C++

کورس کامل

۱۷ درس · مبتدی تا پیشرفته

انگلیسی و دری · رایگان

- کلاس‌ها و آبجکت‌ها ✓
- کپسوله‌سازی و تجرید ✓
- وراثت و چندریختی ✓
- RAII و unique_ptr ✓
- template و STL ✓
- vector · map · الگوریتم‌ها ✓
- استثناها ✓
- ساختارهای دیتا · C++ مدرن ✓

همه را یاد بگیرید @MyWebUniversity-Dari

۱۷ درس این کورس



همه را یاد بگیرید @MyWebUniversity-Dari

۱. از C به C++



همه را یاد بگیرید @MyWebUniversity-Dari

۲. اساسات C++: نوع‌ها، مرجع‌ها، string و auto



همه را یاد بگیرید @MyWebUniversity-Dari

۳. تابع‌ها، اورلود و lambdaها



همه را یاد بگیرید @MyWebUniversity-Dari

۷. وراثت



همه را یاد بگیرید @MyWebUniversity-Dari

۸. چندریختی و تابع‌های مجازی



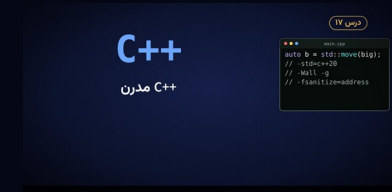
همه را یاد بگیرید @MyWebUniversity-Dari

۹. اورلود کردن عملگرها



همه را یاد بگیرید @MyWebUniversity-Dari

۱۳. تکرارکننده‌ها و الگوریتم‌ها



همه را یاد بگیرید @MyWebUniversity-Dari

۱۴. رشته‌ها و جریان‌های فایل



همه را یاد بگیرید @MyWebUniversity-Dari

۱۵. استثناها و مدیریت خطا

همه را یاد بگیرید @MyWebUniversity-Dari

۶. سازنده‌ها، تخریب‌کننده‌ها و RAII

همه را یاد بگیرید @MyWebUniversity-Dari

۵. کپسوله‌سازی، پنهان‌سازی دیتا و تجرید

همه را یاد بگیرید @MyWebUniversity-Dari

۴. کلاس‌ها و آبجکت‌ها

همه را یاد بگیرید @MyWebUniversity-Dari

۱۲. ظرف‌های STL: vector، map و بیشتر

همه را یاد بگیرید @MyWebUniversity-Dari

۱۱. templateها و برنامه‌نویسی عمومی

همه را یاد بگیرید @MyWebUniversity-Dari

۱۰. حافظه و اشاره‌گرهای هوشمند

همه را یاد بگیرید @MyWebUniversity-Dari

۱۷. C++ مدرن، پروژه‌ها و قدم‌های بعدی

همه را یاد بگیرید @MyWebUniversity-Dari

۱۶. ساختارهای دیتا در C++

یک کورس رایگان از MyWebUniversity

برنامه‌نویسی C++

درس ۱

از C به C++

C++ چه اضافه می‌کند، معیارها، و اولین پروگرام



C++ چیست؟

- رویه‌ای — مثل C
- شیءگرا — کلاس‌ها، آبجکت‌ها
- عمومی — templateها



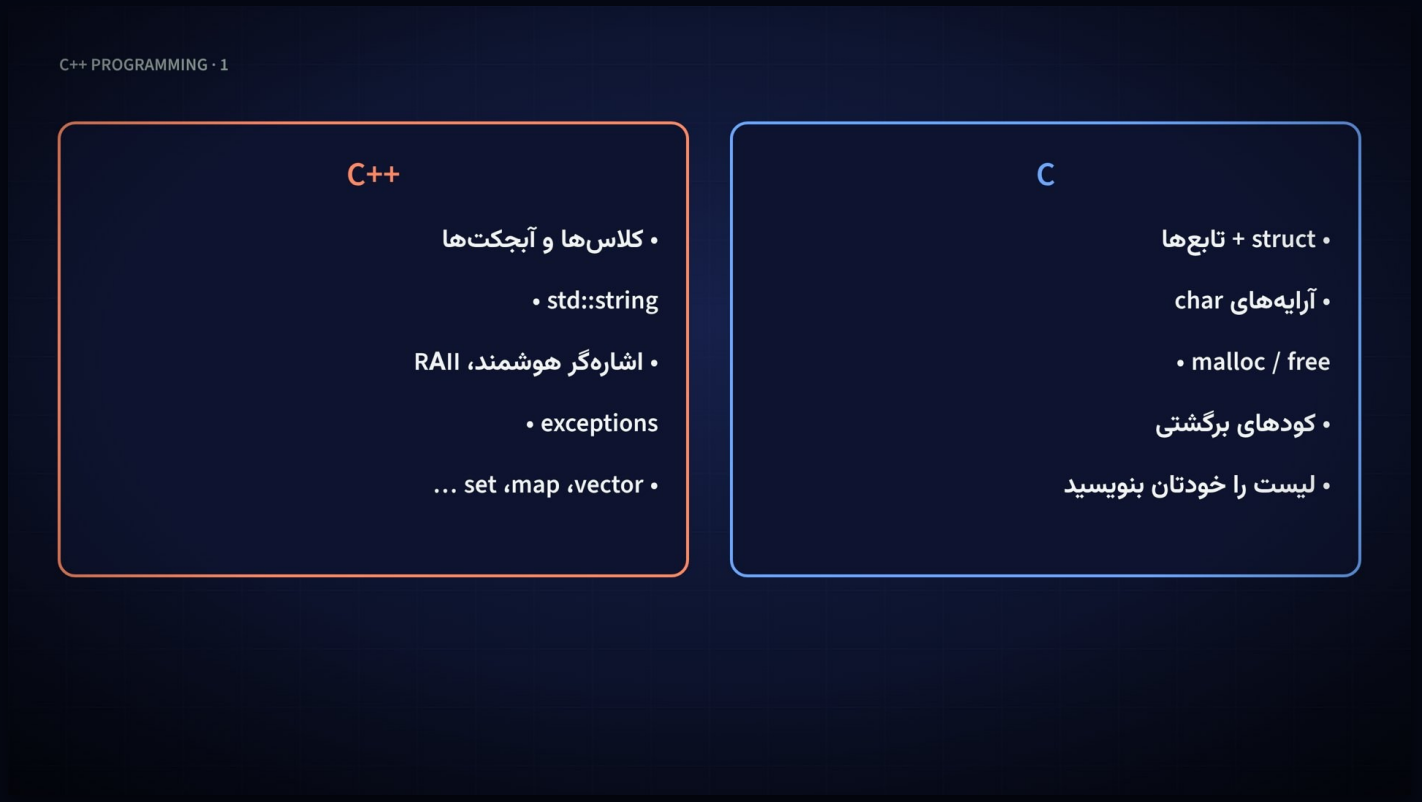
۱.۰۱

C++ به C چه اضافه می‌کند

C++ کلاس‌ها و آبجکت‌ها را اضافه می‌کند تا دیتا و رفتار یکجا زندگی کنند. مرجع‌ها، ...

کتابخانهٔ استاندارد آن ظرف‌های آماده مثل vector و map، یک نوع رشتهٔ واقعی، الگوریتم‌ها، و ...

تقریباً همهٔ کود C، C++ معتبر هم است، پس هر چیزی که در کورس C یاد گرفتید هنوز به ...



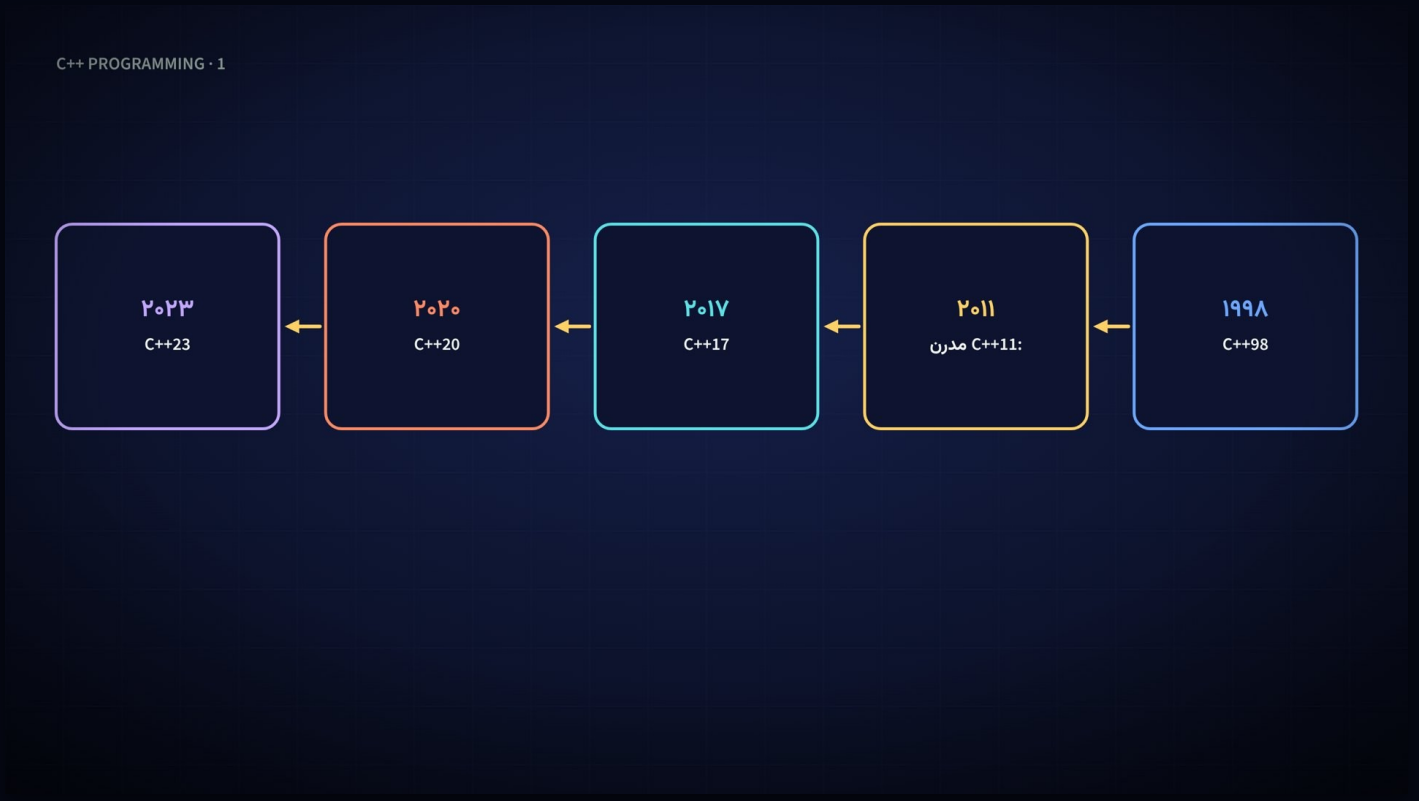
۱.۰۲

معیارهای C++

C++98 — ۱۹۹۸

C++11: — ۲۰۱۱ مدرن

C++17 — ۲۰۱۷



۱.۰۳

سلام، C++

<iostream>> — ورودی و خروجی

std::cout << — چاپ روی صفحه

std:: — namespace استاندارد

C++ PROGRAMMING · 1

hello.cpp

```cpp
#include <iostream>

int main() {
    std::cout << "Hello, C++!" << std::endl;
    return 0;
}
```

<iostream>
ورودی و خروجی

std::cout <<
چاپ روی صفحه

std::
namespace استاندارد

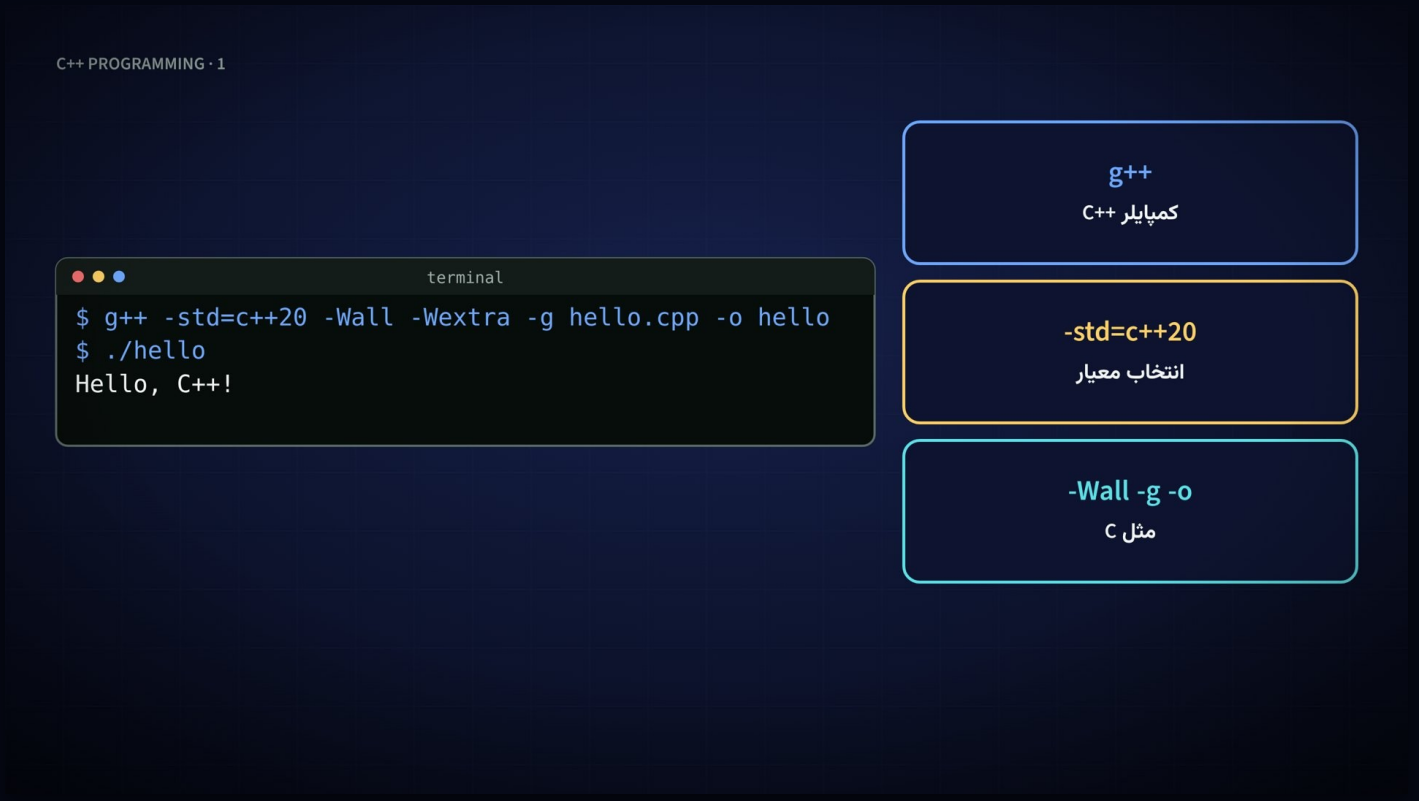
۱۰۰۴

کمپایل با g++

g++ — کمپایلر C++

-std=c++20 — انتخاب معیار

-Wall -g -o — مثل C



۱.۰۵

ورودی با cin

std دو نقطه دو نقطه cin ورودی را با دو علامت بزرگ‌تر می‌خواند. نوع هر متغیر را ...

برای خواندن یک سطر کامل با فاصله‌ها، از getline با یک std::string استفاده کنید.

C++ PROGRAMMING · 1

input.cpp

```cpp
#include <iostream>
#include <string>

int main() {
    std::string name;
    int age;
    std::cout << "Full name: ";
    std::getline(std::cin, name);
    std::cout << "Age: ";
    std::cin >> age;
    std::cout << "Hello, " << name << "! Next year: " << age + 1 << "\n";
}
```

۱.۰۶

نقشهٔ کورس

۱–۳ — اساسات · تابع‌ها

۴–۹ — کلاس‌ها · چهار ستون

۱۰–۱۳ — حافظه · STL · template



۱.۰۷

یک کورس رایگان از MyWebUniversity

برنامه‌نویسی C++

درس ۲

اساسات C++: نوع‌ها، مرجع‌ها، string و auto

ابزارهای روزمرهٔ C++ مدرن



نوع‌ها و مقداردهی

C++ همهٔ نوع‌های C را دارد، و یک نوع bool واقعی با true و false.

C++ مدرن مقداردهی با قوس را ترجیح می‌دهد. برای هر نوع کار می‌کند و تبدیل‌های کاهنده ...

