Problem Solving (C30)

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 Code Optimizer

Scenario:

Gammal Tech, a vanguard in the software industry, known for its innovative team and advanced office facilities, is facing a unique challenge. They need to develop a system that optimizes the storage of their vast array of software projects. Each project is uniquely identified by a string name.

Problem Statement:

Gammal Tech's software repository consists of N software projects, each with a unique string identifier. The identifiers are stored in an array of strings. Due to the increasing number of projects, Gammal Tech wants to optimize the storage by grouping projects based on the length of their string identifiers.

Your task is to write a program that sorts the array of project identifiers based on the length of the string. If two identifiers have the same length, they should be sorted lexicographically (alphabetically). After sorting, your program should output the sorted array, displaying the projects in the order they should be stored.

Input:

- The first line of the input contains a single integer N, the number of software projects.
- Each of the next N lines contains a string representing the name of a software project.

Output:

- Output N lines, each containing a string, representing the sorted order of the projects.

Constraints:

- $1 \leq N \leq 1000$
- Each string will have a length between 1 and 100 characters.
- All characters will be lowercase English letters.

Sample Input:

```
4
codebase
devtool
ai
compiler
```

Sample Output:

```
ai
devtool
compiler
codebase
```

Explanation:

In the sample input, "ai" has the shortest length, followed by "devtool" and "compiler" which have the same length but are sorted lexicographically, and finally "codebase".

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>

#define MAX_PROJECTS 1000
#define MAX_LENGTH 101

// Function to compare two strings based on the problem's requirement
int compare(const void *a, const void *b) {
    char *strA = *(char **)a;
    char *strB = *(char **)b;
    int lenA = strlen(strA);
    int lenB = strlen(strB);
    if (lenA == lenB) return strcmp(strA, strB);
    return lenA - lenB;
}

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

    char *projects[MAX_PROJECTS];
    for (int i = 0; i < N; i++) {
        projects[i] = malloc(MAX_LENGTH);
        scanf("%s", projects[i]);
    }

    qsort(projects, N, sizeof(char *), compare);

    for (int i = 0; i < N; i++) {
        printf("%s\n", projects[i]);
        free(projects[i]);
    }

    return 0;
}
```

In this solution, we use `qsort` from the C standard library with a custom comparator to sort the array of strings based on the length and lexicographic order. The program first reads the number of projects and then their names, sorts them, and finally prints them in the desired order.