Problem 10: Image Perimeters

Source filename:	`image.(cpp|java)`
Input filename:	`image.in`
Output filename:	`image.out`

Technicians in a pathology lab analyze digitized images of slides. Objects on a slide are selected for analysis by a mouse click on the object. The perimeter of the boundary of an object is one useful measurement of the object. Your task is to determine the perimeter of selected objects.

The digitized slides will be represented by a rectangular grid of periods, '`.`', indicating empty space, and the capital letter '`X`', indicating part of an object. Simple examples are:

Grid 1
```
XX
XX
```

Grid 2
```
.XXX
.XXX
.XXX
...X
..X.
X...
```

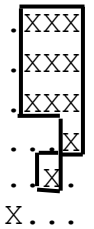
An `X` in a grid square indicates that the entire grid square, including its boundaries, lies in some object. The `X` in the center of the grid below is *adjacent* to the `X` in any of the 8 positions around it. The grid squares for any 2 adjacent `X`'s overlap on an edge or corner, so they are connected.

```
.....
.XXX.
.XXX.
.XXX.
.....
```

(Central `X` is adjacent to 8 other `X`'s; and together they indicate 1 object.)

An object consists of the grid squares of all `X`'s that can be linked to one another through a sequence of adjacent `X`'s. In Grid 1, the whole grid is filled by one object. In Grid 2, there are two objects. One in Grid 1 contain only the lower left grid square. The remaining `X`'s belong to the other object.

The technician will always click on an `X`, selecting the object containing that `X`. Rows and columns are numbered starting with 1 for the upper left hand corner. The technician could select the object in Grid 1 by clicking on row 2 and column 2. The larger object in Grid 2 could be selected by clicking on row 2, column 3. The click could not be on row 4, column 3.



The technician wants to know the perimeter of the selected object. Assume each X corresponds to a square one unit on each side. Hence the object in Grid 1 has perimeter 8 (2 on each of four sides). The perimeter for the larger object in Grid 2 is illustrated in the figure at the left. The perimeter length is 18.

Objects will not contain any totally enclosed holes, so the leftmost grid patterns shown below could NOT appear. On the other hand, the variations on the right could appear.

Impossible

```
XXXX
X..X
XX.X
XXXX

.....
..X..
.X.X.
..X..
.....
```

Possible

```
XXXX      XXXX      XXXX
XXXX      X...      X...
XXXX      XX.X      XX.X
XXXX      XXXX      XX.X

.....     .....     .....
..X..     ..X..     ..X..
.XXX.     .X...     .....
..X..     ..X..     ..X..
.....     .....     .....
```

The input file contains 1 or more grids. The first line contains 4 integers, *m*, *n*, *x*, and *y* separated by 1 or more spaces. The integers, *m* and *n*, represent the number of rows and columns in the grid to follow. The integers, *x* and *y*, represent the x-coordinate and y-coordinate of the grid square selected by the technician by "clicking". The values for *m* and *n* will between 1 and 100, inclusive; and $1<=x<=m$, $1<=y<=n$. The x and y coordinates will correspond to a grid square that contains part of an object (indicated by an 'X').

Following the 4 integers will be *m* lines of *n* characters each. Each character corresponds to one grid square, and is either '.', representing a space, or 'X', representing a part of an object.

After the *m* lines containing the grid, there will be 4 more integers represented a new grid.

Values of 0 for *m*, *n*, *x*, and *y* indicate the end of the input file.

For each grid in the input file, the output file should report the perimeter of the object selected by the given x and y coordinate.

Example input

```
6 4 2 3
.XXX
.XXX
.XXX
...X
..X.
X...
6 4 6 1
.XXX
.XXX
.XXX
...X
..X.
X...
6 4 5 3
.XXX
.XXX
.XXX
...X
..X.
X...
5 6 3 3
.XXXX.
X....X
..XX.X
.X...X
..XXX.
0 0 0 0
```

Example output

```
Perimeter of object 1 at (2,3) = 18
Perimeter of object 2 at (6,1) = 4
Perimeter of object 3 at (5,3) = 18
Perimeter of object 4 at (3,3) = 40
```