

Our Solution(s)

Run Code

Your Solutions

Run Code

Solution 1

```
1 // Copyright © 2020 AlgoExpert, LLC. All rights reserved.
2
3 class Program {
4     // O(b + s) time | O(b + s) space - where b is the length of the big input string
5     // input string and s is the length of the small input string
6     func smallestSubstringContaining(_ bigString: String, _ smallString: String) -> String {
7         let targetCharCounts = getCharCounts(smallString)
8         let substringBounds = getSubstringBounds(bigString, targetCharCounts)
9         return getStringFromBounds(bigString, substringBounds)
10    }
11
12    func getCharCounts(_ str: String) -> [Character: Int] {
13        var charCounts = [Character: Int]()
14        for char in str {
15            changeCharCount(char, &charCounts, 1)
16        }
17        return charCounts
18    }
19
20    func changeCharCount(_ char: Character, _ charCounts: inout [Character: Int], _ change: Int) {
21        if let count = charCounts[char] {
22            charCounts.updateValue(count + change, forKey: char)
23            return
24        }
25        charCounts[char] = change
26    }
27
28    func getSubstringBounds(_ str: String, _ targetCharCounts: [Character: Int]) -> (Int, Int) {
29        var substringBounds = [0, Int.max]
30        var substringCharCounts = [Character: Int]()
31        var numUniqueChars = targetCharCounts.count
32        var numUniqueCharsDone = 0
33        var leftIdx = 0
```

Solution 1

Solution 2

Solution 3

```
1 class Program {
2     func smallestSubstringContaining(_ bigString: String, _ smallString: String) -> String {
3         // Write your code here.
4         return ""
5     }
6 }
7
```

Our Tests

Custom Output

Submit Code

```
1 // Copyright © 2020 AlgoExpert, LLC. All rights reserved.
2
3 func smallestSubstringContaining(_ bigString: String, _ smallString: String) -> String {
4     let targetCharCounts = getCharCounts(smallString)
5     let substringBounds = getSubstringBounds(bigString, targetCharCounts)
6     return getStringFromBounds(bigString, substringBounds)
7 }
8
9 func getCharCounts(_ str: String) -> [Character: Int] {
10    var charCounts = [Character: Int]()
11    for char in str {
12        changeCharCount(char, &charCounts, 1)
13    }
14    return charCounts
15 }
16
17 func changeCharCount(_ char: Character, _ charCounts: inout [Character: Int], _ change: Int) {
18    if let count = charCounts[char] {
19        charCounts.updateValue(count + change, forKey: char)
20        return
21    }
22    charCounts[char] = change
23 }
24
25 func getSubstringBounds(_ str: String, _ targetCharCounts: [Character: Int]) -> (Int, Int) {
26    var substringBounds = [0, Int.max]
27    var substringCharCounts = [Character: Int]()
28    var numUniqueChars = targetCharCounts.count
29    var numUniqueCharsDone = 0
30    var leftIdx = 0
31    var rightIdx = 0
32    while numUniqueCharsDone < numUniqueChars {
33        rightIdx += 1
34        let char = str[rightIdx]
35        changeCharCount(char, &substringCharCounts, 1)
36        if let count = substringCharCounts[char] {
37            if count == targetCharCounts[char] {
38                numUniqueCharsDone += 1
39            }
40        }
40        while numUniqueCharsDone == numUniqueChars {
41            substringBounds = [rightIdx - leftIdx, substringBounds[0]]
42            let char = str[leftIdx]
43            changeCharCount(char, &substringCharCounts, -1)
44            leftIdx += 1
45        }
46    }
47    return substringBounds
48 }
```

Run or submit code when you're ready.