@LeetCode

Given an array of characters, compress it in-place.

The length after compression must always be smaller than or equal to the original array.

Every element of the array should be a **character** (not int) of length 1.

After you are done **modifying the input array** in-place, return the new length of the array.

Follow up:
Could you solve it using only O(1) extra space?

Example 1:

```
Input:

["a","a","b","b","c","c","c"]



Output:

Return 6, and the first 6 characters of the input array should be:
["a","2","b","2","c","3"]



Explanation:

"aa" is replaced by "a2". "bb" is replaced by "b2". "ccc" is replaced by "c3".
```

Example 2:

```
Input:

["a"]



Output:

Return 1, and the first 1 characters of the input array should be: ["a"]
```

```
Explanation:

Nothing is replaced.
```

Example 3:

```
Input:

["a","b","b","b","b","b","b","b","b","b","b","b","b"]



Output:

Return 4, and the first 4 characters of the input array should be: ["a","b","1","2"].



Explanation:

Since the character "a" does not repeat, it is not compressed. "bbbbbbbbbbbb" is
replaced by "b12".

Notice each digit has it's own entry in the array.
```

Note:

1. All characters have an ASCII value in `[35, 126]`.
2. `1 <= len(chars) <= 1000`.