





Grokking the Coding Interview: Patterns for Coding Questions

4% completed C

Q Search Course

Pattern: Cyclic Sort

- Introduction (/courses/grokking-the-codinginterview/YVjXo6J9xN9)
- Cyclic Sort (easy) (/courses/grokking-the-codinginterview/B8qXVqVwDKY)
- Find the Missing Number (easy) (/courses/grokking-the-coding-interview/JPnp17NYXE9)

Find all Missing Numbers
(easy)
(/courses/grokking-the-coding-interview/Y52qNM0ljWK)

Longest Substring with Same Letters after Replacement (hard)

We'll cover the following

- Problem Statement
- Try it yourself
- Solution
- Code
 - Time Complexity
 - Space Complexity

Problem Statement

Given a string with lowercase letters only, if you are allowed to **replace no more than** 'k' letters with any letter, find the **length of the longest substring having the same** letters after replacement.

Example 1:

Grokking the Coding Interview: Patterns for Coding Questions

4% completed

Q Search Course

Pattern: Cyclic Sort

Introduction (/courses/grokking-the-codinginterview/YVjXo6J9xN9)

Cyclic Sort (easy) (/courses/grokking-the-codinginterview/B8gXVqVwDKY)

Find the Missing Number (easy)

(/courses/grokking-the-coding-interview/JPnp17NYXE9)

Find all Missing Numbers (easy) (/courses/grokking-the-codinginterview/Y52qNM0ljWK) Input: String="aabccbb", k=2

Output: 5

Explanation: Replace the two 'c' with 'b' to have a longest repeating substring

"bbbbb".

Example 2:

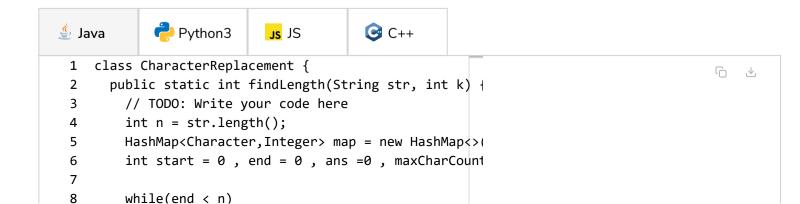
Input: String="abbcb", k=1
Output: 4
Explanation: Replace the 'c' with 'b' to have a longest repeating substring "bbb".

Example 3:

Input: String="abccde", k=1
Output: 3
Explanation: Replace the 'b' or 'd' with 'c' to have the longest repeating substring "ccc".

Try it yourself

Try solving this question here:



Grokking the Coding Interview: Patterns for Coding Questions

4% completed

Q Search Course

Pattern: Cyclic Sort

Introduction (/courses/grokking-the-codinginterview/YVjXo6J9xN9)

Cyclic Sort (easy) (/courses/grokking-the-codinginterview/B8qXVqVwDKY)

Find the Missing Number (easy)

(/courses/grokking-the-coding-interview/JPnp17NYXE9)

Find all Missing Numbers (easy) (/courses/grokking-the-coding-interview/Y52gNM0ljWK)

```
9
          char temp = str.charAt(end);
10
          map.put(temp,map.getOrDefault(temp,0)+1);
11
          maxCharCount = Math.max(maxCharCount,map.get(
12
13
          if(end-start+1 - maxCharCount > k)
14
            char temp2 = str.charAt(start);
15
            map.put(temp2, map.get(temp2)-1);
16
17
            start++;
18
          ans = Math.max(ans, end - start +1);
19
20
          end++;
21
22
23
        return ans;
24
25
26
```

Test

Save *

Reset

X

[]

Show Results

Show Console

3 of 3 Tests Passed

| Result | Input | Expected Output | Actual Output | Reason |
|----------|------------------------|-----------------|---------------|-----------|
| ✓ | findLength(aabccbb, 2) | 5 | 5 | Succeeded |
| ~ | findLength(abbcb, 1) | 4 | 4 | Succeeded |
| ~ | findLength(abccde, 1) | 3 | 3 | Succeeded |





Grokking the Coding Interview: Patterns for Coding Questions

4% completed C

Pattern: Cyclic Sort

Introduction (/courses/grokking-the-codinginterview/YVjXo6J9xN9)

Cyclic Sort (easy) (/courses/grokking-the-codinginterview/B8qXVqVwDKY)

Find the Missing Number (easy) (/courses/grokking-the-coding-interview/JPnp17NYXE9)

Find all Missing Numbers (easy) (/courses/grokking-the-coding-interview/Y52gNM0ljWK)

Solution

This problem follows the **Sliding Window** pattern, and we can use a similar dynamic sliding window strategy as discussed in No-repeat Substring (https://www.educative.io/collection/page/5668639101419520/5671464854355968/54850 10335301632/). We can use a HashMap to count the frequency of each letter.

