Problem F. Palindromes and Superpower — 2

Input file:	`input.txt`
Output file:	`output.txt`
Time limit:	**1.2 seconds**
Memory limit:	**150 mebibytes**

Dima adds letters s_1, ..., s_n one by one to the end of a word. After each letter, he asks Misha to tell him how many new palindrome substrings appeared when he added that letter. Two substrings are considered distinct if they are different as strings. Which n numbers will be said by Misha if it is known that he is never wrong?

Oh, by the way, don't forget to carefully consider **TL**, **ML** and maximum string length. You might have already seen this problem with different limitations. We have done everything to make your old solution unacceptable :)

Input

The input contains a string $s_1 \dots s_n$ consisting of letters '`a`' and '`b`' ($1 \le n \le 5\,000\,000$).

Output

Print n numbers without spaces: i-th number must be the number of palindrome substrings of the prefix $s_1 \dots s_i$ minus the number of palindrome substrings of the prefix $s_1 \dots s_{i-1}$. The first number in the output should be one.

Example

input.txt	output.txt
abbbba	111111

Note

The only Java solution we have for this problem handles input and output very carefully. We can guarantee that it meets **ML** and **TL**/2.

We are against separate **TL** and **ML** for different programming languages. We also do not consider lack of a Java solution for a problem as normal. Anyway, if you choose to write your code in Java, you must surely know particular ways to deal with memory, input and output in this language.

Just in case, here is a tip from us: we handled input and output using **FileReader** and **FileWriter** only.

```
FileReader in = new FileReader("input.txt");
FileWriter out = new FileWriter("output.txt");
s = new char[SIZE];
result = new char[SIZE];
int slen = in.read(s, 0, SIZE);
...
out.write(result, 0, slen);
out.flush();
```