Find the smallest window in a string containing all characters of another string

Given two strings string1 and string2, find the smallest substring in string1 containing all characters of string2 efficiently.

For Example:

Input string1: “this is a test string”
Input string2: “tist”
Output string: “t stri”

Method 1 (Brute force solution)
a) Generate all substrings of string1 (“this is a test string”)
b) For each substring, check whether the substring contains all characters of string2 (“tist”)
c) Finally print the smallest substring containing all characters of string2.

Method 2 (Efficient Solution)
1) Build a count array count[] for string 2. The count array stores counts of characters.
count['i'] = 1
count['t'] = 2
count['s'] = 1

2) Scan the string1 from left to right until we find all the characters of string2. To check if all the characters are there, use count[] built in step 1. So we have substring “this is a t” containing all characters of string2. Note that the first and last characters of the substring must be present in string2. Store the length of this substring as min_len.

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

3) Now move forward in string1 and keep adding characters to the substring “this is a t”. Whenever a character is added, check if the added character matches the left most character of substring. If matches, then add the new character to the right side of substring and remove the leftmost character and all other extra characters after left most character. After removing the extra characters, get the length of this substring and compare with min_len and update min_len accordingly.

Basically we add ‘e’ to the substring “this is a t”, then add ‘s’ and then’t’. ‘t’ matches the left most character, so remove ‘t’ and ‘h’ from the left side of the substring. So our current substring becomes “is a test”. Compare length of it with min_len and update min_len.
Again add characters to current substring “is a test”. So our string becomes “is a test str”. When we add ‘i’, we remove leftmost extra characters, so current substring becomes “t stri”. Again, compare length of it with min_len and update min_len. Finally add ‘n’ and ‘g’. Adding these characters doesn’t decrease min_len, so the smallest window remains “t stri”.

4) Return min_len.

Please write comments if you find the above algorithms incorrect, or find other ways to solve the same problem.

Source: http://geeksforgeeks.org/forum/topic/find-smallest-substring-containing-all-characters-of-a-given-word

Geometric Algorithms

Popular Posts

All permutations of a given string

Memory Layout of C Programs

Related Tpoics:

- 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
- Dynamic Programming | Set 29 (Longest Common Substring)

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

69 Comments GeeksforGeeks

Sort by Newest

AlienOnEarth • 3 days ago

Good explanation with source code:
http://leetcode.com/2010/11/fi...

Time complexity: O(m+n)

Prasenjit Mondal • 20 days ago

Please check the code here:
https://ideone.com/8lzJcC

I have considered all the cases.

The approach mentioned above does not work because the initial window may (which are in target string). Those extra characters can be removed after findi

• Reply • Share ›

PB • a month ago

C# implementation with comments for explanation. Works perfectly.

/*Given two strings string1 and string2, find the smallest substring in string1 c count) of string2 efficiently.*/

public static void FindSmallestSbstrContainingStr2(string str1, string str2)

{

//first create a dictionary of the count of chars in str 2

Dictionary<char, int=""> charCount2 = new Dictionary<char, int="">();

foreach (char c in str2)

{

if (!charCount2.ContainsKey(c))

charCount2[c] = 0;

see more

• Reply • Share ›

695

Subscribe

Recent Comments

affiszerv Your example has two 4s on row 3, that's why it...

Backtracking | Set 7 (Sudoku) · 20 minutes ago

RVM Can someone please elaborate this Qs from above...

Flipkart Interview | Set 6 · 40 minutes ago

Vishal Gupta I talked about as an Interviewer in general,...

Software Engineering Lab, Samsung Interview | Set 2 · 40 minutes ago

@meya Working solution for question 2 of 4f2f round....

Amazon Interview | Set 53 (For SDE-1) · 1 hour ago

sandeep void rearrange(struct node *head) {...

Given a linked list, reverse alternate nodes and append at the end · 2 hours ago

Neha I think that is what it should return as, in...

Find depth of the deepest odd level leaf node · 2 hours ago

raim_dtu • 4 months ago

```
#include<iostream>
#include<cstdio>
#include<cstdlib>
#include<climits>
#include<string>
using namespace std;
int main()
{
string str1,str2;
int count1[256]={0}, count2[256]={0},curr=0,beg=0,minimum=INT_MAX,no,no
getline(cin,str1);
getline(cin,str2);
while(str1[curr])
{
count1[str1[curr]]++;
curr++;
}
no=curr;
```

see more

Reply • Share ›

Anonymous • 4 months ago

What and how do you detect "extra characters" ?

