TCS NQT 2025

25 March 2025 Shift 1

Question1

Write a program that processes a given string to determine:

1. **The first non-repeating character (if present).**
2. **The most repeated character in the string.**
3. **If multiple characters have the same highest frequency, print the first non-repeating character first, then the most repeated character.**
4. **If the input string is empty, print "Invalid Input".**
5. **If all characters in the string are repeating, print "None" followed by the most repeating character.**

TestCases:

```
Enter a string: swissmississippi
```

Output

```
First Non-Repeating Character: w
Most Repeated Character: s (appears 7 times)
```

```
Enter a string: aabbcc
```

Output

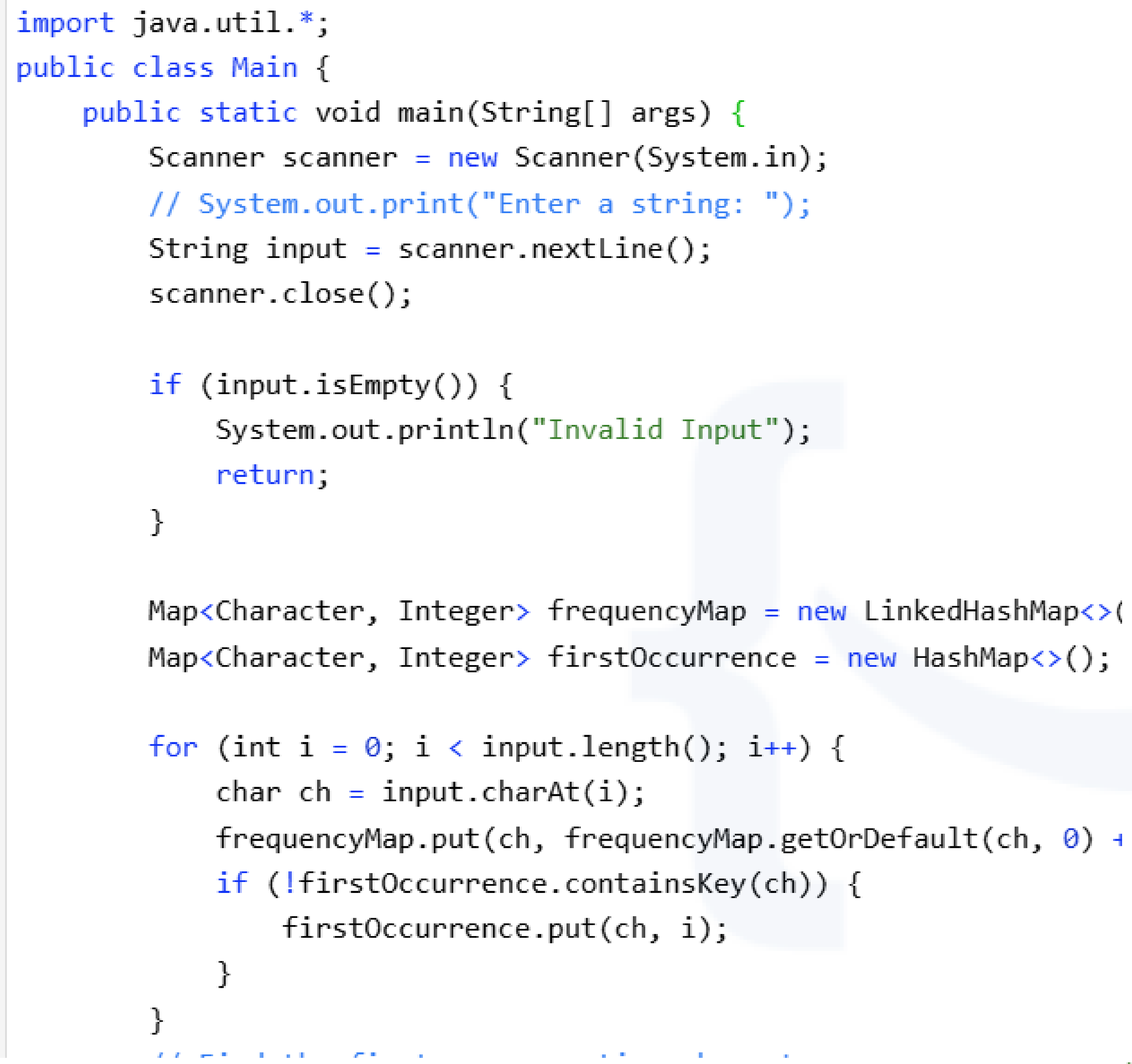
```
First Non-Repeating Character: None
Most Repeated Character: a (appears 2 times)
```

```
Enter a string: aabbccd
```

