Longest Palindromic Substring | Set 1

Given a string, find the longest substring which is palindrome. For example, if the given string is “forgeeksskeegfor”, the output should be “geeksskeeg”.

Method 1 (Brute Force)
The simple approach is to check each substring whether the substring is a palindrome or not. We can run three loops, the outer two loops pick all substrings one by one by fixing the corner characters, the inner loop checks whether the picked substring is palindrome or not.

Time complexity: O (n^3)
Auxiliary complexity: O (1)

Method 2 (Dynamic Programming)
The time complexity can be reduced by storing results of subproblems. The idea is similar to this post. We maintain a boolean table[n][n] that is filled in bottom up manner. The value of table[i][j] is true, if the substring is palindrome, otherwise false. To calculate table[i][j], we first check the value of table[i+1][j-1], if the value is true and str[i] is same as str[j], then we make table[i][j] true. Otherwise, the value of table[i][j] is made false.

```
// A dynamic programming solution for longest palindr.
// This code is adopted from following link
// http://www.leetcode.com/2011/11/longest-palindromic-substring-part-

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

// A utility function to print a substring str[low..high]
void printSubStr( char* str, int low, int high )
{
    for( int i = low; i <= high; ++i )
```


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

```
        printf("%c", str[i]);
}
```

```
// This function prints the longest palindrome substring of str[].
// It also returns the length of the longest palindrome
int longestPalSubstr( char *str )
{
    int n = strlen( str ); // get length of input string

    // table[i][j] will be false if substring str[i..j] is not palindr
    // Else table[i][j] will be true
    bool table[n][n];
    memset( table, 0, sizeof( table ) );

    // All substrings of length 1 are palindromes
    int maxLength = 1;
    for( int i = 0; i < n; ++i )
        table[i][i] = true;

    // check for sub-string of length 2.
    int start = 0;
    for( int i = 0; i < n-1; ++i )
    {
        if( str[i] == str[i+1] )
        {
            table[i][i+1] = true;
            start = i;
            maxLength = 2;
        }
    }

    // Check for lengths greater than 2. k is length of substring
    for( int k = 3; k <= n; ++k )
    {
        // Fix the starting index
        for( int i = 0; i < n - k + 1 ; ++i )
        {
            // Get the ending index of substring from starting index i
            int j = i + k - 1;

            // checking for sub-string from ith index to jth index iff
            // to str[j-1] is a palindrome
            if( table[i+1][j-1] && str[i] == str[j] )
            {
                table[i][j] = true;

                if( k > maxLength )
```


Geometric Algorithms

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

```
                {
                    start = i;
                    maxLength = k;
                }
            }
        }
    }

    printf("Longest palindrome substring is: ");
    printSubStr( str, start, start + maxLength - 1 );

    return maxLength; // return length of LPS
}

// Driver program to test above functions
int main()
{
    char str[] = "forgeeksskeegfor";
    printf("\nLength is: %d\n", longestPalSubstr( str ) );
    return 0;
}
```

Output:

```
Longest palindrome substring is: geeksskeeg
Length is: 10
```

Time complexity: O (n^2)
Auxiliary Space: O (n^2)

We will soon be adding more optimized methods as separate posts.

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

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.

30 Comments GeeksforGeeks

Sort by Newest

Recent Comments

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

Backtracking | Set 7 (Sudoku) · 16 minutes ago

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

Flipkart Interview | Set 6 · 35 minutes ago

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

Software Engineering Lab, Samsung Interview | Set 2 · 36 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

Join the discussion…

hours ago

Lohith Ravi • 11 days ago

This is also DP is it not ? please correct me if im wrong

```
main()

{

for(int i=0;i<n;i++) {="" for(int="" j="i+1;j&amp;lt;n;j++)" {="" if(j-i+1="">> max){

if(isPalindrome(i,j,array))

{

max = j-i+1;

resultLowIndex=i;

resultHighIndex=j

}

}
```

see more

• Reply • Share ›

sukisukimo • 2 months ago

```
/*

* Program to find the longest palindrome
* The algo time complexity is O(n^2) and space complexity is O(1)

*/
```

```
static String getBigPalindrome(String str)

{

int i = 0, j = 0, tempi = 0;

int maxLen = 0, currentLen = 0;

String maxpalin = null;

String currentpalin = null;

char[] carr = str.toCharArray();
```

see more

• Reply • Share ›

neelabhsingh • 7 months ago

Very Important CHECK this condition

```
for( int k = 3; k <= n; ++k )
{
// Fix the starting index
for( int i = 0; i < n - k + 1 ; ++i )
{
// Get the ending index of substring from starting index i and length k
int j = i + k - 1;
// checking for sub-string from ith index to jth index iff str[i+1]
// to str[j-1] is a palindrome
if( table[i+1][j-1] && str[i] == str[j] )
{
table[i][j] = true;
if( k > maxLength )
```

```
{
start = i;
maxLength = k;
```

see more

• Reply • Share ›

baba ➦ neelabhsingh • 5 months ago

table[i][i] have all been set to true.

