

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

Given two strings 'X' and 'Y', find the length of the longest common substring. For example, if the given strings are "GeeksforGeeks" and "GeeksQuiz", the output should be 5 as longest common substring is "Geeks"

Let m and n be the lengths of first and second strings respectively.

A **simple solution** is to one by one consider all substrings of first string and for every substring check if it is a substring in second string. Keep track of the maximum length substring. There will be  $O(m^2)$  substrings and we can find whether a string is substring on another string in  $O(n)$  time (See [this](#)). So overall time complexity of this method would be  $O(n * m^2)$

**Dynamic Programming** can be used to find the longest common substring in  $O(m*n)$  time. The idea is to find length of the longest common suffix for all substrings of both strings and store these lengths in a table.

The longest common suffix has following optimal substructure property

$$\text{LCSuff}(X, Y, m, n) = \begin{cases} \text{LCSuff}(X, Y, m-1, n-1) + 1 & \text{if } X[m-1] = Y[n-1] \\ 0 & \text{Otherwise (if } X[m-1] \neq Y[n-1]) \end{cases}$$

The maximum length Longest Common Suffix is the longest common substring.

$$\text{LCSubStr}(X, Y, m, n) = \text{Max}(\text{LCSuff}(X, Y, i, j)) \text{ where } 1 \leq i \leq m \text{ and } 1 \leq j \leq n$$

Following is C++ implementation of the above solution.

```
/* Dynamic Programming solution to find length of the longest common s
```

Google™ Custom Search



GeeksforGeeks



52,731 people like [GeeksforGeeks](#).



[Interview Experiences](#)

[Advanced Data Structures](#)

[Dynamic Programming](#)

[Greedy Algorithms](#)

[Backtracking](#)

[Pattern Searching](#)

[Divide & Conquer](#)

[Mathematical Algorithms](#)

[Recursion](#)

[Geometric Algorithms](#)

```

#include<iostream>
#include<string.h>
using namespace std;

// A utility function to find maximum of two integers
int max(int a, int b)
{   return (a > b)? a : b; }

/* Returns length of longest common substring of X[0..m-1] and Y[0..n-1]
int LCSuffStr(char *X, char *Y, int m, int n)
{
    // Create a table to store lengths of longest common suffixes of
    // substrings.  Notethat LCSuff[i][j] contains length of longest
    // common suffix of X[0..i-1] and Y[0..j-1]. The first row and
    // first column entries have no logical meaning, they are used only
    // for simplicity of program
    int LCSuff[m+1][n+1];
    int result = 0; // To store length of the longest common substring

    /* Following steps build LCSuff[m+1][n+1] in bottom up fashion. */
    for (int i=0; i<=m; i++)
    {
        for (int j=0; j<=n; j++)
        {
            if (i == 0 || j == 0)
                LCSuff[i][j] = 0;

            else if (X[i-1] == Y[j-1])
            {
                LCSuff[i][j] = LCSuff[i-1][j-1] + 1;
                result = max(result, LCSuff[i][j]);
            }
            else LCSuff[i][j] = 0;
        }
    }
    return result;
}

/* Driver program to test above function */
int main()
{
    char X[] = "OldSite:GeeksforGeeks.org";
    char Y[] = "NewSite:GeeksQuiz.com";

    int m = strlen(X);
    int n = strlen(Y);

```

**Visual Studio Extension**  
**For Faster, Smarter Coding**  
Telerik JustCode™

System.Drawing.Color  
public float GetSaturation()  
{  
 float r = (float)this.R / 255f;  
 float g = (float)this.G / 255f;  
 float b = (float)this.B / 255f;  
 float single = 0f;  
 float single1 = r;  
 float single2 = r;  
}

Autos  
Name Value Type

**Download free trial**

Telerik®

## 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

```
cout << "Length of Longest Common Substring is " << LCSubStr(X, Y,  
return 0;  
}
```

Output:

Length of Longest Common Substring is 10

Time Complexity:  $O(m*n)$

Auxiliary Space:  $O(m*n)$

**References:** [http://en.wikipedia.org/wiki/Longest\\_common\\_substring\\_problem](http://en.wikipedia.org/wiki/Longest_common_substring_problem)

The longest substring can also be solved in  $O(n+m)$  time using Suffix Tree. We will be covering Suffix Tree based solution in a separate post.

**Exercise:** The above solution prints only length of the longest common substring. Extend the solution to print the substring also.

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



Better Than Hadoop.

