxxxxxx
xxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxx
xxxxxxxx3112xxxxxxx
xxxxxx211..1xxxxxxx
xxxx211....1xxxxxxx
xxxx2......1xxxxxxx
xxxx2111.112xxxxxxx
xxxxxxx1.1xxxxxxxxx
xxxxxxx1.1xxxxxxxxx
```