• Reply • Share ›

Sumit Monga • 7 months ago

To find whether a substring starting at index 'i' and ending at index 'j' is a palinc regarding 'i+1' and 'j-1'. So the solution to the problem is building the table star from the first index in the string .The below code finds the length as well as pri the given string:

```
#include<stdio.h>

#include<string.h>

int lps(char * str)

{

int n = strlen(str);

int i,j,start_ind,end_ind;

int max = 0;

bool max_pal[n][n];
```

see more

Divya • 8 months ago

Is the following code correct? Can somebody please authenticate?

```
int main()

{

char *input = "abforgeeksskeegforba";

char outputarr[100] = {NULL};

//printf("strlen is %d\n", strlen(input));

int i = 0;

int j = strlen(input) - 1;

int k = 0;

bool isPal = false;

while (i <= j)
```


jayasurya j ➚ Divya • 2 months ago

i guess its wrong

Sanjith Sakthivel • 8 months ago

Follow the manacher&#039s Algorithm that&#039s simple and takes less spa

prity • 10 months ago

It seems O(n) solution is available. Refer this http://codeinterview.blogspot....

• Reply • Share ›

Shweta prity • 6 months ago

The solution given at the link is not O(n).

• Reply • Share ›

Purushotham • 11 months ago

I agree the above solution is using DP. However this problem has a better solu
Below is the code using Greedy approach.

```
public static int Lps(String s){
char[] c = s.toCharArray();
int[] lps = new int[s.length()];
int tmp, max_len = 1;
lps[0] = 1;
for(int i = 1; i 0 && c[i] == c[tmp])
lps[i] = lps[i-1] + 2;
else if(c[i] == c[i-1])
lps[i] = 2;
else
lps[i] = 1;

if(lps[i] > max_len) max_len = lps[i];
}

return max_len;
}
```

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

Debashish Ghosh ➚ Purushotham • 5 months ago

i have trouble understanding your code. Please remove some errors fr

▲ | ▼ • Reply • Share ›

Sarthak Mall 'shanky' • 11 months ago

I thing a simple brute force can help in O(n^2) and in O(1) space...

```
#include<iostream>
using namespace std;.
void lps(string x);.
int main()
{

string x = "forgeeksskeegfor";.

//string x = "aba";.

lps(x);.

system("pause");.

return 0;.
}
void lps(string x).
{
```

see more

▲ | ▼ • Reply • Share ›

pritybhudolia • 11 months ago

@GeeksforGeeks
Can we use this approach as it works in O(n) I think?

common sub-sequence between the given string and it's reverse.

For example: Given string is STR = {ABBAGEEKSSKEEG};
reverse string is REV = {GEEKSSKEEGABBA};
reverse string is OUTPUT = {GEEKSSKEEGABBA};
INDEX[] = {10,11,12,13,0,1,2,3,4,5,6,7,8,9};
we maintain an another array INDEX[] which searches for the occurence of ea
stores its index in INDEX[],
as soon as a character's first occurence in REV[] is found, the element is cha
repeated element
it doesn't store same index.

Find the longest increasing subsequence for INDEX{0,1,2,3,4,5,6,7,8,9};
Index this value in the OUTPUT[]. Finally we get "GEEKSSKEEG" which is lon

```
/* #include<stdio.h>
```

see more

1 ▲ ▼ • Reply • Share ›

Suchandrim 'Sucho' Sarkar ➚ pritybhudolia • 20 days ago
This solution isn't working!!!!

▲ ▼ • Reply • Share ›

Aashish ➚ pritybhudolia • 11 months ago
This method finds the Longest Palindrome Subsequence.
Please consider the input string: ABBCA
Its reverse would be: ACBBA
Index[] would be: 0, 3, 1, 2, 4
Longest Increasing Subsequence is 0, 1, 2, 4.
The corresponding string is ABBA which is a subsequence.
Also, finding LIS takes NlogN time at its best.

