You can view this report online at : https://www.hackerrank.com/x/tests/1242137/candidates/33152213/report

Full Name:	Rohan Raj
Email:	rr07656@st.habib.edu.pk
Test Name:	**CS 101 - Lab 9 - Fall 2021**
Taken On:	27 Nov 2021 17:03:10 PKT
Time Taken:	2574 min 8 sec/ 10080 min
Work Experience:	< 1 years
Invited by:	Aisha
Skills Score:	
Tags Score:	CS101 220/220 Strings 220/220 for-in loop 40/40

100%
625/625

scored in **CS 101 - Lab 9 - Fall 2021** in 2574 min 8 sec on 27 Nov 2021 17:03:10 PKT

Recruiter/Team Comments:

No Comments.

Question Description	Time Taken	Score	Status
Q1 Split into Unicode > Coding	27 min 59 sec	40/ 40	✓
Q2 Recursively slow conceal > Coding	54 min 29 sec	40/ 40	✓
Q3 Third or fifth > Coding	34 min 37 sec	50/ 50	✓
Q4 Count vowels in a string > Coding	13 min 36 sec	40/ 40	✓
Q5 Stretch a string > Coding	8 min 2 sec	40/ 40	✓
Q6 Split email address > Coding	8 min 44 sec	40/ 40	✓
Q7 Rotate a string > Coding	20 min 2 sec	60/ 60	✓
Q8 Devowelify > Coding	5 min 16 sec	40/ 40	✓
Q9 Is it a palindrome? > Coding	23 min 40 sec	45/ 45	✓
Q10 Intersection > Coding	23 min 33 sec	80/ 80	✓
Q11 My Way > Coding	4 hour 19 min 48 sec	40/ 40	✓
Q12 Find ourselves > Coding	5 min 6 sec	50/ 50	✓
Q13 Problem Solving - Pattern 1 > Coding	22 min 36 sec	20/ 20	✓

Q14	Problem Solving - Pattern 2 › Coding	38 min 41 sec	20/ 20	✓
Q15	Problem Solving - Pattern 3 › Coding	1 hour 25 min 35 sec	20/ 20	✓

QUESTION 1

Correct Answer

Score 40

Split into Unicode › Coding Strings CS101

QUESTION DESCRIPTION

Challenge

Write a function `split_into_unicode(s)` that prints out the Unicode value of each letter in `s`.

Sample

```
>>> split_into_unicode('Pizza!')
80
105
122
122
97
33
```

Unicode

Each character in Python has a unique unicode assigned to it. For instance, the letter 'A' has the unicode 65 assigned to it, while 'Z' has the unicode 90 assigned to it. Note that 'a' and 'z' have a different unicode numbers as compared to the uppercase letters (97 and 122 respectively). The unicode values will be studied in more depth in the future, but for now, all you are required to know is that you can obtain the unicode for a given character using the ord() function. For instance:
print(ord('A')) # This will print 65 onto the screen

INTERVIEWER GUIDELINES

Solution

```
def split_into_unicode(s):
    for letter in s:
        print(ord(letter))
```

CANDIDATE ANSWER

Language used: **Python 3**

```
1 def split_into_unicode(s):
2     for i in range (0 , len (s)):
3         print (ord(s[i]))
4
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	Sample case	Success	10	0.0328 sec	7.93 KB
Testcase 1	Easy	Sample case	Success	10	0.0233 sec	7.94 KB
Testcase 2	Easy	Sample case	Success	10	0.0321 sec	7.98 KB
Testcase 3	Easy	Sample case	Success	10	0.022 sec	7.94 KB

No Comments

QUESTION 2

Correct Answer

Score 40

Recursively slow conceal › Coding

QUESTION DESCRIPTION

Challenge

Write a **recursive** function `slow_conceal(s)` that prints the lines in reverse order, i.e., the entire string `s` is printed first, then the string `s` with the last letter missing, then the last two letters missing, and so on, until only the first letter in `s` is printed.

Note: you may not use while or for loops.

Sample