Output

```
First Non-Repeating Character: d
Most Repeated Character: a (appears 2 times)
```

Java

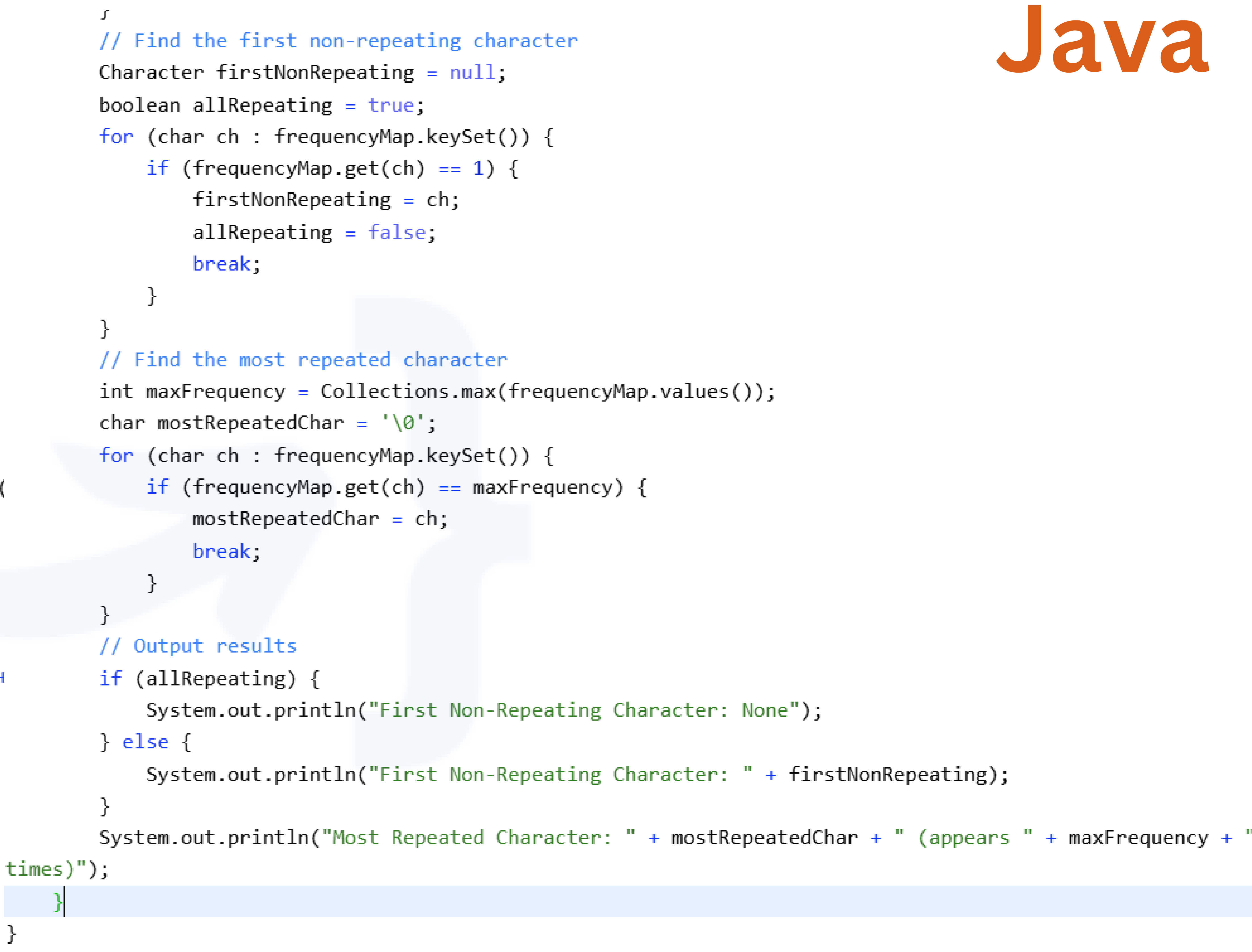


```java
import java.util.*;
public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        // System.out.print("Enter a string: ");
        String input = scanner.nextLine();
        scanner.close();

        if (input.isEmpty()) {
            System.out.println("Invalid Input");
            return;
        }

        Map<Character, Integer> frequencyMap = new LinkedHashMap<>(
        Map<Character, Integer> firstOccurrence = new HashMap<>();

        for (int i = 0; i < input.length(); i++) {
            char ch = input.charAt(i);
            frequencyMap.put(ch, frequencyMap.getOrDefault(ch, 0) +
            if (!firstOccurrence.containsKey(ch)) {
                firstOccurrence.put(ch, i);
            }
        }
```



```java
        }
        // Find the first non-repeating character
        Character firstNonRepeating = null;
        boolean allRepeating = true;
        for (char ch : frequencyMap.keySet()) {
            if (frequencyMap.get(ch) == 1) {
                firstNonRepeating = ch;
                allRepeating = false;
                break;
            }
        }
        // Find the most repeated character
        int maxFrequency = Collections.max(frequencyMap.values());
        char mostRepeatedChar = '\0';
        for (char ch : frequencyMap.keySet()) {
            if (frequencyMap.get(ch) == maxFrequency) {
                mostRepeatedChar = ch;
                break;
            }
        }
        // Output results
        if (allRepeating) {
            System.out.println("First Non-Repeating Character: None");
        } else {
            System.out.println("First Non-Repeating Character: " + firstNonRepeating);
        }
        System.out.println("Most Repeated Character: " + mostRepeatedChar + " (appears " + maxFrequency + "
times)");
    }
}
```