• Reply • Share ›

pritybhudolia ↗ Aashish • 11 months ago

@Aashish

No, it works as finding longest continuously increasing subsequ

▲ | ▼ • Reply • Share ›

pritybhudolia ↗ pritybhudolia • 11 months ago

@Ashish

I got your point. I think, I have fixed those errors, you car

```
 #include<stdio.h>
#include<conio.h>
int main()
{
int i,j,k=0;
int upper=0,lower=0,max=0,up=0,down=0;
  char str[100] = "ABBAGEEKSSKEEG";
  char rev[100],output[100];
  int index[100];
  for(i=(strlen(str)-1),j=0;i>=0;i--)
  {
         rev[j]=str[i];
         output[j]=str[i];
         j++;
  }
```

see more

▲ | ▼ • Reply • Share ›

prity bhudolia ↗ pritybhudolia • 10 months ago

yeah , i realized that earlier, but there is no option to del

Anyways, thanks for this :)

Aashish ➚ pritybhudolia • 10 months ago

Prity, your approach doesn't seem to be working for few
"STUCTS".
See here: http://ideone.com/oezBPM

In fact, the approach wont work for set of strings where
exists and part of its length is greater than the longest s
above example, it outputs the part of a longest palindror

pritybhudolia • 11 months ago

Longest palindromic sub string can also be obtained by reducing the problem
sequence between the given string and it's reverse.
For example: Given string is STR = {ABBAGEEKSSKEEG};
reverse string is REV = {GEEKSSKEEGABBA};
reverse string is OUTPUT = {GEEKSSKEEGABBA};
INDEX[] = {10,11,12,13,0,1,2,3,4,5,6,7,8,9};
we maintain an another array INDEX[] which searches for the occurence of ea
stores its index in INDEX[],as soon as a character's first occurence in REV[] is
when it searches for the repeated element it doesn't store same index.

Find the longest increasing subsequence for INDEX{0,1,2,3,4,5,6,7,8,9};
Index this value in the OUTPUT[]. Finally we get "GEEKSSKEEG" which is lon

```
#include
#include
int main()
{
int i,j,k=0,upper=0,lower=0,max=0,up=0,down=0;
```

char str[100] = "abbagcakaakcag";

see more

• Reply • Share ›

abhishek08aug • a year ago

Intelligent :D

• Reply • Share ›

javanetbeans • a year ago

As suggested by Nikhil, Suffix Tree yields the solution in O(n) instead of going
For suffix tree building in O(n) , please see this resource

http://www.cs.ucf.edu/~shzhang...

• Reply • Share ›

Nikhil • 2 years ago

Instead of dynamic programming we can go with the string matching

1) Lets us assume given string is str
2) Reverse the string and store it in str2
3) Now find the common substring using the suffix tree.

Time Complexity : O(n)

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

1 • Reply • Share ›

abhishek • 2 years ago

```
/* A O(n) iterative program for construction of BST from preorder tra
#include <stdio.h>
```

```
#include <limits.h>
//enum bool {true, false}boolean;
int LPS(char *str, int beg, int end, int token);
int max(int a, int b);

int LPS(char *str, int beg, int end, int token){
    if(beg > end)
        return 0;
    if(beg == end)
        return 1;
    if(str[beg] == str[end]) {
        return max(2+LPS(str, beg+1, end-1, 1), max(LPS(str, beg+1, e
    }
    else {
```

see more

• Reply • Share ›

suman • 2 years ago

[sourcecode language="java"]
public class LongestPalidromString {
static int leftIndex = 0;
static int rightIndex = 0;
public static void calculatePalidromLength(int left, int right, String str){
int length = 0;
while(left >=0 && right < str.length()){
if (str.charAt(left) == str.charAt(right)){
length++;
leftIndex = left;
rightIndex = right;
left--;

```
}else{
break;
}
}
}
```

see more

Reply • Share ›

Sharad Garg suman • a year ago

Time complexity is still O(n^2)

Reply • Share ›

sourabh jain • 2 years ago

```
#include
#include
using namespace std;
string fun(string s,int i)
{
int l1=0,l2=1,j,k;
string str="";
if(i>0 && s[i]==s[i-1])
{
j=i-2,k=i+1;
while(j>=0 && k<s.length() && s[j]==s[k])
{
j--;
k++;
}
l1=k-j-1;

}
```

see more

• Reply • Share ›

Duke ➚ sourabh jain • 2 years ago

Dude,
there are many silly mistakes in your code, first correct them .

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

• Reply • Share ›

sourabh jain ➚ Duke • 2 years ago

can u plz tell me case for which it is giving wrong answer

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

• Reply • Share ›

Subscribe Add Disqus to your site