```
>>> slow_conceal('Pizza!')
Pizza!
Pizza
Pizz
Piz
Pi
P
```

INTERVIEWER GUIDELINES

Solution

```
def slow_conceal(s):
    if s == "":
        return ""
    else:
        print(s)
        slow_conceal(s[:-1])
```

CANDIDATE ANSWER

Language used: **Python 3**

```
1 def slow_conceal(s):
2     a = len(s)
3     print (s[0: a])
4     a -= 1
5     if a > 0:
6         slow_conceal(s[0 : a])
7 s = input().strip()
8 slow_conceal(s)
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	Sample case	Success	10	0.0243 sec	7.95 KB
Testcase 1	Easy	Sample case	Success	10	0.0288 sec	8.08 KB
Testcase 2	Easy	Sample case	Success	10	0.0222 sec	8.01 KB
Testcase 3	Easy	Sample case	Success	10	0.0231 sec	7.79 KB

No Comments

QUESTION 3

Correct Answer

Score 50

Third or fifth › Coding

QUESTION DESCRIPTION

Challenge

Write a function `third_or_fifth(s)` that returns every third and every fifth letter in the string `s`.

Sample

```
>>> print(third_or_fifth('123456789012345678901234567890'))
35690258014570
>>> print(third_or_fifth('pomegranate'))
mgrat
```

CANDIDATE ANSWER

Language used: **Python 3**

```
# Feel free to modify the code below, discussed in the class lecture:
def third_or_fifth(s):
    result = ''
    index = 0
    while index < len(s):
        letter = s[index]
        if (index + 1) % 3 == 0 or (index + 1) % 5 == 0:
            result = result + letter
        index = index + 1
    return result

```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	Sample case	Success	10	0.0311 sec	7.95 KB
Testcase 1	Easy	Sample case	Success	10	0.0232 sec	8 KB
Testcase 2	Easy	Sample case	Success	10	0.0225 sec	7.95 KB
Testcase 3	Easy	Sample case	Success	10	0.0453 sec	7.9 KB
Testcase 4	Easy	Sample case	Success	10	0.0274 sec	7.84 KB

No Comments

QUESTION 4

Correct Answer

Score 40

Count vowels in a string › Coding

QUESTION DESCRIPTION

Challenge

Write a function `count_vowels(s)` that returns the number of vowels (a, e, i, o, u) in the string `s`.

Sample

```
>>> print(count_vowels('The quick brown fox jumps over the lazy dog.'))
8
```

CANDIDATE ANSWER

Language used: **Python 3**

```
def count_vowels(s):
    count = 0
    for i in range (0 , len (s)):
        if s [i] == "a" or s [i] == "e" or s [i] == "i" or s [i] == "o" or s
[i] == "u" or s [i] == "A" or s [i] == "E" or s [i] == "I" or s [i] == "O" or
s [i] == "U":
            count = count + 1
    return count
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	Sample case	Success	10	0.0238 sec	7.97 KB
Testcase 1	Easy	Sample case	Success	10	0.0243 sec	7.93 KB
Testcase 2	Easy	Hidden case	Success	10	0.0323 sec	7.92 KB
Testcase 3	Easy	Hidden case	Success	10	0.0241 sec	7.92 KB

No Comments

QUESTION 5

Correct Answer

Score 40

Stretch a string › Coding CS101 Strings

QUESTION DESCRIPTION

Challenge

Write a function `stretch(s)` that takes a string argument `s` and returns a new string such that the first character appears once, the second character is repeated twice, the third character is repeated thrice, and so on.

Sample

```
>>> print(stretch('Gum'))
Guummm
>>> print(stretch('Pizza!'))
Piizzzzzzzaaaaa!!!!!!
```

INTERVIEWER GUIDELINES

Solution

```
def stretch(s):
    newStr = ""
    i = 1
    for st in s:
        newStr = newStr + (st*i)
        i = i + 1
    return newStr
```

CANDIDATE ANSWER

Language used: **Python 3**

