

Rajalakshmi Engineering College

Name: Nitin Aakash
Email: 240701370@rajalakshmi.edu.in
Roll no: 240701370
Phone: 9498349045
Branch: REC
Department: I CSE FD
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

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

```
def check_password_strength(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((not c.isalnum()) for c in password)

    character_types_count = sum([has_lower, has_upper, has_digit, has_special])
    length = len(password)

    if length < 6 or character_types_count < 2:
        strength = "Weak"
    elif length >= 10 and character_types_count == 4:
        strength = "Strong"
    else:
        strength = "Moderate"
```

```
print(f"{password} is {strength}")
if __name__ == "__main__":
    pwd = input()
    check_password_strength(pwd)
```

Status : Correct

Marks : 10/10

2. 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 not isinstance(weight, float) or weight <= 0:
```

```
        print("Invalid weight. Weight must be greater than 0.")
```

```
        return None
```

```
    if destination == "Domestic":
```

```
        rate = DOMESTIC_RATE
```

```
    elif destination == "International":
```

```
        rate = INTERNATIONAL_RATE
```

```
    elif destination == "Remote":
```

```
        rate = REMOTE_RATE
```

```
    else:
```

```
        print("Invalid destination.")
```

```
        return None
```

```
    return weight * rate
```

```
try:
```

```
    weight_input = input().strip()
```

```
    destination = input().strip()
```

```
    weight = float(weight_input)
```

```
shipping_cost = calculate_shipping(weight, destination)
except ValueError:
    print("Invalid weight. Weight must be greater than 0.")
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

3. Problem Statement

Imagine you are tasked with developing a function for calculating the total cost of an item after applying a sales tax. The sales tax rate is equal to 0.08 and it is defined as a global variable.

The function should accept the cost of the item as a parameter, calculate the tax amount, and return the total cost.

Additionally, the program should display the item cost, sales tax rate, and total cost to the user.

Function Signature: `total_cost(item_cost)`

Input Format

The input consists of a single line containing a positive floating-point number representing the cost of the item.

Output Format

The output consists of three lines:

"Item Cost:" followed by the cost of the item formatted to two decimal places.

"Sales Tax Rate:" followed by the sales tax rate in percentage.

"Total Cost:" followed by the calculated total cost after applying the sales tax, formatted to two decimal places.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 50.00

Output: Item Cost: \$50.00

Sales Tax Rate: 8.0%

Total Cost: \$54.00

Answer

```
#
```

```
SALES_TAX_RATE = 0.08
```

```
def total_cost(item_cost):
```

```
    tax_amount = item_cost * SALES_TAX_RATE
```

```
    total = item_cost + tax_amount
```

```
    return total
```

```
if __name__ == "__main__":
```

```
    item_cost = float(input().strip())
```

```
    total = total_cost(item_cost)
```

```
total_cost = total_cost(item_cost)
```

```
print(f"Item Cost: ${item_cost:.2f}")
```

```
print(f"Sales Tax Rate: {SALES_TAX_RATE * 100}%")
```

```
print(f"Total Cost: ${total_cost:.2f}")
```

Status : Correct

Marks : 10/10

4. Problem Statement

Arjun is working on a mathematical tool to manipulate lists of numbers. He needs a program that reads a list of integers and generates two lists: one containing the squares of the input numbers, and another containing the cubes. Arjun wants to use lambda functions for both tasks.

Write a program that computes the square and cube of each number in the input list using lambda functions.

Input Format

The input consists of a single line of space-separated integers representing the list of input numbers.

Output Format

The first line contains a list of the squared values of the input numbers.

The second line contains a list of the cubed values of the input numbers.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 1 2 3

Output: [1, 4, 9]
[1, 8, 27]

Answer

```
input_numbers = input().strip().split()
numbers = list(map(int, input_numbers))
squares = list(map(lambda x: x ** 2, numbers))
cubes = list(map(lambda x: x ** 3, numbers))
print(squares)
print(cubes)
```

Status : Correct

Marks : 10/10