

# Rajalakshmi Engineering College

Name: Shivani R J  
Email: 240701500@rajalakshmi.edu.in  
Roll no: 2116240701500  
Phone: 9962571492  
Branch: REC  
Department: I CSE FE  
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

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 tri(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:
```

```
    s1=int(input())
```

```
    s2=int(input())
```

```
    s3=int(input())
```

```
    if(tri(s1,s2,s3)):
```

```
        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**

## 2. 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 !@#\$%^&\* 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#nhøj

Output: Valid Password

### ***Answer***

```
# You are using Python
def val_password(password):
```

```

if not(10<=len(password)<=20):
    raise ValueError("Should be a minimum of 10 characters and a maximum of 20 characters")
if not any(char.isdigit() for char in password):
    raise ValueError("Should contain at least one digit")
if not any(char in "!@#$%^&*" for char in password):
    raise ValueError("It should contain at least one special character")
return True
try:
    name=input()
    no=int(input())
    user_name=input()
    password=input()
    if val_password(password):
        print("Valid Password")
except ValueError as v:
    print(v)

```

**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**

# You are using Python

```
import datetime
```

```
s1=input()
```

```
s2=input()
```

```
try:
```

```
    datetime.datetime.strptime(s1,"%Y-%m-%d %H:%M:%S")
```

```
    datetime.datetime.strptime(s2,"%Y-%m-%d %H:%M:%S")
```

```
    print(s1)
```

```
    print(s2)
```

```
except ValueError:
```

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

**Status : Correct**

**Marks : 10/10**

## **4. 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=="q"):
        break
    names.append(name)

for i in (sorted(names)):
    print(i)
```

**Status :** Correct

**Marks : 10/10**