```
1  def stretch(s):
2      y = ""
3      for i in range (0 , len(s)):
4          x = s [i] * (i + 1)
5          y = y + str(x)
6      return y
7
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	Sample case	Success	10	0.0471 sec	8 KB
Testcase 1	Easy	Sample case	Success	10	0.0311 sec	7.95 KB
Testcase 2	Easy	Hidden case	Success	10	0.0237 sec	7.9 KB
Testcase 3	Easy	Hidden case	Success	10	0.0248 sec	7.95 KB

No Comments

QUESTION 6

Correct Answer

Split email address › Coding

QUESTION DESCRIPTION

Score 40

Challenge

Write a function `split_email(email)` that extracts and outputs the user name and domain from a given email address. The function will be passed an `email` (string) as parameter

Rules

An email address will always be given as follows: 'username@domain'. Some examples are 'martin@mars.com', 'robert@space.com'.

Sample

```
>>> split_email('martin@mars.com')
username: martin
domain: mars.com
>>> split_email('robert@space.com')
username: robert
domain: space.com
```

INTERVIEWER GUIDELINES

Solution

```
def split_email(email):

    index = email.find('@')

    username = email[:index]
    domain = email[(index+1):]

    print('username:', username)
    print('domain:', domain)
```

CANDIDATE ANSWER

Language used: **Python 3**

```
1 def split_email(email):
2     i = 0
3     c = 0
4     while email [i] != "@":
5         c = c + 1
6         i += 1
7     print ("username: " + str(email[0 : c]) )
8     print ("domain: " + str(email[c + 1 : len (email)]))
9
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	Sample case	Success	10	0.0316 sec	7.91 KB
Testcase 1	Easy	Sample case	Success	10	0.0323 sec	7.95 KB
Testcase 2	Easy	Hidden case	Success	10	0.0356 sec	8 KB
Testcase 3	Easy	Hidden case	Success	10	0.022 sec	7.96 KB

No Comments

QUESTION 7

Rotate a string › Coding Strings CS101

Correct Answer

Score 60

QUESTION DESCRIPTION

Challenge

Given a word, a rotation is found by moving one letter from the head to the tail of the word. For example, the word 'orange' rotated once is 'rangeo', rotated twice is 'angeor', rotated thrice is 'ngeora'.

Write a function `rotate(s, n)` that takes two arguments, a string `s` and an int `n`, that rotates a given string `s`, `n` times.

Sample

```
>>> print(rotate('potato', 1))
otatop
>>> print(rotate('potato', 3))
atopot
>>> print(rotate('potato', -2))
topota
```

Input/Output

Input and output will be handled by HackerRank.

Constraints

`s` is a string.
`t` is a integer.

INTERVIEWER GUIDELINES

Solution

```
def rotate(s, n):
    """Returns s rotated n times.

    Args:
    - s (str): the string to be rotated
    - n (int): the number of times to rotate the string

    Observations:
    - s rotated n times moves the first n letters of s to the end.
    - If n is the length of s, then the rotated string is the same as s.

    Returns:
    str: s rotated n times.
    """
    # Take away useless rotations.
    n %= len(s)
    # Get the first n letters of s and the remainder of s.
    first = s[:n]
    rest = s[n:]
    # The rotation is the first n letters at the end.
    return rest + first
```

CANDIDATE ANSWER

Language used: **Python 3**

```
1 def rotate(s, n):
2     while n > len (s):
3         n = n - len (s)
4     while n < - (len (s)):
5         n = n + len(s)
6     if n >= 0:
```

```
        x = s [n: len (s)]
        return (x + str(s[0:n]))
    elif n < 0:
        x = s [0 : n]
        return (str(s[len(s) + n] + x))

```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	Sample case	Success	10	0.0319 sec	8.04 KB
Testcase 1	Easy	Sample case	Success	10	0.0232 sec	8.07 KB
Testcase 2	Easy	Sample case	Success	10	0.0235 sec	8.18 KB
Testcase 3	Easy	Hidden case	Success	10	0.021 sec	8.07 KB
Testcase 4	Easy	Hidden case	Success	10	0.0212 sec	8.07 KB
Testcase 5	Easy	Hidden case	Success	10	0.0257 sec	8.23 KB

No Comments

QUESTION 8

Correct Answer

Score 40

Devowelify › Coding

Strings | CS101

QUESTION DESCRIPTION

Challenge

Write a function `devowelify(s)` that takes a string argument `s` and returns a string identical to s, except with all the vowels (a, e, i, o, u) removed.

Sample

