Green University of Bangladesh
Department of Computer Science and Engineering (CSE)
Faculty of Sciences and Engineering
Semester: Fall , Year:2024), B.Sc. in CSE (Day)
Lab Report NO 01
Course Title: Machine Learning Lab
Course Code: CSE 412 Section: 213_D1

Lab Experiment Name:

1. Write a Python program that calculates the frequency of each character in a given input string. Your program should count how many times each character appears in the input and display the results.

2. You are given the task of calculating the total tax amount for a person's income based on the following tax brackets: - Income up to $10,000 is taxed at a rate of 5%. - Income from $10,001 to $50,000 is taxed at a rate of 10% - Income from $50,001 to $100,000 is taxed at a rate of 20%. - Income over $100,000 is taxed at a rate of 30%.Your python program should take the user's income as input and calculate the total tax amount they owe based on the provided tax brackets.

Student Details

	Name	ID
1.	Md. Rabby Khan	213902037

Lab Date : 03-09-2024
Submission Date : 10-09-2024
Course Teacher's Name : Sadia Afroze

Lab Report Status

Marks: …… Signature:.....
Comments:..... Date:.....

1. TITLE OF THE LAB REPORT EXPERIMENT

1. Write a Python program that calculates the frequency of each character in a given input string. Your program should count how many times each character appears in the input and display the results.

2. You are given the task of calculating the total tax amount for a person's income based on the following tax brackets: - Income up to $10,000 is taxed at a rate of 5%. - Income from $10,001 to $50,000 is taxed at a rate of 10% - Income from $50,001 to $100,000 is taxed at a rate of 20%. - Income over $100,000 is taxed at a rate of 30%.Your python program should take the user's income as input and calculate the total tax amount they owe based on the provided tax brackets.

2. OBJECTIVES/AIM

- **Enhance Char_Frequency**: Make the function case-insensitive and return the frequency data as a dictionary for increased flexibility.
- **Refine Calculate_Tax:** Add validation for inputs and support customizable tax brackets.
- **Increase Readability**: Add comments and docstrings to make the code easier to understand.
- **Improve User Experience**: Implement error handling and provide clearer feedback to users

3. PROCEDURE / ANALYSIS / DESIGN

Task_1:

Algorithm for Character Frequency Calculation:

Steps:

1. Start
2. Input the string.
3. Create an empty dictionary frequency.
4. For each character in the string:
   - If the character is in frequency, increment the value.
   - Else, add the character to frequency with an initial value of 1.
5. Print each character and its frequency.
6. End

Task_2:
Algorithm: Calculate_Tax(income)
Steps:
1. Initialize tax to 0.
2. If income ≤ 10,000:
   - Tax = income 0.05
3. If 10,001 ≤ income ≤ 50,000:
   - Tax = 500 + (income - 10,000) * 0.10
4. If 50,001 ≤ income ≤ 100,000:
   - Tax = 4500 + (income - 50,000) * 0.20
5. If income > 100,000:
   - Tax = 14500 + (income - 100,000) * 0.30
6. Return the total tax.

4. IMPLEMENTATION

Task_1:

```
def char_frequency(input_string):
    frequency = {}  # Dictionary to hold the frequency of each character
    for char in input_string:
        if char in frequency:
            frequency[char] += 1
        else:
            frequency[char] = 1

    # Display the results
    for char, count in frequency.items():
        print(f"'{char}': {count}")

# Example usage
input_string = input("Enter a string: ")
char_frequency(input_string)
```

5. TEST RESULT / OUTPUT

```
Enter a string: hello world
'h': 1
'e': 1
'l': 3
'o': 2
' ': 1
'w': 1
'r': 1
'd': 1
```

4. IMPLEMENTATION

```
Task_2:
def calculate_tax(income):
    tax = 0

    if income <= 10000:
        tax = income * 0.05
    elif income <= 50000:
        tax = (10000 * 0.05) + ((income - 10000) * 0.10)
    elif income <= 100000:
        tax = (10000 * 0.05) + (40000 * 0.10) + ((income - 50000) * 0.20)
    else:
        tax = (10000 * 0.05) + (40000 * 0.10) + (50000 * 0.20) + ((income - 100000) * 0.30)

    return tax

# Example usage
income = float(input("Enter your income: $"))
tax = calculate_tax(income)
print(f"The total tax amount is: ${tax:.2f}")
```

5. TEST RESULT / OUTPUT

```
Enter your income: $30000
The total tax amount is: $2500.00
```

6. ANALYSIS AND DISCUSSION

The **char_frequency function** efficiently counts the occurrences of each character in a string, using a dictionary to store and update frequencies. It processes characters in linear time, ensuring swift results even for longer strings. This function is straightforward but could benefit from case normalization to combine upper and lower case characters. Additionally, returning the frequency dictionary instead of printing it would enhance its versatility for further use.

The **calculate_tax** function calculates tax based on progressive income brackets, applying different rates for various income ranges. It accurately computes taxes by considering each bracket stepwise. While effective, the function could be improved with input validation and flexibility to handle custom tax brackets. Enhancing these aspects would make the function more robust and adaptable for different scenarios.

7. SUMMARY

The **char_frequency** function efficiently counts characters but could be improved with case normalization and by returning results for flexibility. Similarly, the **calculate_tax** function works well but needs input validation and customizable brackets for more adaptability.