Reply • Share ›

hero • 5 months ago

This is a good solution. It is in O(n) time. For those who can't appreciate its gc
contains all we need but it may not be the smallest substring so reduce it to th
different set by moving the second pointer to the back and reduce it ... so on s

end.

One similar problem: find all possible sets of consecutive integers whose sum

 • Reply • Share ›

Harjit Singh • 7 months ago

I think this does not work. Consider the following case string1="abdcba" and s
"bdcba" as the solution, but "cba" is the actual solution. Any comment plz?

 • Reply • Share ›

hero ➚ Harjit Singh • 5 months ago

you are simply wrong. Vinod gives you the correct answer

 • Reply • Share ›

Vinod ➚ Harjit Singh • 6 months ago

Please read : "remove the leftmost character and all other extra charac

1  • Reply • Share ›

hero ➚ Vinod • 5 months ago

I can prove your algorithm is O(n) we have two pointers each m
2n QED

 • Reply • Share ›

dheeraj ➚ Vinod • 5 months ago

can you please elaborate the process of removal from leftmost
In this statement:
Basically we add ‘e’ to the substring “this is a t”, then add ‘s’ an
character, so remove ‘t’ and ‘h’ from the left side of the substrin
“is a test”. Compare length of it with min_len and update min_le
Again add characters to current substring “is a test”.
.....
I think after removal of t h from "this is a test" it should be "is is

I don't understand how you are removing i,s after t,h .
please explain.....

• Reply • Share ›

Harshit Sharan • 8 months ago

A simple approach - http://coding-interview-archiv...

• Reply • Share ›

asr • 8 months ago

One possible solution:

Starting from index 0 find the first letter that is in string 2

Then find the smallest window for which our condition is satisfied(i.e window h

Now when we move to the next index we have a smaller window to check.

for eg. string1="abecaadaeb" and string2="aadb"

we have window [0,6].

Now for starting index 1, we'll only need to check till index 6.

Similarly we continue for all possible indices and update minimum window siz

Please tell me if this algorithm seems correct.

• Reply • Share ›

TheBondSuperman • 8 months ago

What do you mean by "extra chracters" in "remove the leftmost character and
most character" ?
If I follow your algorithm right, then there are going to be a lot of other substring
basis you remove "extra characters"

3 ^ | v • Reply • Share ›

Marsha Donna ➦ TheBondSuperman • 3 months ago

i think extra chars means all chars int string1 upto the next character w

^ | v • Reply • Share ›

raushanraj • 9 months ago

```
 /* Paste your code here (You may delete these lines if not writing c
#include<stdio.h>
int* createtable(char* s,int k,int j)
{
int* table=calloc(256,sizeof(int));
int i;
for(i=k;i<j;i++)
{
    table[s[i]]++;
}
return table;
}
int getMinimumWindow(char* s1,char* s2)
{
int* s2table=createtable(s2,0,strlen(s2));

int start=0;
int end=0;
```

see more

^ | v • Reply • Share ›

Avinash Abhi • 9 months ago

there is a small issue here,

once we get a window with all characters of string2.

if we add, a character at the right which matches the leftmost character of the character and the extra characters,
till here it is fine,
at this position we are comparing the minlength.
in this case, when comparing the minlength, suppose at the leftmost index the between the window, this wont be the min length,
here we will need to remove the leftmost character again, if it is repeated som characters again.

till we find the character at leftmost index that isn&#039t repeated in between.
and now the length can give the min length if it is min length.

3 ^ | v • Reply • Share ›

Avinash Abhi • 9 months ago

there is a small issue here,

once we get a window in string2 with all characters of string1.
if we add, a character at the right which matches the leftmost character of the character and the extra characters,
till here it is fine,
at this position we are comparing the minlength.
in this case, when comparing the minlength, suppose at the leftmost index the between the window, this wont be the min length,
here we will need to remove the leftmost character again, if it is repeated som characters again.

till we find the character at leftmost index that isn&#039t repeated in between.
and now the length can give the min length if it is min length.

^ | v • Reply • Share ›

darkpassenger • 9 months ago

there is bug in the algo .....

for ex:string:afghijbzzzcyydba and secondstring:abcd....
o/p :10
correct o/p:6

• Reply • Share ›

jeswanth darkpassenger • 6 months ago

http://coding-interview-archiv...