C++

```cpp
#include<bits/stdc++.h>
using namespace std;

int main() {
    string input;
    getline(cin, input);

    if (input.empty()) {
        cout << "Invalid Input" << endl;
        return 0;
    }

    map<char, int> frequencyMap;
    unordered_map<char, int> firstOccurrence;

    for (int i = 0; i < input.length(); i++) {
        char ch = input[i];
        frequencyMap[ch]++;
        if (firstOccurrence.find(ch) == firstOccurrence.end()) {
            firstOccurrence[ch] = i;
        }
    }

    char firstNonRepeating = '\0';
    int firstNonRepeatingIndex = 1e9;
    bool allRepeating = true;
    for ( auto pair : frequencyMap) {

        if (pair.second == 1 && firstOccurrence[pair.first] < firstNonRepeatingIndex) {
            firstNonRepeating = pair.first;
            firstNonRepeatingIndex = firstOccurrence[pair.first];
            allRepeating = false;
        }
    }
    int maxFrequency = 0;
    char mostRepeatedChar = '\0';
    int firstMostRepeatedIndex = 1e9;
    for (auto pair : frequencyMap) {
        if (pair.second > maxFrequency || (pair.second == maxFrequency && firstOccurrence[pair.first] <
firstMostRepeatedIndex)) {
            maxFrequency = pair.second;
            mostRepeatedChar = pair.first;
            firstMostRepeatedIndex = firstOccurrence[pair.first];
        }
    }
    if (allRepeating) {
        cout << "First Non-Repeating Character: None" << endl;
    } else {
        cout << "First Non-Repeating Character: " << firstNonRepeating << endl;
    }
    cout << "Most Repeated Character: " << mostRepeatedChar << " (appears " << maxFrequency << " times)"
endl;
    return 0;
}
```

Python

```python
from collections import OrderedDict

def analyze_string(input_str):
    if not input_str:
        print("Invalid Input")
        return

    frequency_map = {}
    first_occurrence = OrderedDict()

    for i, ch in enumerate(input_str):
        frequency_map[ch] = frequency_map.get(ch, 0) + 1
        if ch not in first_occurrence:
            first_occurrence[ch] = i

    first_non_repeating = None
    first_non_repeating_index = float('inf')
    all_repeating = True

    for ch, count in frequency_map.items():
        if count == 1 and first_occurrence[ch] < first_non_repeating_index:
            first_non_repeating = ch
            first_non_repeating_index = first_occurrence[ch]
            all_repeating = False

    max_frequency = 0
    most_repeated_char = None
    first_most_repeated_index = float('inf')

    for ch, count in frequency_map.items():
        if count > max_frequency or (count == max_frequency and first_occurrence[ch] <
first_most_repeated_index):
            max_frequency = count
            most_repeated_char = ch
            first_most_repeated_index = first_occurrence[ch]

    if all_repeating:
        print("First Non-Repeating Character: None")
    else:
        print(f"First Non-Repeating Character: {first_non_repeating}")

    print(f"Most Repeated Character: {most_repeated_char} (appears {max_frequency} times)"

# Example usage
input_str = input()
analyze_string(input_str)
```

