Problem Solving (C61)

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 Smart Database Manager

Background

Gammal Tech, a pioneer in the software industry, renowned for its cutting-edge office facilities and innovative team, is now venturing into developing a new technology for efficient database management systems (DBMS). This new system, named "Smart DB Manager," aims to optimize data storage and retrieval by intelligently categorizing data.

Scenario

As a software engineer at Gammal Tech, you are tasked with designing a basic version of the Smart DB Manager. The system should be able to process a series of data entries and categorize them into different tables based on a simple categorization rule. Each data entry will consist of an identifier and a numerical value. The categorization rule is straightforward: if the numerical value is even, the data goes into the "EvenTable"; otherwise, it goes into the "OddTable".

Objective

Your goal is to design a system that processes a series of N data entries and categorizes them correctly. The system should output the contents of both EvenTable and OddTable at the end of the processing.

Input Format

- The first line contains an integer N, the number of data entries.
- The following N lines each contain a string and an integer, representing the identifier and numerical value of the data entry, respectively.

Output Format

- The first line should display the contents of EvenTable: a list of identifiers whose numerical values are even, separated by spaces.
- The second line should display the contents of OddTable: a list of identifiers whose numerical values are odd, separated by spaces.

Constraints

- $1 \leq N \leq 1000$
- The numerical values are non-negative and less than 10,000.

Sample Input:

```
5
Data1 4
Data2 7
Data3 8
Data4 3
Data5 2
```

Sample Output:

```
Data1 Data3 Data5
Data2 Data4
```

Explanation

- In the sample input, Data1, Data3, and Data5 have even numerical values and thus are categorized into EvenTable. Data2 and Data4 have odd numerical values and are categorized into OddTable.

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

C Programming Solution:

```c
#include <stdio.h>
#include <string.h>

typedef struct {
    char identifier[50];
    int value;
} DataEntry;

int main() {
    int n, i;
    scanf("%d", &n);

    DataEntry entries[n];
    for(i = 0; i < n; i++) {
        scanf("%s %d", entries[i].identifier, &entries[i].value);
    }

    printf("EvenTable: ");
    for(i = 0; i < n; i++) {
        if(entries[i].value % 2 == 0) {
            printf("%s ", entries[i].identifier);
        }
    }

    printf("\nOddTable: ");
    for(i = 0; i < n; i++) {
        if(entries[i].value % 2 != 0) {
            printf("%s ", entries[i].identifier);
        }
    }

    return 0;
}
```