```
>>> print(devowelify('Defenestrate'))
Dfnstrt
>>> print(devowelify('The quick brown fox jumps over the lazy dog.'))
Th qck brwn fx jmps vr th lzy dg.
```

INTERVIEWER GUIDELINES

Solution

```
def devowelify(s):
    result = ''
    for letter in s:
        if letter not in 'aeiouAEIOU':
            result = result + letter
    return result
```

CANDIDATE ANSWER

Language used: **Python 3**

```
1 def devowelify(s):
2     x = ""
3     for i in range (0 , len (s)):
4         if s[i] != "a" and s[i] != "e" and s[i] != "i" and s[i] != "o" and
5 s[i] != "u" and s[i] != "A" and s[i] != "E" and s[i] != "I" and s[i] != "O"
6 and s[i] != "U":
7             x = x + str(s[i])
    return x
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	Sample case	Success	10	0.0251 sec	7.97 KB
Testcase 1	Easy	Sample case	Success	10	0.0239 sec	7.93 KB
Testcase 2	Easy	Sample case	Success	10	0.0214 sec	7.92 KB
Testcase 3	Easy	Sample case	Success	10	0.0244 sec	7.91 KB

No Comments

QUESTION 9

Correct Answer

Score 45

Is it a palindrome? › Coding

QUESTION DESCRIPTION

Challenge

Write a function is_palindrome(s) that returns True if the given string s is a palindrome (reads the same forward and backward), False otherwise.

Sample

```
>>> print(is_palindrome('racecar'))
True
>>> print(is_palindrome('racecars'))
False
```

CANDIDATE ANSWER

Language used: **Python 3**

```
def is_palindrome(s):
    a= ""
    l = len (s)
    for i in range (1, l):
        a = a + str(s[-i])
    a = a + str(s[0])
    if a == s:
        return True
    else:
        return False

```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	Sample case	Success	10	0.0268 sec	7.88 KB
Testcase 1	Easy	Sample case	Success	5	0.023 sec	7.78 KB
Testcase 2	Easy	Sample case	Success	10	0.0228 sec	7.96 KB
Testcase 3	Easy	Sample case	Success	5	0.0362 sec	7.97 KB
Testcase 5	Easy	Sample case	Success	5	0.0229 sec	8.01 KB
Testcase 6	Easy	Sample case	Success	10	0.0231 sec	8.01 KB

No Comments

QUESTION 10

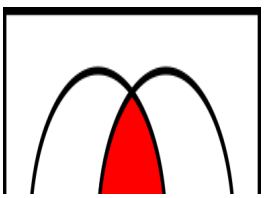
Correct Answer

Score 80

Intersection › Coding

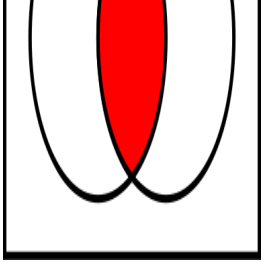
QUESTION DESCRIPTION

Problem



Python provides the `in` operator to easily check membership, e.g.

```
>>> 'l' in 'Hello World'
True
>>> 'X' in 'Hello World'
False
```



Credit: Wikipedia

```
>>> 'World' in 'Hello World'
True
```

Write a function called `common` that takes parameters `s1` and `s2` and uses the `in` operator to return a string containing the unique common letters in `s1` and `s2`.

Sample

```
>>> common('telephone', 'telegraph')
'telph'
>>> common('apple', 'orange')
'ae'
>>> common('you', 'me')
''
>>> common('work', 'play')
''
>>> common('antelope', 'eagle')
'ael'
>>> common('eagle', 'antelope')
'eal'
```

Input Format

The input contains `s1` and `s2` on separate lines.

