Rajalakshmi Engineering College

Name: NEIL DANIEL A
Email: 240701356@rajalakshmi.edu.in
Roll no: 240701356
Phone: 8925059757
Branch: REC
Department: I CSE FD
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(c):
    return c.isalpha() and c not in 'aeiou'

def is_perfect_string(s):
    if len(s) == 0 or not is_consonant(s[0]):
        return False

    consonant_seen = set()
    i = 0
    length = len(s)

    while i < length:
        current = s[i]

        if is_consonant(current):
            if current in consonant_seen:
```

```
            return False  # Consonant appears more than once
        consonant_seen.add(current)

        if i > 0 and is_consonant(s[i - 1]):
            return False  # Two consonants in a row
    else:  # current is a vowel
        if i > 0 and is_consonant(s[i - 1]):
            return False  # A vowel immediately follows a consonant
        while i < length and not is_consonant(s[i]):
            i += 1
        continue  # Skip to the next iteration
    i += 1

  return True

# Input
input_string = input().strip()

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

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

2. Problem Statement

Raj wants to write a program that takes a list of strings as input and returns the longest word in the list. If there are multiple words with the same length, the program should return the first one encountered.

Help Raj in his task.

Input Format

The input consists of a single line of space-separated strings.

Output Format

The output prints a string representing the longest word in the given list.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: cat dog elephant lion tiger giraffe
Output: elephant

Answer

```
def longest_word(s):
    words = s.split()
    longest = ""
    for word in words:
        if len(word) > len(longest):
            longest = word
    return longest
```

```
input_string = input().strip()

print(longest_word(input_string))
```

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

3. Problem Statement

A company is creating email accounts for its new employees. They want to use a naming convention for email addresses that consists of the first letter of the employee's first name, followed by their last name, followed by @company.com.

The company also has a separate email domain for administrative employees.

Write a program that prompts the user for their first name, last name, role, and company and then generates their email address using the appropriate naming convention based on their role. This is demonstrated in the below examples.

Note:

The generated email address should consist of the first letter of the first name, the last name in lowercase, and a suffix based on the role and company, all in lowercase.

Input Format

The first line of input consists of the first name of an employee as a string.

The second line consists of the last name of an employee as a string.

The third line consists of the role of the employee as a string.

The last line consists of the company name as a string.

Output Format

The output consists of a single line containing the generated email address for

the employee, following the specified naming convention.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: John
Smith
admin
iamNeo
Output: jsmith@admin.iamneo.com

Answer

```
first_name = input().strip()

last_name = input().strip()

role = input().strip()

company = input().strip()


email_prefix = first_name[0].lower() + last_name.lower()

if role.lower() == "admin":

    email_suffix = f"@{role.lower()}.{company.lower()}.com"

else:

    email_suffix = f"@{company.lower()}.com"


email_address = email_prefix + email_suffix

print(email_address)
```

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