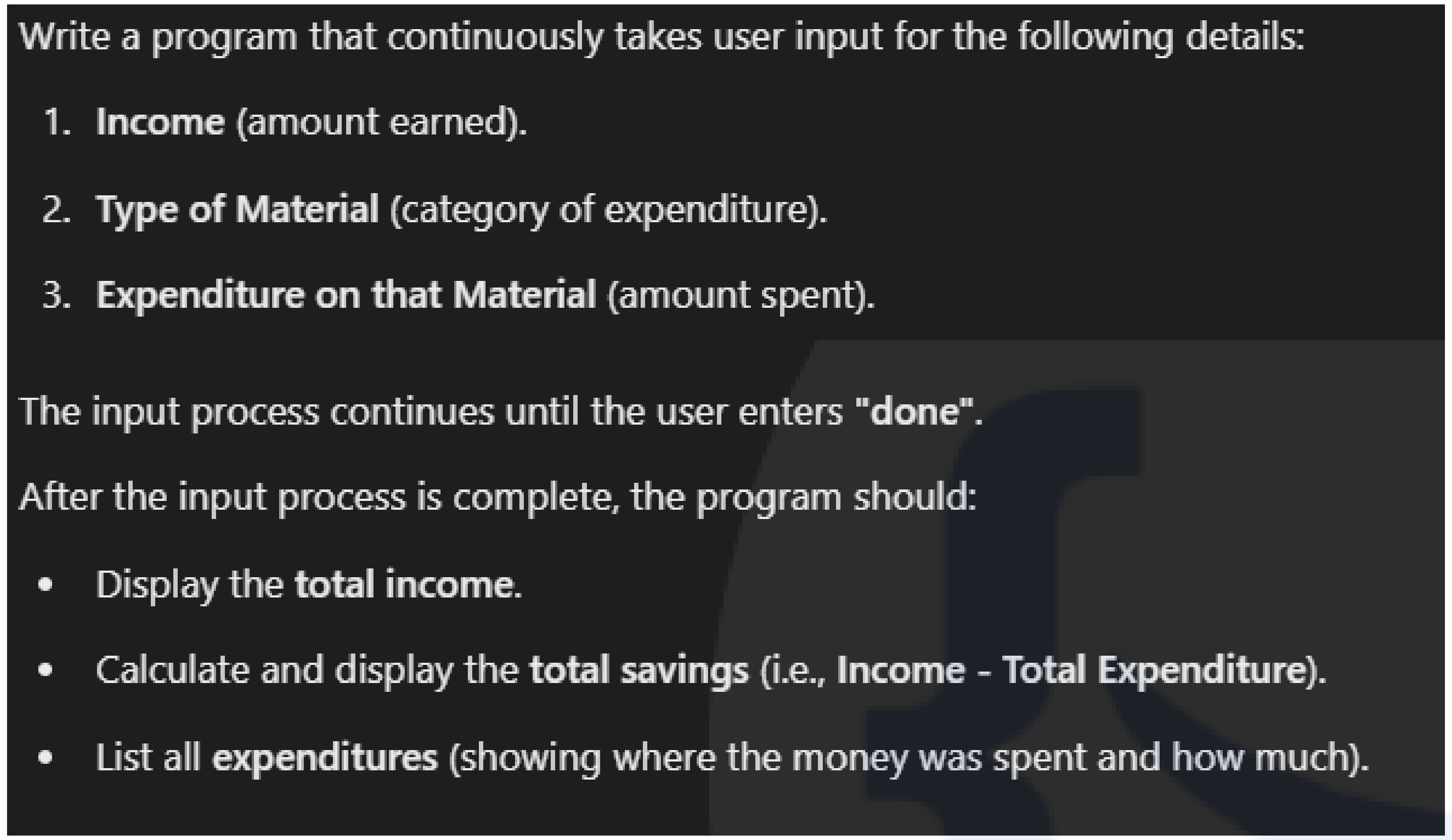
MASSIVE SUCCESS RATE

"Transform Your Interview Opportunity into an Offer Letter and Make Your Parents Proud!"

