



Full Name: Joshnavi Podapati

Email: joshnavipodapati@gmail.com

Test Name: Mock Test

Taken On: 23 Aug 2025 08:29:30 IST

Time Taken: 18 min 47 sec/ 90 min

Invited by: Ankush

Invited on: 23 Aug 2025 08:29:13 IST

Skills Score:

Tags Score:

100%

280/280

scored in **Mock Test** in 18 min
47 sec on 23 Aug 2025 08:29:30
IST

- Algorithms280/280
- Core CS280/280
- Data Structures105/105
- Easy280/280
- LCM105/105
- Least Common Multiple105/105
- Math105/105
- Problem Solving105/105
- Strings175/175
- gcd105/105
- greatest common divisor105/105
- problem-solving280/280
- sets105/105

Recruiter/Team Comments:

No Comments.

	Question Description	Time Taken	Score	Status
Q1	Palindrome Index > Coding	4 min 18 sec	105/ 105	✔
Q2	Between Two Sets > Coding	5 min 40 sec	105/ 105	✔
Q3	Anagram > Coding	5 min 59 sec	70/ 70	✔

QUESTION 1

✔

Correct Answer

Score 105

Palindrome Index > Coding

Problem Solving

Strings

Algorithms

Easy

problem-solving

Core CS

QUESTION DESCRIPTION

Given a string of lowercase letters in the range `ascii[a-z]`, determine the index of a character that can be removed to make the string a [palindrome](#). There may be more than one solution, but any will do. If the word is already a palindrome or there is no solution, return `-1`. Otherwise, return the index of a character to remove.

Example

s = "bcb**c**"

Either remove 'b' at index **0** or 'c' at index **3**.

Function Description

Complete the *palindromeIndex* function in the editor below.

palindromeIndex has the following parameter(s):

- *string s*: a string to analyze

Returns

- *int*: the index of the character to remove or **-1**

Input Format

The first line contains an integer ***q***, the number of queries.

Each of the next ***q*** lines contains a query string ***s***.

Constraints

- $1 \leq q \leq 20$
- $1 \leq \text{length of } s \leq 10^5 + 5$
- All characters are in the range `ascii[a-z]`.

Sample Input

STDIN	Function
3	q = 3
aaab	s = 'aaab' (first query)
baa	s = 'baa' (second query)
aaa	s = 'aaa' (third query)

Sample Output

```
3
0
-1
```

Explanation

Query 1: "aaab"

Removing 'b' at index **3** results in a palindrome, so return **3**.

Query 2: "baa"

Removing 'b' at index **0** results in a palindrome, so return **0**.

Query 3: "aaa"

This string is already a palindrome, so return **-1**. Removing any one of the characters would result in a palindrome, but this test comes first.

Note: The custom checker logic for this challenge is available [here](#).

CANDIDATE ANSWER

Language used: **Python 3**

1 #

```

2 # Complete the 'palindromeIndex' function below.
3 #
4 # The function is expected to return an INTEGER.
5 # The function accepts STRING s as parameter.
6 #
7
8 def palindromeIndex(s):
9     n=len(s)
10    for left in range(n//2):
11        right=n-1-left
12        if s[left]!=s[right]:
13            if validPalindrome(s,left,right-1):
14                return right
15            if validPalindrome(s,left+1,right):
16                return left
17        return -1
18    return -1
19 def validPalindrome(text,starting,ending):
20     length=ending-starting+1
21     for k in range(length//2):
22         if text[starting+k]!=text[ending-k]:
23             return False
24     return True
25

