Rajalakshmi Engineering College

Name: n.madhu narayanan
Email: 240701295@rajalakshmi.edu.in
Roll no: 240701295
Phone: 8870065218
Branch: REC
Department: I CSE AH
Batch: 2028
Degree: B.E - CSE

Scan to verify results

NeoColab_REC_CS23221_Python Programming

REC_Python_Week 6_COD

Attempt : 1
Total Mark : 50
Marks Obtained : 48.5

Section 1 : Coding

1. Problem Statement

Tara is a content manager who needs to perform case conversions for various pieces of text and save the results in a structured manner.

She requires a program to take a user's input string, save it in a file, and then retrieve and display the string in both upper-case and lower-case versions. Help her achieve this task efficiently.

File Name: text_file.txt

Input Format

The input consists of a single line containing a string provided by the user.

Output Format

The first line displays the original string read from the file in the format: "Original String: {original_string}".

The second line displays the upper-case version of the original string in the format: "Upper-Case String: {upper_case_string}".

The third line displays the lower-case version of the original string in the format: "Lower-Case String: {lower_case_string}".

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: #SpecialSymBoLs1234
Output: Original String: #SpecialSymBoLs1234
Upper-Case String: #SPECIALSYMBOLS1234
Lower-Case String: #specialsymbols1234

Answer

```
# You are using Python
def main():
    # Get user input
    user_input = input().strip()

    # Save the input string to a file
    with open("text_file.txt", "w") as file:
        file.write(user_input)

    # Read the string back from the file
    with open("text_file.txt", "r") as file:
        original_string = file.read().strip()

    # Convert to upper-case and lower-case
    upper_case_string = original_string.upper()
    lower_case_string = original_string.lower()

    # Print the results
    print(f"Original String: {original_string}")
    print(f"Upper-Case String: {upper_case_string}")
    print(f"Lower-Case String: {lower_case_string}")
```

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

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

2. Problem Statement

Sophie enjoys playing with words and wants to count the number of words in a sentence. She inputs a sentence, saves it to a file, and then reads it from the file to count the words.

Write a program to determine the number of words in the input sentence.

File Name: sentence_file.txt

Input Format

The input consists of a single line of text containing words separated by spaces.

Output Format

The output displays the count of words in the sentence.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: Four Words In This Sentence
Output: 5

Answer

```
# You are using Python
def main():
    # Get user input
    user_input = input().strip()

    # Save the input sentence to a file
    with open("sentence_file.txt", "w") as file:
```

```
    file.write(user_input)

# Read the sentence back from the file
with open("sentence_file.txt", "r") as file:
    sentence = file.read().strip()

# Count the number of words in the sentence
if sentence:
    word_count = len(sentence.split())
else:
    word_count = 0

# Print the word count
print(word_count)

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

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

3. Problem Statement

A retail store requires a program to calculate the total cost of purchasing a product based on its price and quantity. The program performs validation to ensure valid inputs and handles specific error conditions using exceptions:

Price Validation: If the price is zero or less, raise a ValueError with the message: "Invalid Price".Quantity Validation: If the quantity is zero or less, raise a ValueError with the message: "Invalid Quantity".Cost Threshold: If the total cost exceeds 1000, raise RuntimeError with the message: "Excessive Cost".

Input Format

The first line of input consists of a double value, representing the price of a product.

The second line consists of an integer, representing the quantity of the product.

Output Format

If the calculation is successful, print the total cost rounded to one decimal place.

If the price is zero or less prints "Invalid Price".

If the quantity is zero or less prints "Invalid Quantity".

If the total cost exceeds 1000, prints "Excessive Cost".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 20.0
5
Output: 100.0

Answer

```
# You are using Python
def main():
    try:
        # Read the price and quantity from input
        price = float(input().strip())
        quantity = int(input().strip())

        # Validate price
        if price <= 0:
            raise ValueError("Invalid Price")

        # Validate quantity
        if quantity <= 0:
            raise ValueError("Invalid Quantity")

        # Calculate total cost
        total_cost = price * quantity

        # Check for excessive cost
        if total_cost > 1000:
            raise RuntimeError("Excessive Cost")

        # Print the total cost rounded to one decimal place
```

```
        print(f"{total_cost:.1f}")

    except ValueError as ve:
        print(ve)
    except RuntimeError as re:
        print(re)

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

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

4. Problem Statement

Write a program that calculates the average of a list of integers. The program prompts the user to enter the length of the list (n) and each element of the list. It performs error handling to ensure that the length of the list is a non-negative integer and that each input element is a numeric value.

Input Format

The first line of the input is an integer n, representing the length of the list as a positive integer.

The second line of the input consists of an element of the list as an integer, separated by a new line.

Output Format

If the length of the list is not a positive integer or zero, the output displays "Error: The length of the list must be a non-negative integer."

If a non-numeric value is entered for the length of the list, the output displays "Error: You must enter a numeric value."

If a non-numeric value is entered for a list element, the output displays "Error: You must enter a numeric value."

If the inputs are valid, the program calculates and prints the average of the provided list of integers with two decimal places: "The average is: [average]".

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: -2
1
2
Output: Error: The length of the list must be a non-negative integer.

Answer

```
def main():
    try:
        n_input = input()
        n = int(n_input)
        if n <= 0 or n > 20:
            print("Error: The length of the list must be a non-negative integer.")
            return
    except ValueError:
        print("Error: You must enter a numeric value.")
        return

    numbers = []
    for _ in range(n):
        try:
            num_input = input()
            num = int(num_input)
            numbers.append(num)
        except ValueError:
            print("Error: You must enter a numeric value.")
            return

    average = sum(numbers) / n
    print(f"The average is: {average:.2f}")
main()
```

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

5. Problem Statement

In a voting system, a person must be at least 18 years old to be eligible to vote. If a user enters an age below 18, the system should raise a user-defined exception indicating that they are not eligible to vote.

Input Format

The input contains a positive integer representing age.

Output Format

If the age is less than 18, the output displays "Not eligible to vote".

Otherwise, the output displays "Eligible to vote".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 18
Output: Eligible to vote

Answer

```
class NotEligibleToVote(Exception):
    pass

def check_voting_eligibility(age):
    if age < 18:
        raise NotEligibleToVote("Not eligible to vote")
    else:
        return "Eligible to vote"

def main():
    try:

        age = int(input().strip())

        if age < 1 or age > 100:
            print("Invalid age. Age must be between 1 and 100.")
```

```
        return

    result = check_voting_eligibility(age)
    print(result)

  except NotEligibleToVote as e:
    print(e)
  except ValueError:
    print("Invalid input. Please enter a positive integer.")

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

Status : Partially correct ***Marks : 8.5/10***