

المحاضرة 01 · التفكير الحاسوبي

01

لماذا نبرمج؟ من المشكلة إلى البرنامج

لماذا لا يكفي أن نمتلك حاسوبًا سريعًا لحل أي مشكلة؟

تطبيق عملي

أمثلة C++17

رسم توضيحي

فهم قبل الحفظ

مستوحاة من برمجة 1 · صفحات المصدر 2-8 · إعداد تربوي: م.براءة علي رحيمو

++C

01 · المشكلة أولاً

قبل التعريف... لماذا نحتاج هذه الفكرة؟

لماذا لا يكفي أن نمتلك حاسوبًا سريعًا لحل أي مشكلة؟

مشهد من الواقع

تخيّل موظفًا لديه درجات 100 طالب. الأرقام موجودة، لكن استخراج نسبة النجاح يدويًا بطيء ومعرض للأخطاء.

أين المشكلة في الطريقة الساذجة؟

المعطيات الخام لا تتحول وحدها إلى قرار. كما أن كتابة الكود قبل فهم المطلوب تجعل البرنامج ينفّذ شيئًا غير المطلوب بدقة!

كيف تساعدنا الفكرة؟

نحدّد المدخلات ونصف المعالجة والمخرجات ونختبر النتائج؛ تصبح الآلة منقّذًا لخطة نفهمها.

شَبَّهها بما تعرفه: الحاسوب كطاهٍ سريع جدًا. وجود المكونات لا يكفي؛ نحتاج وصفة دقيقة ومقادير واضحة.

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.

Hardware

المكونات المادية التي تنفذ العمليات، ولا تختار هدف المهمة بنفسها.

Software

تعليمات منظّمة توضّح ماذا ينفذ الحاسوب.

Data → Information

درجات الطلاب بيانات؛ متوسطها ونسبة النجاح معلومات بعد المعالجة.

خريطة المفهوم: من الفكرة إلى التنفيذ



السؤال الأهم: لماذا لا يكفي أن نمتلك حاسوبًا سريعًا لحل أي مشكلة؟

03 · مثال عملي أول

شغل وافهم كل سطر

من درجتين إلى قرار مفهوم

C++17 / source.cpp

```
#include <iostream>
using namespace std;
int main() {
    int math = 70, programming = 90;
    double average = (math + programming) / 2.0;
    cout << "Average = " << average << '\n';
    cout << (average >= 60 ? "Pass" : "Review");
}
```

الخرج المتوقع Average = 80 | Pass

لماذا كتبنا البرنامج بهذا الشكل؟

قبل الكود: قررنا أن المعدل يساوي مجموع العلامتين مقسومًا على 2؛ ثم ترجمنا القاعدة إلى تعليمات.

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

04 · مثال إضافي

نفس المبدأ، حالة استخدام مختلفة

مثال مختلف: تحويل السعر إلى فاتورة

C++17 / source.cpp

```
#include <iostream>
using namespace std;
int main() {
    double price = 150, quantity = 3;
    double subtotal = price * quantity;
    double tax = subtotal * 0.10;
    cout << "Total = " << subtotal + tax;
}
```

الخرج المتوقع Total = 495

ما الذي نتعلمه من المثال الثاني؟

المدخلات: السعر والكمية؛ المعالجة: ضرب ثم ضريبة؛ المخرجات: قيمة واحدة يفهمها المستخدم.

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 انتباهًا

```
cout << total; // total قبل حساب
```

تصحيح وتفسير

```
double total = 150 * 3;  
cout << total;
```

لماذا يُعد هذا خطأً؟

استخدام قيمة قبلتهيئتها قد ينتج سلوكًا غير صحيح. افهم العلاقة بين البيانات والنتائج.

جرب بنفسك

تحدي 01

حوّل مسألة حساب خصم 20% إلى: مدخلات، معالجة، مخرجات.

تحدي 02

اسأل: كيف نعرف أن برنامج حساب المتوسط صحيح؟ جرب 0 و 100.

تحدي 03

ناقش الفرق بين بيانات 45 ساعة عمل ومعلومة مفيدة عن ساعات العمل.

06 · تثبيت الفهم

ماذا يجب أن يبقى في ذاكرتك؟

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

✓ ابدأ بفهم المشكلة قبل الترميز.

✓ الحاسوب ينفذ التعليمات؛ لا يستنتج نيتك.

✓ الاختبار والتصحيح جزء من البرمجة لا خطوة ترفيحية.

خريطة المفهوم: من الفكرة إلى التنفيذ



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 التالية سنصف الحل كخوارزمية حتى قبل أن نكتب ++C.

المرجع: «برمجة 1» لم. بسمة سراقبي، الصفحات 2-8 من الملف المرفق. الأمثلة والشرح الإضافية صيغت للأغراض التعليمية وتستخدم C++17.