Rajalakshmi Engineering College

Name: Giridharan A
Email: 241501056@rajalakshmi.edu.in
Roll no:
Phone: 9042220911
Branch: REC
Department: I AIML AD
Batch: 2028
Degree: B.E - AI & ML

Scan to verify results

NeoColab_REC_CS23221_Python Programming

REC_Python_Week 6_CY

Attempt : 1
Total Mark : 40
Marks Obtained : 40

Section 1 : Coding

1. Problem Statement

Write a program to obtain the start time and end time for the stage event show. If the user enters a different format other than specified, an exception occurs and the program is interrupted. To avoid that, handle the exception and prompt the user to enter the right format as specified.

Start time and end time should be in the format 'YYYY-MM-DD HH:MM:SS'If the input is in the above format, print the start time and end time.If the input does not follow the above format, print "Event time is not in the format "

Input Format

The first line of input consists of the start time of the event.

The second line of the input consists of the end time of the event.

Output Format

If the input is in the given format, print the start time and end time.

If the input does not follow the given format, print "Event time is not in the format".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 2022-01-12 06:10:00
2022-02-12 10:10:12

Output: 2022-01-12 06:10:00
2022-02-12 10:10:12

Answer

```
# You are using Python
from datetime import datetime

def validate_time_format(time_str):
    try:
        datetime.strptime(time_str, '%Y-%m-%d %H:%M:%S')
        return True
    except ValueError:
        return False

def main():
    start_time = input().strip()
    end_time = input().strip()

    if validate_time_format(start_time) and validate_time_format(end_time):
        print(f"{start_time} {end_time}")
    else:
        print("Event time is not in the format")

if __name__ == "__main__":
    main()
```

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

2. Problem Statement

Implement a program that checks whether a set of three input values can form the sides of a valid triangle. The program defines a function is_valid_triangle that takes three side lengths as arguments and raises a ValueError if any side length is not a positive value. It then checks whether the sum of any two sides is greater than the third side to determine the validity of the triangle.

Input Format

The first line of input consists of an integer A, representing side1.

The second line of input consists of an integer B, representing side2.

The third line of input consists of an integer C, representing side3.

Output Format

The output prints either "It's a valid triangle" if the input side lengths form a valid triangle,

or "It's not a valid triangle" if they do not.

If there is a ValueError, it should print "ValueError: <error_message>".

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 3
4
5

Output: It's a valid triangle

Answer

```
# You are using Python
def is_valid_triangle(side1, side2, side3):
    # Check if any side is non-positive
    if side1 <= 0 or side2 <= 0 or side3 <= 0:
        raise ValueError("Side lengths must be positive")

    # Check the triangle inequality theorem
    if side1 + side2 > side3 and side1 + side3 > side2 and side2 + side3 > side1:
        return True
    return False

def main():
    try:
        # Read the three sides from the user
        side1 = int(input().strip())
        side2 = int(input().strip())
        side3 = int(input().strip())

        # Check if the sides form a valid triangle
        if is_valid_triangle(side1, side2, side3):
            print("It's a valid triangle")
        else:
            print("It's not a valid triangle")

    except ValueError as e:
        # Handle ValueError for invalid side lengths or other issues
        print(f"ValueError: {e}")

if __name__ == "__main__":
    main()
```

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

3. Problem Statement

Bob, a data analyst, requires a program to automate the process of analyzing character frequency in a given text. This program should allow the user to input a string, calculate the frequency of each character within the text, save these character frequencies to a file named "char_frequency.txt," and display the results.

Input Format

The input consists of the string.

Output Format

The first line prints "Character Frequencies:".

The following lines print the character frequency in the format: "X: Y" where X is the character and Y is the count.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: aaabbbccc

Output: Character Frequencies:
a: 3
b: 3
c: 3

Answer

```
from collections import Counter

def calculate_frequency(input_string):
    return Counter(input_string)

def save_frequencies_to_file(frequencies):
    with open("char_frequency.txt", "w") as file:
        for char, count in frequencies.items():
            file.write(f"{char}: {count}\n")

def main():
    input_string = input().strip()

    frequencies = calculate_frequency(input_string)

    print("Character Frequencies:")
    for char, count in frequencies.items():
        print(f"{char}: {count}")

    save_frequencies_to_file(frequencies)
```

```
if __name__ == "__main__":
    main()
```

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

4. Problem Statement

Alex is creating an account and needs to set up a password. The program prompts Alex to enter their name, mobile number, chosen username, and desired password. Password validation criteria include:

Length between 10 and 20 characters.At least one digit.At least one special character from !@#$%^&amp;* set. Display "Valid Password" if criteria are met; otherwise, raise an exception with an appropriate error message.

Input Format

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

The second line of the input consists of the mobile number as a string.

The third line of the input consists of the username as a string.

The fourth line of the input consists of the password as a string.

Output Format

If the password is valid (meets all the criteria), it will print "Valid Password"

If the password is weak (fails any one or more criteria), it will print an error message accordingly.

Refer to the sample outputs for the formatting specifications.

Sample Test Case

```
Input: John
9874563210
john
```

john1#nhoj
Output: Valid Password

Answer

```
import re

name = input()
mobile_number = input()
username = input()
password = input()

if not (10 <= len(password) <= 20):
    print("Should be a minimum of 10 characters and a maximum of 20 
characters")
elif not any(char.isdigit() for char in password):
    print("Should contain at least one digit")
elif not any(char in "!@#$%^&*" for char in password):
    print("It should contain at least one special character")
else:
    print("Valid Password")
```

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