Lexicographic rank of a string

Given a string, find its rank among all its permutations sorted lexicographically. For example, rank of “abc” is 1, rank of “acb” is 2, and rank of “cba” is 6.

For simplicity, let us assume that the string does not contain any duplicated characters.

One simple solution is to initialize rank as 1, generate all permutations in lexicographic order. After generating a permutation, check if the generated permutation is same as given string, if same, then return rank, if not, then increment the rank by 1. The time complexity of this solution will be exponential in worst case. Following is an efficient solution.

Let the given string be “STRING”. In the input string, ‘S’ is the first character. There are total 6 characters and 4 of them are smaller than ‘S’. So there can be 4 * 5! smaller strings where first character is smaller than ‘S’, like following

R X X X X X
I X X X X X
N X X X X X
G X X X X X

Now let us Fix S’ and find the smaller strings staring with ‘S’.

Repeat the same process for T, rank is 4*5! + 4*4! +…

Now fix T and repeat the same process for R, rank is 4*5! + 4*4! + 3*3! +…

Now fix R and repeat the same process for I, rank is 4*5! + 4*4! + 3*3! + 1*2! +…

Now fix I and repeat the same process for N, rank is 4*5! + 4*4! + 3*3! + 1*2! + 1*1! +…

Google™ Custom Search

GeeksforGeeks

53,525 people like GeeksforGeeks.

Interview Experiences
Advanced Data Structures
Dynamic Programming
Greedy Algorithms
Backtracking
Pattern Searching
Divide & Conquer
Mathematical Algorithms
Recursion
Geometric Algorithms

open in browser PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

Now fix N and repeat the same process for G, rank is 4*5! + 4*4 + 3*3! + 1*2! + 1*1! + 0*0!

Rank = 4*5! + 4*4! + 3*3! + 1*2! + 1*1! + 0*0! = 597

Since the value of rank starts from 1, the final rank = 1 + 597 = 598

```
#include <stdio.h>
#include <string.h>

// A utility function to find factorial of n
int fact(int n)
{
    return (n <= 1)? 1 :n * fact(n-1);
}

// A utility function to count smaller characters on right
// of arr[low]
int findSmallerInRight(char* str, int low, int high)
{
    int countRight = 0, i;

    for (i = low+1; i <= high; ++i)
        if (str[i] < str[low])
            ++countRight;

    return countRight;
}

// A function to find rank of a string in all permutations
// of characters
int findRank (char* str)
{
    int len = strlen(str);
    int mul = fact(len);
    int rank = 1;
    int countRight;

    int i;
    for (i = 0; i < len; ++i)
    {
        mul /= len - i;

        // count number of chars smaller than str[i]
        // fron str[i+1] to str[len-1]
        countRight = findSmallerInRight(str, i, len-1);
```

SAMSUNG
SAMSUNG GEAR APP CHALLENGE
Are you GEARed up?
Create the best Samsung Gear App and change your universe.

Popular Posts

All permutations of a given string
Memory Layout of C Programs
Understanding “extern” keyword in C
Median of two sorted arrays
Tree traversal without recursion and without stack!
Structure Member Alignment, Padding and Data Packing
Intersection point of two Linked Lists
Lowest Common Ancestor in a BST.
Check if a binary tree is BST or not
Sorted Linked List to Balanced BST

open in browser PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

```
        rank += countRight * mul ;
    }

    return rank;
}

// Driver program to test above function
int main()
{
    char str[] = "string";
    printf ("%d", findRank(str));
    return 0;
}
```

Output

```
598
```

The time complexity of the above solution is O(n^2). We can reduce the time complexity to O(n) by creating an auxiliary array of size 256. See following code.

```
// A O(n) solution for finding rank of string
#include <stdio.h>
#include <string.h>
#define MAX_CHAR 256

// A utility function to find factorial of n
int fact(int n)
{
    return (n <= 1)? 1 :n * fact(n-1);
}

// Construct a count array where value at every index
// contains count of smaller characters in whole string
void populateAndIncreaseCount (int* count, char* str)
{
    int i;

    for( i = 0; str[i]; ++i )
        ++count[ str[i] ];

    for( i = 1; i < 256; ++i )
        count[i] += count[i-1];
}

// Removes a character ch from count[] array
```

HIGH-PERFORMANCE COMPUTING ON A UNIVERSITY BUDGET

The quest to find tailored server capabilities on a shoestring

To conduct successful university research and crunch massive amounts of data, you need the highest-performing hardware on the market. Unfortunately, the price tag of high performance can be daunting for researchers who rely on grants to fund their projects.

There are many off-the-shelf server options available, of course, and your department may even have a standardized, prescribed server.

The best bang for your buck, however, may be to configure a custom server to fit your exact high-performance computing needs.

