

المحاضرة 11 · الجداول والنصوص

11

المصفوفات الثنائية والنصوص: جدول بدل قائمة

كيف نمثل درجات مجموعة طلاب في عدة مواد؟

تطبيق عملي

أمثلة C++17

رسم توضيحي

فهم قبل الحفظ

مستوحاة من برمجة 1 · صفحات المصدر 117-128 · إعداد تربوي: م.براءة علي رحيمو

++C

01 · المشكلة أولاً

قبل التعريف... لماذا نحتاج هذه الفكرة؟

كيف نمثل درجات مجموعة طلاب في عدة مواد؟

مشهد من الواقع

لديك ثلاثة طلاب وكل منهم يدرس مادتين. البيانات ليست سلسلة واحدة بل جدول؛ وصَفَّ إضافي لا يعني تغيير منطق الحساب كله.

أين المشكلة في الطريقة الساذجة؟

إذا تعاملنا مع درجات كل طالب منفردًا سنكرر العمليات، وإذا تجاهلنا فهرس الصف والعمود سنقرأ بيانات خاطئة.

كيف تساعدنا الفكرة؟

نستخدم array ثنائية البعد [col][row] مع حلقتين؛ ولأسماء نستخدم std::string بدل إدارة محارف منخفضة المستوى حين يكون هذا هو الأنسب.

شَبَّهها بما تعرفه: جدول المدرسة: الصف يدل على الطالب والعمود يدل على المادة؛ تقاطع الاثنين يعطي الدرجة.

لا تحفظ كلمة جديدة قبل أن تعرف المشكلة التي وُجدت لتساعد في حلها.

02 · النموذج العقلي

اجعل المفهوم مرئيًا

ثلاث أفكار رئيسية ثم مسار واضح يربط بينها.

grades[3][2]

جدول 3 صفوف وعمودين، فهارسهما تبدأ من الصفر.

Nested loops

حلقة خارجية للصفوف وأخرى داخلية للأعمدة.

std::string

تخزن نصًا ديناميكيًا أسهل معالجة من مصفوفة محارف C التقليدية.

الرسم التوضيحي: كيف تعمل الفكرة؟

grades[3][2] · صفوف ومواد		
	المادة 1	المادة 2
الطالب 1	70	80
الطالب 2	90	100
الطالب 3	60	80

السؤال الأهم: كيف نمثل درجات مجموعة طلاب في عدة مواد؟

03 · مثال عملي أول

شغل وافهم كل سطر

متوسط كل صف في جدول درجات

C++17 / source.cpp

```
#include <iostream>
using namespace std;
int main() {
    int g[3][2] = {{70,80},{90,100},{60,80}};
    for (int row=0; row<3; ++row) {
        int sum = 0;
        for (int col=0; col<2; ++col)
            sum += g[row][col];
        cout << sum / 2.0 << ' ';
    }
}
```

الخرج المتوقع 75 95 70

لماذا كتبنا البرنامج بهذا الشكل؟

حلقة داخل الصف تجمع المادتين، ثم تبدأ عملية جديدة للطالب التالي.

💡 قبل تجربة البرنامج: حاول توقع النتيجة على الورق، ثم قارن توقعك بالخرج الفعلي.

04 · مثال إضافي

نفس المبدأ، حالة استخدام مختلفة

قراءة الاسم الكامل بطريقة صحيحة

C++17 / source.cpp

```
#include <iostream>
#include <string>
using namespace std;
int main() {
    string fullName = "Baraa Ali";
    cout << "Name: " << fullName << '\n';
    cout << "Letters: " << fullName.size();
}
```

الخرج المتوقع Name: Baraa Ali | Letters: 9

ما الذي نتعلمه من المثال الثاني؟

سلسلة string تحتفظ بالمسافة بين الاسم الأول والثاني؛ cin>>name وحدها تقف عند المسافة، ولإدخال سطر كامل نستخدم getline.

جرب تغيير قيمة واحدة فقط، ثم توقّع أثرها: هل تغيّر مسار البرنامج أم الناتج فقط؟

05 · مختبر الأخطاء

الخطأ ليس للحفظ... اكتشف سببه

قارن الكود الخاطئ بالصحيح ولا تكتفِ بمعرفة الإجابة.

الكود الذي يحتاج انتباهاً

```
int g[2][2]={ {1,2},{3,4}}; cout<<g[2][0];
```

تصحيح وتفسير

```
int g[2][2]={ {1,2},{3,4}}; cout<<g[1][0];
```

لماذا يُعد هذا خطأ؟

حتى في الجدول يوجد حد لكل بُعد؛ الصف رقم 2 خارج جدول يحوي صفين.

جرب بنفسك

تحدي 01

صمّم جدول مبيعات 3 أيام × 2 أصناف، ثم احسب مجموع كل يوم.

تحدي 02

اشرح لماذا نحتاج حلقتين في المصفوفة الثنائية.

تحدي 03

جرب `getline(cin,name)` عندما يحوي الاسم مسافة.

06 · تثبيت الفهم

ماذا يجب أن يبقى في ذاكرتك؟

لا تقرأ القواعد فقط؛ اربط كل قاعدة بخطأ تمنعه أو مشكلة تحلها.

- ✓ لكل بُعد حد مستقل يجب احترامه.
- ✓ الحلقة الداخلية تعالج الأعمدة لكل صف.
- ✓ `std::string` أفضل لأغلب نصوص ++C الحديثة مقارنةً بـ `char[]` عند عدم الحاجة لتفاصيل منخفضة المستوى.

الرسم التوضيحي: كيف تعمل الفكرة؟

`grades[3][2]` · صفوف ومواد

الطالب 1	70	80
الطالب 2	90	100
الطالب 3	60	80
	المادة 1	المادة 2

تحضير الكويز التفاعلي

قبل فتح اختبار هذه المحاضرة، حاول شرح المثال الأول لشخص آخر دون النظر إلى الشيفرة. إذا فهم لماذا تعمل الخطوات، فأنت جاهز.

الجسر إلى المحاضرة التالية

في المحاضرة المقبلة سننقل العمليات المتكررة إلى دوال واضحة.

المرجع: «برمجة 1» لم. بسمة سراقبي، الصفحات 117–128 من الملف المرفق. الأمثلة والشروح الإضافية صيغت للأغراض التعليمية وتستخدم ++17C.