Constraints

- `isinstance(s1, str)` is `True`
- `isinstance(s2, str)` is `True`

INTERVIEWER GUIDELINES

Solution

```
s1 = input()
s2 = input()

def common(s1, s2):
    common = ''
    for i in s1:
        if i in s2 and i not in common:
            common += i
    return common
```

CANDIDATE ANSWER

Language used: **Python 3**

```
s1 = input()
s2 = input()
def common (s1, s2):
    b = 0
    x = ""
    for i in range(len(s1)):
        for j in range (len(s2)):
            if s1[i] == s2[j]:
                a = s1[i]
                b = 0
```

```
                for k in range (len(x)):
                    if a == x[k]:
                        b = 1
                if b == 0:
                    x = x + str(a)
    return x
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
TestCase 0	Easy	Sample case	Success	10	0.042 sec	7.83 KB
TestCase 1	Easy	Sample case	Success	10	0.0214 sec	7.94 KB
TestCase 2	Easy	Sample case	Success	10	0.0323 sec	7.91 KB
TestCase 3	Easy	Sample case	Success	10	0.0226 sec	7.93 KB
TestCase 4	Easy	Sample case	Success	10	0.0276 sec	7.98 KB
TestCase 5	Easy	Sample case	Success	10	0.0228 sec	8.03 KB
TestCase 6	Easy	Sample case	Success	10	0.0233 sec	7.82 KB
TestCase 7	Easy	Sample case	Success	10	0.0208 sec	7.96 KB

No Comments

QUESTION 11

Correct Answer

Score 40

My Way › Coding Strings CS101 for-in loop

QUESTION DESCRIPTION

Problem

Write a function called `length` that takes a parameters `s` and returns the length of `s`. Do not use `len()`.

Sample

```
>>> length('Hello World')
11
>>> length('Yohsin')
6
>>> length('kilometer')
9
```

Input Format

The input contains `s` on the first line.

Constraints

- `isinstance(s, str)` is `True`

INTERVIEWER GUIDELINES

Solution

```
s = input()
def length(s):
    a = 0
    for i in s:
        a += 1
    return a
```

CANDIDATE ANSWER

Language used: **Python 3**

```
s = input ()
def length(s):
    c = 0
    for x in s:
        c += 1
    return c

print(length(s))
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
TestCase 0	Easy	Sample case	Success	10	0.0284 sec	7.78 KB
TestCase 1	Easy	Hidden case	Success	10	0.0225 sec	7.88 KB
TestCase 2	Easy	Hidden case	Success	10	0.0253 sec	7.82 KB
TestCase 3	Easy	Hidden case	Success	10	0.0271 sec	7.89 KB

No Comments

QUESTION 12

Correct Answer

Score 50

Find ourselves › Coding

QUESTION DESCRIPTION

Challenge

Write a function find(s, t) that returns the index of the location where t is found in s, or -1 if t is not found in s, starting from the leftmost letter in s.
Hint: Use built-in function find().

Sample

```
>>> print(find('The quick brown fox jumps over the lazy dog.', 'he'))
1
>>> print(find('The quick brown fox jumps over the lazy dog.', 'she'))
-1
```

CANDIDATE ANSWER

Language used: **Python 3**

```
1 def find(s, t):
2     return (s.find(t))
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	Sample case	Success	10	0.0248 sec	7.9 KB
Testcase 1	Easy	Sample case	Success	10	0.0217 sec	7.9 KB
Testcase 2	Easy	Sample case	Success	10	0.0235 sec	7.99 KB
Testcase 3	Easy	Sample case	Success	10	0.0269 sec	7.83 KB
Testcase 4	Easy	Sample case	Success	10	0.0287 sec	7.89 KB

No Comments

QUESTION 13

Correct Answer

Score 20

Problem Solving - Pattern 1 › Coding

QUESTION DESCRIPTION

Problem

Write an *iterative* function named `pattern` to generate the following pattern for a given parameter, `n`.

Sample