```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 1	Easy	Sample case	✔ Success	0	0.0298 sec	10 KB
Testcase 2	Medium	Hidden case	✔ Success	5	0.0341 sec	10.1 KB
Testcase 3	Medium	Hidden case	✔ Success	5	0.0243 sec	10.3 KB
Testcase 4	Medium	Hidden case	✔ Success	5	0.0261 sec	10.3 KB
Testcase 5	Medium	Hidden case	✔ Success	5	0.0236 sec	10.1 KB
Testcase 6	Medium	Hidden case	✔ Success	5	0.0479 sec	10.4 KB
Testcase 7	Medium	Hidden case	✔ Success	5	0.0477 sec	10.3 KB
Testcase 8	Medium	Hidden case	✔ Success	5	0.0489 sec	10.3 KB
Testcase 9	Hard	Hidden case	✔ Success	10	0.0326 sec	10.4 KB
Testcase 10	Hard	Hidden case	✔ Success	10	0.0339 sec	10.4 KB
Testcase 11	Hard	Hidden case	✔ Success	10	0.0368 sec	10.3 KB
Testcase 12	Hard	Hidden case	✔ Success	10	0.0244 sec	10.1 KB
Testcase 13	Hard	Hidden case	✔ Success	10	0.0321 sec	10.4 KB
Testcase 14	Hard	Hidden case	✔ Success	10	0.0326 sec	10.5 KB
Testcase 15	Hard	Hidden case	✔ Success	10	0.0367 sec	10.4 KB

No Comments

QUESTION 2



Correct Answer

Score 105

Between Two Sets > Coding Math Algorithms Easy gcd Data Structures LCM sets

problem-solving Core CS greatest common divisor Least Common Multiple

QUESTION DESCRIPTION

There will be two arrays of integers. Determine all integers that satisfy the following two conditions:

1. The elements of the first array are all factors of the integer being considered

2. The integer being considered is a factor of all elements of the second array

These numbers are referred to as being *between* the two arrays. Determine how many such numbers exist.

Example

$a = [2, 6]$
 $b = [24, 36]$

There are two numbers between the arrays: **6** and **12**.

$6\%2 = 0$, $6\%6 = 0$, $24\%6 = 0$ and $36\%6 = 0$ for the first value.

$12\%2 = 0$, $12\%6 = 0$ and $24\%12 = 0$, $36\%12 = 0$ for the second value. Return **2**.

Function Description

Complete the `getTotalX` function in the editor below. It should return the number of integers that are between the sets.

`getTotalX` has the following parameter(s):

- `int a[n]`: an array of integers
- `int b[m]`: an array of integers

Returns

- `int`: the number of integers that are between the sets

Input Format

The first line contains two space-separated integers, n and m , the number of elements in arrays a and b .

The second line contains n distinct space-separated integers $a[i]$ where $0 \leq i < n$.

The third line contains m distinct space-separated integers $b[j]$ where $0 \leq j < m$.

Constraints

- $1 \leq n, m \leq 10$
- $1 \leq a[i] \leq 100$
- $1 \leq b[j] \leq 100$

Sample Input

```
2 3
2 4
16 32 96
```

Sample Output

```
3
```

Explanation

2 and 4 divide evenly into 4, 8, 12 and 16.

4, 8 and 16 divide evenly into 16, 32, 96.

4, 8 and 16 are the only three numbers for which each element of a is a factor and each is a factor of all elements of b .

CANDIDATE ANSWER

Language used: **Python 3**

```
1
2 #
3 # Complete the 'getTotalX' function below.
4 #
5 # The function is expected to return an INTEGER.
6 # The function accepts following parameters:
7 # 1. INTEGER_ARRAY a
```

```

8 # 2. INTEGER_ARRAY b
9 #
10
11 def getTotalX(a, b):
12     limit=min(b)
13     valid_numbers=[]
14     for num in range(max(a),limit+1):
15         if all(num%x==0 for x in a):
16             if all(y%num==0 for y in b):
17                 valid_numbers.append(num)
18     return len(valid_numbers)
19

```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 1	Easy	Sample case	✔ Success	0	0.0291 sec	10.3 KB
Testcase 2	Easy	Hidden case	✔ Success	15	0.0245 sec	10.1 KB
Testcase 3	Easy	Hidden case	✔ Success	15	0.0275 sec	10.3 KB
Testcase 4	Easy	Hidden case	✔ Success	15	0.026 sec	10.1 KB
Testcase 5	Easy	Hidden case	✔ Success	15	0.0252 sec	10.3 KB
Testcase 6	Easy	Hidden case	✔ Success	15	0.023 sec	10.3 KB
Testcase 7	Easy	Hidden case	✔ Success	15	0.0235 sec	10.3 KB
Testcase 8	Easy	Hidden case	✔ Success	15	0.0342 sec	10.3 KB
Testcase 9	Easy	Sample case	✔ Success	0	0.0334 sec	10.1 KB

No Comments

QUESTION 3



Correct Answer

Score 70

Anagram > Coding

Strings

Algorithms

Easy

problem-solving

Core CS

QUESTION DESCRIPTION

Two words are *anagrams* of one another if their letters can be rearranged to form the other word.