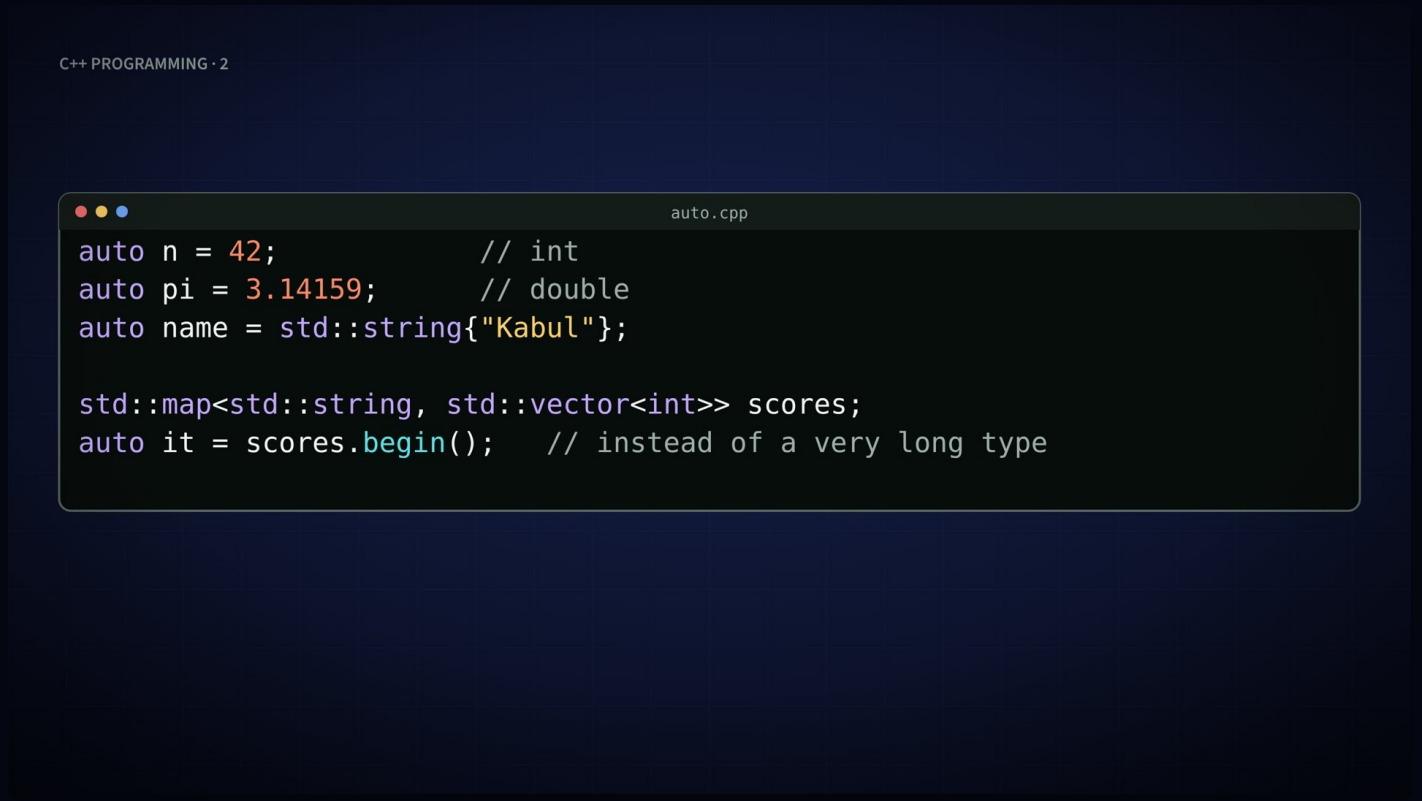


۲.۰۱

auto

auto به کمپایلر اجازه می‌دهد نوع را از مقدار حدس بزند. برای نام‌های طولانی نوع، مخصوصاً با ...

وقتی نوع واضح یا بی‌اهمیت است از آن استفاده کنید. متغیر هنوز یک نوع ثابت دارد؛ C++ ...



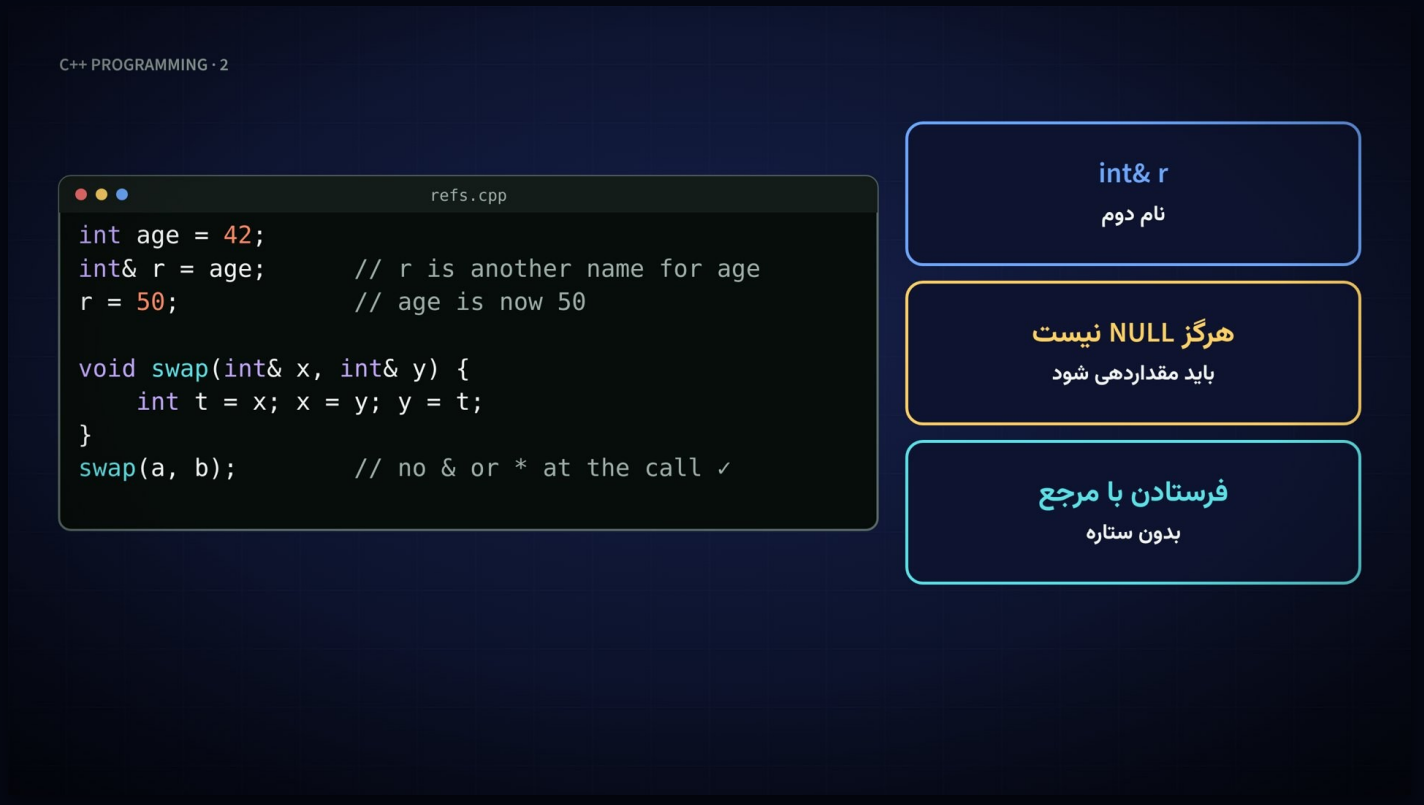
۲.۰۲

مرجع‌ها

int& r — نام دوم

هرگز NULL نیست — باید مقداردهی شود

فرستادن با مرجع — بدون ستاره

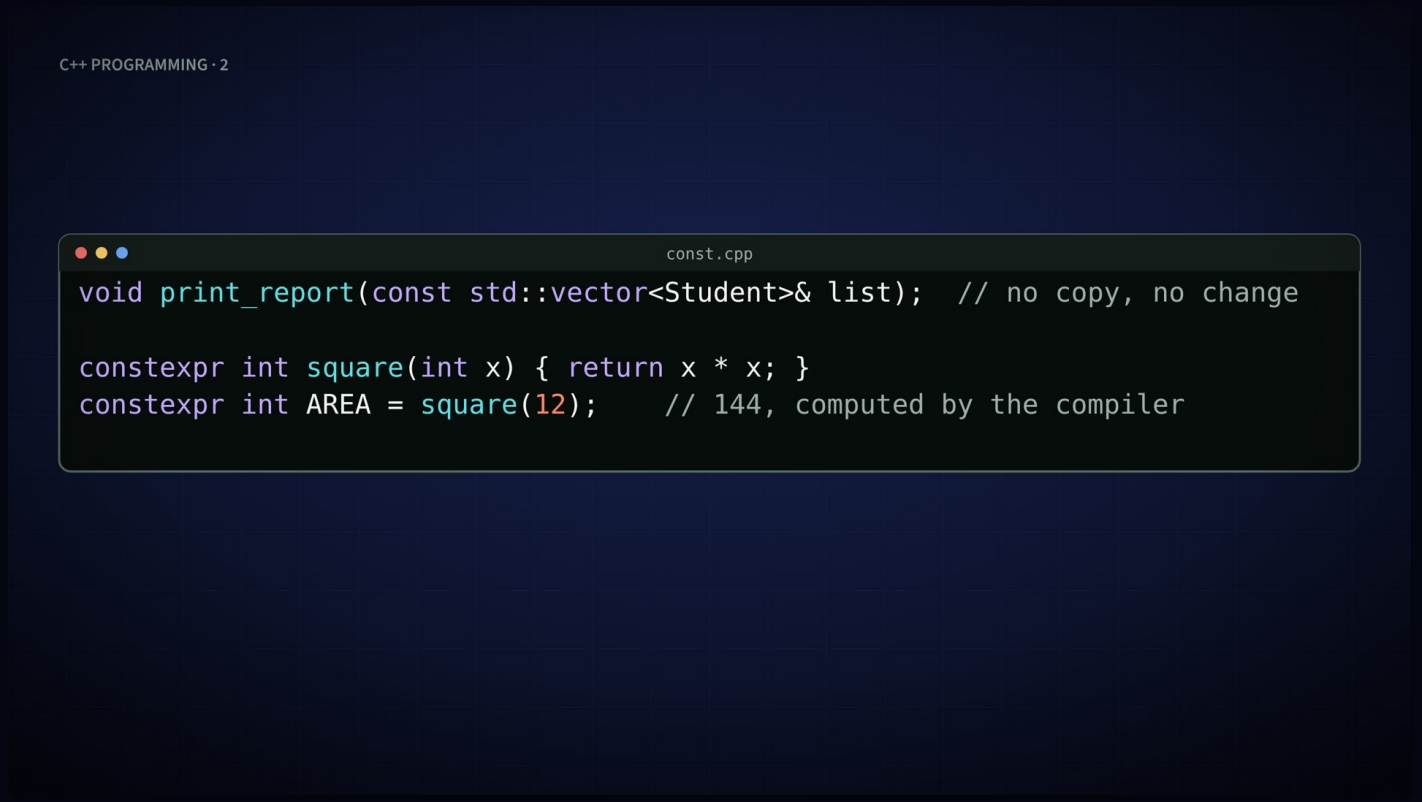


۲.۰۳

const و const expr

const یعنی یک مقدار تغییر نمی‌کند. فرستادن آبجکت‌های بزرگ به عنوان مرجع const از کاپی ...

const expr یک قدم بیشتر می‌رود: مقدار در وقت کمپایل حساب می‌شود، پس هنگام اجرای پروگرام ...



۲.۰۴

std::string

+ — یکجا کردن

== — مقایسهٔ متن

.size() .find() — طول، جستجو

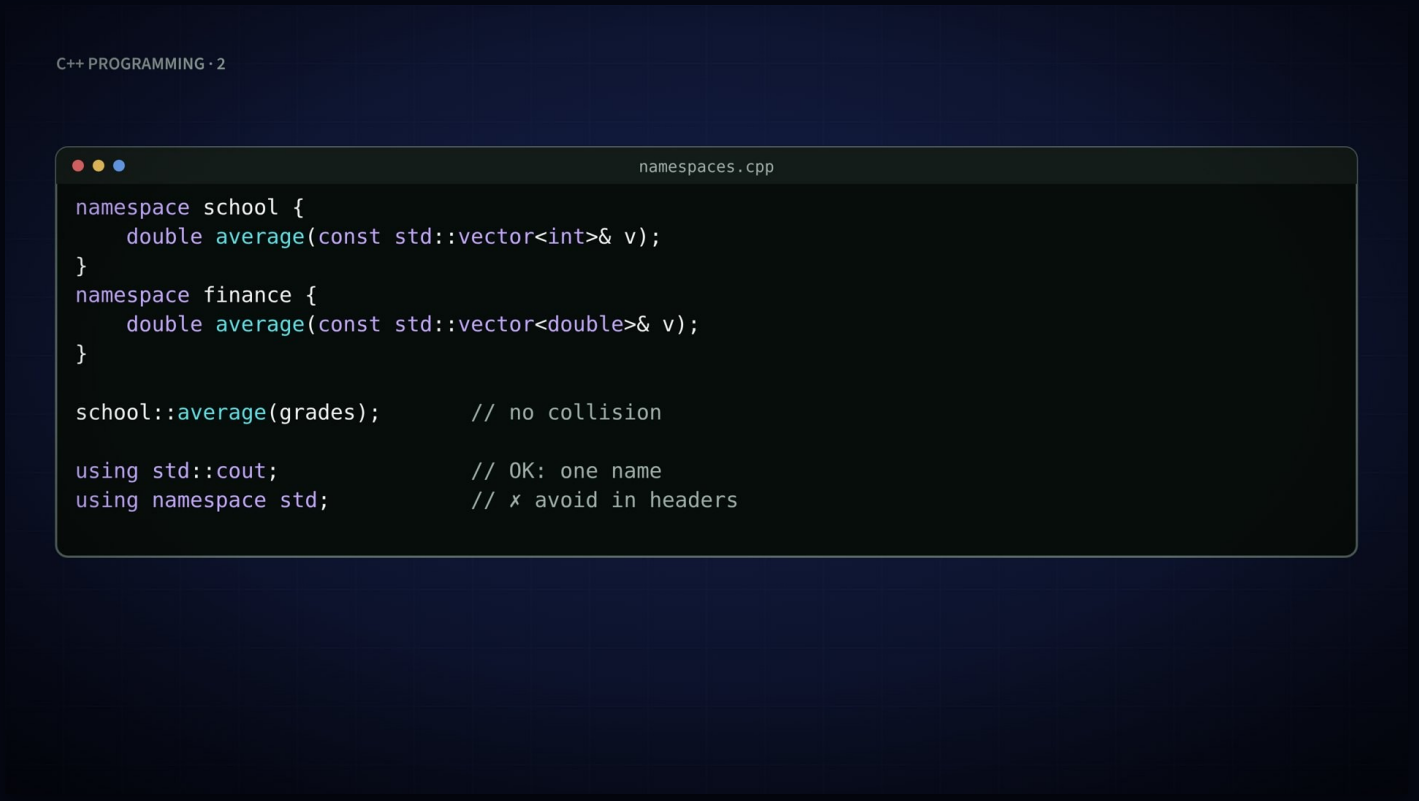


۲.۰۵

namespaceها

namespace نام‌ها را گروه می‌کند تا از برخورد جلوگیری شود. دو کتابخانه هر دو می‌توانند ...

کتابخانهٔ استاندارد در namespace std زندگی می‌کند. نوشتن using namespace اِس تی ...

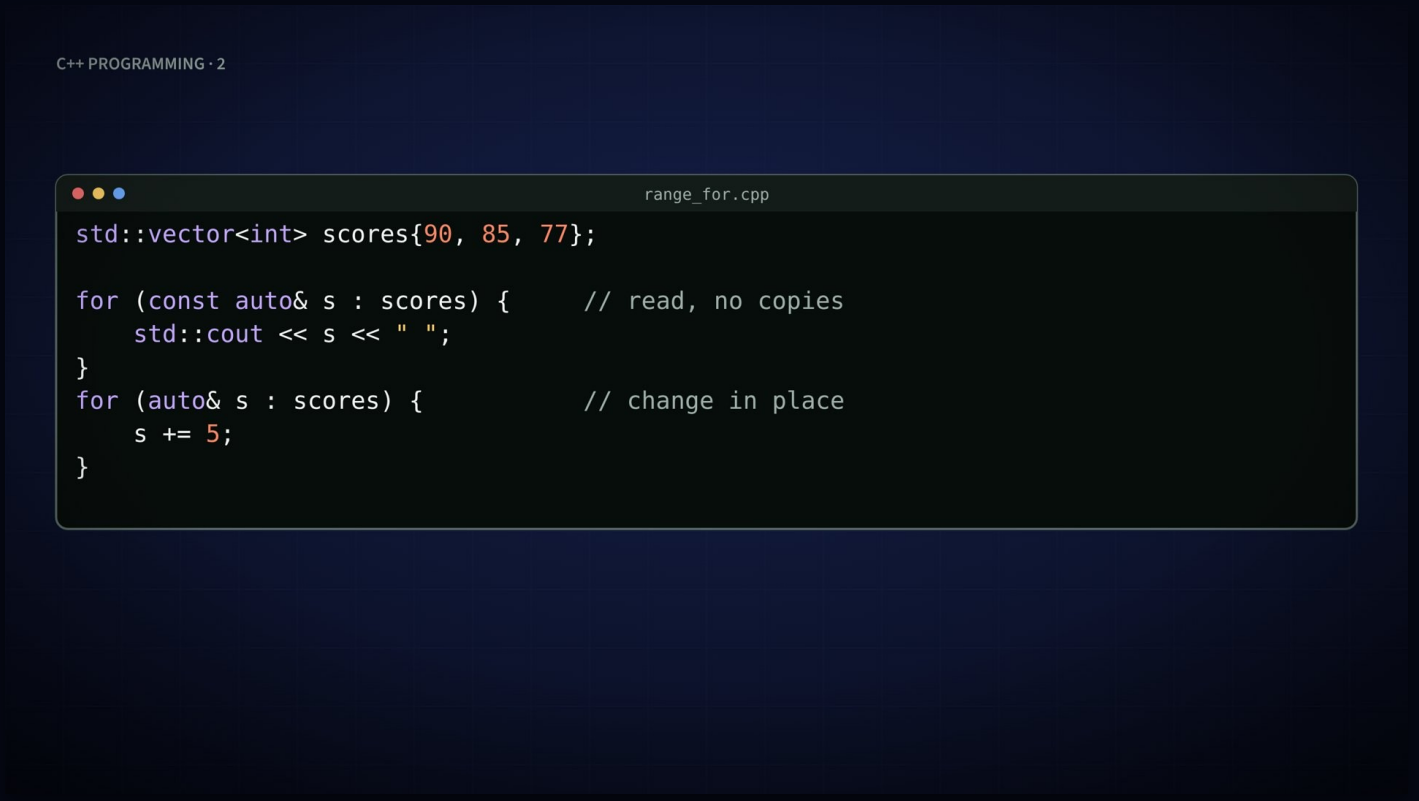


۲.۰۶

حلقهٔ for بر اساس محدوده

C++ مدرن یک حلقهٔ for ساده‌تر دارد که از هر عنصر یک ظرف دیدن می‌کند. به اندکس ضرورت ...

برای خواندن عناصر بدون کاپی از مرجع const استفاده کنید، و برای تغییر آن‌ها از مرجع عادی.



۲.۰۷

یک کورس رایگان از MyWebUniversity

برنامه‌نویسی C++

درس ۳

تابع‌ها، اورلود و lambdaها

آرگومان‌های پیش‌فرض، تابع‌های inline و عبارت‌های lambda



اورلود کردن تابع

در C، هر تابع به یک نام یکتا ضرورت دارد. در C++، چند تابع می‌توانند یک نام داشته ...

