Problem 3597: Partition String

Problem Information

Difficulty: Medium
Acceptance Rate: 58.10%
Paid Only: No
Tags: Hash Table, String, Trie, Simulation

Problem Description

Given a string `s`, partition it into **unique segments** according to the following procedure:

* Start building a segment beginning at index 0. * Continue extending the current segment character by character until the current segment has not been seen before. * Once the segment is unique, add it to your list of segments, mark it as seen, and begin a new segment from the next index. * Repeat until you reach the end of `s`.

Return an array of strings `segments`, where `segments[i]` is the `ith` segment created.

Example 1:

Input: s = "abbccccd"

Output: ["a","b","bc","c","cc","d"]

Explanation:

Index | Segment After Adding | Seen Segments | Current Segment Seen Before? | New Segment | Updated Seen Segments ---|---|---|---|---|--- 0 | "a" | [] | No | "" | ["a"] 1 | "b" | ["a"] | No | "" | ["a", "b"] 2 | "b" | ["a", "b"] | Yes | "b" | ["a", "b"] 3 | "bc" | ["a", "b"] | No | "" | ["a", "b", "bc"] 4 | "c" | ["a", "b", "bc"] | No | "" | ["a", "b", "bc", "c"] 5 | "c" | ["a", "b", "bc", "c"] | Yes | "c" | ["a", "b", "bc", "c"] 6 | "cc" | ["a", "b", "bc", "c"] | No | "" | ["a", "b", "bc", "c", "cc"] 7 | "d" | ["a", "b", "bc", "c", "cc"] | No | "" | ["a", "b", "bc", "c", "cc", "d"] Hence, the final output is `["a", "b", "bc", "c", "cc", "d"]`.

Example 2:

Input: s = "aaaa"

Output: ["a","aa"]

Explanation:

Index | Segment After Adding | Seen Segments | Current Segment Seen Before? | New Segment | Updated Seen Segments ---|---|---|---|---|--- 0 | "a" | [] | No | "" | ["a"] 1 | "a" | ["a"] | Yes | "a" | ["a"] 2 | "aa" | ["a"] | No | "" | ["a", "aa"] 3 | "a" | ["a", "aa"] | Yes | "a" | ["a", "aa"] Hence, the final output is `["a", "aa"]`.

Constraints:

* `1 <= s.length <= 105` * `s` contains only lowercase English letters.

Code Snippets

C++:

```
class Solution {
public:
vector<string> partitionString(string s) {

}
};
```

Java:

```
class Solution {
public List<String> partitionString(String s) {

}
}
```

Python3:

```
class Solution:
def partitionString(self, s: str) -> List[str]:
```