HPCC Systems is Big Data Processing and Analytics  
*Open Source. Proven. Trusted.*

 LexisNexis® [Learn More](#) 

**HIGHLY RECOMMENDED**

# Download & Update All Your Windows Drivers!

Windows 8 / 7 / Vista / XP

Free Scan

Boost Windows Performance



**START  
DOWNLOAD**



## Related Topics:

- [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](#)
- [Write your own atoi\(\)](#)



32



Tweet

3



3

Writing code in comment? Please use [ideone.com](#) and share the link here.

36 Comments

GeeksforGeeks

Sort by Newest ▼



Join the discussion...



**AlienOnEarth** · 5 days ago

@GeeksforGeeks:

I think the recurrence relation is not written properly. It says max of Max(LCSul only 1 parameter for Max(a,b)

^ | v · Reply · Share ›



**GeeksforGeeks** Mod → AlienOnEarth · 5 days ago

Please take a look at the link and know from 4 to 11 is wrong



695



Subscribe

## Recent Comments

[affiszerv](#) Your example has two 4s on row 3, that's why it...

[Backtracking | Set 7 \(Sudoku\)](#) · 13 minutes ago

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

[Flipkart Interview | Set 6](#) · 33 minutes ago

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

[Software Engineering Lab, Samsung Interview | Set 2](#) · 33 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

 Please take a closer look. Here i and j vary from 1 to n. It is maximum 1 know if this clarifies.

^ | v • Reply • Share ›



**Anjaneya Alluri** • 21 days ago

Hi, Can anyone provide the Memoization version of this in Java

^ | v • Reply • Share ›



**sawan** • 2 months ago

The Problem can be solved with  $O(m \cdot n)$  time and  $O(m)$  space complexity. In the below program the result array is basically result[2], to store the start an

```
void LCS(char str1[],char str2[], int result[]) {  
  
    int len1=strlen(str1),len2=strlen(str2);  
  
    char temp[2][len2];  
  
    int z=0;  
  
    for(int i=0;i<len1;i++){ for(int="" j="0;j<len2;j++){ if(str1[i]==str2[j]){ if(i=="0'  
temp[i%2][j]="temp[(i%2+1)%2][j-1]+1;" if(temp[i%2][j]=="">z){  
  
z=temp[i%2][j];  
  
result[0]=i-z+1;  
  
result[1]=i;  
  
,
```

see more

^ | v • Reply • Share ›



**Ronak Hingar** • 3 months ago


Here is the code in java

AdChoices 

► [Graph C++](#)

► [Java to C++](#)


► [Graph Java](#)

AdChoices 

► [Substring](#)

► [C++ Example](#)

► [Compare C++ Code](#)

AdChoices 

► [C++ Program](#)

► [Programming C++](#)

► [STD String C++](#)



Here is the code in Java

```
public class LCS {  
  
    public String LongestCommonSubsequence(String x, String y, int i, int j) {  
  
        String s = "";  
  
        if (i == x.length() || j == y.length()) {  
  
            return "";  
  
        }  
  
        if (x.charAt(i) == y.charAt(j)) {  
  
            s += x.charAt(i) + LongestCommonSubsequence(x, y, i + 1, j + 1);  
  
            return s;  
  
        } else {
```

[see more](#)

1 ^ | v • Reply • Share ›



**Anjaneya Alluri** → Ronak Hingar • 21 days ago

isn't creating so many strings in the process of finding substring a bad  
we don't create so many strings.

^ | v • Reply • Share ›



**prashant** • 4 months ago

```
int fun(char st1[],char st2[],int low1,int low2)  
  
{
```

```
int max=0;
```

```

if((low1>=strlen(st1))||(low2>=strlen(st2)))

return max;

for(int i=low1;i<strlen(st1);i++) {="" int="" k="search(st1[i],st2,low2);" if(k!="-1)'
if(p="">max)

max=p;

}

}

return max;

}

```

^ | v • Reply • Share ›



**jeremy** • 4 months ago

excellent job!

