Rajalakshmi Engineering College

Name: Nekhita Sri
Email: 241801184@rajalakshmi.edu.in
Roll no: 241801184
Phone: 8637459907
Branch: REC
Department: I AI & DS FC
Batch: 2028
Degree: B.E - AI & DS

Scan to verify results

NeoColab_REC_CS23221_Python Programming

REC_Python_Week 6_CY

Attempt : 1
Total Mark : 40
Marks Obtained : 37.5

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
s=input()
e=input()
try:
   s=datetime.strptime(s,"%Y-%m-%d %H:%M:%S")
   e=datetime.strptime(e,"%Y-%m-%d %H:%M:%S")
   print(s)
   print(e)
except ValueError:
   print("Event time is not in the format")
```

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

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
n=int(input())
lst=list(map(int,input().split()))
m=int(input())
if n>30 or m>=200:
   print("Exceeding limit!")
else:
   for i in lst:
      print(i*m)
```

Status : *Partially correct* ***Marks : 7.5/10***

3. 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
r=input()
m=input()
def valid(r):
  if r[:2].isdigit() and r[2:5].isalpha() and r[5:].isdigit():
    return 1
try:
  if len(r)!=9:
    raise ValueError("Register Number should have exactly 9 characters.")
  if len(m)!=10:
    raise ValueError("Mobile Number should have exactly 10 characters.")
  if not m.isdigit():
    raise ValueError("Mobile Number should only contain digits.")
  if not r.isalnum():
    raise ValueError("Register Number should only contain alphabets and 
digits")
  if not valid(r):
    raise ValueError("Register Number should have the format: 2 numbers, 3 
characters, and 4 numbers.")
  print("Valid")
except ValueError as e:
  print(f"Invalid with exception message: {e}")
```

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

4. 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
lst=[]
su=0
j=0
while True:
    s=input()
    if s=='done':
        break
    else:
        lst.append(s)
for i in lst:
    if i.isdigit():
        su+=int(i)
        j+=1
avg=su/j
print("{:.2f}".format(avg))
```

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