کمپایلر نسخهٔ درست را از روی آرگومان‌هایی که می‌فرستید انتخاب می‌کند. این اولین نوع ...

overload.cpp

```cpp
int    area(int side)              { return side * side; }
int    area(int w, int h)          { return w * h; }
double area(double radius)         { return 3.14159 * radius * radius; }

area(4);        // 16      → square
area(3, 5);     // 15      → rectangle
area(2.0);      // 12.566  → circle
```

۳.۰۱

آرگومان‌های پیش‌فرض

یک پارامتر می‌تواند مقدار پیش‌فرض داشته باشد که وقتی صدا کننده آن را نفرستد استفاده می‌شود. ...

این اکثراً چند اورلود را با یک تابع روشن جایگزین می‌کند.

defaults.cpp

```cpp
void greet(const std::string& name, const std::string& greeting = "Hello") {
    std::cout << greeting << ", " << name << "!\n";
}

greet("Mariam");            // Hello, Mariam!
greet("Ahmad", "Salaam");   // Salaam, Ahmad!
```

۳.۰۲

انتخاب شیوهٔ فرستادن

این یک قاعدهٔ ساده است. نوع‌های کوچک مثل int و double را با مقدار بفرستید. آبجکت‌های بزرگی ...

وقتی تابع باید آبجکت صدا کننده را تغییر دهد، با مرجع غیر const بفرستید. اشاره‌گرها برای ...

پارامتر	کی
`int x`	مقدارهای کوچک: کاپی ارزان است
`const std::string& s`	آبجکت‌های بزرگ فقط خواندنی
`std::vector<int>& v`	تابع آن را تغییر می‌دهد
`Node* p`	اختیاری، شاید nullptr

۳.۰۳

تابع‌های inline

صدا کردن یک تابع هزینهٔ کوچکی دارد. برای تابع‌های خیلی کوچک، کمپایلر می‌تواند بدنه را ...

کمپایلرهای مدرن هنگام بهینه‌سازی این را خودشان تصمیم می‌گیرند. امروز کلمهٔ inline بیشتر ...

C++ PROGRAMMING · 3

math_utils.h

```
#pragma once

inline int clamp(int v, int lo, int hi) {
    return v < lo ? lo : (v > hi ? hi : v);
}

// safer than the C macro: real types, arguments evaluated once
```

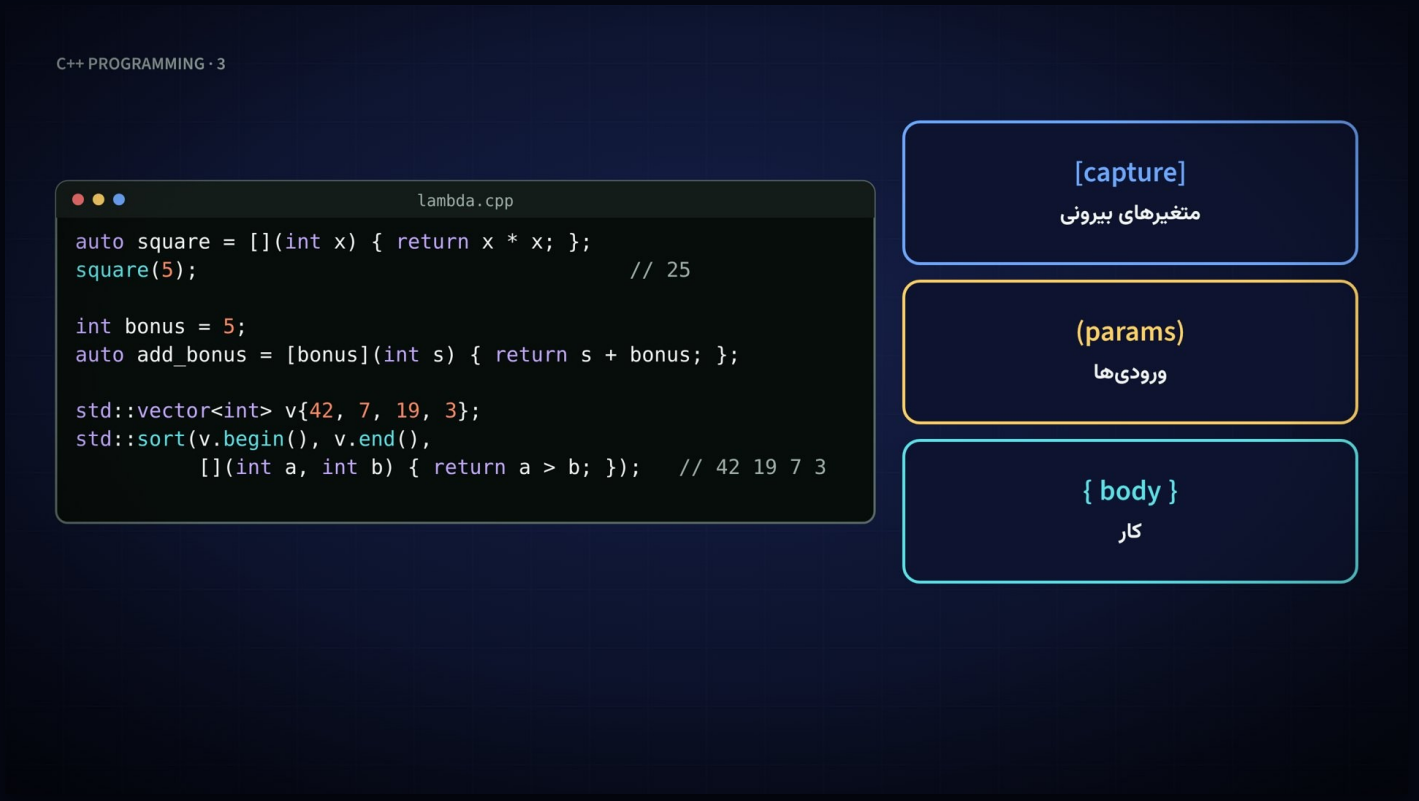
۳.۰۴

عبارت‌های lambda

[capture] — متغیرهای بیرونی

(params) — ورودی‌ها

{ body } — کار



۳.۰۵

گرفتن با مقدار و با مرجع

گرفتن با مقدار متغیر را در lambda کاپی می‌کند.
گرفتن با اَمپرسَند از اصلی استفاده می‌کند، ...

مواظب باشید: لامبدایی که با مرجع می‌گیرد نباید بیشتر از متغیرهایی که به آن‌ها اشاره می‌کند ...



۳.۰۶

یک کورس رایگان از MyWebUniversity

برنامه‌نویسی C++

درس ۴

کلاس‌ها و آبجکت‌ها

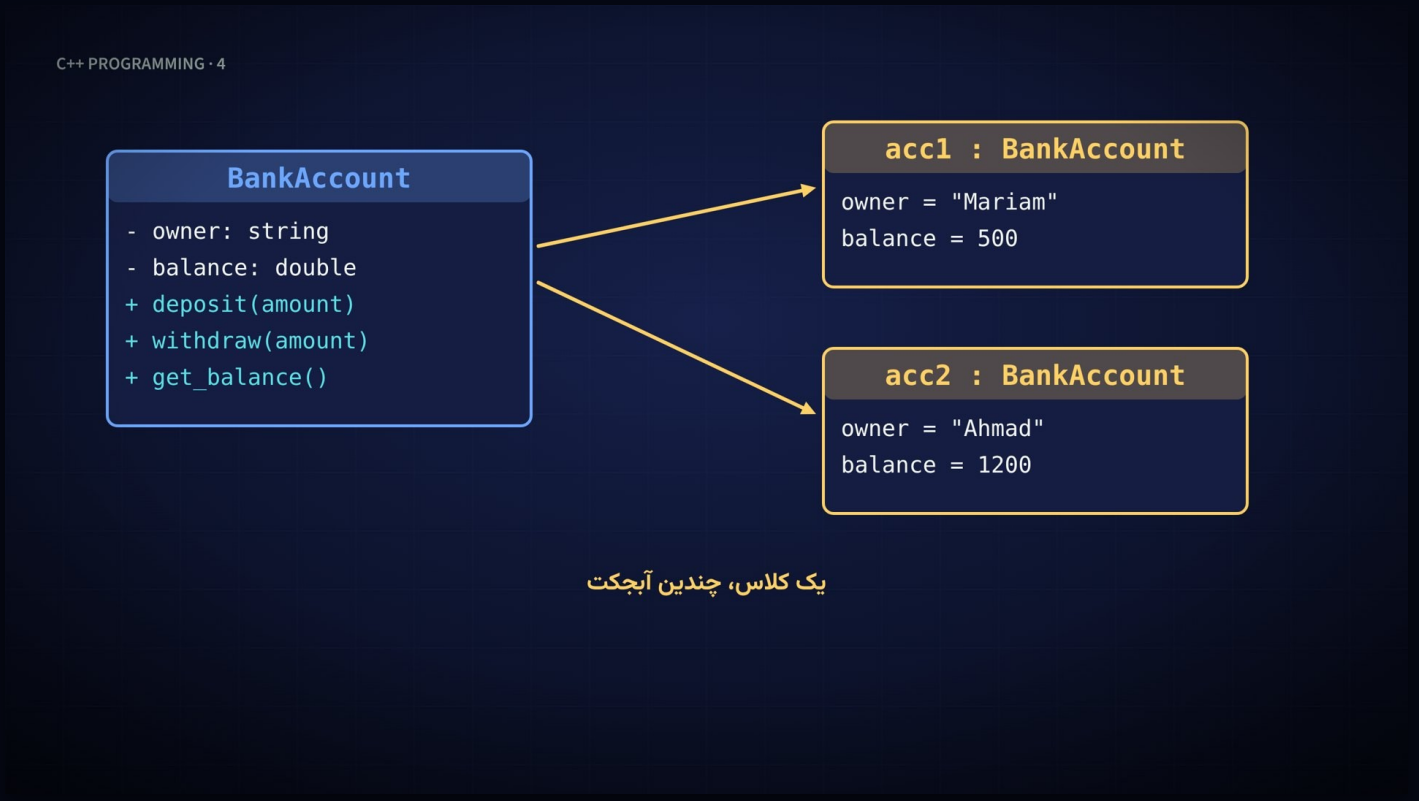
یکجا کردن دیتا و رفتار



کلاس: یک نقشه

کلاس نقشه‌ای است که یک نوع چیز را توصیف می‌کند: چه دیتایی نگه می‌دارد و چه کارهایی انجام ...

یک کلاس حساب بانکی می‌تواند هزاران آبجکت حساب بسازد، هر کدام با مالک و موجودی خود، اما همه ...



۴.۰۱

تعریف یک کلاس

متغیرهای عضو — دیتا

تابع‌های عضو — رفتار

public / private — چه کسی از چه استفاده کند



۴.۰۲

ساختن و استفاده از آبجکت‌ها

ساختن یک آبجکت مثل اعلان یک متغیر است که نوع آن کلاس باشد. متودها را با عملگر نقطه صدا کنید.

تلاش برای دست زدن به یک عضو خصوصی از بیرون خطای کمپایل است. کلاس دیتای خود را حفظ می‌کند: ...

main.cpp

```
BankAccount acc;
acc.deposit(500);
acc.deposit(-90);              // ignored by the class rule
std::cout << acc.get_balance();  // 500

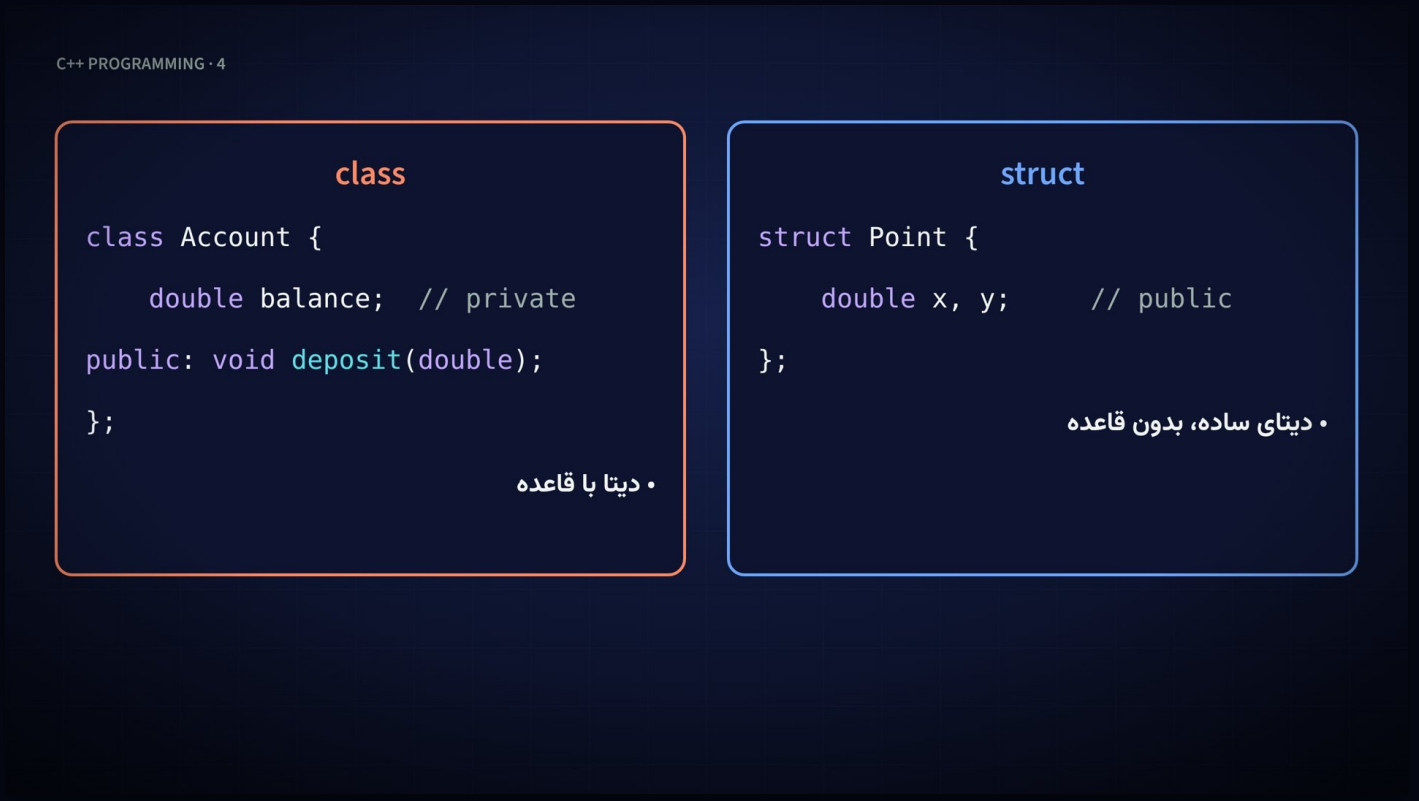
acc.balance = -1000;   // ✗ error: 'balance' is private
```

۴.۰۳

کلاس در برابر struct

در C++، struct و کلاس تقریباً یکی هستند. تنها فرق: اعضای struct به طور پیش‌فرض عمومی ...

به رسم معمول، برای بسته‌های سادهٔ دیتا بدون قاعده از struct استفاده کنید، و وقتی آبجکت ...



۴۰۴

اشاره‌گر this و متودهای const

داخل یک تابع عضو، this اشاره‌گری به آبجکتی است که تابع روی آن صدا شده. همین طور متود می‌داند ...

متودی که const علامه زده شود وعده می‌دهد آبجکت را تغییر ندهد. گِترها باید const باشند تا ...

C++ PROGRAMMING · 4

this.cpp

```cpp
class Counter {
public:
    Counter& add(int n) { this->value += n; return *this; }
    int get() const { return value; }      // const: read only
private:
    int value = 0;
};

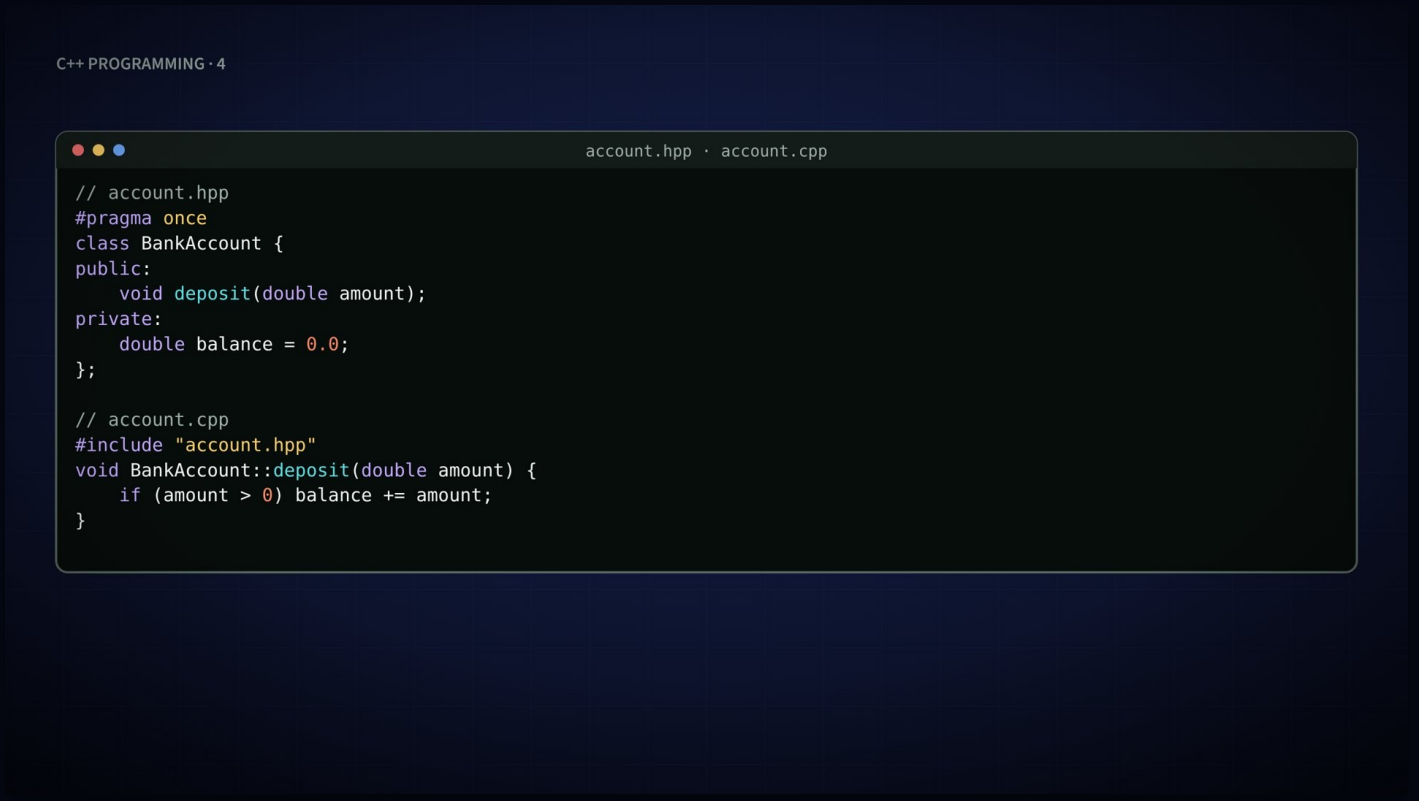
Counter c;
c.add(2).add(3);          // chaining via *this
std::cout << c.get();     // 5
```

۴.۰۵

فایل‌های هِدر و منبع

پروژه‌های واقعی یک کلاس را دو بخش می‌کنند. هِدر، با پسوند .h یا .hpp، تعریف …

فایل منبع، که با .cpp تمام می‌شود، بدنهٔ متودها را نگه می‌دارد که با نام کلاس و دو …



۴.۰۶

یک کورس رایگان از MyWebUniversity

برنامه‌نویسی C++

درس ۵

کپسوله‌سازی، پنهان‌سازی دیتا و تجرید

دو ستون از چهار ستون برنامه‌نویسی شیءگرا



چهار ستون

کپسوله‌سازی — امروز

تجرید — امروز

وراثت — درس ۷

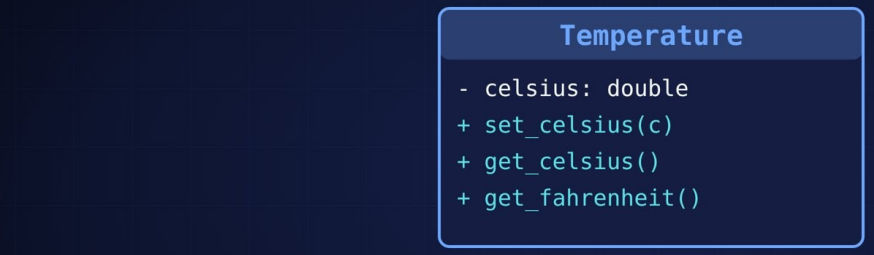


۵.۰۱

کپسوله‌سازی

کپسوله‌سازی یعنی یکجا کردن دیتا با تابع‌هایی که روی آن کار می‌کنند، داخل یک کلاس، مثل دوا ...

