Problem 6: The Speckled Letter

With inexpensive scanners and computing systems, optical character recognition, or OCR, is now frequently used and very reliable. Most OCR systems perform some recognition tasks by matching the outline of a scanned character against the outline of known samples. Unfortunately, spots on a scanned image have a significant negative impact on this matching process. In this problem you'll consider how to "despeckle" the image of a character to remove these extraneous spots.

The image of a scanned character will be provided as a rectangular bitmap indicating the light and dark *pixels*, or picture elements, comprising the image. The dark pixels represent regions containing ink (or extraneous marks), and the light pixels represent the unmarked background on which the character was placed. For this problem, 0 and 1 represent light and dark regions, respectively. For example, on the left below is a bitmap with 13 rows and 15 columns representing the capital letter A with two extraneous pixels (speckles, or bad spots) that we wish to remove. If we number the rows 0 to 12 from the top to the bottom, and the columns 0 to 14 from the left to the right, then the extraneous pixels are at row 1, column 2 and row 11, column 14.

```
000000000000000        000000000000000
001000000000000        001.00000000000
000000010000000        000.0000.0000000
000000111000000        000.000111.000000
000001101100000        000.001101.100000
000011000110000        000.011000.110000
000011000110000        000.011000.110000
000011111110000        000.011111.110000
000011000110000        000.011000.110000
000011000110000        000.011000.110000
000011000110000        000.011000.110000
000000000000001        000000000000.0.1
000000000000000        000000000000000
```

To identify the unwanted pixels, first determine the area of the smallest convex polygon that encloses all the dark pixels, treating each dark pixel as a point. This polygon can be visualized as that resulting from stretching a rubber band around the set of dark pixels in the bitmap. This is illustrated by the dotted lines superimposed on the bitmap shown on the right above.

Next determine if removing any of the dark pixels will reduce the area of the convex polygon by at least a specified percentage. If it does, assume that pixel was an unwanted "speckle", remove it from the bitmap (that is, set it to 0), and repeat the steps until there are no pixels that can be removed to reduce the area by at least the specified percentage. If there are multiple pixels that could be removed to suitably reduce the area, select the pixel that reduces the area by the largest percentage. Resolve any remaining tie by first removing the pixel with the smallest row number, and then the pixel with the smallest column number, assuming pixels are numbered as described above.

Input

There will be multiple input cases to consider. The input for each case begins with a line containing two integers *NR* and *NC* ($1 \le NR, NC \le 100$) giving the number of rows and columns in the bitmap, and a real number between 0 and 100 giving the percentage by which a pixel's removal must reduce the area of the enclosing convex polygon for it to be removed. This line is then followed by the bitmap, given as *NR* lines each containing *NC* characters, each 0 or 1, followed by the end of line character. The input for the last case is followed by a line containing three zeroes.

Output

For each case, display the case number (1, 2, …) and the list of pixels to be removed in the order they should be removed. If no pixels are to be removed, explicitly state that as shown in the sample output. Your output should be formatted as shown in the sample below.

Sample Input	Output for the Sample Input
13 15 5.0 000000000000000 001000000000000 000000010000000 000000111000000 000001101100000 000011000110000 000011000110000 000011111110000 000011000110000 000011000110000 000011000110000 000000000000001 000000000000000 13 15 5.0 000000000000000 000111110000000 000110011000000 000110001100000 000110100110000 000110001100000 000111111000000 000110001100000 000110000110000 000110100011000 000110000110000 000111111100000 000000000000000 0 0 0	Case 1: Delete pixel at (1,2) Delete pixel at (11,14) Case 2: No pixels deleted.