ACM ICPC 2017 DHAKA REGIONAL (MOCK CONTEST)

Finished

THE CONTEST HAS ENDED.

C. Rounding

Score: 1

CPU: 3s
Memory: 1500MB

Suppose you have a floating-point number with **m (m > 0)** digits after the decimal point. Now if you want to round it up to **d (0 ≤ d < m)** digits after the decimal point, sometimes you need to round up and sometimes you need to round down and sometimes nothing is needed. Sometimes the situation is ambiguous (when rounding up and down causes equal absolute error). Given the number you will have to find for how many values of **d (0 ≤ d < m)** (i) you need to round up, (ii) you need to round down (iii) no change is needed (iv) the situation is ambiguous. For example if the number is 3.141592653589790 we can explain using the table below:

Given a positive number with at most 50 digits after the decimal point, you will count out how many ROUND UP, ROUND DOWN, NO CHANGE and AMBIGUOUS situations occur for all possible values of **d**.

Input

Input file contains at most **10000** test cases. Each of the next **T** lines contain a single positive floating-point number **F (0 < F ≤ 10000)**. This number has at least one and at most **50** digits after the decimal point **(0 < m ≤ 50)**.

Input is terminated by a line containing a single **#**.

Output

For each line of input produce 5 lines of outputs. First line contains the serial of output. The next four lines report the total number of ROUND UP, ROUND DOWN, NO CHANGE and AMBIGIOUS cases considering all possible values of **d (0 ≤ d < m)**. Meaning of d is described in the first paragraph. **Print a blank line after the output for each test case**. Look at the output for sample input for details and strictly follow it.

Sample

Input	Output
3.141592653590 4.00 #	Case 1: ROUND UP: 6 ROUND DOWN: 5 NO CHANGE: 1 AMBIGUOUS: 0 Case 2: ROUND UP: 0 ROUND DOWN: 0 NO CHANGE: 2 AMBIGUOUS: 0