دنیای بیرون تنها از طریق متودهای عمومی با آبجکت صحبت می‌کند. آبجکت تصمیم می‌گیرد دیتای آن ...



دیتا و تابع‌هایی که از آن استفاده می‌کنند، در یک واحد

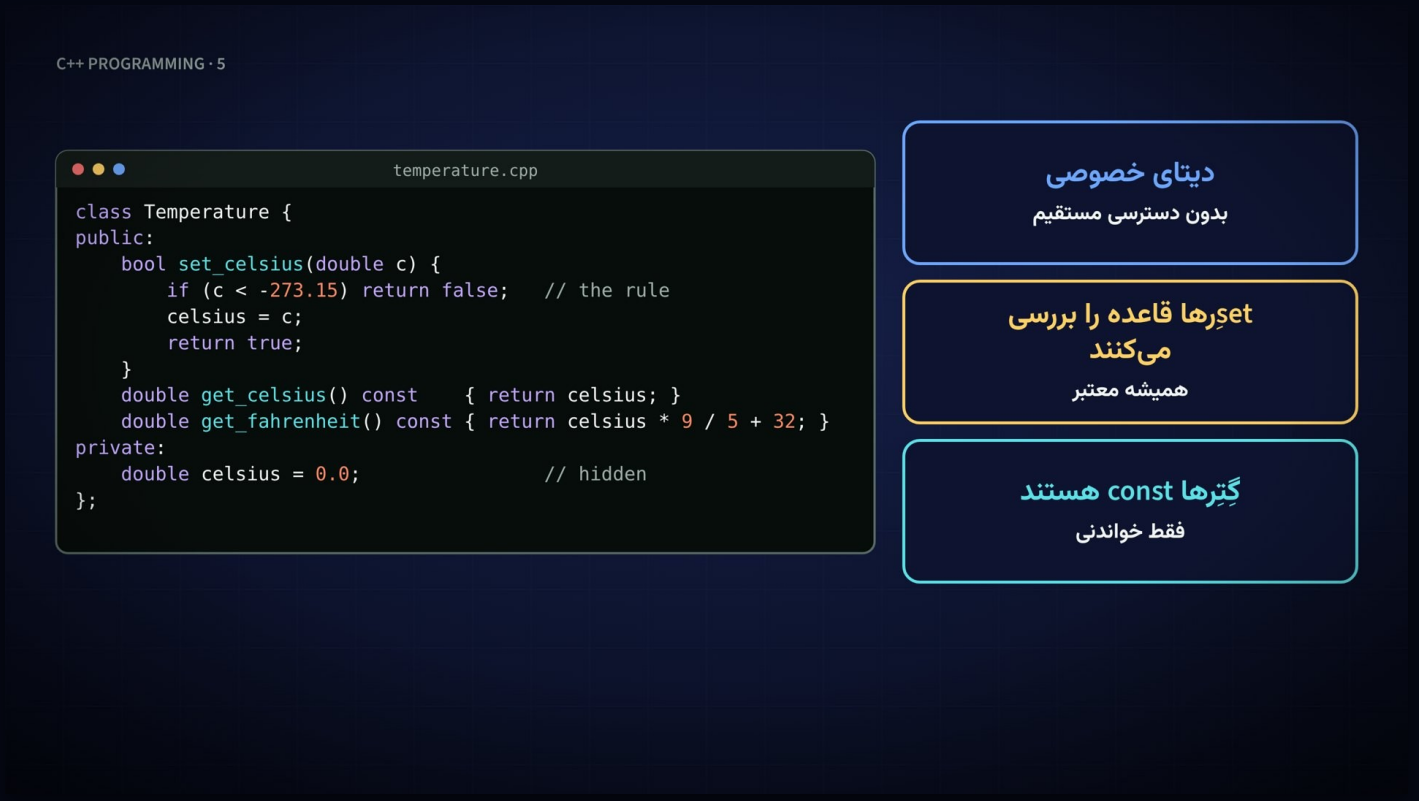
۵.۰۲

پنهان‌سازی دیتا با private

دیتای خصوصی — بدون دسترسی مستقیم

set_ها قاعده را بررسی می‌کنند — همیشه معتبر

گِترها const هستند — فقط خواندنی



۵.۰۳

protected و public، private

C++ سه سطح دسترسی دارد. اعضای public را هر کسی استفاده کرده می‌تواند. اعضای private ...

اعضای protected مثل private هستند، اما کلاس‌های مشتق هم می‌توانند از آن‌ها استفاده کنند؛ این ...

دسترسی	خود کلاس	کلاس مشتق	همه
public	✓	✓	✓
protected	✓	✓	✗
private	✓	✗	✗

۵.۰۴

تجرید

تجرید یعنی تنها نشان دادن آنچه استفاده‌کننده لازم دارد، و پنهان کردن اینکه چطور کار می‌کند. ...

std::vector یک مثال عالی است. push_back و size را صدا می‌کنید، و هرگز به دوباره‌رزرو کردن ...



۵.۰۵

رابط‌های تجریدی

می‌توانید یک تجرید را به شکل یک رابط توصیف کنید: کلاسی که تنها تابع‌های مجازی خالص دارد که ...

کودی که از رابط استفاده می‌کند با هر کلاسی که آن را پیاده کند کار می‌کند. در درس چندریختی ...

storage.hpp

```cpp
class Storage {                          // an interface
public:
    virtual ~Storage() = default;
    virtual void save(const std::string& key, const std::string& value) = 0;
    virtual std::string load(const std::string& key) = 0;
};

class FileStorage  : public Storage { /* saves to files    */ };
class CloudStorage : public Storage { /* saves to a server */ };
```

۵.۰۶

چرا مهم است

باگ کمتر — دیتا همیشه معتبر

تغییر آسان — داخل قابل بازنویسی

کار تیمی — رابط‌های روشن



۵.۰۷

یک کورس رایگان از MyWebUniversity

برنامه‌نویسی C++

درس ۶

سازنده‌ها، تخریب‌کننده‌ها و RAII

آبجکت‌ها چطور ساخته، کاپی، منتقل و نابود می‌شوند



سازنده‌ها

همان نام کلاس — بدون نوع برگشتی

: member{value} — لیست مقداردهی



۶.۰۱

تخریب‌کننده‌ها

تخریب‌کننده وقتی آبجکت نابود می‌شود خودکار اجرا می‌شود: وقتی از حوزه بیرون می‌رود، یا وقتی ...

کار آن پاک‌کاری است: بستن فایل‌ها، آزاد کردن حافظه، باز کردن قفل‌ها. آبجکت‌ها به ترتیب ...

lifetime.cpp

```cpp
class Logger {
public:
    Logger()  { std::cout << "start\n"; }
    ~Logger() { std::cout << "end\n"; }       // destructor
};

void work() {
    Logger log;       // prints: start
    /* ... */
}                     // prints: end  (automatic!)
```

۶.۰۲

RAII: مهم‌ترین ایده

RAII یعنی گرفتن منبع، مقداردهی است. یعنی:
منبع را در سازنده بگیر، و در تخریب‌کننده ...

چون تخریب‌کننده‌ها خودکار اجرا می‌شوند، منبع
همیشه آزاد می‌شود، حتی اگر تابع زود برگردد یا ...

std::string، vector، جریان‌های فایل و
اشاره‌گرهای هوشمند همین طور حافظه و فایل‌ها را ...



۶.۰۳

کاپی کردن آبجکت‌ها

وقتی یک آبجکت را کاپی می‌کنید، سازندهٔ کاپی اجرا می‌شود. وقتی یک آبجکت موجود را به دیگری ...

برای اعضایی مثل رشته‌ها و vectorها، کاپی پیش‌فرض کار درست را می‌کند. اما اگر کلاس شما مالک ...

C++ PROGRAMMING · 6

copy.cpp

```cpp
Student a{"Mariam", 1};
Student b = a;          // copy constructor
Student c{"Ahmad", 2};
c = a;                  // copy assignment

class Buffer {          // ✗ dangerous
    int* data = new int[100];
    ~Buffer() { delete[] data; }
};                      // default copy → double delete!
```

۶.۰۴

قاعدهٔ صفر، سه و پنج

قاعدهٔ صفر ✓ — string، vector، اشاره‌گر هوشمند

قاعدهٔ سه — تخریب‌کننده + سازندهٔ کاپی + کاپی =

قاعدهٔ پنج — + سازندهٔ انتقال + انتقال =



۶.۰۵

مفهوم انتقال

کاپی کردن یک vector بزرگ هر عنصر را کاپی می‌کند. اما وقتی منبع به هر حال در حال نابود شدن ...

std::move می‌گوید دیگر به یک آبجکت ضرورت ندارید، پس می‌توان از آن انتقال داد. برگرداندن ...

C++ PROGRAMMING · 6

move.cpp

```cpp
std::vector<int> big(10'000'000, 1);

std::vector<int> a = big;              // copy: 10 million ints
std::vector<int> b = std::move(big);   // move: just pointers
                                       // big is now empty
std::vector<int> make_data() {
    std::vector<int> v(1000);
    return v;                          // moved, not copied
}
```

۶.۰۶

یک کورس رایگان از MyWebUniversity

برنامه‌نویسی C++

درس ۷

وراثت

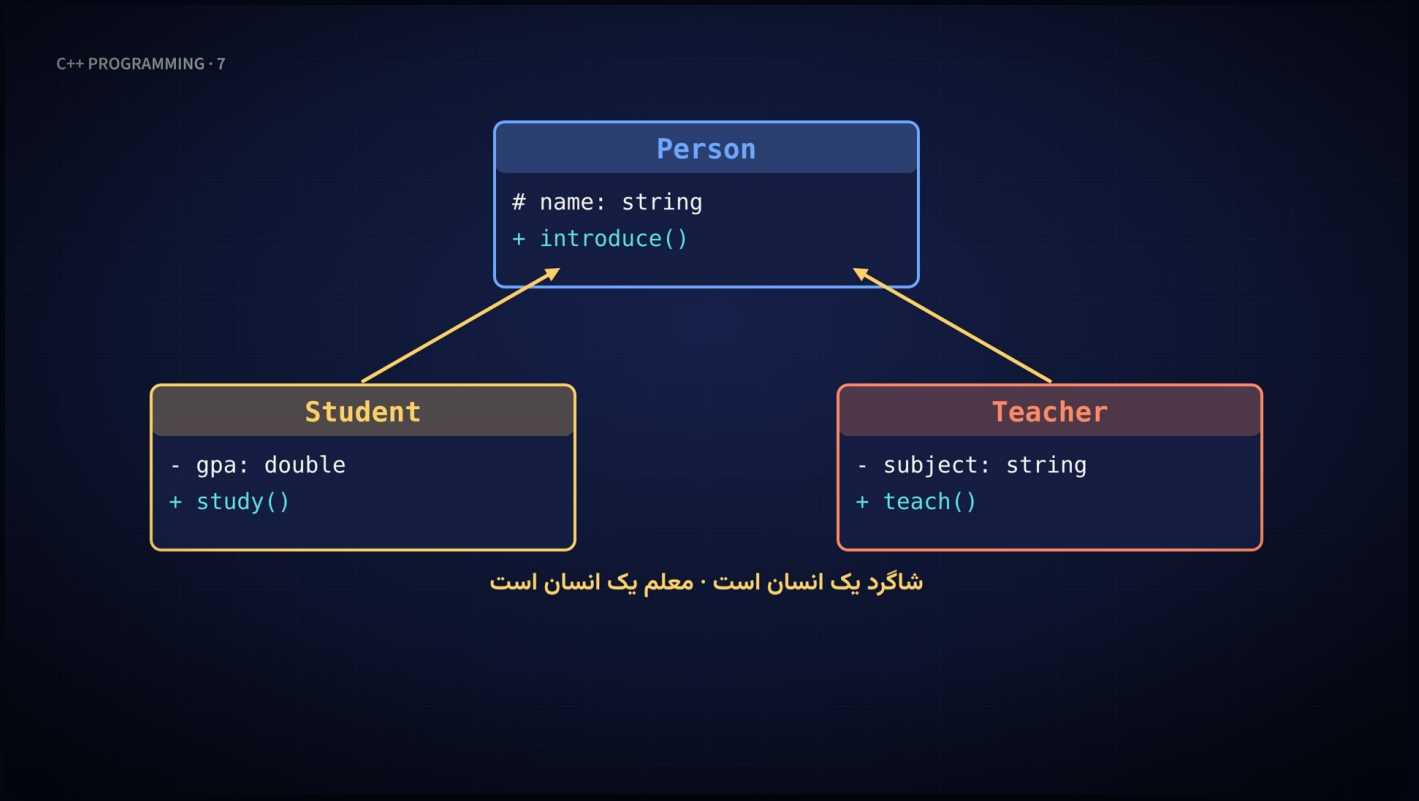
استفادهٔ دوباره و گسترش کلاس‌ها



رابطهٔ «یک نوع از»

وراثت به یک کلاس نو، که کلاس مشتق گفته می‌شود، اجازه می‌دهد همه چیز را از یک کلاس پایهٔ ...

از وراثت برای رابطهٔ «یک نوع از» استفاده کنید. حساب پس‌انداز یک نوع حساب بانکی است. دایره ...



۷.۰۱

مشتق کردن یک کلاس

بعد از نام کلاس یک دو نقطه و public بنویسید، بعد کلاس پایه را. کلاس مشتق همهٔ اعضای public ...

سازندهٔ آن باید اول سازندهٔ پایه را در لیست مقداردهی صدا کند، تا بخش پایه قبل از بخش نو ...

C++ PROGRAMMING · 7

person.cpp

```cpp
class Person {
public:
    Person(std::string n) : name{std::move(n)} {}
    void introduce() const { std::cout << "I am " << name << "\n"; }
protected:
    std::string name;
};

class Student : public Person {
public:
    Student(std::string n, double g) : Person{std::move(n)}, gpa{g} {}
    void study() const { std::cout << name << " is studying\n"; }
private:
    double gpa;
};
```

۷.۰۲

استفاده از آبجکت‌های مشتق

یک آبجکت شاگرد می‌تواند اینترودیوس را که به ارث برده، و اِستادی را که اضافه کرده، صدا کند.

وقتی یک شاگرد ساخته می‌شود، اول بخش انسان ساخته می‌شود، بعد بخش شاگرد. وقتی نابود می‌شود، ...

main.cpp

```cpp
Student s{"Mariam", 3.9};
s.introduce();     // inherited from Person: I am Mariam
s.study();         // added by Student:      Mariam is studying

// construction:  Person → Student
// destruction:   ~Student → ~Person
```

۷.۰۳

تغییر رفتار

یک کلاس مشتق می‌تواند یک تابع کلاس پایه را با نسخهٔ خود جایگزین کند. تابع پایه را virtual و ...

override از کمپایلر می‌خواهد بررسی کند که واقعاً یک تابع پایه را جایگزین می‌کنید. یک اشتباه ...

C++ PROGRAMMING · 7

override.cpp

```cpp
class Person {
public:
    virtual void introduce() const { std::cout << "I am " << name << "\n"; }
    virtual ~Person() = default;
protected:
    std::string name;
};
class Teacher : public Person {
public:
    void introduce() const override {
        std::cout << "I teach " << subject << "\n";
    }
private:
    std::string subject;
};
```

۷.۰۴

«یک نوع از» در برابر «دارای»

هر رابطه نباید از وراثت استفاده کند. موتر یک انجن دارد؛ خودش انجن نیست. برای رابطه‌های ...

یک قاعدهٔ خوب: ترکیب را ترجیح دهید، و تنها وقتی از وراثت استفاده کنید که آزمون «یک نوع از» ...



