Problem 2213: Longest Substring of One Repeating Character

Problem Information

Difficulty: Hard
Acceptance Rate: 33.89%
Paid Only: No
Tags: Array, String, Segment Tree, Ordered Set

Problem Description

You are given a **0-indexed** string `s`. You are also given a **0-indexed** string `queryCharacters` of length `k` and a **0-indexed** array of integer **indices** `queryIndices` of length `k`, both of which are used to describe `k` queries.

The `ith` query updates the character in `s` at index `queryIndices[i]` to the character `queryCharacters[i]`.

Return _an array_ `lengths` _of length_`k` _where_ `lengths[i]` _is the**length** of the **longest substring** of _`s` _consisting of**only one repeating** character **after** the_ `ith` _query_ _is performed._

Example 1:

Input: s = "babacc", queryCharacters = "bcb", queryIndices = [1,3,3] **Output:** [3,3,4] **Explanation:** - 1st query updates s = "_b**b** b_acc". The longest substring consisting of one repeating character is "bbb" with length 3. - 2nd query updates s = "bbb _**c** cc_". The longest substring consisting of one repeating character can be "bbb" or "ccc" with length 3. - 3rd query updates s = "_bbb**b**_ cc". The longest substring consisting of one repeating character is "bbbb" with length 4. Thus, we return [3,3,4].

Example 2:

Input: s = "abyzz", queryCharacters = "aa", queryIndices = [2,1] **Output:** [2,3] **Explanation:** - 1st query updates s = "ab**a** _zz_ ". The longest substring consisting of one repeating character is "zz" with length 2. - 2nd query updates s = "_a**a** a_zz". The longest substring consisting of one repeating character is "aaa" with length 3. Thus, we return [2,3].

Constraints:

* `1 <= s.length <= 105` * `s` consists of lowercase English letters. * `k == queryCharacters.length == queryIndices.length` * `1 <= k <= 105` * `queryCharacters` consists of lowercase English letters. * `0 <= queryIndices[i] < s.length`

Code Snippets

C++:

```
class Solution {
public:
vector<int> longestRepeating(string s, string queryCharacters, vector<int>&
queryIndices) {

}
};
```

Java:

```
class Solution {
public int[] longestRepeating(String s, String queryCharacters, int[]
queryIndices) {

}
}
```

Python3:

```
class Solution:
def longestRepeating(self, s: str, queryCharacters: str, queryIndices:
List[int]) -> List[int]:
```