Rajalakshmi Engineering College

Name: Mithran VT
Email: 240701312@rajalakshmi.edu.in
Roll no:
Phone: 9952919350
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_PAH

Attempt : 1
Total Mark : 30
Marks Obtained : 21

Section 1 : Coding

1. Problem Statement

Reeta is playing with numbers. Reeta wants to have a file containing a list of numbers, and she needs to find the average of those numbers. Write a program to read the numbers from the file, calculate the average, and display it.

File Name: user_input.txt

Input Format

The input file will contain a single line of space-separated numbers (as a string).

These numbers may be integers or decimals.

Output Format

If all inputs are valid numbers, the output should print: "Average of the numbers is: X.XX" (where X.XX is the computed average rounded to two decimal places)

If the input contains invalid data, print: "Invalid data in the input."

Refer to the sample output for format specifications.

Sample Test Case

Input: 1 2 3 4 5

Output: Average of the numbers is: 3.00

Answer

```
# You are using Python
def save_and_calculate_average(filename):
    try:
        user_input = input().strip()
        with open(filename, 'w') as file:
            file.write(user_input)

        with open(filename, 'r') as file:
            data = file.readline().strip()

        numbers = data.split()
        num_list = []

        for num in numbers:
            try:
                num_list.append(float(num))
            except ValueError:
                print("Invalid data in the input.")
                return

        avg = sum(num_list) / len(num_list)
        print(f"Average of the numbers is: {avg:.2f}")

    except Exception as e:
        print(f"Error: {e}")

save_and_calculate_average("user_input.txt")
```

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

2. Problem Statement

Peter manages a student database and needs a program to add students. For each student, Alex inputs their ID and name. The program checks for duplicate IDs and ensures the database isn't full.

If a duplicate or a full database is detected, an appropriate error message is displayed. Otherwise, the student is added, and a confirmation message is shown. The database has a maximum capacity of 30 students, and each student must have a unique ID.

Input Format

The first line contains an integer n, representing the number of students to be added to the school database.

The next n lines each contain two space-separated values, representing the student's ID (integer) and the student's name (string).

Output Format

The output will depend on the actions performed in the code.

If a student is added to the database, the output will display: "Student with ID [ID number] added to the database."

If there is an exception due to a duplicate student ID, the output will display: "Exception caught. Error: Student ID already exists."

If there is an exception due to the database being full, the output will display: "Exception caught. Error: Student database is full."

Refer to the sample outputs for the formatting specifications.

Sample Test Case

Input: 3
16 Sam
87 Sabari
43 Dani

Output: Student with ID 16 added to the database.
Student with ID 87 added to the database.
Student with ID 43 added to the database.

Answer

```
# You are using Python
class StudentDatabase:
    MAX_CAPACITY = 30

    def __init__(self):
        self.students = {}

    def add_student(self, student_id, student_name):
        if len(self.students) >= self.MAX_CAPACITY:
            print("Exception caught. Error: Student database is full.")
            return

        if student_id in self.students:
            print("Exception caught. Error: Student ID already exists.")
            return

        self.students[student_id] = student_name
        print(f"Student with ID {student_id} added to the database.")

def main():
    student_db = StudentDatabase()

    try:
        n = int(input().strip())
        if n <= 0 or n > StudentDatabase.MAX_CAPACITY:
            print("Error: Number of students must be between 1 and 30.")
            return

        for _ in range(n):
            entry = input().strip().split(maxsplit=1)
```

```
        if len(entry) != 2:
            print("Error: Invalid input format.")
            return

        student_id, student_name = entry

        if not student_id.isdigit() or int(student_id) <= 0:
            print("Error: Student ID must be a positive integer.")
            return

        if len(student_name) < 1 or len(student_name) > 100:
            print("Error: Student name must be between 1 and 100 characters.")
            return

        student_db.add_student(int(student_id), student_name)

    except ValueError:
        print("Error: Invalid input.")

# Run the program
main()
```

Status : Partially correct **Marks : 6/10**

3. Problem Statement

John is a data analyst who often works with text files. He needs a program that can analyze the contents of a text file and count the number of times a specific character appears in the file.

John wants a simple program that allows him to specify a file and a character to count within that file.

Input Format

The first line of input consists of the file's name to be analyzed.

The second line of the input consists of the string they want to write within the file.

The third line of the input consists of a character to count within the file.

Output Format

If the character is found, the output displays "The character 'X' appears {Y} times in the file." where X is the character and Y i the count,

If the character does not appear in the file, the output displays "Character not found."

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: test.txt
This is a test file to check the character count.
e

Output: The character 'e' appears 5 times in the file.

Answer

```
# You are using Python
def analyze_character_count(filename, content, character):
    try:
        if len(content) < 5 or len(content) > 500:
            print("Error: Input string length must be between 5 and 500 characters.")
            return

        if len(character) != 1:
            print("Error: Please provide a single character to count.")
            return

        with open(filename, 'w') as file:
            file.write(content)

        with open(filename, 'r') as file:
            file_content = file.read()

        count = file_content.count(character)
```

```
        if count > 0:
            print(f"The character '{character}' appears {count} times in the file.")
        else:
            print("Character not found in the file.")

    except Exception as e:
        print(f"Error: {e}")

filename = input().strip()
content = input().strip()
character = input().strip()

analyze_character_count(filename, content, character)
```

Status : Partially correct **Marks : 5/10**