

Rajalakshmi Engineering College

Name: Karolin Auxilia J
Email: 240701228@rajalakshmi.edu.in
Roll no: 240701228
Phone: 9977228588
Branch: REC
Department: I CSE FC
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

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_datetime(dt_str):
    try:
        datetime.strptime(dt_str, '%Y-%m-%d %H:%M:%S')
        return True
    except ValueError:
        return False
try:
    start_time=input().strip()
    end_time=input().strip()
except Exception:
    print("Event time is not in the format")
if validate_datetime(start_time)and validate_datetime(end_time):
    print(start_time)
    print(end_time)
else:
    print("Event time is not in the format")
```

Status : Correct

Marks : 10/10

2. 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
text=input()
frequency={}
for char in text:
    if char in frequency:
        frequency[char]+=1
    else:
        frequency[char]=1
with open("char frequency.txt", "w")as file:
    file.write("Character Frequencies:\n")
    for char in frequency:
```

```
file.write(f"{char}: {frequency[char]}\n")
print("Character Frequencies:")
for char in frequency:
    print(f"{char}: {frequency[char]}")
```

Status : Correct

Marks : 10/10

3. 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())
if N>30:
    print("Exceeding limit!")
else:
    items_sold=list(map(int,input().split()))
    M=int(input())
    with open("sales.txt", "w")as f:
        for count in items_sold:
            total=count*M
            f.write(str(total)+ "\n")
    with open("sales.txt", "r")as f:
        lines=f.readlines()
        for line in lines:
            print(line.strip())
```

Status : Correct

Marks : 10/10

4. Problem Statement

Implement a program that checks whether a set of three input values can form the sides of a valid triangle. The program defines a function `is_valid_triangle` that takes three side lengths as arguments and raises a `ValueError` if any side length is not a positive value. It then checks whether

the sum of any two sides is greater than the third side to determine the validity of the triangle.

Input Format

The first line of input consists of an integer A, representing side1.

The second line of input consists of an integer B, representing side2.

The third line of input consists of an integer C, representing side3.

Output Format

The output prints either "It's a valid triangle" if the input side lengths form a valid triangle,

or "It's not a valid triangle" if they do not.

If there is a ValueError, it should print "ValueError: <error_message>".

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 3

4

5

Output: It's a valid triangle

Answer

```
# You are using Python
```

```
def is_valid_triangle(a,b,c):
```

```
    if a<=0 or b<=0 or c<=0:
```

```
        raise ValueError("Side lengths must be positive")
```

```
    if a+b>c and a+c>b and b+c>a:
```

```
        return True
```

```
    else:
```

```
        return False
```

```
try:
```

```
    a=int(input())
```

```
    b=int(input())
```

```
c=int(input())
if is_valid_triangle(a,b,c):
    print("It's a valid triangle")
else:
    print("It's not a valid triangle")
except ValueError as e:
    print(f"ValueError: {e}")
```

Status : Correct

Marks : 10/10