Rajalakshmi Engineering College

Name: Thamburaj Baskaran
Email: 240701559@rajalakshmi.edu.in
Roll no: 240701559
Phone: 9445306389
Branch: REC
Department: I CSE FF
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

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#nhoj

Output: Valid Password

Answer

```
# You are using Python
class LengthError(Exception):
    pass
class DigitError(Exception):
    pass
class CharError(Exception):
    pass
n=input()
p=int(input())
u=input()
try:
    pas=input().strip()
    d,s=0,0
    for i in pas:
        if i.isdigit():
            d+=1
        if not i.isalnum():
            s+=1
    if len(pas)<10 or len(pas)>20:
```

```
        raise LengthError
    if not any(ch.isdigit() for ch in pas):
        raise DigitError
    if pas.isalnum():
        raise CharError
    print("Valid Password")
except LengthError:
    print("Should be a minimum of 10 characters and a maximum of 20
characters")
except DigitError:
    print("Should contain at least one digit")
except CharError:
    print("It shoud contain at least one special character")
```

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

```
import re
def validate_register_number(reg_no):
    if len(reg_no) != 9:
        raise ValueError("Register Number should have exactly 9 characters.")
    if not re.match(r'^\d{2}[A-Za-z]{3}\d{4}$',reg_no):
        raise ValueError("Register Number should have the format: 2 numbers, 3
characters, and 4 numbers.")
    if not reg_no.isalnum():
        raise Exception("Register Number should only contain digits and
alphabets.")
def validate_mobile_number(mobile_no):
    if len(mobile_no) != 10:
        raise ValueError("Mobile Number should have exactly 10 characters.")
    if not mobile_no.isdigit():
        raise ValueError("Mobile Number should only contain digits.")
def main():
    try:
        reg_no = input().strip()
        mobile_no = input().strip()

        validate_register_number(reg_no)
        validate_mobile_number(mobile_no)

        print("Valid")
    except ValueError as ve:
        print("Invalid with exception message:", ve)
    except Exception as e:
```

```
        print("Invalid with exception message:", e)

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

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

3. 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

```
def is_valid_triangle(a, b, c):
    # Check if all sides are positive
    if a <= 0 or b <= 0 or c <= 0:
        raise ValueError("Side lengths must be positive")

    # Check triangle inequality theorem
    if (a + b > c) and (a + c > b) and (b + c > a):
        return True
    else:
        return False

def main():
    try:
        # Reading input values
        a = int(input().strip())
        b = int(input().strip())
        c = int(input().strip())

        # Check triangle validity
        if is_valid_triangle(a, b, c):
            print("It's a valid triangle")
        else:
            print("It's not a valid triangle")
    except ValueError as ve:
        print("ValueError:Side lengths must be positive")


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

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
with open("sorted_names.txt","w")as f:
    while True:
        e=input()
        if e=='q':
            break
        f.write(e.strip()+'\n')
with open("sorted_names.txt","r")as f:
    l=list(line for line in f)
    l.sort()
    for i in l:
```

```
    print(i,end='')
```

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