Problem # 3: Rich Rhyme(6715)

(Input File: in3.txt, Output: Console Output)

In the land of Rhyme, the King and his people always speak in rhymes. They are so accustomed in conversing using rhymes that at one point it became monotonous. So what was once a source of happiness and amusement is now creating a strange emotion within the heart of the people. The productivity of the people has become less and more importantly, the happiness index of the land has plummeted. The experts have investigated the whole matter and reported that speaking in rhyme is not the problem; it is the monotony of the same thing that is affecting the people. In fact they suggested changing the definition of rhyme slightly. One of their recommendations is to introduce a new concept of "richness" in a sentence.

A sentence would be termed rich if it starts and ends with the same word! The King seems to like this definition. But he is finding it difficult to even say a simple meaningful sentence that is "rich". The King became quite anxious. Your boss, King's ICT adviser, suggested that a competition among the general people should be thrown where the participants would have to submit a piece (story/poem/essay etc.) full of rich sentences. Then the richest pieces would be judged and be awarded. The King made a decree accordingly. Your boss was ordered to formulate a judging scheme for the pieces and write a program immediately to do the judgment. And as usual he formulated something and asked you to implement that. What he has asked you to do is as follows. You will take as input a piece as a character array. And output another array where each position will give the richness of the piece up to that position.

"What is richness?" You asked.

Your boss replied, "The richness is the size of the identical proper substring which is present at the beginning and at the end of the piece. But, the complete string cannot be considered as its substring."

"But you said to calculate the richness at each position"! You inquired.

"Yes, off course! When you calculate up to a position, you will have to assume that the piece ends at that position." Your boss smiled. He seemed to be quite happy in formulating this.

You asked again, "Don't I need to think about the word boundaries?"

"Not now, my son. We will think about it later. Do it without considering word boundaries for now. You will consider the input character by character". He assured you. He seemed all the more content with his own formulation and probably was not seeing the possible pitfalls. He continued, "And one other point; richness of the first position would always be 0."

Meaningful, you thought. But you had to ask a final question. "How would you then finally compute the richness of the total piece using all these information? What will be the formula?"

Your boss seemed perplexed. He did not think this through, as it seemed. So, he said, "You need not worry about that. Do as I said. Just remember that the input size could be quite large."

You knew that you have to stop asking and start coding now. "Yes, boss!" you said. You also realized the implication of the last direction of your boss. Few years back there was another competition on literary works. And the size of some pieces was 4 million characters!

Input

Input consists of several lines. Each line contains a string of lowercase English letters. End of input is indicated by a line containing the string 'End'. No output line should be produced for this line.

Output

For each input line, you must output a line containing several integers, separated by space. The number of integers must be same as the length of the input string. The Integer at a specific position must represent the richness of the input string up to that position. There must not be any trailing space in any output line.

Sample Input

```
oebmgoca
bdblhc
jgojeqp
atofpgrwk
fdcdd
ababbacdbacdbacd
dacc
cbda
accb
dcbb
dacbbcbcbcbcdcbbbcbcbcbcbcbcdacbbcbcdcbbbcbcdcbbdcbb
end
```

Sample Output

```
0 0 0 0 0 1 0 0

0 0 1 0 0 0

0 0 0 1 0 0 0

0 0 0 0 0 0 0 0 0

0 0 0 0 0

0 0 1 2 0 1 0 0 0 1 0 0 0 1 0 0

0 0 0 0

0 0 0 0

0 0 0 0

0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 2 3 4 5 6 7 8 1 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0
```