

GeeksforGeeks

A computer science portal for geeks

GeeksQuiz

- [Home](#)
- [Algorithms](#)
- [DS](#)
- [GATE](#)
- [Interview Corner](#)
- [Q&A](#)
- [C](#)
- [C++](#)
- [Java](#)
- [Books](#)
- [Contribute](#)
- [Ask a Q](#)
- [About](#)

[Array](#)

[Bit Magic](#)

[C/C++](#)

[Articles](#)

[GFacts](#)

[Linked List](#)

[MCQ](#)

[Misc](#)

[Output](#)

[String](#)

[Tree](#)

[Graph](#)

Longest Even Length Substring such that Sum of First and Second Half is same

Given a string 'str' of digits, find length of the longest substring of 'str', such that the length of the substring is 2k digits and sum of left k digits is equal to the sum of right k digits.

Examples:

Input: str = "123123"

Output: 6

The complete string is of even length and sum of first and second half digits is same

Input: str = "1538023"

Output: 4

The longest substring with same first and second half sum is "5380"

A **Simple Solution** is to check every substring of even length. The following is C based implementation of simple approach.

```
// A simple C based program to find length of longest even length
// substring with same sum of digits in left and right
#include<stdio.h>
#include<string.h>

int findLength(char *str)
{
    int n = strlen(str);
    int maxlen =0; // Initialize result

    // Choose starting point of every substring
    for (int i=0; i<n; i++)
    {
        // Choose ending point of even length substring
        for (int j =i+1; j<n; j += 2)
        {
            int length = j-i+1; //Find length of current substr

            // Calculate left & right sums for current substr
            int leftsum = 0, rightsum =0;
            for (int k =0; k<length/2; k++)
            {
                leftsum += (str[i+k]-'0');
                rightsum += (str[i+k+length/2]-'0');
            }

            // Update result if needed
            if (leftsum == rightsum && maxlen < length)
                maxlen = length;
        }
    }
    return maxlen;
}

// Driver program to test above function
int main(void)
{
    char str[] = "1538023";
    printf("Length of the substring is %d", findLength(str));
    return 0;
}
```

Output:

Length of the substring is 4

The time complexity of above solution is $O(n^3)$. The above solution can be optimized to work in $O(n^2)$ using **Dynamic Programming**. The idea is to build a 2D table that stores sums of substrings. The

following is C based implementation of Dynamic Programming approach.

```
// A C based program that uses Dynamic Programming to find length of the
// longest even substring with same sum of digits in left and right half
#include <stdio.h>
#include <string.h>

int findLength(char *str)
{
    int n = strlen(str);
    int maxlen = 0; // Initialize result

    // A 2D table where sum[i][j] stores sum of digits
    // from str[i] to str[j]. Only filled entries are
    // the entries where j >= i
    int sum[n][n];

    // Fill the diagonal values for sunstrings of length 1
    for (int i = 0; i < n; i++)
        sum[i][i] = str[i] - '0';

    // Fill entries for substrings of length 2 to n
    for (int len = 2; len <= n; len++)
    {
        // Pick i and j for current substring
        for (int i = 0; i < n - len + 1; i++)
        {
            int j = i + len - 1;
            int k = len / 2;

            // Calculate value of sum[i][j]
            sum[i][j] = sum[i][j - k] + sum[j - k + 1][j];

            // Update result if 'len' is even, left and right
            // sums are same and len is more than maxlen
            if (len % 2 == 0 && sum[i][j - k] == sum[j - k + 1][j]
                && len > maxlen)
                maxlen = len;
        }
    }
    return maxlen;
}

// Driver program to test above function
int main(void)
{
    char str[] = "153803";
    printf("Length of the substring is %d", findLength(str));
    return 0;
}
```

Output:

Length of the substring is 4

Time complexity of the above solution is $O(n^2)$, but it requires $O(n^2)$ extra space.

