GeeksforGeeks

Array Matrix Strings Hashing Linked List Stack Queue Binary Tree Binary Search

Count of substrings of length K with exactly K distinct characters

Difficulty Level : Medium • Last Updated : 19 Aug, 2021

Given string **str** of the lowercase alphabet and an integer **K**, the task is to count all substrings of length **K** which have exactly **K** distinct characters.

Example:

> ***Input:*** *str = "abcc", K = 2*
> ***Output:*** *2*
> ***Explanation:***
> *Possible substrings of length K = 2 are*
> *ab : 2 distinct characters*
> *bc : 2 distinct characters*
> *cc : 1 distinct character*
> *Only two valid substrings exist {"ab", "bc"}.*
>
> ***Input:*** *str = "aabab", K = 3*
> ***Output:*** *0*
> ***Explanation:***
> *Possible substrings of length K = 3 are*
> *aab : 2 distinct characters*

aba : 2 distinct characters

bab : 2 distinct characters

No substrings of length 3 exist with exactly 3 distinct characters.

Recommended Practice

Substrings of length k with k-1 distinct elements

Try It!

Naive approach:

The idea is to generate all substrings of length **K** and, for each substring count, a number of distinct characters. If the length of a string is **N**, then there can be **N – K + 1** substring of length **K**. Generating these substrings will require **O(N)** complexity, and checking each substring requires **O(K)** complexity, hence making the overall complexity like **O(N*K).**

Efficient approach:

The idea is to use Window Sliding Technique. Maintain a window of size **K** and keep a count of all the characters in the window using a HashMap. Traverse through the string reduces the count of the first character of the previous window and adds the frequency of the last character of the current window in the **HashMap**. If the count of distinct characters in a window of length **K** is equal to **K**, increment the answer by 1.

Below is the implementation of the above approach:

C++

```
// C++ program to find the
// count of k length substrings
// with k distinct characters
// using sliding window
#include <bits/stdc++.h>
using namespace std;

// Function to return the
// required count of substrings
int countSubstrings(string str, int K)
{
    int N = str.size();
    // Store the count
    int answer = 0;

    // Store the count of
    // distinct characters
    // in every window
    unordered_map<char, int> map;

    // Store the frequency of
    // the first K length substring
    for (int i = 0; i < K; i++) {

        // Increase frequency of
        // i-th character
        map[str[i]]++;
    }

    // If K distinct characters
    // exist
    if (map.size() == K)
        answer++;

    // Traverse the rest of the
    // substring
    for (int i = K; i < N; i++) {

        // Increase the frequency
        // of the last character
        // of the current substring
        map[str[i]]++;
        // Decrease the frequency
        // of the first character
        // of the previous substring
        map[str[i - K]]--;
```

```
            // If the character is not present
            // in the current substring
            if (map[str[i - K]] == 0) {
                map.erase(str[i - K]);
            }

            // If the count of distinct
            // characters is 0
            if (map.size() == K) {
                answer++;
            }
        }

        // Return the count
        return answer;
    }

    // Driver code
    int main()
    {
        // string str
        string str = "aabcdabbcdc";

        // integer K
        int K = 3;

        // Print the count of K length
        // substrings with k distinct characters
        cout << countSubstrings(str, K) << endl;

        return 0;
    }
```