WHAT NETWORKING INTERFACES ARE NEEDED?
☐ 1GBe ☐ InfiniBand
☐ 10GBe ☐ Unsure
☐ 40GBe

HOW MANY CPU CORES DO YOU WANT?
☐ 4 ☐ 16
☐ 8 ☐ Unsure
☐ 12

HOW MUCH STORAGE DO YOU WANT?
☐ 1TB — 1PB+
☐ Unsure

WHAT OPERATING SYSTEM ARE YOU RUNNING?
☐ Linux Variant
☐ Windows Server
☐ VMWare
☐ Unsure

Unsure of how to get the best performance out of your application? Do you need assistance answering any of these questions?

Download To See Entire Infographic

SERVERSDIRECT

open in browser PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

```
// constructed by populateAndIncreaseCount()
void updatecount (int* count, char ch)
{
    int i;
    for( i = ch; i < MAX_CHAR; ++i )
        --count[i];
}

// A function to find rank of a string in all permutations
// of characters
int findRank (char* str)
{
    int len = strlen(str);
    int mul = fact(len);
    int rank = 1, i;
    int count[MAX_CHAR] = {0};  // all elements of count[] are initial

    // Populate the count array such that count[i] contains count of
    // characters which are present in str and are smaller than i
    populateAndIncreaseCount( count, str );

    for (i = 0; i < len; ++i)
    {
        mul /= len - i;

        // count number of chars smaller than str[i]
        // fron str[i+1] to str[len-1]
        rank += count[ str[i] - 1] * mul;

        // Reduce count of characters greater than str[i]
        updatecount (count, str[i]);
    }

    return rank;
}

// Driver program to test above function
int main()
{
    char str[] = "string";
    printf ("%d", findRank(str));
    return 0;
}
```

The above programs don’t work for duplicate characters. To make them work for duplicate characters, find all the characters that are smaller (include equal this time also), do the same as

705 Subscribe

Recent Comments

Abhi You live US or India?
Google (Mountain View) interview · 15 minutes ago

Aman Hi, Why arent we checking for conditions...
Write a C program to Delete a Tree. · 55 minutes ago

kzs please provide solution for the problem...
Backtracking | Set 2 (Rat in a Maze) · 58 minutes ago

Sanjay Agarwal bool tree::Root_to_leaf_path_given_sum(tree...
Root to leaf path sum equal to a given number · 1 hour ago

GOPI GOPINATH @admin Highlight this sentence "We can easily...
Count trailing zeroes in factorial of a number · 1 hour ago

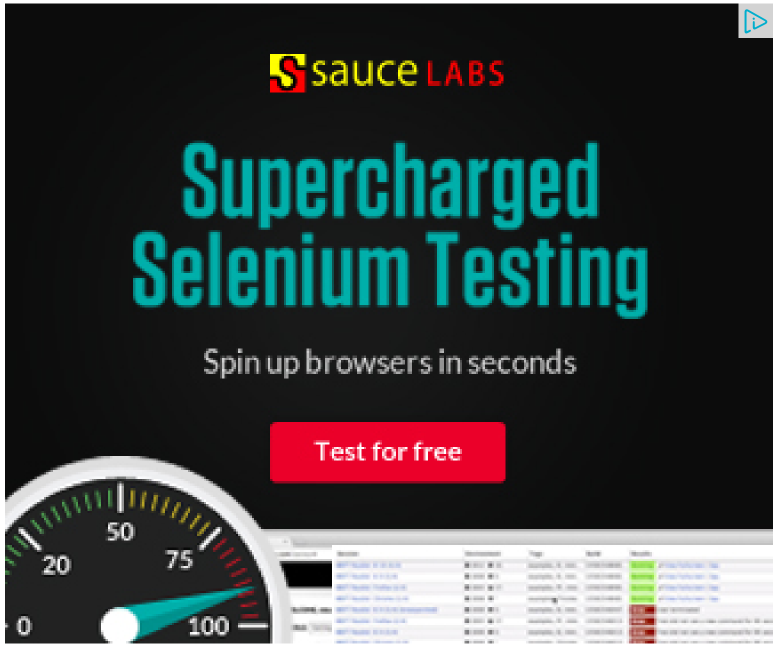
newCoder3006 If the array contains negative numbers also. We...
Find subarray with given sum · 1 hour ago

AdChoices
► String Function
► JavaScript Array
► C++ at String
AdChoices

open in browser PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

above but, this time divide the rank so formed by p! where p is the count of occurrences of the repeating character.

Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.



Related Tpoics:

- Print all possible words from phone digits
- Printing Longest Common Subsequence
- Suffix Array | Set 2 (nLogn Algorithm)
- Rearrange a string so that all same characters become d distance away
- Recursively remove all adjacent duplicates
- Find the first non-repeating character from a stream of characters
- Dynamic Programming | Set 33 (Find if a string is interleaved of two other strings)
- Remove “b” and “ac” from a given string

4 Tweet 0 0

Writing code in comment? Please use **ideone.com** and share the link here.

▶ C# String Split
▶ String Java
▶ String C
AdChoices
▶ Repair String
▶ String Java
▶ String String

@geeksforgeeks, **Some rights reserved** **Contact Us!** Powered by **WordPress** & **MooTools**, customized by geeksforgeeks team

open in browser PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com