This code based on the algorithm above works fine

• Reply • Share ›

aditya gupta • 10 months ago

[sourcecode language="JAVA"]
import java.util.*;

class smallest_window1{
public static void main(String args[]){
String word = "vngk";

String str = "geakstuvpnkhg";

int i,n,temp=0,length,min_index = -1,min=-1,mins=-1,mine=-1;
boolean found = false;

word = word.toLowerCase();
str = str.toLowerCase();

length = word.length();

HashMap<Character,Integer> map = new HashMap<Character,Integer>();

for(i=0;i<word.length();i++){

see more

• Reply • Share ›

sairam • 10 months ago

sairam:may be it is easy

```
/* Paste your code here (You may delete these lines if not writing code) */
[/#include
#include
int main()
{
int i,j,k,l=0,n,m,a,x,y,b=8888,flag=0,p=0,s,t,o=0;
char s1[]="sai rams rihg";
char s2[]="sir";
char s3[20],s4[20];
m=strlen(s1);
n=strlen(s2);
for(j=0;j<n;j++)
{
```

see more

^ | v • Reply • Share ›

anonymous • 11 months ago

i think above algorithm fails as in case
string1:"geakstuvpnkhg"
string2:"vngk"
it should be 6
but i think above algorithm will print 10..

admin or anyone please clarify it if i am wrong..

Reply • Share ›

aygul • a year ago

1) Build a boolean count array count[] of string 2...

This is wrong. you sould build an int count array.

Reply • Share ›

GeeksforGeeks ➚ aygul • a year ago

@aygul: Thanks for pointing this out. We have corrected the statement

Reply • Share ›

nikhil • a year ago

simple and best solution with implementation....
http://leetcode.com/2010/11/fi...

Reply • Share ›

Hanish ➚ nikhil • a year ago

(y)

Reply • Share ›

rahul sundar • a year ago

This algorithm would not find min length for 'this is a tistt'. So here as per our a
t' which contain all characters and updates min length as 11. Then it continues
and substring becomes 'is is a tist' and min still as same and completes scan
be sub string 'tis' length 3.

```
/* Paste your code here (You may delete these lines if not writing c
```

Reply • Share ›

suresh • a year ago

```
/* Paste your code here (You may delete these lines if not writing c
```

#include
#include
#include
#include
#include

bool minWindow(const char* S, const char *T,
int &minWindowBegin, int &minWindowEnd)
{
int sLen = strlen(S);
int tLen = strlen(T);
int needToFind[256] = {0};
int hasFound[256] = {0};
int minWindowLen = INT_MAX;
int count = 0;

for (int i = 0; i < tLen; i++)

see more

• Reply • Share ›

huha • 2 years ago

i think the above solution is not correct..atleast its not clear enough.we need tc
string s also in which we store the count we have encountered so far.

```
// Returns false if no valid window is found. Else returns
// true and updates minWindowBegin and minWindowEnd with the
// starting and ending position of the minimum window.
bool minWindow(const char* S, const char *T,
               int &minWindowBegin, int &minWindowEnd) {
  int sLen = strlen(S);
  int tLen = strlen(T);
```

```
int tLen = strlen(T);
int needToFind[256] = {0};

for (int i = 0; i < tLen; i++)
  needToFind[T[i]]++;

int hasFound[256] = {0};
int minWindowLen = INT_MAX;
int count = 0;
```

see more

• Reply • Share ›

hmmm ➦ huha • 2 years ago

source:
http://www.leetcode.com/2010/1...

decent explanation...

• Reply • Share ›

ddarbari • 2 years ago

[sourcecode language="JAVA"]
package StringPattern;

import java.util.HashMap;
import java.util.Iterator;
import java.util.Map;
public class SmallestWindow
{
String text;
String pattern;
Map<Character, Integer> pm;
Map<Character, Integer> tm;

```
SmallestWindow(String t,String p )
{
text=t;
pattern=p;
pm=new HashMap<Character, Integer>();
tm=new HashMap<Character, Integer>();
```

see more

Reply • Share ›

Shouri • 2 years ago

Let's assume that A is the string in which we need to identify the string B's cha
character of B. Scan through the string A and identify all the characters of B ar

Now, start with the first number in string A and find all the sub strings containir
minimum sliding window approach (from each substring which starts with a cl
identified by the number) and find the sub string containing all the characters c

Reply • Share ›

Rohit Saraf • 3 years ago

Another O(n) solution that is cleaner to state (than the one in solution)

----------*----------**(second pointer at end)
(first pointer * such that all elements in string2 are available between the two p
(second pointer ** at end)
Maintain counts for each element in string2 while moving * towards left.