Java

```
// Java program to find the count
// of k length substrings with k
// distinct characters using
// sliding window
import java.util.*;

class GFG{

// Function to return the
// required count of substrings
public static int countSubstrings(String str,
                                  int K)
{
    int N = str.length();

    // Store the count
    int answer = 0;

    // Store the count of
    // distinct characters
    // in every window
    Map<Character,
        Integer> map = new HashMap<Character,
                                  Integer>();

    // Store the frequency of
    // the first K length substring
    for(int i = 0; i < K; i++)
    {

        // Increase frequency of
        // i-th character
        if (map.get(str.charAt(i)) == null)
        {
            map.put(str.charAt(i), 1);
        }
        else
        {
            map.put(str.charAt(i),
            map.get(str.charAt(i)) + 1);
        }
    }

    // If K distinct characters
    // exist
    if (map.size() == K)
        answer++;

    // Traverse the rest of the
    // substring
    for(int i = K; i < N; i++)
    {

        // Increase the frequency
```

```
            // of the last character
            // of the current substring
            if (map.get(str.charAt(i)) == null)
            {
                map.put(str.charAt(i), 1);
            }
            else
            {
                map.put(str.charAt(i),
                map.get(str.charAt(i)) + 1);
            }

            // Decrease the frequency
            // of the first character
            // of the previous substring
            map.put(str.charAt(i - K),
            map.get(str.charAt(i - K)) - 1);

            // If the character is not present
            // in the current substring
            if (map.get(str.charAt(i - K)) == 0)
            {
                map.remove(str.charAt(i - K));
            }

            // If the count of distinct
            // characters is 0
            if (map.size() == K)
            {
                answer++;
            }
        }

        // Return the count
        return answer;
    }

    // Driver code
    public static void main(String[] args)
    {

        // string str
        String str = "aabcdabbcdc";

        // integer K
        int K = 3;

        // Print the count of K length
        // substrings with k distinct characters
        System.out.println(countSubstrings(str, K));
    }
    }

    // This code is contributed by grand master
```

Python3

```
# Python3 program to find the
# count of k length substrings
# with k distinct characters
# using sliding window

# Function to return the
# required count of substrings
def countSubstrings(str, K):

    N = len(str)

    # Store the count
    answer = 0

    # Store the count of
    # distinct characters
    # in every window
    map = {}

    # Store the frequency of
    # the first K length substring
    for i in range(K):

        # Increase frequency of
        # i-th character
        map[str[i]] = map.get(str[i], 0) + 1

    # If K distinct characters
    # exist
    if (len(map) == K):
        answer += 1
    # Traverse the rest of the
    # substring
    for i in range(K, N):

        # Increase the frequency
        # of the last character
        # of the current substring
        map[str[i]] = map.get(str[i], 0) + 1

        # Decrease the frequency
        # of the first character
        # of the previous substring
        map[str[i - K]] -= 1

        # If the character is not present
        # in the current substring
        if (map[str[i - K]] == 0):
            del map[str[i - K]]

        # If the count of distinct
        # characters is 0
        if (len(map) == K):
            answer += 1

    # Return the count
    return answer

# Driver code
if __name__ == '__main__':

    str = "aabcdabbcdc"

    # Integer K
    K = 3

    # Print the count of K length
    # substrings with k distinct characters
    print(countSubstrings(str, K))

# This code is contributed by mohit kumar 29
```

C#

```
// C# program to find the count
// of k length substrings with k
// distinct characters using
// sliding window
using System;
using System.Collections.Generic;

class GFG{

// Function to return the
// required count of substrings
public static int countSubstrings(string str,
                                  int K)
{
    int N = str.Length;

    // Store the count
    int answer = 0;

    // Store the count of
    // distinct characters
    // in every window
    Dictionary<char,
               int> map = new Dictionary<char,
                                         int>();

    // Store the frequency of
    // the first K length substring
    for(int i = 0; i < K; i++)
    {

        // Increase frequency of
        // i-th character
        if(!map.ContainsKey(str[i]))
        {
            map[str[i]] = 1;
        }
        else
        {
            map[str[i]]++;
        }
    }

    // If K distinct characters
    // exist
    if (map.Count == K)
```

```
            answer++;

        // Traverse the rest of the
        // substring
        for(int i = K; i < N; i++)
        {

            // Increase the frequency
            // of the last character
            // of the current substring
            if(!map.ContainsKey(str[i]))
            {
                map[str[i]] = 1;
            }
            else
            {
                map[str[i]]++;
            }

            // Decrease the frequency
            // of the first character
            // of the previous substring
            map[str[i - K]]--;

            // If the character is not present
            // in the current substring
            if (map[str[i - K]] == 0)
            {
                map.Remove(str[i - K]);
            }

            // If the count of distinct
            // characters is 0
            if (map.Count == K)
            {
                answer++;
            }
        }

        // Return the count
        return answer;
    }

    // Driver code
    public static void Main(string[] args)
```

```
{

    // string str
    string str = "aabcdabbcdc";

    // integer K
    int K = 3;

    // Print the count of K length
    // substrings with k distinct characters
    Console.Write(countSubstrings(str, K));
}
}

// This code is contributed by rutvik_56
```