```
>>> pattern(5)
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5

>>> pattern(1)
1

>>> pattern(2)
1
1 2
```

Input

Input `n` from the console without any prompt.

Constraints

- `isinstance(n, int)` is `True`
- n >= 1

CANDIDATE ANSWER

Language used: **Python 3**

```
n = int(input())
def pattern(n):
    r = "1"
    print (r)
    if n >= 2:
        for i in range (2 , n + 1):
            r = r + " " + str(i)
            print (r)

```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
TestCase 0	Easy	Sample case	Success	2.5	0.0322 sec	8.04 KB
TestCase 1	Easy	Hidden case	Success	2.5	0.0282 sec	7.92 KB
TestCase 2	Easy	Hidden case	Success	2.5	0.0308 sec	8.14 KB
TestCase 3	Easy	Sample case	Success	2.5	0.0237 sec	8.14 KB
Testcase 4	Easy	Sample case	Success	10	0.0282 sec	8.2 KB

No Comments

QUESTION 14

QUESTION 14

Correct Answer

Score 20

Problem Solving - Pattern 2 › Coding

QUESTION DESCRIPTION

Problem

Write an *iterative* function named `pattern` to generate the following pattern for a given parameter, `n`.

Sample

```
>>> pattern(5)
0 1 2 3 4 5
0 1 2 3 4
0 1 2 3
0 1 2
0 1

>>> pattern(1)
0 1

>>> pattern(2)
0 1 2
0 1
```

Input

Input `n` from the console without any prompt.

Constraints

- `isinstance(n, int)` is `True`
- n >= 1

CANDIDATE ANSWER

Language used: **Python 3**

```
n = int(input())
def pattern(n):
    r = "0"
    if n >= 1:
        for i in range (1 , n + 1):
            r = r + " " + str(i)
        if len(r) > 20:
            for j in range (len(r), 20, -3):
                print (r [0:j])
            for j in range (19, 2, -2):
                print (r [0:j])
        else:
            for j in range (len(r), 2, -2):
                print (r [0:j])

```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
TestCase 0	Easy	Sample case	Success	2.5	0.0313 sec	8 KB
TestCase 1	Easy	Hidden case	Success	2.5	0.0332 sec	8.17 KB
TestCase 2	Easy	Hidden case	Success	2.5	0.0299 sec	8.07 KB
TestCase 3	Easy	Sample case	Success	2.5	0.0317 sec	8.16 KB
Testcase 4	Easy	Sample case	Success	10	0.0237 sec	8.04 KB

No Comments

QUESTION 15

Correct Answer

Score 20

Problem Solving - Pattern 3 › Coding

QUESTION DESCRIPTION

Problem

Write an *iterative* function named `pattern` to generate the following pattern for a given parameter, `n`.

Sample

```
>>> pattern(3)
1
2 3 4
5 6 7 8 9

>>> pattern(1)
1

>>> pattern(2)
1
2 3 4

>>> pattern(6)
1
2 3 4
5 6 7 8 9
10 11 12 13 14 15 16
17 18 19 20 21 22 23 24 25
26 27 28 29 30 31 32 33 34 35 36
```

Input

Input `n` from the console without any prompt.

Constraints

- `isinstance(n, int)` is `True`
- n >= 1

CANDIDATE ANSWER

Language used: **Python 3**

```
n = int(input())
def pattern(n):
    l = 1
    i = 0
    while l <= n:
        i += 1
        if i < l * l:
            print (i, end = " ")

        else:
            l += 1
            print (i)

```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
TestCase 0	Easy	Sample case	Success	2.5	0.0357 sec	8.03 KB

TestCase 1	Easy	Hidden case	Success	2.5	0.0291 sec	8.2 KB
TestCase 2	Easy	Hidden case	Success	2.5	0.0361 sec	8.17 KB
TestCase 3	Easy	Sample case	Success	2.5	0.0229 sec	7.96 KB
Testcase 4	Easy	Sample case	Success	10	0.0242 sec	7.98 KB

No Comments

PDF generated at: 22 Dec 2021 09:10:49 UTC