^ | v • Reply • Share ›



**Guest** • 4 months ago

/\*find the length of Longest Common Substring \*/

```
#include<iostream>
```

```
#include<string>
```

```
using namespace std;
```

```
string s="OldSite:GeeksforGeeks.org";
```

```
string str="NewSite:GeeksQuiz.com";
```

```
int main()

{

int i,index=0,max_size=0,count=0,l,k;

for( i=0; i<s.length(); i++)="" {="" for(int="" j="0;" j<str.length();="" j++)="" {="" if
k="j;" while(s[l]=="str[k]" &&="" l="">=0 && k>=0)
```

[see more](#)

^ | v • Reply • Share ›



**Guest** • 4 months ago

```
/*find the length of Longest Common Substring */
#include<iostream>
#include<string>
using namespace std;

string s="OldSite:GeeksforGeeks.org";
string str="NewSite:GeeksQuiz.com";

int main()
{
int i,index=0,max_size=0,count=0,l,k;
for( i=0; i<s.length(); i++)="" {="" for(int="" j="0;" j<str.length();="" j++)="" {="" if
k="j;" while(s[l]=="str[k]" &&="" l="">=0 && k>=0)
{
count+=1;
l--;
k--;
}
if(count>max_size)
```

[see more](#)



^ | v · Reply · Share ›



**eragon** · 5 months ago

fails on input harry sally answer is 2 for ay. Code gives 1

^ | v · Reply · Share ›



**Guest** → eragon · 2 months ago

ay is a subsequence.. we are looking for a substring

^ | v · Reply · Share ›



**jaaaames** → eragon · 2 months ago

the code is for longest contiguous string

^ | v · Reply · Share ›



**wgpshashank** · 5 months ago

Please correct wht you have written There will be  $O(m^2)$  substrings actually t  
for string of length n.

^ | v · Reply · Share ›



**Kartik** → wgpshashank · 5 months ago

Nothing seems to be wrong to me. For a string of length m, there are a

^ | v · Reply · Share ›



**dharmendra** · 8 months ago

Nice explanation, I really liked it.

There is another variant of this question where the interviewer asks you to find  
subsequence instead of substring.

I wrote an article for the same, if anybody came here searching for the topic. [h](#)

^ | v · Reply · Share ›



**AlienOnEarth** → dharmendra · 5 days ago

LCS is already present on GeeksforGeeks

^ | v · Reply · Share ›



**learner** · 9 months ago

In case someone is a java guy :)

```
[sourcecode language="JAVA"]
public class LCM {
    public static void main(String[] args) {
        System.out.println(lcm(args[0],args[1]));
    }

    static String lcm(String s1, String s2){
        int result = 0;
        int[][] lcmdp = new int[s1.length()+1][s2.length()+1];
        int index=0;

        for(int i=0;i<s1.length();i++){
            for(int j=0;j<s2.length();j++){
                if(s1.charAt(i) == s2.charAt(j)){
                    lcmdp[i+1][j+1] = lcmdp[i][j] + 1;
                    //result = Math.max(lcmdp[i+1][j+1], result);
                    if(result < lcmdp[i+1][j+1]){
```

[see more](#)

^ | v · Reply · Share ›



**Vikas Gupta** · 9 months ago

Munish first take two string and find the LCS of both string and let name it as l  
find LCS of this string and Max\_LCS and update the Max\_LCS. This way proc  
Max\_LCS will be the answer..Hope u got it..:)

^ | v · Reply · Share ›

1 | v · Reply · Share ›



**Munish Singla** · 10 months ago

When more then two strings are given, how to find the longest common subst

1 ^ | v · Reply · Share ›



**gargsanjay** · 10 months ago

we can do this through hash

create hashtable of size (small string \* chars)

store indexes of occurrence of every character in first string

now pick elements of 2nd string one by one and try to match them with ev  
hash table.

$O(m*n)$

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

^ | v · Reply · Share ›



**Vignesh Miriyala** · 10 months ago

Santosh Kumar the string is not sorted to do binary search ;).

^ | v · Reply · Share ›



**Santosh Kumar** · 10 months ago

```
for(i=0;i<str2.length;i++){
```

```
int index=searchBinary(str1, str2[i]);
```

```
while(str1[index++]=str[j++]).
```

```
{.
```

```
result+ = str1[index];
```

```
}.
```

```
if(result.length()>finalLCM.length()).
```

```
finalLCM=result;
```

```
}
```

time complexity:  $O(n \cdot \text{lcm} \cdot \log n)$ .

^ | v • Reply • Share ›



**firoz** • 10 months ago

```
#include<stdio.h>

void main()
{
    int i,j,k,n=0,m,t;
    char ar[50],arr[50];
    clrscr();
    printf("enter first string:\t");
    gets(ar);
    printf("\nenter Second String:\t");
    gets(arr);
    for(i=0;ar[i]!='&#92;&#48';i++)
    {
        for(j=0,k=i;arr[j]!='&#92;&#48';j++)
        {
            if(ar[k]==arr[j])
            {
                k++; t=k;
            }
        }
    }
}
```

see more

^ | v • Reply • Share ›



**tushar** → firoz • 10 months ago

Nice one , algo do not uses any extra memory and working perfectly.

