

المحاضرة 14 · الذاكرة والعناوين

14

المؤشرات: ما الفرق بين القيمة وعنوانها؟

إذا كانت القيمة داخل خانة ذاكرة، كيف نشير إلى موقع الخانة نفسها؟

تطبيق عملي

أمثلة C++17

رسم توضيحي

فهم قبل الحفظ

مستوحاة من برمجة 1 · صفحات المصدر 152-156 · إعداد تربوي: م.براءة علي رحيمو

++C

01 · المشكلة أولاً

قبل التعريف... لماذا نحتاج هذه الفكرة؟

إذا كانت القيمة داخل خانة ذاكرة، كيف نشير إلى موقع الخانة نفسها؟

مشهد من الواقع

تريد دالة قراءة قيمة ثم تعديلها عبر عنوانها، أو فهم كيف تشير بنى البيانات إلى عناصر مخزنة في مواقع مختلفة.

أين المشكلة في الطريقة الساذجة؟

عند الخلط بين المؤشر والقيمة التي يشير إليها قد نقرأ عنواناً بدل العدد أو ن فك إشارة مؤشر غير صالح.

كيف تساعدنا الفكرة؟

نتعلم & لاستخراج العنوان و * لفك الإشارة، ومبدأ `nullptr` والتحقق قبل الوصول.

شبهها بما تعرفه: القيمة هي محتوى صندوق؛ والمؤشر ورقة مكتوب عليها عنوان الصندوق، لا محتواه.

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.

value&

يعطي عنوان المتغير value في الذاكرة.

int* pointer

متغير يستطيع تخزين عنوان int متوافق.

pointer / nullptr

فك الإشارة للوصول إلى القيمة، وnullptr يعني أنه لا يشير إلى كائن.

الرسم التوضيحي: كيف تعمل الفكرة؟



السؤال الأهم: إذا كانت القيمة داخل خانة ذاكرة، كيف نشير إلى موقع الخانة نفسها؟

03 · مثال عملي أول

شغل وافهم كل سطر

مؤشر إلى عدد صحيح

C++17 / source.cpp

```
#include <iostream>
using namespace std;
int main() {
    int score = 10;
    int* ptr = &score;
    *ptr = 25;
    cout << score;
}
```

الخرج المتوقع 25

لماذا كتبنا البرنامج بهذا الشكل؟

ptr يحمل عنوان score؛ إسناد قيمة إلى *ptr يغيّر محتوى score لا عنوانه.

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

04 · مثال إضافي

نفس المبدأ، حالة استخدام مختلفة

فحص المؤشر قبل الاستخدام

C++17 / source.cpp

```
#include <iostream>
using namespace std;
int main() {
    int number = 7;
    int* ptr = nullptr;
    if (ptr == nullptr) ptr = &number;
    if (ptr != nullptr) cout << *ptr;
}
```

الخرج المتوقع 7

ما الذي نتعلمه من المثال الثاني؟

نبدأ بمؤشر واضح الحالة؛ ولا نفك إشارته قبل التأكد من أنه يشير إلى شيء مناسب.

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

```
int* ptr = nullptr; cout << *ptr;
```

تصحيح وتفسير

```
int n = 7; int* ptr = &n; cout << *ptr;
```

لماذا يُعد هذا خطأ؟

فك الإشارة لمؤشر فارغ سلوك غير معرّف وخطر؛ لا تفترض أن nullptr عنوان قابل للقراءة.

جرب بنفسك

تحدي 01

ارسم صندوقًا يحمل 10 وسهمًا من ptr إليه.

تحدي 02

فسّر اختلاف cout<<ptr عن cout<<*ptr.

تحدي 03

اكتب دالة تزيد عددًا باستخدام *int وتتحقق من nullptr.

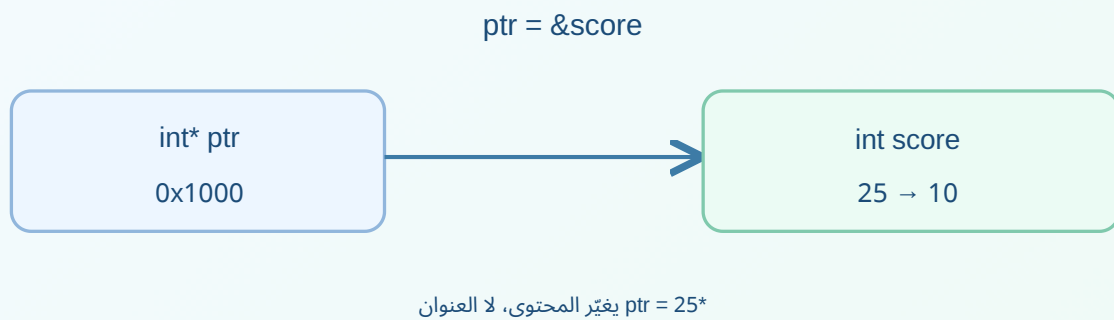
06 · تثبيت الفهم

ماذا يجب أن يبقى في ذاكرتك؟

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

- ✓ المؤشر ليس القيمة بل عنوان لموضعها.
- ✓ * تستخدم للتصريح عن المؤشر وأيضاً لفك الإشارة بحسب السياق.
- ✓ التحقق من صلاحية المؤشر قبل فك الإشارة جزء من السلامة.

الرسم التوضيحي: كيف تعمل الفكرة؟



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

هكذا اكتملت رحلة برمجة 1: من المشكلة إلى خوارزمية ثم برنامج وإدارة بياناته.

المرجع: «برمجة 1» لم. بسمة سراقبي، الصفحات 152–156 من الملف المرفق. الأمثلة والشروح الإضافية صيغت للأغراض التعليمية وتستخدم C++17.