Our Solution(s) Run Code

Solution 1

```java
// Copyright © 2020 AlgoExpert, LLC. All rights reserved.

import java.util.*;

class Program {
  // O(b + s) time | O(b + s) space - where b is the length of the bi
  // input string and s is the length of the small input string
  public static String smallestSubstringContaining(String bigString, S
    Map<Character, Integer> targetCharCounts = getCharCounts(smallStri
    List<Integer> substringBounds = getSubstringBounds(bigString, targ
    return getStringFromBounds(bigString, substringBounds);
  }

  public static Map<Character, Integer> getCharCounts(String string) {
    Map<Character, Integer> charCounts = new HashMap<Character, Intege
    for (int i = 0; i < string.length(); i++) {
      increaseCharCount(string.charAt(i), charCounts);
    }
    return charCounts;
  }

  public static List<Integer> getSubstringBounds(
      String string, Map<Character, Integer> targetCharCounts) {
    List<Integer> substringBounds = new ArrayList<Integer>(Arrays.asLi
    Map<Character, Integer> substringCharCounts = new HashMap<Characte
    int numUniqueChars = targetCharCounts.size();
    int numUniqueCharsDone = 0;
    int leftIdx = 0;
    int rightIdx = 0;
    // Move the rightIdx to the right in the string until you've count
    // all of the target characters enough times.
    while (rightIdx < string.length()) {
      char rightChar = string.charAt(rightIdx);
```

Your Solutions Run Code

Solution 1 Solution 2 Solution 3

```java
import java.util.*;

class Program {
  public static String smallestSubstringContaining(String bigString, S
    // Write your code here.
    return "";
  }
}

```

Our Tests

Custom Output Submit Code

Run or submit code when you're ready.