Problem B. String Algorithm

Input file:	*standard input*
Output file:	*standard output*
Time limit:	20 seconds
Memory limit:	512 mebibytes

We give you a string s of length n.

Let's fix some k ($1 \le k \le n$). Create $m = \left\lfloor \frac{n}{k} \right\rfloor$ strings of length k, the i-th of them being a substring of s starting with position $(i-1)k+1$: $p_i = s_{(i-1)k+1}s_{(i-1)k+2}\ldots s_{ik}$.

In other words, we cut the string s into strings of length k and discard leftovers. Let $f(k) = |\{(i,j) \mid 1 \le i < j \le m,\ dist(p_i, p_j) \le 1\}|$, where *dist* denotes the Hamming distance. In human words, $f(k)$ is the number of pairs of strings p that are different in at most 1 position.

We ask you to devise an *algorithm* to compute $f(k)$ for all k from 1 to n.

Input

The first line contains one positive integer n ($1 \le n \le 2 \cdot 10^5$) — the length of the string.

The second line contains the string s of length n, consisting of lowercase English characters.

Output

Print n numbers, the k-th of them being $f(k)$.

Examples

standard input	standard output
7 kkekeee	21 2 1 0 0 0 0
10 babaiskeke	45 2 0 0 0 0 0 0 0 0
11 aaabaaabaaa	55 10 2 1 0 0 0 0 0 0 0