Rajalakshmi Engineering College

Name: Krishna Kumar
Email: 240701622@rajalakshmi.edu.in
Roll no:
Phone: null
Branch: REC
Department: I CSE FF
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

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

```
# You are using Python
numbers = list(map(int,input().split()))
squares = list(map(lambda x:x**2,numbers))
cubes = list(map(lambda x:x**3, numbers))
print(squares, cubes)
```

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

```
# You are using Python
import string

def check_password_strength(password):
    # Define character sets
    lowercase = string.ascii_lowercase
    uppercase = string.ascii_uppercase
    digits = string.digits
    special_chars = string.punctuation

    # Initialize flags for character types
    has_lower = has_upper = has_digit = has_special = False

    # Check for the presence of each character type
    for char in password:
        if char.islower():
            has_lower = True
        elif char.isupper():
            has_upper = True
        elif char.isdigit():
            has_digit = True
        elif char in special_chars:
            has_special = True

    # Determine password strength
    if len(password) >= 10 and all([has_lower, has_upper, has_digit, has_special]):
        return f"{password} is Strong"
    elif len(password) >= 6 and (has_lower + has_upper + has_digit + has_special)
>= 2:
        return f"{password} is Moderate"
    else:
        return f"{password} is Weak"

# Example usage
password = input()
print(check_password_strength(password))
```

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

3. 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

```
#

# Constants for shipping rates
DOMESTIC_RATE = 5.0
INTERNATIONAL_RATE = 10.0
REMOTE_RATE = 15.0

# Function to calculate shipping cost
def calculate_shipping(weight, destination):
    # Check if the weight is valid (positive float)
    if weight <= 0 or not isinstance(weight, (float, int)):
        return None, "Invalid weight. Weight must be greater than 0."

    # Dictionary to map destination to rate
    destination_rates = {
        "Domestic": DOMESTIC_RATE,
        "International": INTERNATIONAL_RATE,
        "Remote": REMOTE_RATE
    }

    # Check if the destination is valid
    if destination not in destination_rates:
        return None, "Invalid destination."

    # Calculate the shipping cost
    cost = weight * destination_rates[destination]

    # Return the calculated cost
    return cost, None

# Input handling
try:
    weight = float(input())  # Input for weight
    destination = input().strip()  # Input for destination

    # Calculate shipping cost
    shipping_cost, error_message = calculate_shipping(weight, destination)

    # Output the result
    if error_message:
        print(error_message)
except ValueError:
    print("Invalid input. Please enter valid numerical values for weight.")

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***

4. Problem Statement

Create a program for a mathematics competition where participants need to find the smallest positive divisor of a given integer n. Your program should efficiently determine this divisor using the min() function and display the result.

Input Format

The input consists of a single positive integer n, representing the number for which the smallest positive divisor needs to be found.

Output Format

The output prints the smallest positive divisor of the input integer in the format: "The smallest positive divisor of [n] is: [smallest divisor]".

Refer to the sample output for the exact format.

Sample Test Case

Input: 24

Output: The smallest positive divisor of 24 is: 2

Answer

```
# You are using Python
def smallest_divisor(n):
   divisors = [i for i in range(2,n+1)if n % i==0]
   return min(divisors)

n = int(input())
divisor=smallest_divisor(n)
print(f"The smallest positive divisor of{n} is: {divisor}")
```

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