- We'll iterate through the string to add one letter at a time in the window.
- We'll also keep track of the count of the maximum repeating letter in **any** window (let's call it maxRepeatLetterCount).
- So, at any time, we know that we can have a window which has one letter repeating maxRepeatLetterCount times; this means we should try to replace the remaining letters.
- If we have more than 'k' remaining letters, we should shrink the window as we are not allowed to replace more than 'k' letters.

While shrinking the window, we don't need to update <code>maxRepeatLetterCount</code> (which makes it global count; hence, it is the maximum count for ANY window). Why don't we need to update this count when we shrink the window? The answer: In any window, since we have to replace all the remaining letters to get the longest substring having the same letter, we can't get a better answer from any other window even though all occurrences of the letter with frequency <code>maxRepeatLetterCount</code> is not in the current window.

Grokking the Coding Interview: Patterns for **Coding Questions**

4% completed Search Course

Pattern: Cyclic Sort

Introduction (/courses/grokking-the-codinginterview/YVjXo6J9xN9) Cyclic Sort (easy)

(/courses/grokking-the-codinginterview/B8qXVqVwDKY)

Find the Missing Number (easy) (/courses/grokking-the-codinginterview/JPnp17NYXE9)

Find all Missing Numbers (easy) (/courses/grokking-the-codinginterview/Y52qNM0ljWK)

Code





Here is what our algorithm will look like:

```
Java
                                     Python3
                                                                               G C++
                                                                                                                      Js JS
             import java.util.*;
                                                                                                                                                                                                                                                                       \downarrow
    2
             class CharacterReplacement {
                   public static int findLength(String str, int k) {
    4
                         int windowStart = 0, maxLength = 0, maxRepeatLe
                         Map<Character, Integer> letterFrequencyMap = ne
    6
                         // try to extend the range [windowStart, window
    7
    8
                         for (int windowEnd = 0; windowEnd < str.length(</pre>
                               char rightChar = str.charAt(windowEnd);
    9
                               letterFrequencyMap.put(rightChar, l
 10
 11
                               maxRepeatLetterCount = Math.max(maxRepeatLet1
 12
 13
                               // current window size is from windowStart to
 14
                               // repeating 'maxRepeatLetterCount' times, th
                               // repeating 'maxRepeatLetterCount' times and
 15
                               // if the remaining letters are more than 'k'
 16
                               // are not allowed to replace more than 'k' ]
 17
 18
                               if (windowEnd - windowStart + 1 - maxRepeatLe
 19
                                     char leftChar = str.charAt(windowStart);
 20
                                     letterFrequencyMap.put(leftChar, letterFreq
 21
                                     windowStart++;
 22
 23
                               maxLength = Math.max(maxLength, windowEnd - v
 24
 25
 26
 27
                         return maxLength;
 28
                   }
 29
 30
                   public static void main(String[] args) {
                         System.out.println(CharacterReplacement.findLer
 31
      Run
                                                                                                                                                                                                               Save
                                                                                                                                                                                                                                              Reset
```



Grokking the Coding Interview: Patterns for Coding Questions

4% completed C

Q Search Course

Pattern: Cyclic Sort

Introduction (/courses/grokking-the-codinginterview/YVjXo6J9xN9)

Cyclic Sort (easy) (/courses/grokking-the-codinginterview/B8gXVqVwDKY)

Find the Missing Number (easy) (/courses/grokking-the-cod

(/courses/grokking-the-coding-interview/JPnp17NYXE9)

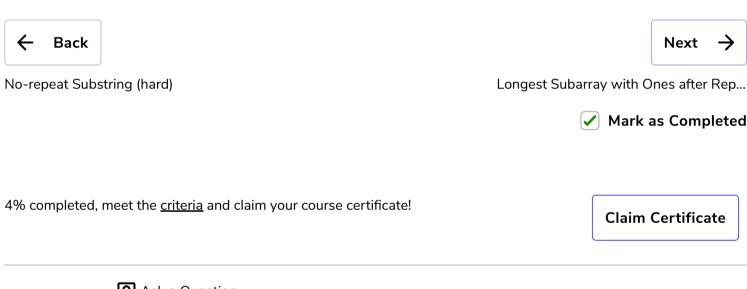
Find all Missing Numbers (easy)
(/courses/grokking-the-coding-interview/Y52qNM0ljWK)

Time Complexity #

The above algorithm's time complexity will be O(N), where 'N' is the number of letters in the input string.

Space Complexity

As we expect only the lower case letters in the input string, we can conclude that the space complexity will be O(26) to store each letter's frequency in the **HashMap**, which is asymptotically equal to O(1).



Reportan Issue

? Ask a Question

 $(https://discuss.educative.io/tag/longest-substring-with-same-letters-after-replacement-hard_pattern-sliding-window_grokking-the-coding-interview-patterns-for-coding-questions)\\$