Given a string, split it into two contiguous substrings of equal length. Determine the minimum number of characters to change to make the two substrings into anagrams of one another.

Example

***s* = abccde**

Break ***s*** into two parts: 'abc' and 'cde'. Note that all letters have been used, the substrings are contiguous and their lengths are equal. Now you can change 'a' and 'b' in the first substring to 'd' and 'e' to have 'dec' and 'cde' which are anagrams. Two changes were necessary.

Function Description

Complete the *anagram* function in the editor below.

anagram has the following parameter(s):

- string s*: a string

Returns

- int*: the minimum number of characters to change or -1.

Input Format

The first line will contain an integer, q , the number of test cases.

Each test case will contain a string s .

Constraints

- $1 \leq q \leq 100$
- $1 \leq |s| \leq 10^4$
- s consists only of characters in the range `ascii[a-z]`.

Sample Input

```
6
aaabbb
ab
abc
mnop
xyyx
xaxbbxx
```

Sample Output

```
3
1
-1
2
0
1
```

Explanation

Test Case #01: We split s into two strings $S1='aaa'$ and $S2='bbb'$. We have to replace all three characters from the first string with 'b' to make the strings anagrams.

Test Case #02: You have to replace 'a' with 'b', which will generate "bb".

Test Case #03: It is not possible for two strings of unequal length to be anagrams of one another.

Test Case #04: We have to replace both the characters of first string ("mn") to make it an anagram of the other one.

Test Case #05: $S1$ and $S2$ are already anagrams of one another.

Test Case #06: Here $S1 = "xaxb"$ and $S2 = "bbxx"$. You must replace 'a' from $S1$ with 'b' so that $S1 = "xbxb"$.

CANDIDATE ANSWER

Language used: **Python 3**

```
1
2 #
3 # Complete the 'anagram' function below.
4 #
5 # The function is expected to return an INTEGER.
6 # The function accepts STRING s as parameter.
7 #
8
9 def anagram(s):
10     n=len(s)
11     if n%2!=0:
12         return -1
13     mid=n//2
14     left=s[:mid]
15     right=s[mid:]
```

```
16     freq=[0]*26
17     for c in left:
18         freq[ord(c)-97]+=1
19     for c in right:
20         idx=ord(c)-97
21         if freq[idx]>0:
22             freq[idx]-=1
23     return sum(freq)
24
25
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 1	Easy	Hidden case	✔ Success	5	0.0344 sec	10.3 KB
Testcase 2	Easy	Hidden case	✔ Success	5	0.0261 sec	10.3 KB
Testcase 3	Easy	Hidden case	✔ Success	5	0.023 sec	10.3 KB
Testcase 4	Easy	Hidden case	✔ Success	5	0.025 sec	10.3 KB
Testcase 5	Easy	Hidden case	✔ Success	5	0.027 sec	10.1 KB
Testcase 6	Easy	Hidden case	✔ Success	5	0.0931 sec	10.3 KB
Testcase 7	Easy	Hidden case	✔ Success	5	0.0535 sec	10.3 KB
Testcase 8	Easy	Hidden case	✔ Success	5	0.086 sec	10.3 KB
Testcase 9	Easy	Hidden case	✔ Success	5	0.0456 sec	10.3 KB
Testcase 10	Easy	Hidden case	✔ Success	5	0.0855 sec	10.3 KB
Testcase 11	Easy	Hidden case	✔ Success	5	0.0443 sec	10 KB
Testcase 12	Easy	Hidden case	✔ Success	5	0.1004 sec	10.1 KB
Testcase 13	Easy	Hidden case	✔ Success	5	0.0838 sec	10.1 KB
Testcase 14	Easy	Hidden case	✔ Success	5	0.0946 sec	10.3 KB
Testcase 15	Easy	Sample case	✔ Success	0	0.0257 sec	10.3 KB
Testcase 16	Easy	Sample case	✔ Success	0	0.0309 sec	10.1 KB

No Comments