2743. Count Substrings Without Repeating Character

Description

You are given a string `s` consisting only of lowercase English letters. We call a substring **special** if it contains no character which has occurred at least twice (in other words, it does not contain a repeating character). Your task is to count the number of **special** substrings. For example, in the string `"pop"`, the substring `"po"` is a **special** substring, however, `"pop"` is not **special** (since `'p'` has occurred twice).

Return *the number of **special** substrings.*

A **substring** is a contiguous sequence of characters within a string. For example, `"abc"` is a substring of `"abcd"`, but `"acd"` is not.

Example 1:

```
Input: s = "abcd"
Output: 10
Explanation: Since each character occurs once, every substring is a special substring. We have 4 substrings of length one, 3 of length two, 2 of length three, and 1 substring of length four. So overall there are 4 + 3 + 2 + 1 = 10 special substrings.
```

Example 2:

```
Input: s = "ooo"
Output: 3
Explanation: Any substring with a length of at least two contains a repeating character. So we have to count the number of substrings of length one, which is 3.
```

Example 3:

```
Input: s = "abab"
Output: 7
Explanation: Special substrings are as follows (sorted by their start positions):
Special substrings of length 1: "a", "b", "a", "b"
Special substrings of length 2: "ab", "ba", "ab"
And it can be shown that there are no special substrings with a length of at least three. So the answer would be 4 + 3 = 7.
```

Constraints:

- `1 <= s.length <= 10^5`
- `s` consists of lowercase English letters