This article is contributed by **Ashish Bansal**. Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above

Related Topics:

- [Recursively print all sentences that can be formed from list of word lists](#)
- [Check if a given sequence of moves for a robot is circular or not](#)
- [Find the longest substring with k unique characters in a given string](#)
- [Function to find Number of customers who could not get a computer](#)
- [Find maximum depth of nested parenthesis in a string](#)
- [Find all distinct palindromic sub-strings of a given string](#)
- [Find if a given string can be represented from a substring by iterating the substring “n” times](#)
- [Suffix Tree Application 6 – Longest Palindromic Substring](#)

Tags: [Dynamic Programming](#)



Tweet

< 3

g+1

< 3

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

19 Comments

GeeksforGeeks

1

Login ▾

♥ Recommend

🔗 Share

Sort by Newest ▾



Join the discussion...



Manoj Saini • 8 days ago

This java code uses the concept of prefix sum and solves it in $O(n^2)$ time with $O(n)$ space complexity.

```
public static void main(String s[]){
```

```
String st = "1538023";
```

```
char ch[] = st.toCharArray();
```

```
int a[] = new int[ch.length];
```

```
for(int i=0;i<ch.length;i++){ a[i]=Integer.parseInt(""+ch[i]); } int="" l="a.length;" for(int="" i="1;i<ch.length;i++){ a[i]=a[i-1] +"" a[i]; } int="" si="0;" int="" ei="1;" int="" midi="0;" int="" max="0;" for(si="0;si<ch.length;si++){ for(int="" len="2;len+si-1<ch.length;len=len+2){ ei="si+len-1;" midi=" (si+ei)/2;" int="" sum1="0;" int="" sum2="0;" if(si-1="">=0){
```

```
sum1 = a[midi] - a[si-1];
```

```
}else{
```

[see more](#)

^ | v • Reply • Share ›



saurabh tiwari • 13 days ago

Problem can be solved in $O(N^2)$ with $O(N)$ extra space.

^ | v • Reply • Share ›



prashant jha • a month ago

<http://ideone.com/VR4N2u>

^ | v • Reply • Share ›



creeping_death • 2 months ago

Ruby solution, slightly different, easier to read than OP's solution, but same time and space complexity

<http://ideone.com/2TN3uM>

^ | v • Reply • Share ›



Guest • 2 months ago

calling the 2nd solution as dp is stretching it. memoization is what you're actually doing here, to avoid recalculating the values

^ | v • Reply • Share ›



Aditya Goel • 2 months ago

Simple solution($O(n^3)$) can be turned into $O(n^2)$ time and $O(n)$ space by using sum array. Instead of calculating left & right sums, make a sum array. Then left & right sums can be found in constant time.

1 ^ | v • Reply • Share ›



tejavadali → Aditya Goel • 2 months ago

+1 exactly :)

^ | v • Reply • Share ›



Aditya Goel • 2 months ago

We can further optimize simple solution by doing this check $maxlen < length$ before calculating left & right sums. Calculate them only if needed.

^ | v • Reply • Share ›



coder • 3 months ago

really neat solution

1 ^ | v • Reply • Share ›



rushiraj chavan · 4 months ago

recurse(..) returns the length of Longest Even Length Substring

```
#include<iostream>
```

```
#include<stdlib.h>
```

```
using namespace std;
```

```
int recurse(char *str, int length, int lsum, int rsum, int lstart) {
```

```
if(lsum == rsum && length % 2 == 0) {
```

```
return length;
```

```
}
```

```
if(length <= 0) {
```

```
return 0;
```

```
}
```

[see more](#)

^ | v · [Reply](#) · [Share](#) ›



apurva jeswal · 4 months ago

Recursion method (C#)

```
static void Main(string[] args)
```

```
{
```

```
Console.WriteLine(GetMaxLength("123123"));
```

```
Console.ReadLine();
```

```
}
```

```
internal static int GetMaxLength(string str)
```

```
{
```

```
int len=0;
```

```
if (str.Length % 2 == 0)
```

```
len = str.Length;
```

[see more](#)

