Rajalakshmi Engineering College

Name: Alvin .B
Email: 240701034@rajalakshmi.edu.in
Roll no: 240701034
Phone: 9677000577
Branch: REC
Department: I CSE FA
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

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().strip()
        if name.lower() == "done":
            break

        subject1 = input().strip()
        grade1 = float(input())
        subject2 = input().strip()
        grade2 = float(input())

        file.write(f"{name} {subject1} {grade1} {subject2} {grade2}\n")

        gpa = (grade1 + grade2) / 2
        print(f"{gpa:.2f}")
```

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

2. Problem Statement

Alice is developing a program called "Name Sorter" that helps users organize and sort names alphabetically.

The program takes names as input from the user, saves them in a file, and then displays the names in sorted order.

File Name: sorted_names.txt.

Input Format

The input consists of multiple lines, each containing a name represented as a string.

To end the input and proceed with sorting, the user can enter 'q'.

Output Format

The output displays the names in alphabetical order, each name on a new line.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: Alice Smith
John Doe
Emma Johnson
q
Output: Alice Smith
Emma Johnson
John Doe

Answer

```
# You are using Python

with open("sorted_names.txt", "w") as file:
    while True:
        name = input().strip()
        if name.lower() == 'q':
```

```
        break
    file.write(name + "\n")

with open("sorted_names.txt", "r") as file:
    names = file.readlines()

names = [name.strip() for name in names]
names.sort()

for name in names:
    print(name)
```

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

3. 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

```
from datetime import datetime

try:
    start_time_str = input().strip()
    end_time_str = input().strip()

    start_time = datetime.strptime(start_time_str, '%Y-%m-%d %H:%M:%S')
    end_time = datetime.strptime(end_time_str, '%Y-%m-%d %H:%M:%S')

    print(start_time_str, end_time_str)

except Exception:
    print("Event time is not in the format")
```

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

4. 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

text = input()

freq = OrderedDict()
for ch in text:
    freq[ch] = freq.get(ch, 0) + 1

with open("char_frequency.txt", "w") as f:
    f.write("Character Frequencies:\n")
    for ch, count in freq.items():
        f.write(f"{ch}: {count}\n")

print("Character Frequencies:", end=" ")
for ch, count in freq.items():
    print(f"{ch}: {count}", end=" ")
print()
```

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