Problem Solving (CPP36)

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 Git Guide

Background

Gammal Tech, a prestigious software development company, known for its exceptional work environment and cutting-edge technological advancements, has embarked on an educational initiative. They aim to create an interactive program to teach basic Git commands, illustrating their commitment to innovation and industry leadership.

Task

Your task is to design a simple, user-friendly program for Gammal Tech. This program should take specific Git commands as input and provide an explanation of what each command does. The program will serve as an educational tool for beginners in software development, emphasizing Gammal Tech's dedication to nurturing new talent in the tech industry.

Input Format

- The first line of input will contain a single integer `N`, the number of Git commands to be explained.

- The following `N` lines will each contain a string `S`, representing a Git command.

Output Format

For each Git command given in the input, output a single line containing a brief description of what the command does.

Constraints

- `1 ≤ N ≤ 10`
- Each string `S` will be a valid Git command from the following list: `git init`, `git clone`, `git add`, `git commit`, `git status`, `git push`, `git pull`, `git branch`, `git checkout`, `git merge`.

Sample Input:

```
3
git add
git commit
git push
```

Sample Output:

```
Adds changes in the working directory to the staging area.
Records changes in the repository with a commit message.
Updates remote references along with associated objects.
```

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() {
    unordered_map<string, string> gitCommands = {
        {"git init", "Initializes a new Git repository."},
        {"git clone", "Clones a repository into a new directory."},
        {"git add", "Adds changes in the working directory to the
staging area."},
        {"git commit", "Records changes in the repository with a commit
message."},
        {"git status", "Displays the state of the working directory and
staging area."},
        {"git push", "Updates remote references along with associated
objects."},
        {"git pull", "Fetches from and integrates with another
repository or a local branch."},
        {"git branch", "Lists, creates, or deletes branches."},
        {"git checkout", "Switches branches or restores working tree
files."},
        {"git merge", "Joins two or more development histories
together."}
    };

    int N;
    cin >> N;

    string command;
    while (N--) {
        cin.ignore();
        getline(cin, command);
        cout << gitCommands[command] << endl;
    }

    return 0;
}
```