Problem Solving (CPP46)

هذا البرنامج التدريبي مُصاغ بعناية لتمكين المتدربين من تطوير قدراتهم الفكرية على غرار المبرمجين المحترفين، والتعاون بكفاءة ضمن فريق محترف في شركة "جمال تك" أو أي مؤسسة متعددة الجنسيات أخرى. نظرًا لأهمية اللغة الإنجليزية في بيئة العمل العالمية، يتم تقديم المحتوى التدريبي بالإنجليزية. لا يشترط إتقان اللغة بشكل كامل، لكن من الضروري امتلاك القدرة الكافية لفهم المتطلبات وتنفيذها بشكل فعّال. يُمكن للمتدربين استخدام مترجم جوجل أو الاستعانة بـ "شات جي بي تي" للتغلب على أية عقبات لغوية، المهم هو الفهم الدقيق للمطلوب وتحقيقه بنجاح.

لتعظيم الاستفادة من التدريب، يُنصح بمحاولة حل التمارين بشكل مستقل لمدة ساعة واحدة على الأقل قبل الرجوع إلى الحل المرفق في نهاية الملف.

قد يتضمن الحل كودًا برمجيًا غير مفسر بعد، والغرض من ذلك هو تشجيعك على محاولة فهم الأكواد البرمجية الجديدة التي لم تتعرض لها من قبل. هذه المهارة ضرورية في سوق العمل، حيث تتطور لغات البرمجة باستمرار ويظهر كل يوم لغات جديدة. ستواجه دائمًا أكوادًا لم تدرسها من قبل، ومن المهم أن تكون قادرًا على فهمها بنفسك دون الحاجة إلى دراسة مسبقة. يمكنك الاستعانة بمحرك البحث جوجل، أو استخدام ChatGPT، أو حتى اللجوء لأصدقائك للمساعدة. الهدف الأساسي هو أن تصل إلى فهم معنى كل كود بأي طريقة ممكنة لتتمكن من إيجاد موقعك في سوق العمل.

إن وجود كود برمجي غير مفسر يشكل تحديًا يتوجب عليك إيجاد حل له. هذا النوع من التدريبات يعد جزءًا أساسيًا من تدريبات 'Problem Solving'، التي تهدف إلى تمكينك من أداء عملك بفاعلية بغض النظر عن التحديات والعقبات. هذه القدرة على حل المشكلات هي ما يتمتع به العاملون في 'جمال تك'، ومن الضروري أن تطور في نفسك هذه المهارة لتصبح عضوًا فعالًا في فريق عمل 'جمال تك'.

# Gammal Tech's Entity Relationship Analyzer

## Background

Gammal Tech, a trailblazer in the software industry, is renowned for its innovative approaches and cutting-edge office facilities. The company's latest project involves optimizing database designs through intelligent entity relationship analysis. This system design task focuses on discerning the relationships between various data entities.

## Problem Statement

You are part of Gammal Tech's elite software team, tasked with developing a program to analyze and classify relationships between data entities in a database. Given a number of data entities and their names, along with pairs of these entities, your program should identify the type of relationship between each pair as one of the following: one-to-one, one-to-many, or many-to-many.

## Input Format

- The first line contains an integer $N$ ($1 \le N \le 100$), the number of data entities.

- The following $N$ lines each contain a string representing the name of a data entity. Entity names consist of lowercase English letters and are up to 10 characters long.
- The next line contains an integer $M$ ($1 \leq M \leq 500$), the number of pairs to analyze.
- Each of the next $M$ lines contains two space-separated strings representing the names of two entities whose relationship needs to be identified.

Output Format

For each of the $M$ pairs, output a line containing the names of the two entities separated by a space, followed by one of the following strings, indicating the type of relationship:

- "One-to-One"
- "One-to-Many"
- "Many-to-Many"

Constraints

- Entity names are unique.
- Relationships are bidirectional (if A is related to B, then B is related to A).
- The types of relationships are mutually exclusive for each pair.

**Sample Input:**

```
4
users
orders
products
categories
3
users orders
orders products
products categories
```

**Sample Output:**

```
users orders One-to-Many
orders products Many-to-Many
products categories One-to-Many
```

Explanation

- "users" to "orders" is a One-to-Many relationship, as one user can have many orders, but each order belongs to one user.
- "orders" to "products" is a Many-to-Many relationship, as an order can contain many products, and a product can be in many orders.
- "products" to "categories" is a One-to-Many relationship, as one product belongs to one category, but a category can have many products.

لتعظيم الاستفادة من التدريب، يُنصح بمحاولة حل التمرين بشكل مستقل لمدة ساعة واحدة على الأقل قبل الرجوع إلى الحل المرفق

## C++ Programming Solution:

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

int main() {
    int N, M;
    cin >> N;

    unordered_map<string, int> entityCount;
    string entity;
    for(int i = 0; i < N; ++i) {
        cin >> entity;
        entityCount[entity] = 0;
    }

    cin >> M;
    string entity1, entity2;
    for(int i = 0; i < M; ++i) {
        cin >> entity1 >> entity2;
        entityCount[entity1]++;
        entityCount[entity2]++;
    }

    for(const auto &pair : entityCount) {
        string relationship;
        if(pair.second == 1) {
            relationship = "One-to-One";
        } else if(pair.second == N - 1) {
            relationship = "Many-to-Many";
        } else {
            relationship = "One-to-Many";
        }
        cout << pair.first << " " << relationship << endl;
    }

    return 0;
}
```