

# 1531. String Compression II

## Description

**Run-length encoding** is a string compression method that works by replacing consecutive identical characters (repeated 2 or more times) with the concatenation of the character and the number marking the count of the characters (length of the run). For example, to compress the string `"aabccc"` we replace `"aa"` by `"a2"` and replace `"ccc"` by `"c3"`. Thus the compressed string becomes `"a2bc3"`.

Notice that in this problem, we are not adding `'1'` after single characters.

Given a string `s` and an integer `k`. You need to delete **at most** `k` characters from `s` such that the run-length encoded version of `s` has minimum length.

Find the *minimum length of the run-length encoded version of `s` after deleting at most `k` characters*.

### Example 1:

**Input:** `s = "aaabcccd", k = 2`

**Output:** 4

**Explanation:** Compressing `s` without deleting anything will give us `"a3bc3d"` of length 6. Deleting any of the characters `'a'` or `'c'` would at most decrease the length of the compressed string to 5, for instance delete 2 `'a'` then we will have `s = "abcccd"` which compressed is `abc3d`. Therefore, the optimal way is to delete `'b'` and `'d'`, then the compressed version of `s` will be `"a3c3"` of length 4.

### Example 2:

**Input:** `s = "aabbbaa", k = 2`

**Output:** 2

**Explanation:** If we delete both `'b'` characters, the resulting compressed string would be `"a4"` of length 2.

### Example 3:

**Input:** `s = "aaaaaaaaaa", k = 0`

**Output:** 3

**Explanation:** Since `k` is zero, we cannot delete anything. The compressed string is `"a11"` of length 3.

### Constraints:

- `1 <= s.length <= 100`
- `0 <= k <= s.length`
- `s` contains only lowercase English letters.

