Rajalakshmi Engineering College

Name: DHARINI BALA MURUGAN .
Email: 241501044@rajalakshmi.edu.in
Roll no: 241501044
Phone: 8754111345
Branch: REC
Department: I AI & ML FA
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 : 36.5

Section 1 : Coding

1. 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 string

def validate_password(password):
    special_characters = "!@#$%^&*"

    # 1. Check for at least one digit first
    if not any(char.isdigit() for char in password):
        raise Exception("Should contain at least one digit")

    # 2. Check for at least one allowed special character
    if not any(char in special_characters for char in password):
        raise Exception("It should contain at least one special character")

    # 3. Check for length constraint
    if not (10 <= len(password) <= 20):
        raise Exception("Should be a minimum of 10 characters and a maximum of
20 characters")

    # 4. Check for whitespace characters
    if any(char.isspace() for char in password):
        raise Exception("Password should not contain any whitespace characters")

    # 5. Ensure all characters are printable (exclude control or non-ASCII
characters)
    if not all(char in string.printable for char in password):
        raise Exception("Password contains invalid characters")

    print("Valid Password")

# Input
name = input()
mobile = input()
username = input()
password = input()

# Validation
try:
    validate_password(password)
except Exception as e:
    print(e)
```

Status : Partially correct ***Marks : 6.5/10***

2. Problem Statement

A shopkeeper is recording the daily sales of an item for N days, where the price of the item remains the same for all days. Write a program to calculate the total sales for each day and save them in a file named sales.txt that can store the data for a maximum of 30 days. Then, read the file and display the total earnings for each day.

Note: Total Earnings for each day = Number of Items sold in that day × Price of the item.

Input Format

The first line of input consists of an integer N, representing the number of days.

The second line of input consists of N space-separated integers representing the number of items sold each day.

The third line of input consists of an integer M, representing the price of the item that is common for all N days.

Output Format

If the number of days entered exceeds 30 (N > 30), the output prints "Exceeding limit!" and terminates.

Otherwise, the code reads the contents of the file and displays the total earnings for each day on separate lines.

Contents of the file: The total earnings for N days, with each day's earnings appearing on a separate line.

Refer to the sample output for the formatting specifications.

Sample Test Case

```
Input: 4
5 10 5 0
20
Output: 100
200
100
0
```

Answer

```
# You are using Python
def main():
    N = int(input())

    if N > 30:
        print("Exceeding limit!")
        return

    items_sold = list(map(int, input().split()))
    M = int(input())  # Price of the item
    # Calculate total earnings per day and write to file
    with open("sales.txt", "w") as file:
        for count in items_sold:
            earning = count * M
            file.write(str(earning) + "\n")

    # Read and display total earnings from the file
    with open("sales.txt", "r") as file:
        for line in file:
            print(line.strip())

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

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

3. 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
names=[]
while True:
    name=input()
    if name.lower()=='q':
        break
    names.append(name)
names.sort()
with open("sorted_names.txt","w") as f:
    for name in names:
        f.write(name+ "\n")
with open("sorted_names.txt","r") as fi:
    content=fi.read()
    print(content)
```

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 get_event_time():
    try:
        start_time = input()
        end_time = input()

        # Try parsing the dates
        start_dt = datetime.strptime(start_time, "%Y-%m-%d %H:%M:%S")
        end_dt = 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")

# Run the function
get_event_time()
```

Status : Correct *Marks : 10/10*