Rajalakshmi Engineering College

Name: Keertana Anjimeeti Srihari
Email: 240701252@rajalakshmi.edu.in
Roll no: 240701252
Phone: 9884916024
Branch: REC
Department: I CSE AH
Batch: 2028
Degree: B.E - CSE

Scan to verify results

NeoColab_REC_CS23221_Python Programming

REC_Python_Week 3_CY

Attempt : 1
Total Mark : 30
Marks Obtained : 25

Section 1 : Coding

1. Problem Statement

Write a program to check if a given string is perfect.

A perfect string must satisfy the following conditions:

The string starts with a consonant.The string alternates between consonants and vowels.Each consonant appears exactly once.Vowels can occur consecutively multiple times but should not be followed immediately by a consonant.

If the string satisfies all these conditions, print “True”; otherwise, print “False”.

Input Format

The input consists of a string.

Output Format

The output prints "True" if the string is perfect. Otherwise, print "False".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: capacitor
Output: True

Answer

```
def is_consonant(char):
    """Checks if a character is a consonant."""
    return 'a' not in char and 'e' not in char and 'i' not in char and 'o' not in char and
'u' not in char

def is_perfect_string(s):
    """
    Checks if a string is perfect according to the given rules.

    Args:
        s: The input string (must be lowercase).

    Returns:
        True if the string is perfect, False otherwise.
    """
    if not s:
        return False  # Empty string is not perfect

    if not is_consonant(s[0]):
        return False  # String must start with a consonant

    consonants = set()
    expected_consonant = True  # Track if next char should be consonant
```

```
    for i, char in enumerate(s):
        if expected_consonant:
            if not is_consonant(char):
                return False  # Expected consonant, found vowel
            if char in consonants:
                return False  # Consonant repeated
            consonants.add(char)
            expected_consonant = False  # Next char should be vowel
        else:
            if is_consonant(char):
                return False # Vowel cannot be followed by consonant.
            # Vowels can repeat consecutively, so no need to change
expected_consonant here.
    return True

if __name__ == "__main__":
    # Get input string
    input_string = input()

    # Check if the string is perfect and print the result
    if is_perfect_string(input_string):
        print("True")
    else:
        print("False")
```

Status : Partially correct **Marks : 5/10**

2. Problem Statement

Emily is a data analyst working for a company that collects feedback from customers in the form of text messages. As part of her data validation tasks, Emily needs to perform two operations on each message:

Calculate the sum of all the digits mentioned in the message.If the sum of the digits is greater than 9, check whether the sum forms a palindrome number.

Your task is to help Emily automate this process by writing a program that extracts all digits from a given message, calculates their sum, and checks

if the sum is a palindrome if it is greater than 9.

Input Format

The input consists of a string s, representing the customer message, which may contain letters, digits, spaces, and other characters.

Output Format

The output prints an integer representing the sum of all digits in the string, followed by a space.

If the sum is greater than 9, print "Palindrome" if the sum is a palindrome, otherwise print "Not palindrome".

If the sum is less than or equal to 9, no palindrome check is required.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 12 books 4 pen
Output: 7

Answer

```
def is_palindrome(n):
    return str(n) == str(n)[::-1]

def process_message(s):

    digit_sum = sum(int(char) for char in s if char.isdigit())

    print(digit_sum, end=' ')

    if digit_sum > 9:
        if is_palindrome(digit_sum):
            print("Palindrome")
        else:
```

```
        print("Not palindrome")
    else:
        print()

s = input()
process_message(s)
```

Status : Correct ***Marks : 10/10***

3. Problem Statement

You have two strings str1 and str2, both of equal length.

Write a Python program to concatenate the two strings such that the first character of str1 is followed by the first character of str2, the second character of str1 is followed by the second character of str2, and so on.

For example, if str1 is "abc" and str2 is "def", the output should be "adbecf".

Input Format

The input consists of two strings in each line.

Output Format

The output displays the concatenated string in the mentioned format.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: abc
def
Output: adbecf

Answer

```
def concatenate_strings(str1, str2):
    """
```

```
    Concatenates two strings, interleaving their characters.

    Args:
        str1: The first string.
        str2: The second string.

    Returns:
        The concatenated string.  Returns an empty string if
        the input strings are not of equal length.
    """
    if len(str1) != len(str2):
        return ""

    result = ""
    for i in range(len(str1)):
        result += str1[i] + str2[i]
    return result

if __name__ == "__main__":

    str1 = input()
    str2 = input()


    concatenated_string = concatenate_strings(str1, str2)

    print(concatenated_string)
```

Status : Correct **Marks : 10/10**