Rajalakshmi Engineering College

Name: CHENTHAN AMUTHAN.D
Email: 240701090@rajalakshmi.edu.in
Roll no:
Phone: null
Branch: REC
Department: I CSE FA
Batch: 2028
Degree: B.E - CSE

Scan to verify results

NeoColab_REC_CS23221_Python Programming

REC_Python_Week 4_CY

Attempt : 1
Total Mark : 40
Marks Obtained : 40

Section 1 : Coding

1. Problem Statement

You are tasked with designing a shipping cost calculator program that calculates the shipping cost for packages based on their weight and destination. The program utilizes different shipping rates for domestic, international, and remote destinations. The rates for each destination type are provided as global constants.

Constant Values:

DOMESTIC_RATE = 5.0

INTERNATIONAL_RATE = 10.0

REMOTE_RATE = 15.0

Function Signature: calculate_shipping(weight, destination)

Formula: shipping cost = weight * destination rate

Input Format

The first line of the input consists of a float representing the weight of the package.

The second line consists of a string representing the destinations(Domestic or International or Remote).

Output Format

The program outputs any one of the following:

1. If the input is valid and the destination is recognized, the output should consist of a single line stating the calculated shipping cost for the given weight and destination in the format: "Shipping cost to [destination] for a [weight] kg package: $[calculated cost]" with two decimal places.
2. If the input weight is not a positive float, print "Invalid weight. Weight must be greater than 0."
3. If the input destination is not one of the valid options, print "Invalid destination."

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 5.5
Domestic
Output: Shipping cost to Domestic for a 5.5 kg package: $27.50

Answer

#

```
DOMESTIC_RATE = 5.0
INTERNATIONAL_RATE = 10.0
REMOTE_RATE = 15.0

def calculate_shipping(weight, destination):
    if weight <= 0:
        print("Invalid weight. Weight must be greater than 0.")
```

```
        return None
    elif destination == "Domestic":
        rate = DOMESTIC_RATE
    elif destination == "International":
        rate = INTERNATIONAL_RATE
    elif destination == "Remote":
        rate = REMOTE_RATE
    else:
        print("Invalid destination.")
        return None

    cost = weight * rate
    return cost

weight = float(input())
destination = input().strip()
shipping_cost = calculate_shipping(weight, destination)


if shipping_cost is not None:
    print(f"Shipping cost to {destination} for a {weight} kg package:
${shipping_cost:.2f}")
```

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

2. Problem Statement

Amrita is developing a password strength checker for her website. She wants the checker to consider the length and the diversity of characters used in the password. A strong password should be long and include a mix of character types: uppercase, lowercase, digits, and special symbols.

She also wants the feedback to be user-friendly, so she wants to include the actual password in the output. Help Amrita finish this password checker using Python's built-in string methods.

Character Types Considered:

Lowercase letters (a-z)Uppercase letters (A-Z)Digits (0-9)Special characters (from string.punctuation, e.g. @, !, #, $)

Input Format

The input consists of a single string representing the user's password.

Output Format

The program prints the strength of the password in this format:

If the password length < 6 characters or fewer than 2 of the 4 character types, the output prints "<password> is Weak"

If password length ≥ 6 and at least 2 different character types, the output prints "<password> is Moderate"

If Password length ≥ 10 and all 4 character types present, the output prints "<password> is Strong"

Refer to the sample output for formatting specifications.

Sample Test Case

Input: password123

Output: password123 is Moderate

Answer

```
import string

def check_password_strength(password):
    length = len(password)
    has_lower = any(c.islower() for c in password)
    has_upper = any(c.isupper() for c in password)
    has_digit = any(c.isdigit() for c in password)
    has_special = any(c in string.punctuation for c in password)

    if length >= 10 and all([has_lower, has_upper, has_digit, has_special]):
        return f"{password} is Strong"
    elif length >= 6 and sum([has_lower, has_upper, has_digit, has_special]) >= 2:
        return f"{password} is Moderate"
    else:
        return f"{password} is Weak"
```

```
# Sample Inputs
s=input()
print(check_password_strength(s))
```

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

3. Problem Statement

Meena is analyzing a list of integers and needs to count how many numbers in the list are even and how many are odd. She decides to use lambda functions to filter the even and odd numbers from the list.

Write a program that takes a list of integers, counts the number of even and odd numbers using lambda functions, and prints the results.

Input Format

The first line contains an integer n, representing the number of integers in the list.

The second line contains n space-separated integers.

Output Format

The first line of output prints an integer representing the count of even numbers.

The second line of output prints an integer representing the count of odd numbers.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 7
12 34 56 78 98 65 23
Output: 5
2

Answer

```
# You are using Python
n = int(input())
numbers = list(map(int, input().split()))

even_count = len(list(filter(lambda x: x % 2 == 0, numbers)))
odd_count = len(list(filter(lambda x: x % 2 != 0, numbers)))

print(even_count)
print(odd_count)
```

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

4. Problem Statement

Develop a text analysis tool that needs to count the occurrences of a specific substring within a given text string.

Write a function count_substrings(text, substring) that takes two inputs: the text string and the substring to be counted. The function should count how many times the substring appears in the text string and return the count.

Function Signature: count_substrings(text, substring)

Input Format

The first line of the input consists of a string representing the text.

The second line consists of a string representing the substring.

Output Format

The output should display a single line of output containing the count of occurrences of the substring in the text string.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: programming is fun and programming is cool
programming

Output: The substring 'programming' appears 2 times in the text.

Answer

```
# You are using Python
def count_substrings(text, substring):
    count = text.count(substring)
    print(f"The substring '{substring}' appears {count} times in the text.")

# Sample Inputs
text = input().strip()
substring = input().strip()
count_substrings(text, substring)
```

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