Rajalakshmi Engineering College

Name: Padma Priya D
Email: 240701377@rajalakshmi.edu.in
Roll no: 240701377
Phone: 8668123104
Branch: REC
Department: I CSE FD
Batch: 2028
Degree: B.E - CSE

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

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

```
# You are using Python
from collections import OrderedDict

# Read input string
input_str = input()

# Use OrderedDict to maintain insertion order
freq = OrderedDict()

# Count frequency of each character
for char in input_str:
    freq[char] = freq.get(char, 0) + 1

# Open file for writing
with open("char_frequency.txt", "w") as file:
    # Print and write output
    print("Character Frequencies:")
    file.write("Character Frequencies:\n")
    for char, count in freq.items():
        print(f"{char}: {count}")
        file.write(f"{char}: {count}\n")
```

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

2. Problem Statement

Write a program to read the Register Number and Mobile Number of a student. Create user-defined exception and handle the following:

If the Register Number does not contain exactly 9 characters in the specified format(2 numbers followed by 3 characters followed by 4 numbers) or if the Mobile Number does not contain exactly 10 characters, throw an IllegalArgumentException. If the Mobile Number contains any character other than a digit, raise a NumberFormatException.If the Register Number contains any character other than digits and alphabets, throw a NoSuchElementException.If they are valid, print the message ‘valid’ or else print an Invalid message.

Input Format

The first line of the input consists of a string representing the Register number.

The second line of the input consists of a string representing the Mobile number.

Output Format

The output should display any one of the following messages:

If both numbers are valid, print "Valid".

If an exception is raised, print "Invalid with exception message: ", followed by the specific exception message.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 19ABC1001
9949596920

Output: Valid

Answer

```
# You are using Python
class IllegalArgumentException(Exception):
    pass
class NoSuchElementException(Exception):
```

```
    pass
reg_no = input()
mobile_no = input()
try:
    if len(reg_no) != 9:
        raise IllegalArgumentException("Register Number should have exactly 9
characters.")
    if not (reg_no[:2].isdigit() and reg_no[2:5].isalpha() and reg_no[5:].isdigit()):
        raise IllegalArgumentException("Register Number should have the format: 2
numbers, 3 characters, and 4 numbers.")
    if not reg_no.isalnum():
        raise NoSuchElementException("Register Number should only contain
letters and digits.")
    if len(mobile_no) != 10:
        raise IllegalArgumentException("Mobile Number should have exactly 10
characters.")
    if not mobile_no.isdigit():
        raise ValueError("Mobile Number should only contain digits.")
    print("Valid")
except (IllegalArgumentException, NoSuchElementException, ValueError) as e:
    print(f"Invalid with exception message: {e}")
```

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

3. Problem Statement

In the enchanted realm of Academia, you, the Academic Alchemist, are bestowed with a magical quill and a parchment to weave the grades of aspiring students into a tapestry of academic brilliance.

The mission is to craft a Python program that empowers faculty members to enter student grades for any two subjects, stores these magical grades in a mystical file, and then, with a wave of your virtual wand, calculates the GPA to unveil the true essence of academic achievement.

Input Format

The input format is a string representing the student's name, any two subjects, and corresponding grades.

After entering grades, they can type 'done' when prompted for the student's

name.

Output Format

The output should display the (average of grades) calculated GPA with a precision of two decimal places.

The magical grades will be saved in a mystical file named "magical_grades.txt".

Refer to the sample output for format specifications.

Sample Test Case

Input: Alice
Math
95
English
88
done

Output: 91.50

Answer

```
# You are using Python
with open("magical_grades.txt", "w") as file:
    while True:
        name = input()
        if name.lower() == "done":
            break
        subject1 = input()
        grade1 = int(input())
        subject2 = input()
        grade2 = int(input())
        file.write(f"{name}: {subject1} - {grade1}, {subject2} - {grade2}\n")
        gpa = (grade1 + grade2) / 2
        print(f"{gpa:.2f}")
```

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

4. 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_event_time():
    try:
        start_time = input()
        end_time = input()
        start = datetime.strptime(start_time, '%Y-%m-%d %H:%M:%S')
```

```
        end = datetime.strptime(end_time, '%Y-%m-%d %H:%M:%S')
        print(start_time)
        print(end_time)
    except ValueError:
        print("Event time is not in the format")
validate_event_time()
```

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