- In-depth Technical Mock
  - Crack coding challenges with real experts.
- HR & Managerial Prep
  - Master behavioral questions and impress TCS Interviewer.
- Full Interview Simulation
  - Ace both technical and HR in one session.
- Resume Review
  - Identify and fix weaknesses for a standout CV.
- Personalized Feedback & Expert Guidance
  - Tailored improvement tips to boost success.

www.primecoding.in

Question2



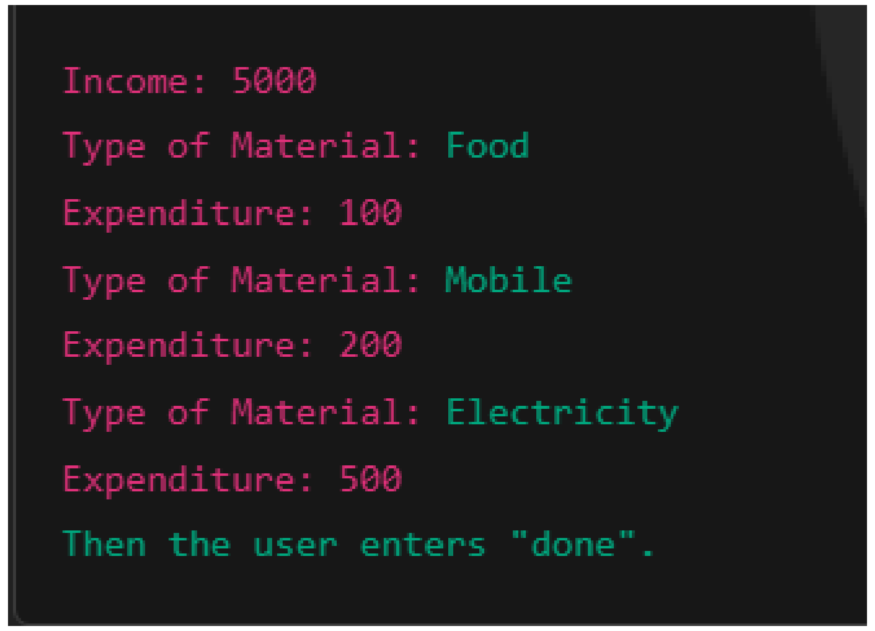
Write a program that continuously takes user input for the following details:

1. **Income** (amount earned).
2. **Type of Material** (category of expenditure).
3. **Expenditure on that Material** (amount spent).

The input process continues until the user enters **"done"**.

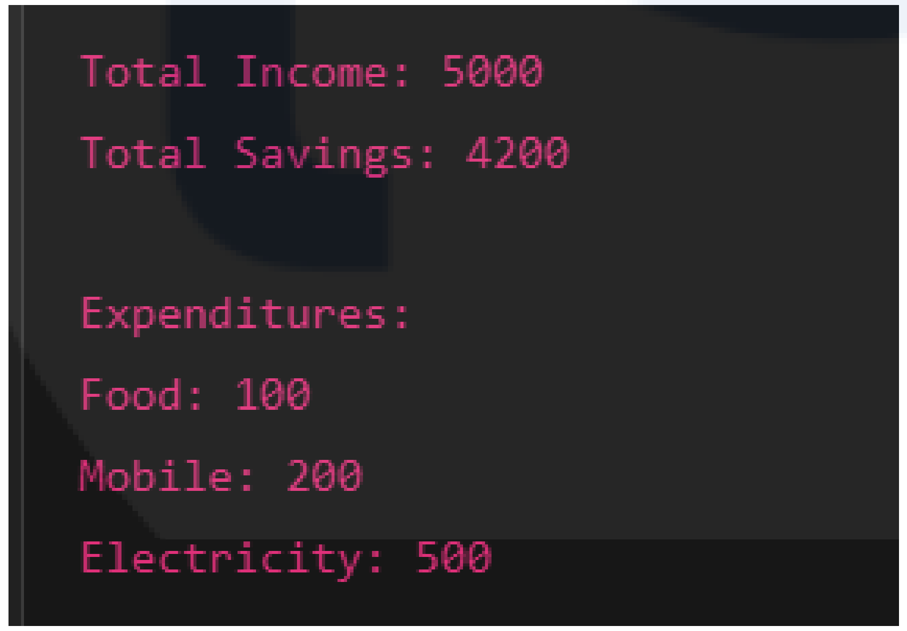
After the input process is complete, the program should:

