# Problem 3167: Better Compression of String

## Problem Information

**Difficulty:** Medium
**Acceptance Rate:** 75.52%
**Paid Only:** Yes
**Tags:** Hash Table, String, Sorting, Counting

## Problem Description

You are given a string `compressed` representing a compressed version of a string. The format is a character followed by its frequency. For example, `"a3b1a1c2"` is a compressed version of the string `"aaabacc"`.

We seek a **better compression** with the following conditions:

1. Each character should appear **only once** in the compressed version. 2. The characters should be in **alphabetical order**.

Return the _better compression_ of `compressed`.

**Note:** In the better version of compression, the order of letters may change, which is acceptable.

**Example 1:**

**Input:** compressed = "a3c9b2c1"

**Output:** "a3b2c10"

**Explanation:**

Characters "a" and "b" appear only once in the input, but "c" appears twice, once with a size of 9 and once with a size of 1.

Hence, in the resulting string, it should have a size of 10.

**Example 2:**

**Input:** compressed = "c2b3a1"

**Output:** "a1b3c2"

**Example 3:**

**Input:** compressed = "a2b4c1"

**Output:** "a2b4c1"

**Constraints:**

* `1 <= compressed.length <= 6 * 104` * `compressed` consists only of lowercase English letters and digits. * `compressed` is a valid compression, i.e., each character is followed by its frequency. * Frequencies are in the range `[1, 104]` and have no leading zeroes.

## Code Snippets

**C++:**

```
class Solution {
public:
string betterCompression(string compressed) {

}
};
```

**Java:**

```
class Solution {
public String betterCompression(String compressed) {

}
}
```

**Python3:**

```
class Solution:
def betterCompression(self, compressed: str) -> str:
```