[see more](#)[^](#) | [v](#) • [Reply](#) • [Share](#) ›**yogi** • 4 months agoI have another $O(n^2)$ solution`String s="121126";``int i;``int maxLengh=-1;`

```
for(i=1;i<s.length()-1;i++){ int="" lsum="0,rsum=0;" int="" l,r;="" l="i;r=i+1;"
for(;l="">=0&&r<s.length();l--,r++){ lsum+="Integer.parseInt(s.charAt(l)+""");"
rsum+="Integer.parseInt(s.charAt(r)+""");" if((lsum==rsum)&&(maxLengh<(r-l)))"
maxlengh="(r-l)+1;" }="" }="" system.out.println(maxlengh);="">
```

[^](#) | [v](#) • [Reply](#) • [Share](#) ›**Koustav Chatterjee** • 4 months ago

I am trying to formulate a recursive solution.

```
public int f1(str, start, end) {
    if (start == end) return 1;
    if (start > end) return 0;
    if (str.length() % 2 == 1) {
        max(f1(str, start, end - 1), f1(str, start + 1, end));
    } else {
        if (sumOfSubstring % 2 == 0 && sumOfSubstring / 2 == sumOfFirstHalf) return end - start + 1;
        else {
            return max(f1(str, start, end - 2), f1(str, start + 2, end));
        }
    }
}
```

Ur comments pls.....

[^](#) | [v](#) • [Reply](#) • [Share](#) ›**apurva jeswal** ➔ **Koustav Chatterjee** • 4 months ago

try this

```
static void Main(string[] args)
{
    Console.WriteLine(GetMaxLength("11"));

    Console.ReadLine();
}
```

```

,

internal static int GetMaxLength(string str)
{
    int len=0;

    if (str.Length % 2 == 0)

        len = str.Length;

```

[see more](#)[^](#) | [v](#) • [Reply](#) • [Share](#) ›**Born Actor** • 4 months ago<http://ideone.com/8L3aAq>[^](#) | [v](#) • [Reply](#) • [Share](#) ›**nishantfirst** • 5 months ago

O(n2) time with O(1) space complexity

just modify

<http://www.geeksforgeeks.org/l...>

use the first while case

2 [^](#) | [v](#) • [Reply](#) • [Share](#) ›**rihansh** • 5 months ago<http://ideone.com/6phT5J>

A little optimised version solution to this problem with O(N) space requirement . .

4 [^](#) | [v](#) • [Reply](#) • [Share](#) ›**random** • 5 months ago

Simple method can be solved in O(n2) too if we store the sum of all the digits in advance.

[^](#) | [v](#) • [Reply](#) • [Share](#) ›**rohit_90** → random • 4 months ago

We don't need to store the sum also. Here is my code for simple approach. Time complexity of it O(n2).

<http://ideone.com/6pRqR8>[^](#) | [v](#) • [Reply](#) • [Share](#) ›

-
-
-
- - [Interview Experiences](#)
 - [Advanced Data Structures](#)
 - [Dynamic Programming](#)
 - [Greedy Algorithms](#)
 - [Backtracking](#)
 - [Pattern Searching](#)
 - [Divide & Conquer](#)
 - [Mathematical Algorithms](#)
 - [Recursion](#)
 - [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](#)

Follow @GeeksforGeeks

• Recent Comments

- lt_k
i need help for coding this function in java...
[Java Programming Language](#) · 2 hours ago

- [Piyush](#)

What is the purpose of else if (recStack[*i])...

[Detect Cycle in a Directed Graph](#) · [2 hours ago](#)

- [Andy Toh](#)

My compile-time solution, which agrees with the...

[Dynamic Programming | Set 16 \(Floyd Warshall Algorithm\)](#) · [2 hours ago](#)

- [lucy](#)

because we first fill zero in first col and...

[Dynamic Programming | Set 29 \(Longest Common Substring\)](#) · [2 hours ago](#)

- [lucy](#)

@GeeksforGeeks i don't n know what is this long...

[Dynamic Programming | Set 28 \(Minimum insertions to form a palindrome\)](#) · [3 hours ago](#)

- [manish](#)

Because TAN is not a subsequence of RANT. ANT...

[Given two strings, find if first string is a subsequence of second](#) · [3 hours ago](#)

•

@geeksforgeeks, [Some rights reserved](#) [Contact Us!](#)

Powered by [WordPress](#) & [MooTools](#), customized by geeksforgeeks team