Rajalakshmi Engineering College

Name: Dhanigivela B
Email: 241501042@rajalakshmi.edu.in
Roll no: 241501042
Phone: 8870800831
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

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

def check_event_times():
    try:
        start_time = input()
        end_time = input()

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

        print(start_time, end_time)

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

check_event_times()
```

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

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

class NumberFormatException(Exception):
```

```python
        pass

class NoSuchElementException(Exception):
    pass

def validate_register_number(reg_no):

    if len(reg_no) != 9:
        raise IllegalArgumentException("Register Number should have exactly 9 characters.")

    if not (reg_no[0:2].isdigit() and
            reg_no[2:5].isalpha() and
            reg_no[5:9].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 digits and alphabets.")

def validate_mobile_number(mobile_no):
    if len(mobile_no) != 10:
        raise IllegalArgumentException("Mobile Number should have exactly 10 characters.")

    if not mobile_no.isdigit():
        raise NumberFormatException("Mobile Number should only contain digits.")

def main():
    reg_no = input().strip()
    mobile_no = input().strip()

    try:
        validate_register_number(reg_no)
        validate_mobile_number(mobile_no)
        print("Valid")
    except (IllegalArgumentException, NumberFormatException, NoSuchElementException) as e:
        print(f"Invalid with exception message: {e}")

if __name__ == "__main__":
```

main()

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

```python
# You are using Python
def academic_alchemy():
    with open("magical_grades.txt", "w") as file:
        while True:
            student_name = input().strip()
            if student_name.lower() == "done":
                break

            subject1 = input().strip()
            grade1 = int(input().strip())

            subject2 = input().strip()
            grade2 = int(input().strip())

            if not (0 <= grade1 <= 100 and 0 <= grade2 <= 100):
                print("Invalid grades. Please enter values between 0 and 100.")
                return

            file.write(f"{student_name}, {subject1}: {grade1}, {subject2}: {grade2}\n")

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

academic_alchemy()
```

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

```python
# You are using Python
from collections import Counter

def analyze_character_frequency():
    text = input().strip()

    frequency = Counter(text)

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

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

analyze_character_frequency()
```

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