۷.۰۵

وراثت چندگانه

C++ اجازه می‌دهد یک کلاس از بیشتر از یک کلاس پایه ارث ببرد. قوی است اما می‌تواند ...

استفادهٔ مطمئن و رایج، ارث بردن از یک کلاس پایهٔ واقعی و رابط‌های کوچکی است که تنها ...

multiple.cpp

```cpp
class Printable { public: virtual void print() const = 0; virtual ~Printable() = default; };
class Saveable  { public: virtual void save()  const = 0; virtual ~Saveable()  = default; };

class Report : public Document, public Printable, public Saveable {
    void print() const override { /* ... */ }
    void save()  const override { /* ... */ }
};
```

۷.۰۶

یک کورس رایگان از MyWebUniversity

برنامه‌نویسی C++

درس ۸

چندریختی و تابع‌های مجازی

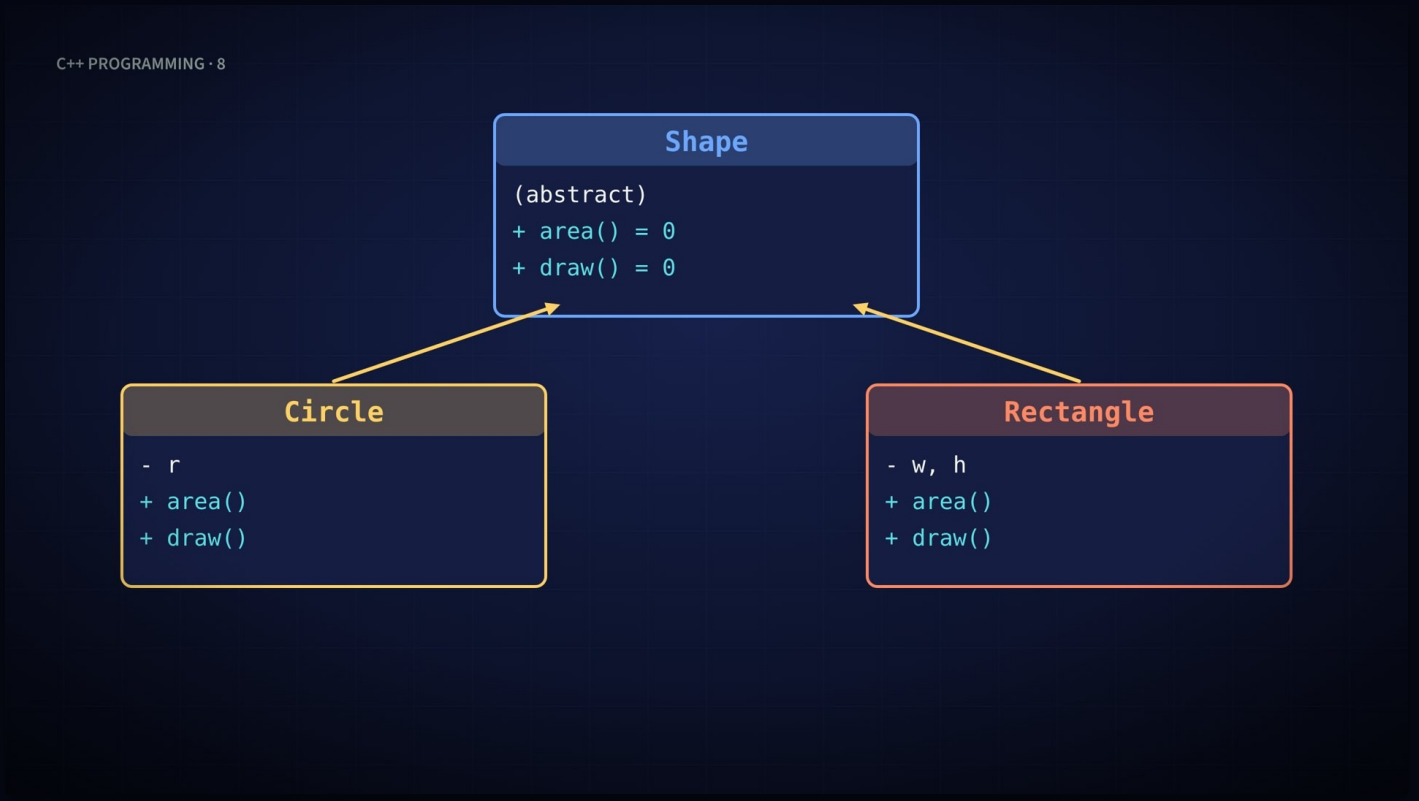
یک رابط، چندین شکل



چندین شکل

چندریختی یعنی چندین شکل. یک قطعه کود می‌تواند با آبجکت‌های کلاس‌های مختلف کار کند، به شرطی ...

یک پروگرام رسم یک لیست از شکل‌ها نگه می‌دارد و به سادگی روی هر کدام دراو را صدا می‌کند. ...

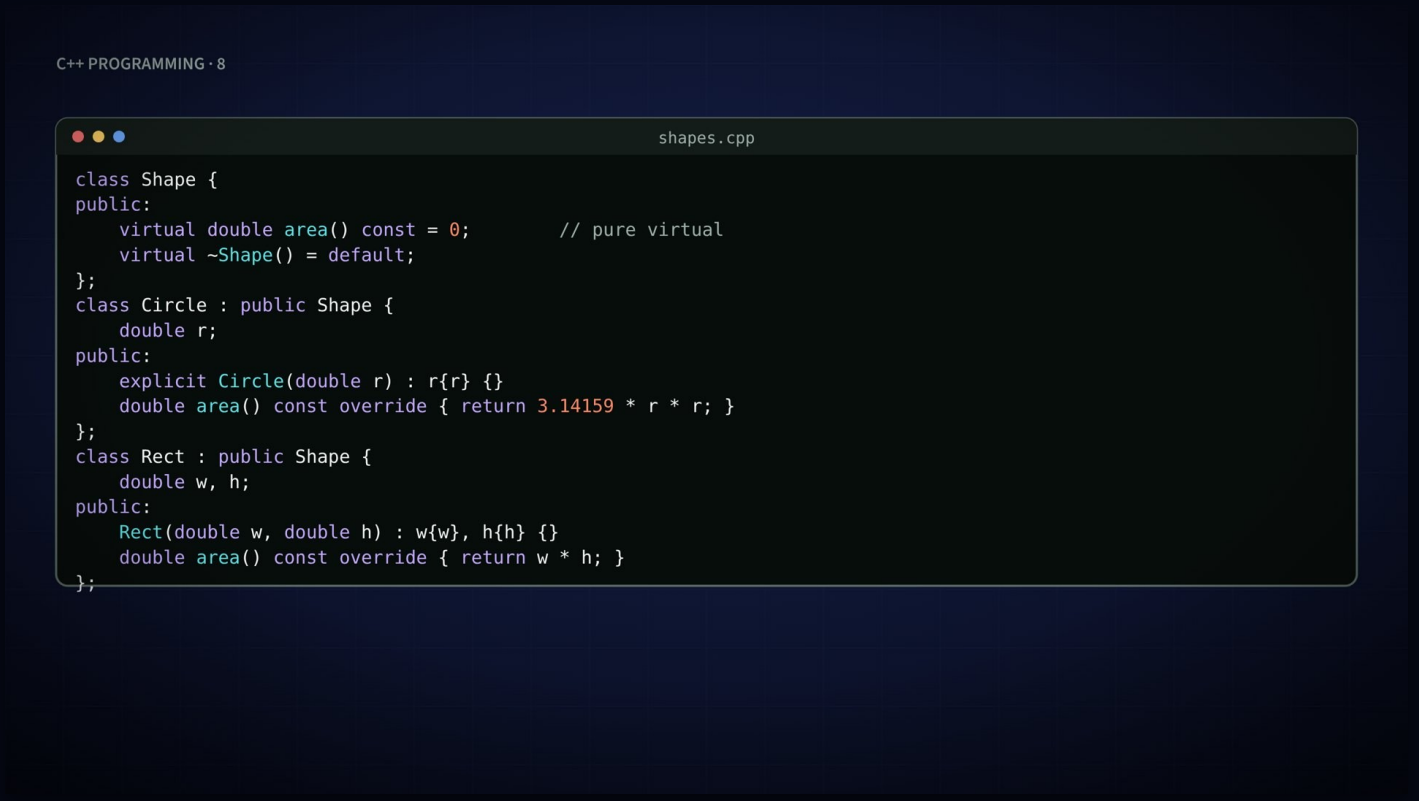


۸.۰۱

تابع‌های مجازی

یک تابع کلاس پایه را virtual علامه بزنید تا کلاس‌های مشتق بتوانند آن را تغییر دهند. آن وقت، ...

بدون virtual، همیشه نسخهٔ پایه اجرا می‌شد. این را ارسال پویا یا چندریختی وقت اجرا می‌گویند.



۸.۰۲

یک حلقه، چندین شکل

حالا شکل‌های مختلف را در یک vector از اشاره‌گرهای هوشمند به شکل ذخیره می‌کنیم و روی هر کدام ...

حلقه نمی‌داند و برایش مهم نیست کدام شکل است. فردا یک کلاس مثلث اضافه کنید، و این حلقه بدون ...

C++ PROGRAMMING · 8

main.cpp

```cpp
std::vector<std::unique_ptr<Shape>> shapes;
shapes.push_back(std::make_unique<Circle>(1.0));
shapes.push_back(std::make_unique<Rect>(2.0, 3.0));

for (const auto& s : shapes) {
    std::cout << s->area() << "\n";   // 3.14159, then 6
}
```

۸.۰۳

مجازی خالص و کلاس‌های تجریدی

تابع مجازی خالص، که با مساوی صفر علامه زده می‌شود، در کلاس پایه بدنه ندارد. کلاسی که حداقل ...

کلاس تجریدی یک قرارداد است. هر کلاس مشتق مشخص باید همهٔ تابع‌های مجازی خالص را پیاده کند، ...

C++ PROGRAMMING · 8

abstract.cpp

```
Shape s;                    // ✗ error: Shape is abstract
Circle c{2.0};              // ✓ Circle implements area()

Shape& ref = c;             // ✓ base reference to a derived object
ref.area();                 // calls Circle::area()
```

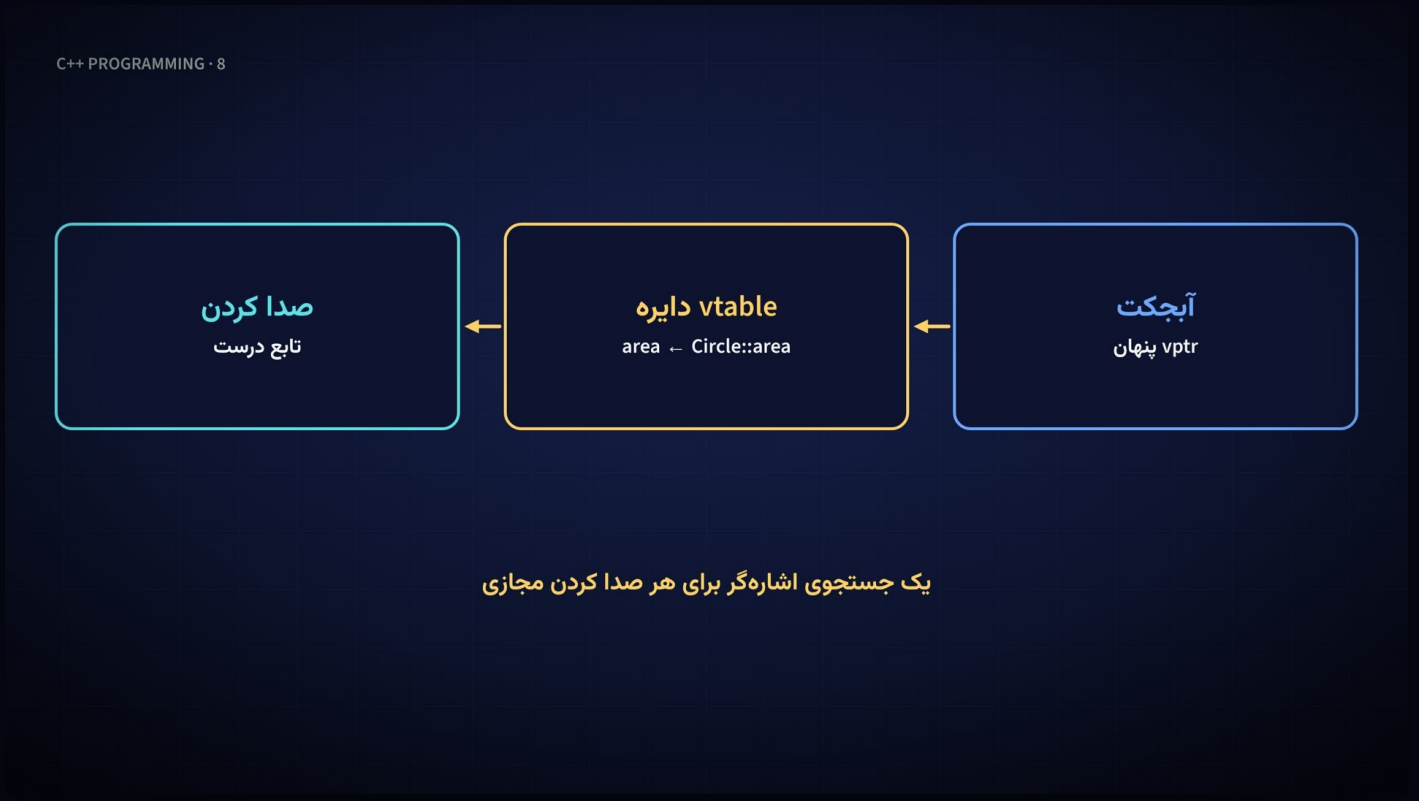
۸.۰۴

چطور کار می‌کند: جدول مجازی

آبجکت — vptr پنهان

vtable دایره — area ← Circle::area

صدا کردن — تابع درست



۸.۰۵

تخریب‌کننده‌های مجازی

اگر یک آبجکت مشتق‌را از طریق اشاره‌گر پایه delete کنید، تخریب‌کنندهٔ کلاس پایه باید virtual ...

قاعده ساده است: هر کلاسی که برای چندریختی ساخته شده به تخریب‌کنندهٔ مجازی ضرورت دارد.

C++ PROGRAMMING · 8

vdtor.cpp

```cpp
class Shape {
public:
    virtual ~Shape() = default;     // ✓ required for polymorphism
};

Shape* p = new Circle{1.0};
delete p;                           // runs ~Circle, then ~Shape
```

۸.۰۶

چندریختی وقت کمپایل

دو نوع چندریختی وجود دارد. چندریختی وقت اجرا از تابع‌های مجازی استفاده می‌کند که هنگام ...

چندریختی وقت کمپایل توسط کمپایلر تصمیم گرفته می‌شود: اورلود کردن تابع، اورلود کردن عملگر، و ...



۸.۰۷

یک کورس رایگان از MyWebUniversity

برنامه‌نویسی C++

درس ۹

اورلود کردن عملگرها

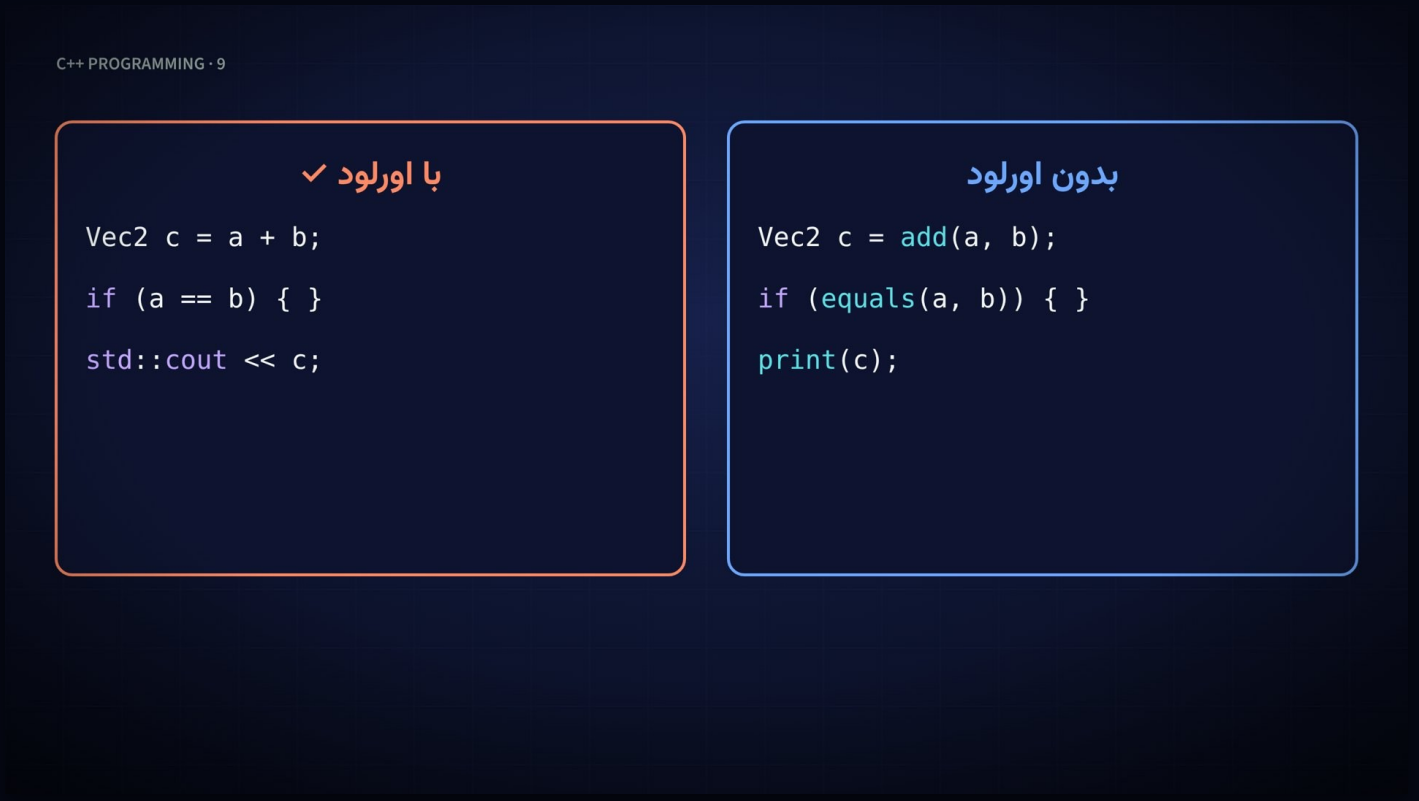
ساختن نوع‌های خودتان مثل نوع‌های داخلی



چرا عملگرها را اورلود کنیم؟

نوع‌های داخلی از عملگرهای طبیعی استفاده می‌کنند: a + b. با اورلود کردن عملگر، نوع‌های ...