Now move ** to right if possible. i.e. the counts permit. If not allowed, store the
** can be further moved left. Keep doing till * hits the start of the array.

I don't know if the same solution is mentioned in the post but that is way too cc

Reply • Share ›

Rohit Saraf • 3 years ago

A beautiful in place O(n) solution which changes the existing strings.

In O(n) you can change the existing strings to contain.

ith character of string2 now contains the location of starting of ith element in th
which is changed such that
all instances of ith character appear before the jth character for any j!=i. (can b
So string1 now contains the index of ith character in the earlier string1.

Now get the maximum of all differences of indices in the string2. Find the char
Increment it's index in string2. Increase it to next index, and update the differer
this does the job. Think why?)

Do this until one of the character region ends. The algorithm is correct. But I d

 • Reply • Share ›

sandygupta • 3 years ago

How will it work for this :
String1 = "eetsst"
String2 = "test"

This will output 6
but the answer is 5

 • Reply • Share ›

Manish Kumar • 3 years ago

Hi Pals,

I have slightly modified the method "subStringSmallest": Please plan to use it

//----------

```
char* subStringSmallest(char* testStr, char* cSet)
```

```
{
char* subString = NULL;
int iSzSet = strlen(cSet) + 1;
int iSzString = strlen(testStr)+ 1;
char* cSetBackUp = new char[iSzSet];
memcpy((void*)cSetBackUp, (void*)cSet, iSzSet);

int iStartIndx = -1;
int iEndIndx = -1;
int iIndexStartNext = -1;

std::vector subStrVec;
int index = 0;
```

see more

• Reply • Share ›

Manish Kumar • 3 years ago

Hey guys please check this out:

```
//-----------------------------------------------------------------------------------------------
bool IsInSet(char ch, char* cSet)
{
char* cSetptr = cSet;
int index = 0;
while (*(cSet+ index) != '\0')
{
if(ch == *(cSet+ index))
{
return true;
}
++index;
}
```

return false;
}

void removeChar(char ch, char* cSet)

see more

• Reply • Share ›

Bhushan • 3 years ago

Hi,

Can you tell me the answer for the following input:

String input1 = "thisisateststring";
String input2 = "ia";

I guess it fails in this case. The initial window which it finds is 'isisa' and thats v
initial window itself is wrong, it should be 'isa'. Am I missing something?

• Reply • Share ›

candis ➚ Bhushan • 3 years ago

see once the string matches....the process is repeated again, this time
leftmosr+1.....v first check of the characters not in the array...skip them
process....but this tym whem match occurs we compare it with the ma

• Reply • Share ›

sourabhjakhar • 3 years ago

this explanation is wrong
consider the string Tecvtsvice
and substring tsvi
it will not give the correct the correct answer

• Reply • Share ›

pranay • 3 years ago

Hi, in your example , after we add 'e' we make it visited , then we add 's' , we r
add 't' , it matches with the left most character of the window , so we add it an
it is not in the given pattern , so the string now is "is is a test" .Now , why is the
neither has it been visited nor its not there in the given pattern("tist").
Thanks.

 • Reply • Share ›

Shubham Maheshwari ➚ pranay • 3 years ago

same doubt ... the explanation given above results in a string of length

 • Reply • Share ›

Dreamer ➚ pranay • 3 years ago

I am also having same doubt??

1  • Reply • Share ›

Algoseekar • 3 years ago

code for 1st program https://ideone.com/pifya..aftr... generating all combinatio
which has all the character of 2nd string check out the link

 • Reply • Share ›

raja • 3 years ago

Very good solution, easily understandable, nicely explained. Thank you.

 • Reply • Share ›

vinay ➚ raja • 3 years ago

This solution is wrong dude.

Check with the following input:

String1: isisisisaisisisis
string2: ia

 • Reply • Share ›

coder ↗ vinay • 2 years ago

It works for this case as well.

```
/* Paste your code here (You may delete these lines if
```

• Reply • Share ›

learner ↗ coder • 2 years ago

Can you please explain how this works?

• Reply • Share ›

learner ↗ coder • 2 years ago

Can you please explain how this works step by step?

• Reply • Share ›

Load more comments