Javascript

```
<script>

// Javascript program to find the
// count of k length substrings
// with k distinct characters
// using sliding window

// Function to return the
// required count of substrings
function countSubstrings(str, K)
{
    var N = str.length;
    // Store the count
    var answer = 0;

    // Store the count of
    // distinct characters
    // in every window
    var map = new Map();

    // Store the frequency of
    // the first K length substring
    for (var i = 0; i < K; i++) {

        // Increase frequency of
        // i-th character
        if(map.has(str[i]))
```

```
                map.set(str[i], map.get(str[i])+1)
            else
                map.set(str[i], 1)
        }

        // If K distinct characters
        // exist
        if (map.size == K)
            answer++;

        // Traverse the rest of the
        // substring
        for (var i = K; i < N; i++) {

            // Increase the frequency
            // of the last character
            // of the current substring
            if(map.has(str[i]))
                map.set(str[i], map.get(str[i])+1)
            else
                map.set(str[i], 1)
            // Decrease the frequency
            // of the first character
            // of the previous substring
            if(map.has(str[i-K]))
                map.set(str[i-K], map.get(str[i-K])-1)


            // If the character is not present
            // in the current substring
            if (map.has(str[i - K]) && map.get(str[i-K])==0) {
                map.delete(str[i - K]);
            }

            // If the count of distinct
            // characters is 0
            if (map.size == K) {
                answer++;
            }
        }

        // Return the count
        return answer;
    }
```

```
// Driver code
// string str
var str = "aabcdabbcdc";
// integer K
var K = 3;
// Print the count of K length
// substrings with k distinct characters
document.write( countSubstrings(str, K) );

</script>
```

Output:

```
5
```

Time Complexity: O(N)

Auxiliary Space: O(N)

Like 13

Previous

Amazon Interview Experience for SDE-1 (Full Time-Referral) 2020

Next

Count number of substrings with exactly k distinct characters

RECOMMENDED ARTICLES

Page : 1 2 3

01 Count number of substrings with exactly k distinct characters
18, Jul 16

02 Find distinct characters in distinct substrings of a string
05, Jul 19

03 Count of substrings containing exactly K distinct vowels
21, Dec 21

04 Minimum length substring with exactly K distinct characters
05, Mar 19

05 Count M-length substrings occurring exactly K times in a string
19, Feb 21

06 Count of K length substrings containing exactly X vowels
21, Dec 21

07 Construct a string of length L such that each substring of length X has exactly Y distinct letters
30, Apr 20

08 Count number of substrings having at least K distinct characters
14, Apr 21

Article Contributed By :

codeku
@codeku

Vote for difficulty

Current difficulty : Medium

Easy Normal Medium Hard Expert

Improved By : mohit kumar 29, grand_master, rutvik_56, rrrtnx, pankajsharmagfg

Article Tags : Amazon, cpp-map, sliding-window, substring, Algorithms, Arrays, Competitive Programming, CS - Placements, Hash, Strings

Practice Tags : Amazon, sliding-window, Arrays, Hash, Strings, Algorithms

Improve Article Report Issue

Writing code in comment? Please use ide.geeksforgeeks.org, generate link and share the link here.

Load Comments

GeeksforGeeks
A-143, 9th Floor, Sovereign Corporate Tower,
Sector-136, Noida, Uttar Pradesh - 201305

feedback@geeksforgeeks.org

NodeJS