این چندریختی وقت کمپایل است: یک عملگر روی چندین نوع کار می‌کند. std::string وقتی متن ...



۹.۰۱

اورلود جمع و جمع مساوی

یک عملگر فقط یک تابع با نام ویژه است: کلمهٔ operator و بعد علامت.

یک الگوی رایج این است که جمع مساوی را به عنوان تابع عضو بنویسید که آبجکت را تغییر می‌دهد، و …

C++ PROGRAMMING · 9

vec2.cpp

```cpp
struct Vec2 {
    double x = 0, y = 0;

    Vec2& operator+=(const Vec2& o) {
        x += o.x; y += o.y;
        return *this;
    }
};

Vec2 operator+(Vec2 a, const Vec2& b) { return a += b; }

Vec2 p{1, 2}, q{3, 4};
Vec2 r = p + q;      // {4, 6}
```

۹.۰۲

عملگرهای مقایسه

دو علامت مساوی را برای مقایسهٔ آبجکت‌ها بر اساس مقدار تعریف کنید. از C++20 به ...

عملگر مقایسهٔ سه‌طرفه، که به خاطر شکلش عملگر سفینهٔ فضایی لقب گرفته، کوچک‌تر، بزرگ‌تر و ...

C++ PROGRAMMING · 9

compare.cpp

```cpp
struct Version {
    int major, minor, patch;
    auto operator<=>(const Version&) const = default;   // C++20
};

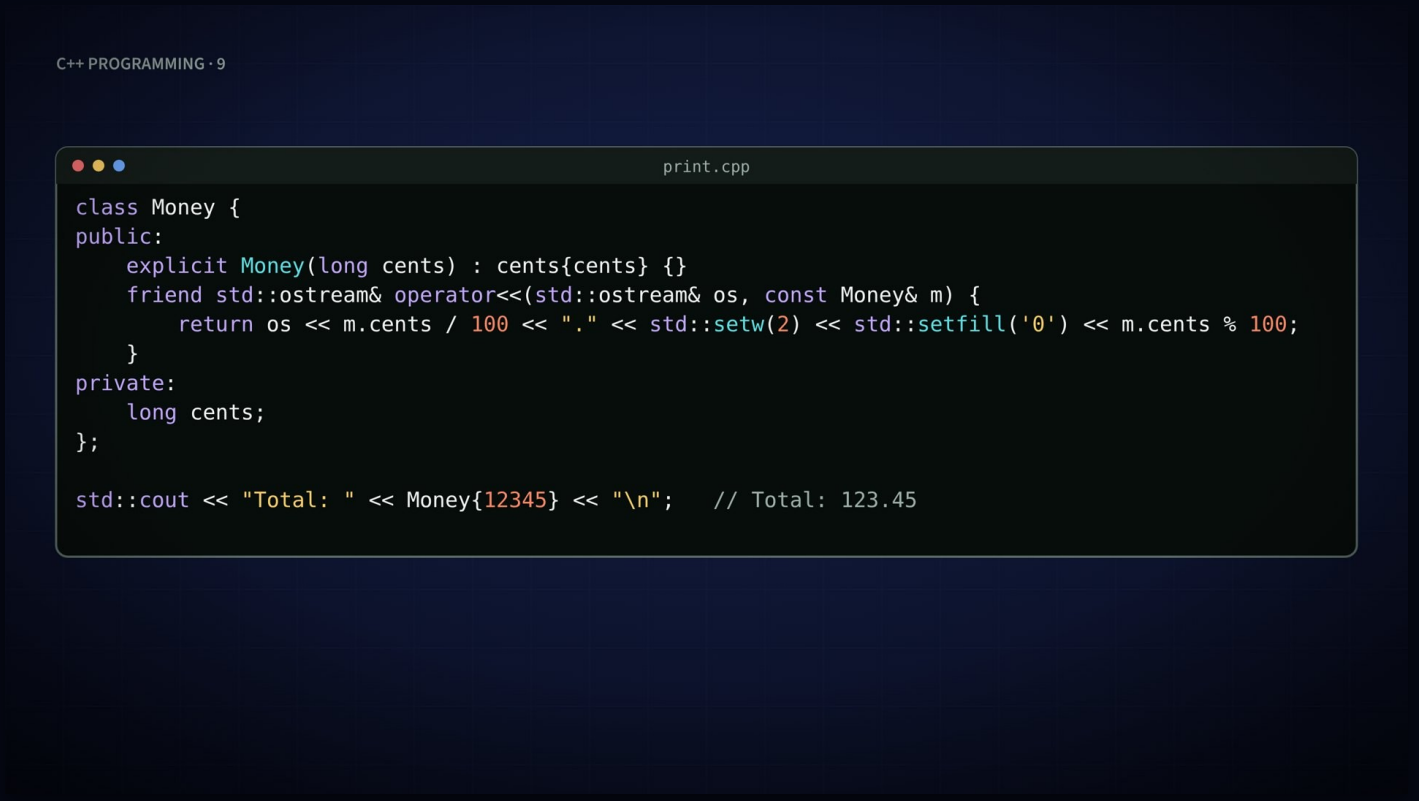
Version a{1, 4, 2}, b{1, 10, 0};
a <  b;    // true
a == b;    // false
a >= b;    // false
```

۹.۰۳

چاپ با علامت کوچک‌تر کوچک‌تر

برای چاپ آبجکت خود با cout، عملگر خروجی را اورلود کنید. یک تابع آزاد است که جریان و آبجکت ...

اگر به اعضای خصوصی ضرورت دارد، آن را داخل کلاس friend اعلان کنید.



۹.۰۴

عملگرهای اندکس و صدا کردن

اورلود کردن قوس‌های مربع کلاس شما را مثل آرایه کارآمد می‌سازد، که vector و map همین طور ...

اورلود کردن عملگر صدا کردن، یعنی پرانتز، آبجکتی می‌سازد که مثل تابع صدا کرده می‌شود، که ...

C++ PROGRAMMING · 9

operators.cpp

```cpp
class Grid {
public:
    int& operator()(int r, int c) { return cells[r * cols + c]; }
private:
    int cols = 10;
    std::vector<int> cells = std::vector<int>(100);
};

struct Multiplier {
    int k;
    int operator()(int x) const { return x * k; }
};
Multiplier times3{3};
times3(7);    // 21
```

۹.۰۵

قواعد سلیقهٔ خوب

معنای واضح — + جمع می‌کند، == مقایسه

هماهنگ بمانید — == با !=، < با >

اورلود نمی‌شوند — . :: ?: sizeof



۹.۰۶

یک کورس رایگان از MyWebUniversity

برنامه‌نویسی C++

درس ۱۰

حافظه و اشاره‌گرهای هوشمند

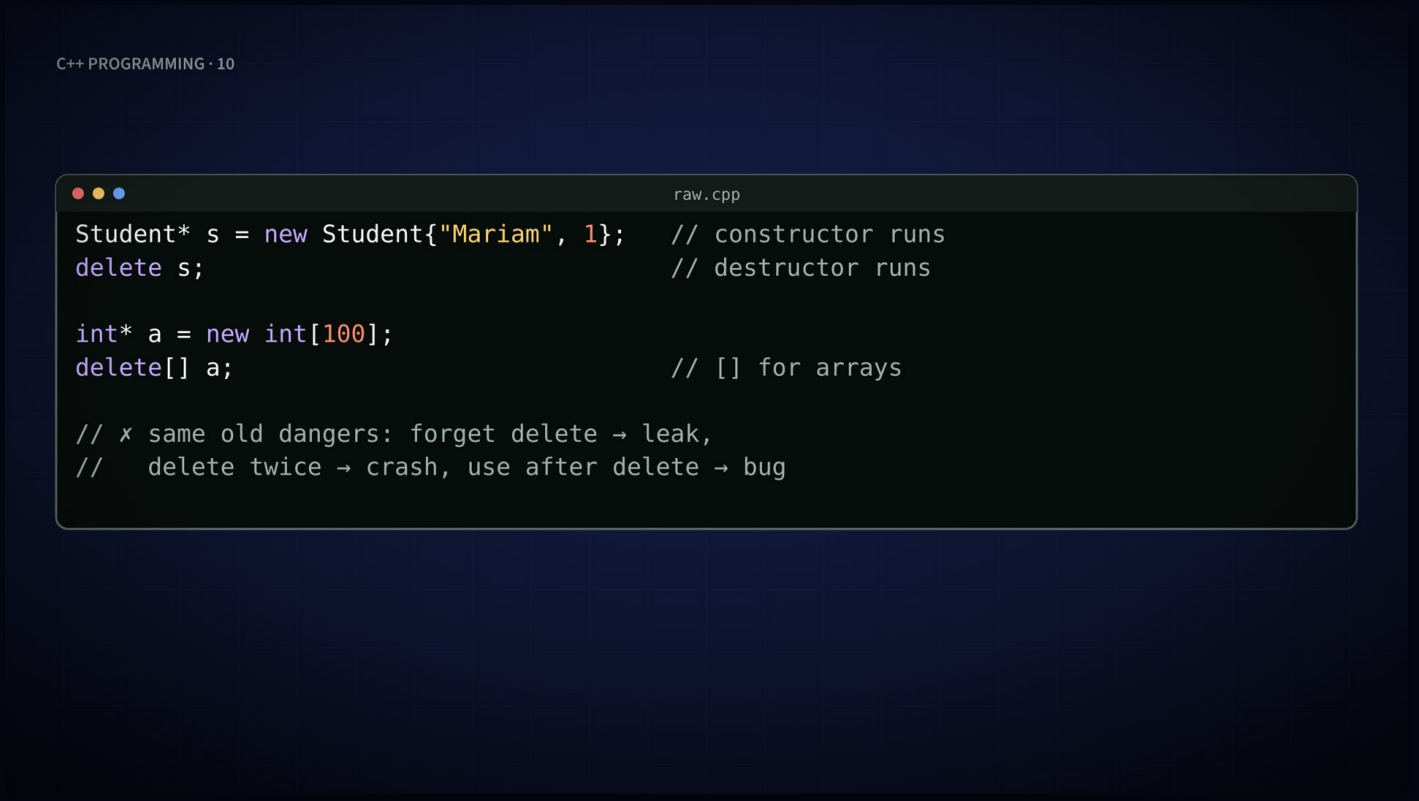
new و delete، unique_ptr، shared_ptr و weak_ptr



new و delete

C++ malloc و free را با new و delete جایگزین می‌کند. new حافظه رزرو می‌کند و سازنده را ...

آرایه‌ها از new با قوس و delete با قوس استفاده می‌کنند. اما new و delete خام هر مشکل C را ...



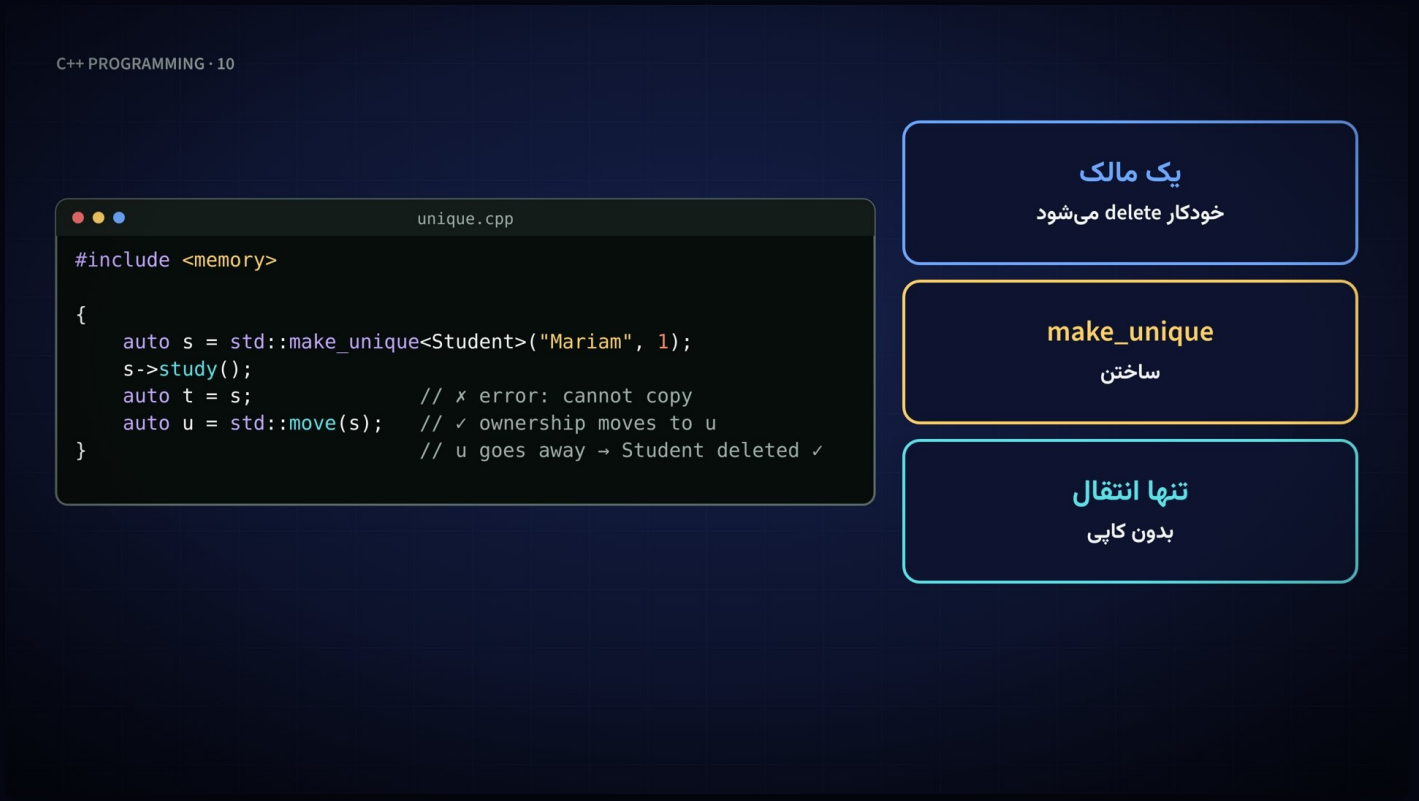
۱۰.۰۱

std::unique_ptr

یک مالک — خودکار delete می‌شود

make_unique — ساختن

تنها انتقال — بدون کاپی



۱۰.۰۲

std::shared_ptr

گاهی چند بخش پروگرام به یک آبجکت ضرورت دارند، و هیچ کدام نمی‌داند کدام آخر تمام می‌کند. ...

هر کاپی شمارش را زیاد می‌کند و هر نابودی آن را کم. وقتی شمارش به صفر برسد، آبجکت delete ...

shared.cpp

```cpp
auto img = std::make_shared<Image>("map.png");   // count = 1
{
    auto viewer = img;                            // count = 2
    auto editor = img;                            // count = 3
}                                                 // count = 1
std::cout << img.use_count();                     // 1
img.reset();                                      // count = 0 → deleted
```

۱۰.۰۳

std::weak_ptr و چرخه‌ها

اگر دو آبجکت شِیرد پوینترهایی به همدیگر نگه دارند، شمارش آن‌ها هرگز به صفر نمی‌رسد و هر دو ...

weak_ptr یک آبجکت را بدون مالکیت نظاره می‌کند، پس شمارش را زیاد نمی‌کند. برای پیوندهای ...

C++ PROGRAMMING · 10

weak.cpp

```cpp
struct Node {
    std::vector<std::shared_ptr<Node>> children;   // owns children
    std::weak_ptr<Node> parent;                    // observes parent
};

if (auto p = node->parent.lock()) {   // still alive?
    p->visit();
}
```

۱۰.۰۴

از کدام اشاره‌گر استفاده کنم؟

این رهنمود مدرن است. آبجکت‌های ساده و ظرف‌ها را ترجیح دهید؛ بیشتر وقت‌ها اصلاً به اشاره‌گر ...

وقتی به یک آبجکت هیپ با یک مالک ضرورت دارید، از unique_ptr استفاده کنید. برای مالکیت ...