^ | v • Reply • Share ›



**madhur** → tushar • 7 months ago



it is not working for all cases.

^ | v · Reply · Share ›



**Born Actor** · 11 months ago

//if a[i][j]=0 when s1[i]!=s2[j], we get the largest contiguous subse  
// if we remove the commented part and comment the lines from 52 to 54

```
#include <iostream>
#include<string>
#include<sstream>
#include<iomanip>
# include <stdio.h>
# include <math.h>
#include <vector>
#include <stdlib.h>
using namespace std;
int m;
int n;
int a[50][50];
int max(int a,int b, int c);
int function();
```

[see more](#)

^ | v · Reply · Share ›



**anonymous** · 11 months ago

Code for printing the substring as well

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

```

void lcsb(char x[],char y[],int m,int n)
{

    int i,j;
    int table[m+1][n+1];

    int max=0,ind=0;
    for(i=0;i<=m;i++)
        for(j=0;j<=n;j++)
        {
            if(i==0 || j==0)
                table[i][j]=0;

```

see more

2 ^ | v • Reply • Share ›



**Balpreet Singh** → anonymous • 8 months ago

The code for printing string is not correct, it will not print complete string

Correct code is

```

printf("maximum LCSUBSTRING IS :");
for(i=ind-max;i<ind;i++) printf("%c",x[i]);="" printf("\n");="">

```

^ | v • Reply • Share ›



**PG** • a year ago

You guys are doing awesome work. It really helps a lot. Keep up the good work

```

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

```

^ | v • Reply • Share ›



**Waqas Khan** · a year ago

can u please tell me code in c.

1 ^ | v · Reply · Share ›



**caijinlong** · a year ago

```
#include <iostream>
#include <vector>
using namespace std;

int location = 0;

int LCSubstring(char *x, char *y, const int m,const int n)
{
    //int LCSbuff[m + 1][n + 1];
    //int **LCSbuff = new int*[m + 1];
    vector<vector<int> > LCSbuff;
    //int* a[4];
    LCSbuff.resize(m + 1);

    int result = 0;
    for (int i = 0; i <= m; i++)
    {
```

[see more](#)

^ | v · Reply · Share ›



**caijinlong** · a year ago

```
#include
#include
using namespace std;
```

```

int location = 0;

int LCSubstring(char *x, char *y, const int m,const int n)
{
    //int LCSbuff[m + 1][n + 1];
    //int **LCSbuff = new int*[m + 1];
    vector<vector> LCSbuff;
    //int* a[4];
    LCSbuff.resize(m + 1);

    int result = 0;
    for (int i = 0; i <= m; i++)
    {
        LCSbuff[i].resize(n + 1);
        //LCSbuff[i][0] = new int [n + 1];
    }
}

```

[see more](#)

^ | v • Reply • Share ›



**Subhajit** • a year ago

Solution which prints the substring also:

```

#include<iostream>
#include<string.h>

using namespace std;
int max(int a,int b)
{
    if(a>b)
        return a;
    else
        return b;
}

```



```

    }
    return 0;
}

int main()
{
    char *s="forGeeksGeeks";
    char *s1="QuizGeeks";

```

[see more](#)

^ | v • Reply • Share ›



**caijinlong** → Subhajit • a year ago

```

#include
#include
using namespace std;

int location = 0;

int LCSubstring(char *x, char *y, const int m,const int n)
{
    //int LCSbuff[m + 1][n + 1];
    //int **LCSbuff = new int*[m + 1];
    vector<vector> LCSbuff;
    //int* a[4];
    LCSbuff.resize(m + 1);

    int result = 0;
    for (int i = 0; i <= m; i++)
    {
        LCSbuff[i].resize(n + 1);
        //LCSbuff[i][0] = new int [n + 1];

```

[see more](#)

^ | v • Reply • Share ›



**Manish Untwal** · a year ago

Initialize the table.

^ | v · Reply · Share ›



Subscribe



Add Disqus to your site

@geeksforgeeks, **Some rights reserved**

**Contact Us!**

Powered by **WordPress** & **MooTools**, customized by geeksforgeeks team