#### **Problem Statements**

#### Event Attendance:

#### A school has two events, and the attendees are stored in two sets. Create 2 sets with names of students. Write a program to:

* Find the students who attended both events.
* Find the students who attended only one of the events.
* Find all students who attended at least one event. ( use ‘|’ operator)

#### **Solution :**

#### 

#### Student Marks Calculation:

You are given a dictionary containing student names as keys and their marks as values.

students = {

"Rohan": 85,

"Spoorthi": 90,

"Aditi": 78,

"Tanya": 92

}

Write a program to:

* Find the student with the highest marks.
* Calculate the average marks for the class.
* Add a new student and their marks to the dictionary.

#### **Solution :**

#### 

#### Sentence Analysis:

Given a sentence, write a program to:

* Count the number of vowels and consonants.
* Find the longest word in the sentence.
* Reverse the sentence.

#### **Solution :**

1. You are given a list of integers , sort the list based on the frequency of the number of occurrences of the elements. Take input of your choice.

#### **Solution :**

1. You went shopping with your family at a local supermarket. Each family member picked up different items independently, creating their own shopping lists. To avoid buying duplicate items, you need to analyze these lists and calculate the final billing amount.

#### **Solution :**

1. You work at a movie theater that keeps track of daily bookings. Each booking record contains the customer's name, selected movie, and seat number in a specific format. The theater needs to analyze these booking records to manage seating and prevent duplicate bookings. Take input from the user.

Input format - "CustomerName-MovieName-SeatNumber"

Hints:

* Consider using string methods like split(), replace()
* Dictionary and set data structures may be used for tracking duplicates

#### **Solution :**

1. Write a Python program that takes two strings and checks if they are anagrams of each other. Ignore spaces and punctuation, and consider the comparison to be case-insensitive. For example, "Astronomer" and "Moon starer" should be identified as anagrams.

#### **Solution :**

1. Write a Python program that takes a string as input and finds the first non-repeating character using a dictionary. If a non-repeating character is found, print it; else, print an appropriate message.

#### **Solution :**

1. Password Validator

Define a simple password validator. The password must:

* Be 8 characters long
* Must contain 1 Upper case character, 1 lower case character and 1 number
* If valid return valid password if not print invalid

#### **Solution :**

1. Isomorphic Strings

Given two strings s and t, determine if they are isomorphic. Two strings s and t are isomorphic if the characters in s can be replaced to get t. All occurrences of a character must be replaced with another character while preserving the order of characters. No two characters may map to the same character, but a character may map to itself.

#### **Solution :**

1. Anonymous Feedback Aggregator

Scenario: Your company collects anonymous feedback from employees with a dictionary storing feedback themes as keys and a list of feedback messages as values.

Use the dictionary feedback = {

"Work Environment": ["Great work culture", "Need more team activities"],

"Salary": ["Fair pay, but bonuses are inconsistent"],

"Management": ["Leadership can improve", "Need more transparency"],

}

Write a program to:

* Add new feedback to the relevant theme.
* Identify the theme with the most feedback.

**Solution :**