C++ PROGRAMMING · 10

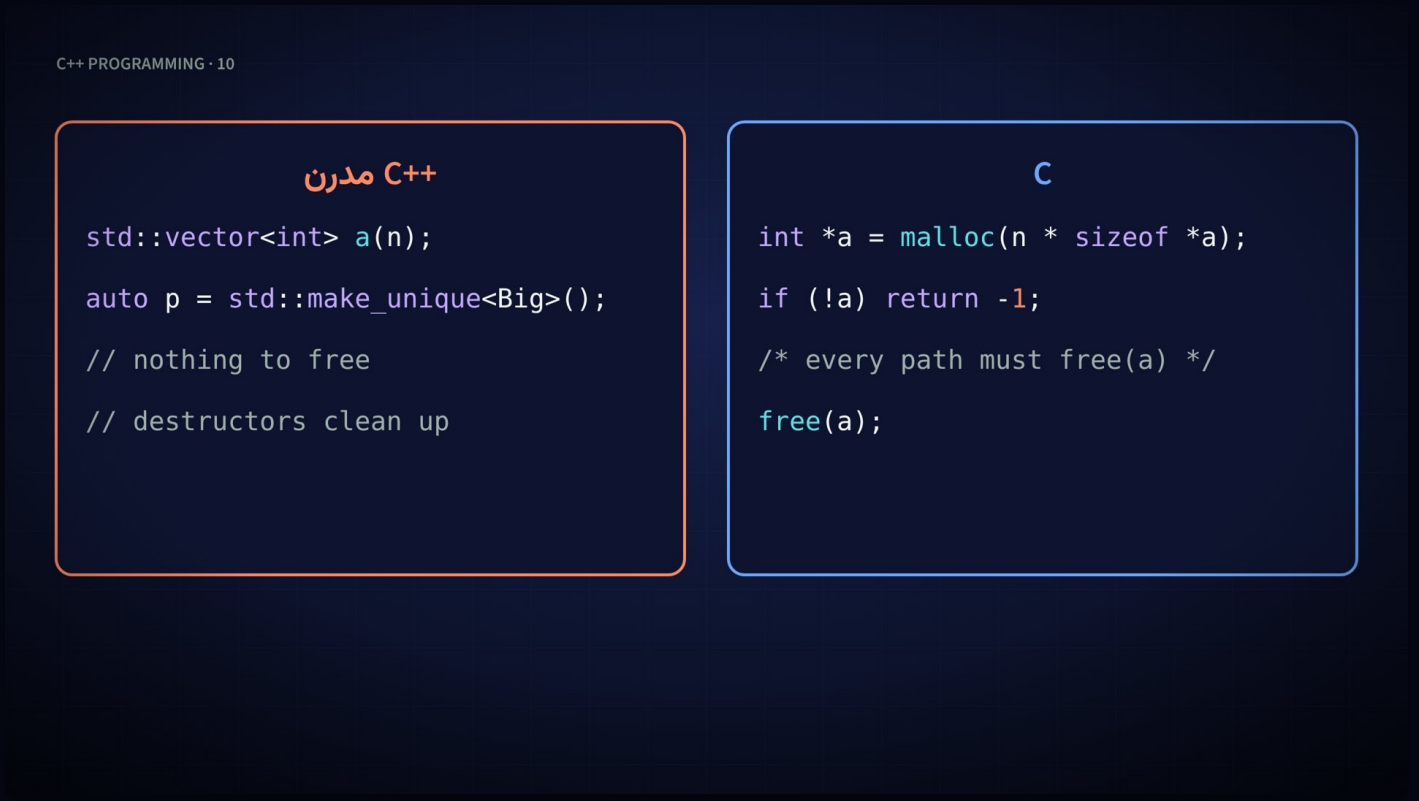
استفاده	کی
`T / std::vector<T>`	بیشتر وقت‌ها: بدون اشاره‌گر
`std::unique_ptr<T>`	یک مالک (پیش‌فرض)
`std::shared_ptr<T>`	مالکیت مشترک
`std::weak_ptr<T>`	نظارهٔ آبجکت مشترک، شکستن چرخه
`T* / T&`	تنها دسترسی بدون مالکیت

۱۰.۰۵

حافظه در C و C++

دو دنیا را مقایسه کنید. در C، هر malloc در هر مسیر به یک free ضرورت دارد، و ابزارها باید ...

در C++ مدرن، مالکیت در نوع‌ها نوشته می‌شود و تخریب‌کننده‌ها خودکار پاک‌کاری می‌کنند. ...



۱۰.۰۶

یک کورس رایگان از MyWebUniversity

برنامه‌نویسی C++

درس ۱۱

templateها و برنامه‌نویسی عمومی

یک بار بنویسید، با هر نوع استفاده کنید



مشکل

در C، تابعی برای پیدا کردن بزرگ‌ترِ دو مقدار باید یک بار برای int، دوباره برای double، و ...

templateها این را حل می‌کنند. کود را یک بار با یک نوع جانشین می‌نویسید، و کمپایلر برای هر ...



۱۱.۰۱

templateهای تابع

یک template تابع با کلمهٔ template و یک پارامتر نوع داخل قوس‌های زاویه‌دار شروع می‌شود که ...

وقتی آن را صدا می‌کنید، کمپایلر T را از آرگومان‌ها حدس می‌زند و نسخهٔ درست را می‌سازد. این ...

C++ PROGRAMMING · 11

max.cpp

```cpp
template <typename T>
T max_of(const T& a, const T& b) {
    return (a > b) ? a : b;
}

max_of(3, 7);                       // T = int     → 7
max_of(2.5, 1.5);                   // T = double  → 2.5
max_of(std::string{"Kabul"}, std::string{"Herat"});   // "Kabul"
```

۱۱.۰۲

templateهای کلاس

کلاس‌ها هم می‌توانند template باشند. یک ستک از T برای intها، رشته‌ها یا شاگردان کار ...

دقیقاً همین طور std::vector، map و تمام کتابخانهٔ استاندارد ساخته شده‌اند: vector int، ...

C++ PROGRAMMING · 11

stack.hpp

```cpp
template <typename T>
class Stack {
public:
    void push(const T& item) { items.push_back(item); }
    T pop() { T top = items.back(); items.pop_back(); return top; }
    bool empty() const { return items.empty(); }
private:
    std::vector<T> items;
};

Stack<int> numbers;     numbers.push(42);
Stack<std::string> words;   words.push("salaam");
```

۱۱.۰۳

مفهوم‌ها: محدود کردن templateها

اگر کسی مَکس آف را با نوعی صدا کند که مقایسه نمی‌شود چه؟ کمپایلرهای قدیم صفحه‌ها خطای ...

C++20 مفهوم‌ها را اضافه کرد که نیازمندی‌های یک template را بیان می‌کنند. اگر یک نوع ...

C++ PROGRAMMING · 11

concepts.cpp

```cpp
#include <concepts>

template <std::totally_ordered T>
T max_of(const T& a, const T& b) { return (a > b) ? a : b; }

template <typename T>
concept Shape = requires(const T& s) { { s.area() } -> std::convertible_to<double>; };

double total_area(const Shape auto& a, const Shape auto& b) {
    return a.area() + b.area();
}
```

۱۱.۰۴

templateها در هِدرها زندگی می‌کنند

در هِدرها — کود کامل دیده شود

بدون هزینهٔ اجرا — در وقت کمپایل ساخته می‌شود

بها — ساختن آهسته‌تر، فایل بزرگ‌تر



۱۱.۰۵

یک کورس رایگان از MyWebUniversity

برنامه‌نویسی C++

درس ۱۲

ظرف‌های STL: vector، map و بیشتر

vector، array، آرایه‌های چندبعدی، map، مجموعه، ستک و صف

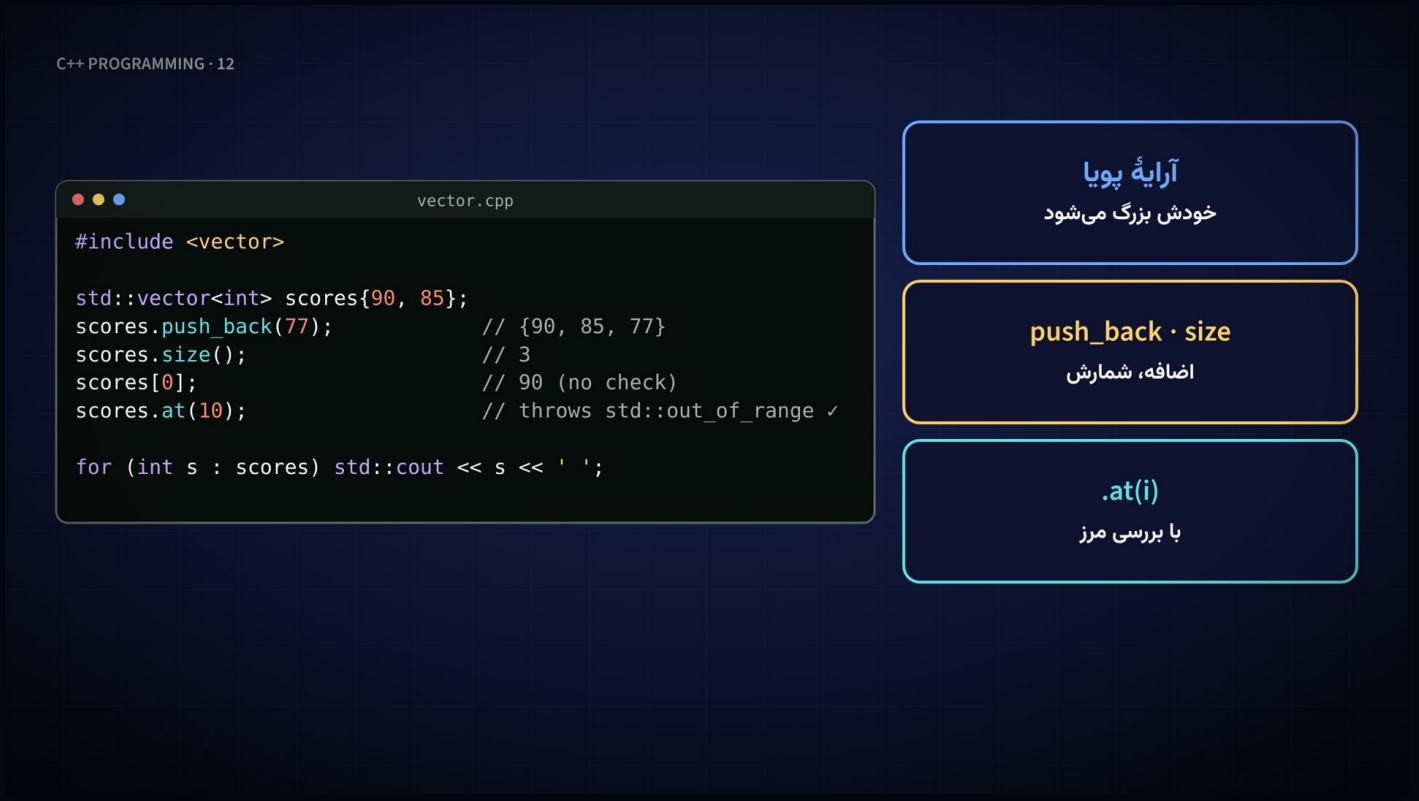


std::vector

آرایهٔ پویا — خودش بزرگ می‌شود

push_back · size — اضافه، شمارش

at(i). — با بررسی مرز



۱۲.۰۱

std::array و آرایه‌های چندبعدی

std::array یک آرایهٔ اندازه‌ثابت است که اندازهٔ خود را می‌داند و با الگوریتم‌های ...

برای شبکه‌ای که اندازه‌اش تنها در وقت اجرا معلوم است، از وکتوری از vectorها استفاده کنید. ...

C++ PROGRAMMING · 12

grids.cpp

```cpp
#include <array>

std::array<int, 5> a{1, 2, 3, 4, 5};      // fixed size, a.size() == 5
std::array<std::array<int, 3>, 3> board{}; // 3×3, e.g. tic-tac-toe

int rows = 4, cols = 6;
std::vector<std::vector<double>> grid(rows, std::vector<double>(cols, 0.0));
grid[2][3] = 1.5;

std::vector<double> flat(rows * cols);      // fast: one block
flat[2 * cols + 3] = 1.5;
```

۱۲.۰۲

std::map و unordered_map

map جفت‌های کلید و مقدار را ذخیره می‌کند، مثل یک قاموس: یک کلمه و معنای آن، یا نام یک شاگرد ...

std::map کلیدها را مرتب نگه می‌دارد و در وقت لگاریتمی پیدا می‌کند. unordered_map از ...

قوس‌های مربع یک کلید موجود نبوده را خودکار اضافه می‌کنند. وقتی تنها می‌خواهید بررسی کنید از ...

C++ PROGRAMMING · 12

map.cpp

```cpp
#include <map>
#include <unordered_map>

std::map<std::string, int> grades;
grades["Mariam"] = 95;
grades["Ahmad"]  = 88;
for (const auto& [name, g] : grades)        // sorted by name
    std::cout << name << ": " << g << "\n";

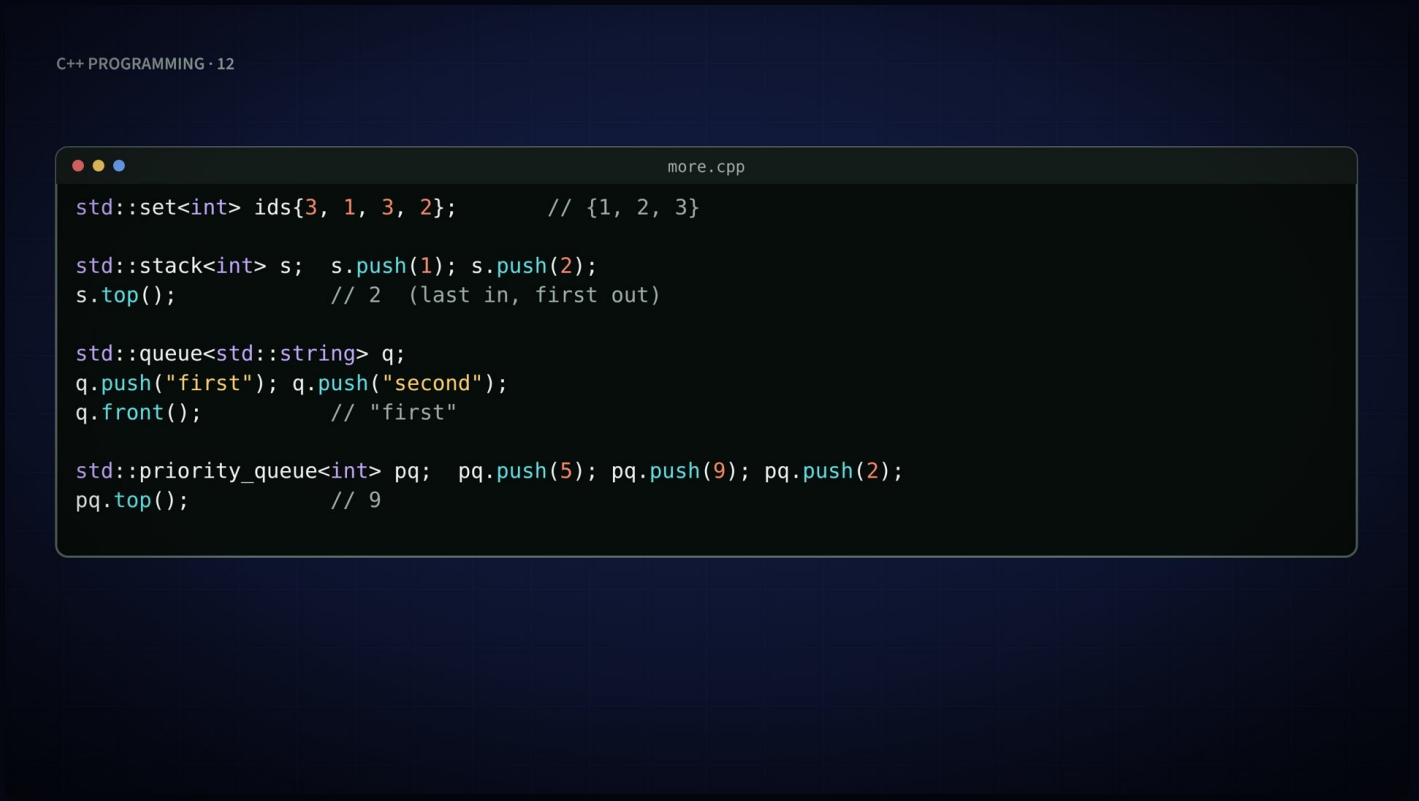
std::unordered_map<std::string, int> word_count;   // hash table
word_count["salaam"]++;
if (word_count.contains("kabul")) { }             // C++20
```

۱۲.۰۳

set، ستک، صف و بیشتر

set مقدارهای یکتا را ذخیره می‌کند و تکراری‌ها را نادیده می‌گیرد. مثل map، نسخه‌های مرتب و ...

ستک دسترسی آخرین ورودی اولین خروجی می‌دهد، صف اولین ورودی اولین خروجی، و صف اولویت همیشه ...



۱۲.۰۴

انتخاب یک ظرف

با vector شروع کنید؛ برای بیشتر کارها سریع‌ترین است، چون عناصر آن در حافظه کنار هم هستند. ...

وقتی در هر دو سر اضافه می‌کنید از deque استفاده کنید، و از لیست تنها وقتی که مدام در وسط ...

ظرف	بهترین برای
vector	پیش‌فرض: سریع، پیوسته
array	اندازهٔ ثابت در وقت کمپایل
map / unordered_map	جستجوی مقدار با کلید
set / unordered_set	مقدارهای یکتا، بررسی سریع
deque	اضافه/حذف در هر دو سر
stack · queue · priority_queue	LIFO · FIFO · بزرگ‌ترین اول

۱۲.۰۵

یک کورس رایگان از MyWebUniversity

برنامه‌نویسی C++

درس ۱۳

تکرارکننده‌ها و الگوریتم‌ها

مرتب‌سازی، جستجو، شمارش، جمع، تبدیل و رنج‌ها



تکرارکننده‌ها

تکرارکننده مثل یک اشاره‌گر هوشمند داخل یک ظرف است. begin به عنصر اول اشاره می‌کند، و end …

هر ظرف تکرارکننده دارد، پس یک الگوریتم روی vector، لیست یا set کار می‌کند. این پیوند بین …



۱۳.۰۱

مرتب‌سازی و جستجو

فایل اَلگوریتم را شامل کنید و ده‌ها الگوریتم آزمایش‌شده و بهینه می‌گیرید. sort یک محدوده را ...

find یک مقدار را جستجو می‌کند و تکرارکننده‌ای به آن برمی‌گرداند، یا end اگر موجود نباشد. ...

C++ PROGRAMMING · 13

algorithms.cpp

```cpp
#include <algorithm>

std::vector<int> v{42, 7, 19, 3};
std::sort(v.begin(), v.end());                    // 3 7 19 42
std::sort(v.begin(), v.end(), std::greater<>());    // 42 19 7 3

auto it = std::find(v.begin(), v.end(), 19);
if (it != v.end()) std::cout << "found";

auto big = std::find_if(v.begin(), v.end(), [](int x) { return x > 10; });
```

۱۳.۰۲

شمارش، جمع و تبدیل

count_if عناصری را که با یک شرط جور می‌آیند می‌شمارد. accumulate، از فایل numeric، همه چیز ...

transform یک تابع را روی هر عنصر اجرا می‌کند و نتیجه‌ها را می‌نویسد. این صدا کردن‌های ...

C++ PROGRAMMING · 13

numeric.cpp

```cpp
#include <numeric>

std::vector<int> scores{90, 55, 77, 64, 98};
auto passed = std::count_if(scores.begin(), scores.end(),
                            [](int s) { return s >= 60; });     // 4
int total = std::accumulate(scores.begin(), scores.end(), 0);   // 384

std::vector<int> curved(scores.size());
std::transform(scores.begin(), scores.end(), curved.begin(),
               [](int s) { return std::min(s + 5, 100); });
```

۱۳.۰۳

رنج‌های C++20

رنج‌های C++20 اجازه می‌دهند به جای begin و end تمام ظرف را بفرستید. همچنین views ...

