Problem Solving (CPP40)

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 Learning Classifier

Background:
Gammal Tech, a pioneer in software development, renowned for its cutting-edge office facilities and innovative team, faces a unique challenge. The company needs a program to classify its diverse software products into categories that require either supervised or unsupervised learning based on their descriptions. This classification is crucial for optimizing the development process and maintaining Gammal Tech's status as an industry leader.

Problem Statement:
Your task is to write a C++ program that reads a software description and determines whether it should be developed using supervised or unsupervised learning methods. Gammal Tech's software descriptions contain key phrases that hint at the nature of the learning required. Supervised learning is indicated by phrases related to "labelled data", "prediction", "regression", or "classification". Unsupervised learning is suggested by phrases like "pattern discovery", "clustering", "association", or "unlabelled data".

Input Format:

- The first line contains an integer N (1 ≤ N ≤ 100), representing the number of software descriptions.
- Each of the next N lines contains a string S describing a software product. S is a single line of text containing up to 1000 characters.

Output Format:

For each software description, output a single line containing either "Supervised Learning" or "Unsupervised Learning", based on the classification determined by your program.

Constraints:

- The software description will contain only English letters, spaces, and standard punctuation.
- Phrases indicating learning type are case-insensitive.

Sample Input:

```
3
Predictive analysis tool using regression models for labelled data.
Clustering algorithm for pattern discovery in large datasets.
Data visualization software with association rules for unlabelled data.
```

Sample Output:

```
Supervised Learning
Unsupervised Learning
Unsupervised Learning
```

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

C++ Programming Solution:

```
#include <iostream>
#include <string>
#include <vector>
#include <sstream>
#include <algorithm>
using namespace std;

bool containsKeyword(const string& description, const vector<string>&
keywords) {
    for (const string& keyword : keywords) {
        if (description.find(keyword) != string::npos) {
            return true;
        }
    }
    return false;
}
string classifyLearningType(const string& description) {
    // Convert to lower case for case-insensitive comparison
    string lowerDesc = description;
    transform(lowerDesc.begin(), lowerDesc.end(), lowerDesc.begin(),
::tolower);

    vector<string> supervisedKeywords = {"labelled data", "prediction",
"regression", "classification"};
    vector<string> unsupervisedKeywords = {"pattern discovery",
"clustering", "association", "unlabelled data"};

    if (containsKeyword(lowerDesc, supervisedKeywords)) {
        return "Supervised Learning";
    }
    if (containsKeyword(lowerDesc, unsupervisedKeywords)) {
        return "Unsupervised Learning";
    }
    return "Unknown Learning Type";
}
int main() {
    int n;
    cin >> n;
    cin.ignore(); // Ignore newline after reading n

    string description;
    for (int i = 0; i < n; ++i) {
        getline(cin, description);
        cout << classifyLearningType(description) << endl;
    }
    return 0;
}
```