Problem D: Contour Tracing

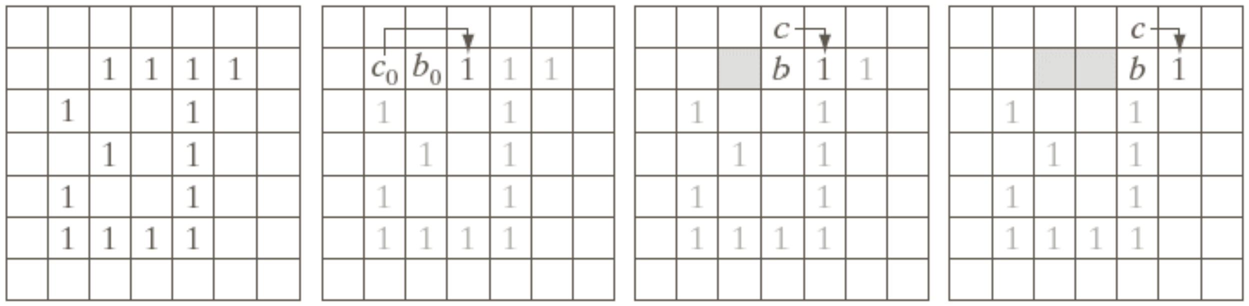
Source: `contour.{c,cpp,java}`
Input: `console {stdin,cin,System.in}`
Output: `console {stdout,cout,System.out}`

In computer vision, objects of interest are often represented as regions of 1's in a binary image (bitmap). An important task in the identification of objects is to trace the contour (also called border and boundary) in an object.

We assume that the bitmap does not contain any 1's on the borders. To trace the contour of a single object, we can use a procedure known as the Moore boundary tracking algorithm as follows:

1. Scan from the top row to the bottom row, from left to right in each row, until an object pixel is found. Call this object pixel b0, and its west background neighbour c0.
2. Examine the 8 neighbours of b0, starting at c0 and proceeding in clockwise direction. Let b1 denote the first neighbour object pixel encountered, and c1 be the background neighbour immediately preceeding b1. Store the locations of b0 and b1, and append b0 and b1 to the contour.
3. Let b = b1 and c = c1
4. Let the 8 neighbours of b, starting at c and proceeding in a clockwise direction, be denoted as n1, n2, ..., n8. Find the first object neighbour nk in this sequence.
5. Let b = nk, c = n(k-1). Append b to the contour.
6. Repeat steps 4 and 5 until b = b0 and the next contour point found is b1. The last two points b0 and b1 are repeated and should not be appended to the contour again.

The first steps of the algorithm is illustrated in the figure below (only the pixels on the boundary are labelled 1 in this bitmap):



In this problem, you will be given a bitmap with at most 200 rows and 200 columns. The bitmap contains a number of objects. You are to determine the length of the contour for each object in the bitmap using the procedure above. In the bitmap, a pixel intensity of 0 represents the background, and a pixel intensity of 1 represents an object pixel. Two pixels with intensity 1 belong to the same object if there is a path from one pixel to another consisting of only 1's, and the path is allowed to follow in any of the 8 compass directions. The borders of the bitmap (first row, last row, first column, last column) contain only background pixels. Any object consisting of fewer than 5 pixels should be ignored as it is most likely noise. None of the objects in the bitmap has holes. Equivalently, there exists a path of background pixels following only the 4 main compass directions (N, S, E, W) for every pair of background pixels in the bitmap.

Input

The input consists of a number of cases. Each case starts with two positive integers on a line, indicating the number of rows (R) and the number of columns (C) in the bitmap. The bitmap is given in the following R lines each containing a string of 0's and 1's of length C. The input is terminated by a case starting with R = C = 0. The last case should not be processed.

Output

For each case, print the case number on its own line. In the next line, print the lengths of the contours of all objects found in the bitmap, sorted in ascending order. Separate the contour lengths by a single space on that line. If the bitmap has no object that have at least 5 pixels, output the line 'no objects found' instead.

Sample Input

```
7 7
0000000
0011110
0111100
0011100
0111100
0111100
0000000
16 7
0000000
0011110
0111100
0011100
0111100
0111100
0000000
0011000
0100110
0000000
0001000
0010100
0010000
0111000
0111000
0000000
4 4
0000
0000
0010
0000
0 0
```

Sample Output

```
Case 1
14
Case 2
8 12 14
Case 3
no objects found
```