5243 Bit Maps

The bitmap is a data structure that arises in many areas of computing. In the area of graphics, for example, a bitmap can represent an image by having a '1' represent a black pixel and a '0' represent a white pixel.

Consider the following two ways of representing a rectangular bit map. In the first, it is simply represented as a two dimensional array of 1's and 0's. The second is based on a decomposition technique. First, the entire bit map is considered. If all bits within it are 1, a '1' is output. If all bits within it are 0, a '0' is output. Otherwise, a D is output, the bit map is divided into quarters (as described below), and each of those is processed in the same way as the original bit map. The quarters are processed in top left, top right, bottom left, bottom right order. Where a bit map being divided has an even number of rows and an even number of columns, all quarters have the same dimensions. Where the number of columns is odd, the left quarters have one more column than the right. Where the number of rows is odd the top quarters have one more row than the bottom. Note that if a region having only one row or one column is divided then two halves result, with the top half processed before the bottom where a single column is divided, and the left half before the right if a single row is divided.

Write a program that will read in bitmaps of either form and transform them to the other form.

Input

Input will consist of a series of bit maps. Each bit map begins with a line giving its format ('B' or 'D') and its dimensions (rows and columns). Neither dimension will be greater than 200. There will be at least one space between each of the items of information. Following this line will be one or more lines containing the sequence of '1', '0' and 'D' characters that represent the bit map, with no intervening spaces. Each line (except the last, which may be shorter) will contain 50 characters. A 'B' type bitmap will be written left to right, top to bottom.

The file will be terminated by a line consisting of a single '#'.

Output

Output will consist of a series of bitmaps, one for each bit map of the input. Output of each bit map begins on a new line and will be in the same format as the input. The width and height are to be output right justified in fields of width four.

Sample Input

```
B 3  4
001000011011
D  2   3
DD10111
#
```

Sample Output

```
D   3   4
D0D1001D101
B   2   3
```

```
101111
```