Problem Solving (C28)

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 Efficient String Analyzer

Background:

At Gammal Tech, where only the brightest minds tackle complex software challenges, efficiency and smart solutions are highly valued. String manipulation and analysis form a core part of many high-level programming tasks, especially in data processing and communication systems. To excel in such an environment, one must be adept at using available tools and libraries efficiently, such as the string header file in C.

Task:

You are required to develop a C program for Gammal Tech that efficiently analyzes a given string. The program should read a string (up to 200 characters) and perform the following tasks using functions from the string header file:

1. Count the number of words in the string.
2. Determine the length of the longest word in the string.
3. Compare the input string with another string input by the user and report if they are identical.

Objective:

1. Implement a program that utilizes functions from the string header file in C.
2. Apply these functions to analyze the input string as per the specified requirements.
3. Display the analysis results to the user.

Constraints:

- The solution must efficiently use the functions available in the string header file.
- Ensure the program is robust and handles different types of string inputs effectively.

Sample Input:

```
Enter the first string: Gammal Tech solves complex problems
Enter the second string to compare: Gammal Tech innovates
```

Sample Output:

```
Number of words in the first string: 5
Length of the longest word: 7 (complex)
The two strings are not identical.
```

لتعظيم 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>
#include <ctype.h>

#define MAX_LENGTH 200

int wordCount(const char *str) {
    int count = 0, inWord = 0;
    while (*str) {
        if (isspace(*str)) {
            inWord = 0;
        } else if (!inWord) {
            inWord = 1;
            count++;
        }
        str++;
    }
    return count;
}

int longestWordLength(const char *str) {
    int maxLength = 0, length = 0;
    while (*str) {
        if (isspace(*str)) {
            if (length > maxLength) {
                maxLength = length;
            }
            length = 0;
        } else {
            length++;
        }
        str++;
    }
    return (length > maxLength) ? length : maxLength;
}

int main() {
    char str1[MAX_LENGTH], str2[MAX_LENGTH];

    // Reading the first string input
    printf("Enter the first string: ");
    fgets(str1, MAX_LENGTH, stdin);
    str1[strcspn(str1, "\n")] = 0; // Remove newline character

    // Reading the second string for comparison
    printf("Enter the second string to compare: ");
    fgets(str2, MAX_LENGTH, stdin);
    str2[strcspn(str2, "\n")] = 0; // Remove newline character
```

```
    // String analysis
    printf("Number of words in the first string: %d\n",
wordCount(str1));
    printf("Length of the longest word: %d\n", longestWordLength(str1));
    printf("The two strings are %s.\n", (strcmp(str1, str2) == 0) ?
"identical" : "not identical");

    return 0;
}
```