I. Yet Another String Matching Problem

time limit per test: 4 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Suppose you have two strings s and t, and their length is equal. You may perform the following operation any number of times: choose two different characters c_1 and c_2, and replace every occurence of c_1 in both strings with c_2. Let's denote the *distance* between strings s and t as the minimum number of operations required to make these strings equal. For example, if s is `abcd` and t is `ddcb`, the *distance* between them is 2 — we may replace every occurence of `a` with `b`, so s becomes `bbcd`, and then we may replace every occurence of `b` with `d`, so both strings become `ddcd`.

You are given two strings S and T. For every substring of S consisting of $|T|$ characters you have to determine the *distance* between this substring and T.

Input

The first line contains the string S, and the second — the string T ($1 \le |T| \le |S| \le 125000$). Both strings consist of lowercase Latin letters from `a` to `f`.

Output

Print $|S|$ - $|T| + 1$ integers. The i-th of these integers must be equal to the *distance* between the substring of S beginning at i-th index with length $|T|$ and the string T.

Example

input

```
abcdefa
ddcb
```

output

```
2 3 3 3
```