








































# Practice Arena

Practice problems aimed to improve your coding skills.

-  PRACTICE-02\_SCAN-PRINT
-  PRACTICE-03\_TYPES
-  LAB-PRAC-02\_SCAN-PRINT
-  LAB-PRAC-01
-  PRACTICE-04\_COND
-  BONUS-PRAC-02
-  LAB-PRAC-03\_TYPES
-  PRACTICE-05\_COND-LOOPS
-  LAB-PRAC-04\_COND
-  LAB-PRAC-05\_CONDDLOOPS
-  PRACTICE-07\_LOOPS-ARR
-  LAB-PRAC-06\_LOOPS
-  LAB-PRAC-07\_LOOPS-ARR
-  LABEXAM-PRAC-01\_MIDSEM
-  PRACTICE-09\_PTR-MAT
-  LAB-PRAC-08\_ARR-STR
  -  Il fratello di Fibonacci
  -  Hidden Palindrome
  -  Hush Hush Hash
  -  Maximum Match
  -  El secreto de sus l
  -  Star Replacement
  -  Stronger together
  -  Rigorous and repeated redaction
  -  strnrev
  -  Monster and Mini Multiply
  -  Clash of the Substrings
  -  Personalizing Emails
-  PRACTICE-10\_MAT-FUN
-  LAB-PRAC-09\_PTR-MAT
-  LAB-PRAC-10\_MAT-FUN
-  PRACTICE-11\_FUN-PTR
-  LAB-PRAC-11\_FUN-PTR
-  LAB-PRAC-12\_FUN-STRUC
-  LABEXAM-PRAC-02\_ENDSEM
-  LAB-PRAC-13\_STRUC-NUM
-  LAB-PRAC-14\_SORT-MISC

# Hidden Palindrome

## LAB-PRAC-08\_ARR-STR

**Hidden Palindrome [20 marks]**

---

**Problem Statement**

The input will contain a string containing  $n$  characters (we promise that  $n$  will be equal to or less than 49). Read this string into a character array (lets call it `str`) (i.e. 1st character of the string is stored in `str[0]`, 2nd character in `str[1]`, and last character in `str[n-1]`. Additionally, `s[n] = '\0'` should be ensured by Mr C if you read the string from the input properly).

Find the length of the largest substring of `str` (including `str` itself) which is a palindrome. Print this length as the first line of your output. In the second and third lines of your output, print the indices of the starting character and the ending character of the this largest substring.

If there are two or more such substrings which are palindromes and of largest length, print the starting and ending indices of the substring with the least starting index.

We promise we will not give you a string that contains whitespace or non-printable characters like space/tab/newline. We also promise we will never give you an empty string as input.

**Caution**

1. A single letter/character is trivially a palindrome too, although a very short one.
2. Be careful about extra/missing lines and extra/missing spaces in your output.

**HINT:** The `strlen()` function can help you find the length of a string. Given a character array, it returns the number of characters in the array upto, but not including, the first occurrence of the null character in the array. Include the header file `string.h` to use this function.

---

**EXAMPLE 1 :**

INPUT

cbcdc

OUTPUT:

3

0

2

**Explanation:** "cbc" and "cdc" both are substrings in the given string which are palindromes. Since there are no substrings of size 4 or larger which are palindromes, the output is 3. The substring with the smallest starting index is "cbc", with starting index 0 and ending index 2.

**EXAMPLE 2 :**

INPUT

abcde

OUTPUT:

1  
0  
0

**Explanation:** The only substrings of the given string which are palindromes are of size 1 (i.e. the trivial palindromes "a","b","c","d","e"). Hence, the length of the largest palindrome is 1. The substring with the smallest starting index is "a", with starting index 0 and ending index 0.

-----

**Grading Scheme:**

Total marks: **[20 Points]**

There will be partial grading in this question. There are three lines in your output. Printing each line correctly, in the correct order, carries some weightage. The first line carries 50% weightage and the next two lines carry 25% weightage each. Each visible test case is worth 2 points and each hidden test case is worth 4 points. There are 2 visible and 4 hidden test cases.

Please remember, however, that when you press Submit/Evaluate, you will get a green bar only if all parts of your answer are correct. Thus, if your answer is only partly correct, Prutor will say that you have not passed that test case completely, but when we do autograding afterwards, you will get partial marks.

 **Start Solving! (/editor/practice/6164)**