- Display the **total income**.
- Calculate and display the **total savings** (i.e., **Income - Total Expenditure**).
- List all **expenditures** (showing where the money was spent and how much).



```
Income: 5000
Type of Material: Food
Expenditure: 100
Type of Material: Mobile
Expenditure: 200
Type of Material: Electricity
Expenditure: 500
Then the user enters "done".
```

Input



```
Total Income: 5000
Total Savings: 4200

Expenditures:
Food: 100
Mobile: 200
Electricity: 500
```

Expected Output

Note : Expenditure Sequence is not mentioned

Java

```java
import java.util.*;

public class Main {
  public static void main(String[] args) {
    Scanner scanner = new Scanner(System.in);

    // Take income input
    System.out.print("Enter Income: ");
    int income = scanner.nextInt();
    scanner.nextLine(); // Consume newline

    Map<String, Integer> expenditures = new HashMap<>();
    int totalExpenditure = 0;

    // Take expense details
    while (true) {
      System.out.print("Enter Type of Material (or 'done' to finish): ");
      String material = scanner.nextLine();
      if (material.equals("done")) {
        break;
      }

      System.out.print("Enter Expenditure on " + material + ": ");
      int expense = scanner.nextInt();
      scanner.nextLine(); // Consume newline

      expenditures.put(material, expenditures.getOrDefault(material, 0) + expense);
      totalExpenditure += expense;
    }

int totalSavings = income - totalExpenditure;

    // Display results
    System.out.println("\nTotal Income: " + income);
    System.out.println("Total Savings: " + totalSavings);
    System.out.println("\nExpenditures:");

    for (Map.Entry<String, Integer> entry : expenditures.entrySet()) {
      System.out.println(entry.getKey() + ": " + entry.getValue());
    }

    scanner.close();
  }
}
```

C++

```cpp
#include <iostream>
#include <map>
#include <string>
using namespace std;

int main() {
   int income;
   cout << "Enter Income: ";
   cin >> income;
   cin.ignore(); // Consume newline

   map<string, int> expenditures;
   int totalExpenditure = 0;

   while (true) {
      cout << "Enter Type of Material (or 'done' to finish): ";
      string material;
      getline(cin, material);

      if (material == "done") {
         break;
      }

      int expense;
      cout << "Enter Expenditure on " << material << ": ";
      cin >> expense;
      cin.ignore(); // Consume newline

      expenditures[material] += expense;
      totalExpenditure += expense;
   }
int totalSavings = income - totalExpenditure;

   cout << "\nTotal Income: " << income << endl;
   cout << "Total Savings: " << totalSavings << endl;
   cout << "\nExpenditures:" << endl;

   for (const auto& entry : expenditures) {
      cout << entry.first << ": " << entry.second << endl;
   }

   return 0;
}
```

Python

```
def main():
    # Take income input
    income = int(input("Enter Income: "))

    expenditures = {}
    total_expenditure = 0

    # Take expense details
    while True:
        material = input("Enter Type of Material (or 'done' to finish): ")
        if material.lower() == "done":
            break

        expense = int(input(f"Enter Expenditure on {material}: "))

        expenditures[material] = expenditures.get(material, 0) + expense
        total_expenditure += expense

    # Calculate savings
    total_savings = income - total_expenditure

    # Display results
    print(f"\nTotal Income: {income}")
    print(f"Total Savings: {total_savings}")
    print("\nExpenditures:")

    for material, expense in expenditures.items():
        print(f"{material}: {expense}")

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

leetcode playground: CLICK HERE

Codes are available in all languages