نتیجه مثل یک جمله خوانده می‌شود: نمره‌ها را بگیر، قبول‌شده‌ها را نگه دار، پنج اضافه کن، و ...



۱۳.۰۴

چرا از الگوریتم‌ها استفاده کنیم؟

آزمایش‌شده — توسط میلیون‌ها

سریع — بهینه‌شده توسط متخصصان

روشن — نام می‌گوید چه می‌کند



۱۳.۰۵

یک کورس رایگان از MyWebUniversity

برنامه‌نویسی C++

درس ۱۴

رشته‌ها و جریان‌های فایل

متودهای string، جریان فایل، جریان رشته و قالب‌بندی



بیشتر دربارهٔ string

std::string متودهای مفید زیادی دارد: substr بخشی را می‌بُرد، find جستجو می‌کند، ...

اعداد را با to_string به متن، و متن را با stoi و stod به عدد تبدیل کنید. string ...

C++ PROGRAMMING · 14

strings.cpp

```cpp
std::string email = "wahid@mywebuniversity.com";
auto at = email.find('@');                    // 5
std::string user = email.substr(0, at);       // "wahid"
bool edu = email.ends_with(".com");           // true (C++20)

int n = std::stoi("2026");                    // text → int
std::string s = std::to_string(3.5);          // "3.500000"

void greet(std::string_view name);            // no copy
```

۱۴.۰۱

نوشتن فایل با ofstream

فایل fstream را شامل کنید. یک آبجکت ofstream یک فایل را برای نوشتن باز می‌کند، و ...

به لطف RAII، فایل وقتی آبجکت از حوزه بیرون می‌رود خودکار بسته می‌شود. با یک if روی ...

C++ PROGRAMMING · 14

write.cpp

```cpp
#include <fstream>

{
    std::ofstream out{"scores.txt"};
    if (!out) {
        std::cerr << "cannot open file\n";
        return 1;
    }
    out << "Mariam " << 95 << "\n";
    out << "Ahmad "  << 88 << "\n";
}                       // closed automatically (RAII)
```

۱۴.۰۲

خواندن فایل با ifstream

ifstream یک فایل را می‌خواند. از getline در یک حلقه برای خواندن یک سطر در هر بار ...

شرط حلقه خود خواندن است، پس در پایان فایل به شکل تمیز متوقف می‌شود.

read.cpp

```cpp
std::ifstream in{"scores.txt"};
std::string name;
int score;
while (in >> name >> score) {            // word, then number
    std::cout << name << ": " << score << "\n";
}

std::ifstream text{"notes.txt"};
std::string line;
while (std::getline(text, line)) { /* one line at a time */ }
```

۱۴.۰۳

جریان‌های رشته

یک جریان رشته یک رشته را مثل یک جریان رفتار می‌کند. از istringstream برای تقسیم یک سطر ...

برای پردازش سطرهای CSV و ساختن پیام‌ها خیلی مفید هستند.

C++ PROGRAMMING · 14

csv.cpp

```cpp
#include <sstream>

std::istringstream row{"Herat,2026,95.5"};
std::string city, year, score;
std::getline(row, city, ',');
std::getline(row, year, ',');
std::getline(row, score, ',');

std::ostringstream msg;
msg << city << " scored " << score;
std::string text = msg.str();    // "Herat scored 95.5"
```

۱۴.۰۴

قالب‌بندی خروجی

iomanip دستکاری‌کننده‌های جریان مثل setw برای عرض ستون و setprecision برای جاهای ...

C++20 std::format را اضافه کرد که مثل Python از جای‌نماهای قوس کورلی استفاده ...

C++ PROGRAMMING · 14

format.cpp

```cpp
#include <iomanip>
#include <format>

std::cout << std::setw(10) << "Mariam" << std::fixed << std::setprecision(2) << 95.456;
//     Mariam95.46

std::string s = std::format("{:<10}{:>6.2f}", "Ahmad", 88.5);   // C++20
std::println("Total: {}", 384);                                 // C++23
```

۱۴.۰۵

یک کورس رایگان از MyWebUniversity

برنامه‌نویسی C++

درس ۱۵

استثناها و مدیریت خطا

try، catch، throw، استثناهای استاندارد، noexcept و ایمنی RAII



شیوهٔ C: کودهای بازگشتی

در C، یک تابع با برگرداندن یک مقدار خاص، مثل منفی یک یا یک اشاره‌گر NULL، خطا را گزارش ...

برنامه‌نویسان بررسی را فراموش می‌کنند، و خطا بی‌صدا ادامه پیدا می‌کند و جایی دورتر باعث ...

c_errors.c

```c
FILE *f = fopen("data.txt", "r");
if (f == NULL) { return -1; }          // check
char *buf = malloc(1000);
if (buf == NULL) { fclose(f); return -1; }   // check + clean up
if (fread(buf, 1, 1000, f) == 0) {
    free(buf); fclose(f); return -1;           // check + clean up again
}
/* ... the real work is buried in checks ... */
```

۱۵.۰۱

throw، try و catch

C++ مسیر خطا را از مسیر عادی جدا می‌کند. وقتی چیزی خراب شود، کود یک استثنا پرتاب ...

استثنا در ستک صدا کردن‌ها بالا می‌رود تا یک بلوک catch مناسب آن را مدیریت کند. کود میانی تمیز ...

exceptions.cpp

```cpp
double divide(double a, double b) {
    if (b == 0)
        throw std::invalid_argument("division by zero");
    return a / b;
}

try {
    std::cout << divide(10, 0);
} catch (const std::invalid_argument& e) {
    std::cerr << "Error: " << e.what() << "\n";
}
```

۱۵.۰۲

انواع استاندارد استثنا

کتابخانهٔ استاندارد یک خانوادهٔ کلاس‌های استثنا دارد که همه از std::exception به ارث ...

اول نوع‌های دقیق‌تر را بگیرید، و همیشه با رفرنس ثابت بگیرید تا از کاپی و بُرش جلوگیری شود.

استثنا	چه وقت پرتاب می‌شود
std::out_of_range	vector اَت با اندیس نادرست
std::invalid_argument	stoi با ورودی نادرست
std::bad_alloc	new حافظه پیدا نمی‌کند
std::runtime_error	خطاهای عمومی هنگام اجرا
std::exception	کلاس پایه: همه را می‌گیرد

۱۵.۰۳

استثناهای خودتان

می‌توانید با ارث‌بری از std::runtime_error کلاس استثنای خودتان را بسازید. این به صدا ...

تابع‌هایی را که هرگز استثنا پرتاب نمی‌کنند با noexcept علامت بزنید. این یک وعده به کمپایلر ...

bank.cpp

```cpp
class InsufficientFunds : public std::runtime_error {
public:
    InsufficientFunds()
        : std::runtime_error("not enough money") {}
};

void withdraw(double amount) {
    if (amount > balance) throw InsufficientFunds{};
    balance -= amount;
}

double get_balance() const noexcept { return balance; }
```

۱۵.۰۴

استثنا + RAII = کود ایمن

وقتی یک استثنا از یک تابع بیرون می‌رود، C++ هر آبجکت محلی را در راه خروج نابود ...

پس اگر منابع شما در آبجکت‌های RAII مثل fstream، string، vector و unique_ptr ...

safe.cpp

```cpp
void process() {
    std::ifstream f{"data.txt"};          // closes itself
    std::vector<char> buf(1000);          // frees itself
    auto p = std::make_unique<Parser>();  // deletes itself

    if (!f) throw std::runtime_error("cannot open");
    // ... real work, no clean-up code ...
}   // error or not: everything is released
```

۱۵.۰۵

یک کورس رایگان از MyWebUniversity

برنامه‌نویسی C++

درس ۱۶

ساختارهای دیتا در C++



کلاس لیست پیوندی، ستک، صف، درخت جستجوی دودویی، جدول هش و Big-O

یک کلاس لیست پیوندی

در C، لیست پیوندی ما یک struct و تابع‌های جدا بود. در C++، یک کلاس می‌شود که مالک ...

با unique_ptr برای نِکست، هر گره مالک گرهٔ بعدی است، پس تمام زنجیره خودش آزاد می‌شود. یک ...

list.cpp

```cpp
template <typename T>
class LinkedList {
    struct Node {
        T data;
        std::unique_ptr<Node> next;
    };
    std::unique_ptr<Node> head;
public:
    void push_front(T value) {
        head = std::make_unique<Node>(
            Node{value, std::move(head)});
    }
};
```

۱۶.۰۱

ستک و صف

ستک آخرین ورودی، اولین خروجی است، مثل یک دسته بشقاب: روی سر push کن، از سر pop کن. دکمه‌های ...

صف اولین ورودی، اولین خروجی است، مثل یک صف در بانک: از پشت اضافه کن، از جلو بیرون کن. ...



۱۶.۰۲

درخت جستجوی دودویی

در درخت جستجوی دودویی، هر گره حداکثر دو فرزند دارد. مقدارهای کوچک‌تر به چپ، و مقدارهای ...

برای یافتن یک مقدار، در هر قدم به چپ یا راست می‌روید، پس هر بار نصف درخت را رد می‌کنید. یک ...

C++ PROGRAMMING · 16

bst.cpp

```cpp
struct TreeNode {
    int value;
    std::unique_ptr<TreeNode> left, right;
};

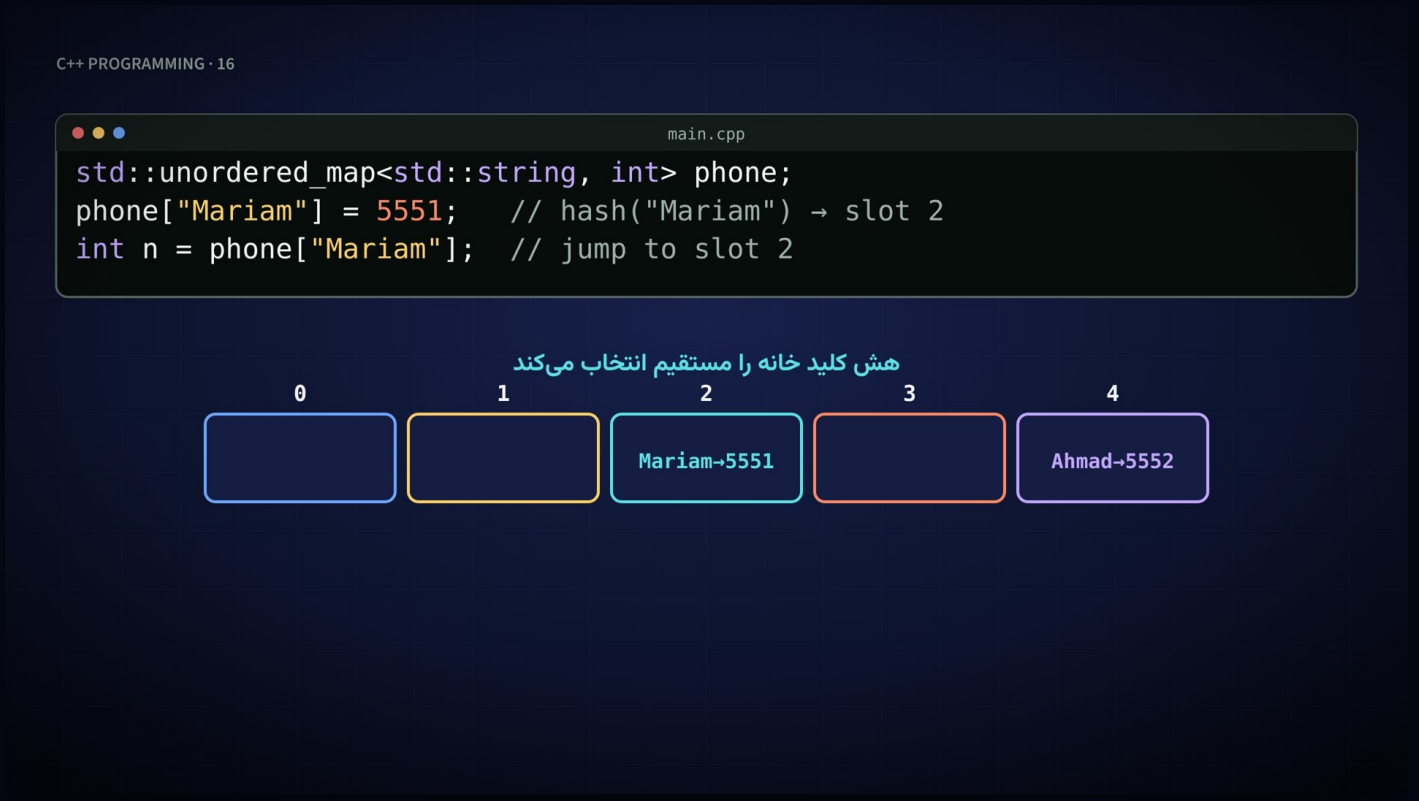
bool contains(const TreeNode* n, int x) {
    while (n) {
        if (x == n->value) return true;
        n = (x < n->value) ? n->left.get()     // go left
                           : n->right.get();   // go right
    }
    return false;
}
```

۱۶.۰۳

جدول‌های هش

یک جدول هش با یک تابع هش یک کلید را به یک اندیس آرایه تبدیل می‌کند، پس مستقیم به خانهٔ درست ...

جستجو به طور میانگین تقریباً زمان ثابت می‌گیرد، سریع‌ترین از همه. در C++، std ...



۱۶.۰۴

انتخاب با Big-O

نشانه‌گذاری Big-O توضیح می‌دهد که با بزرگ شدن دیتا کار چطور زیاد می‌شود. زمان ثابت هموار ...

این جدول راهنمای شماست: ساختاری را انتخاب کنید که عملیات سریعش با کاری که برنامهٔ شما بیشتر ...

C++ PROGRAMMING · 16

ساختار	جستجو	اضافه	دسترسی
vector	O(n)	O(1) در آخر	O(1)
list	O(n)	O(1)	O(n)
map / set	O(log n)	O(log n)	—
unordered_map	O(1)	O(1)	—
stack / queue	—	O(1)	فقط سر

۱۶.۰۵

یک کورس رایگان از MyWebUniversity

برنامه‌نویسی C++

درس ۱۷

C++ مدرن، پروژه‌ها و قدم‌های بعدی

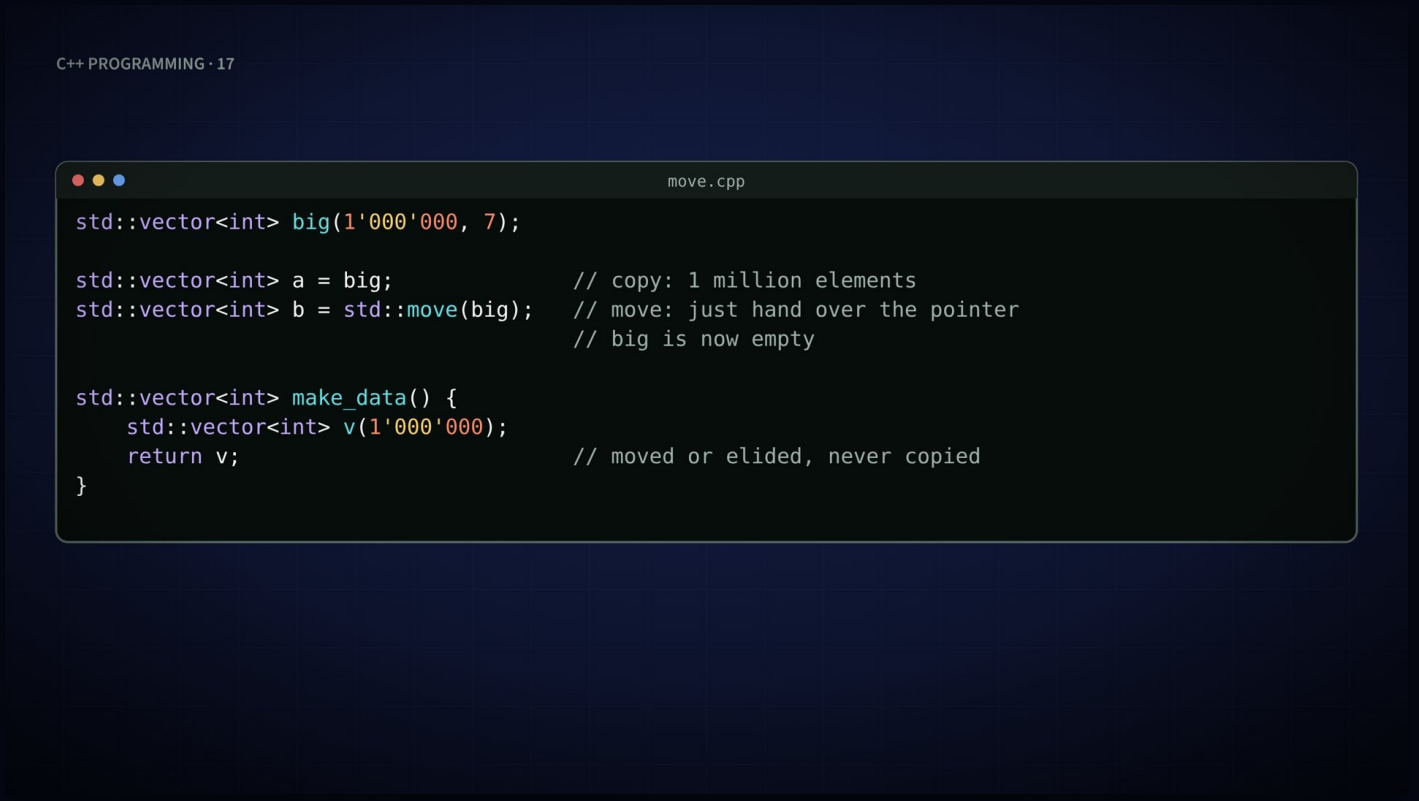


مفهوم جابه‌جایی، پروژه‌های چندفایلی، CMake، فلگ‌های کمپایلر، sanitizers و مسیر کاری

مفهوم جابه‌جایی

کاپی کردن یک vector بزرگ هر عنصر را کاپی می‌کند که آهسته است. از C++11، یک آبجکت ...

std::move می‌گوید: کارم با این تمام شد، منابعش را بگیر. تابع‌ها هم می‌توانند آبجکت‌های ...

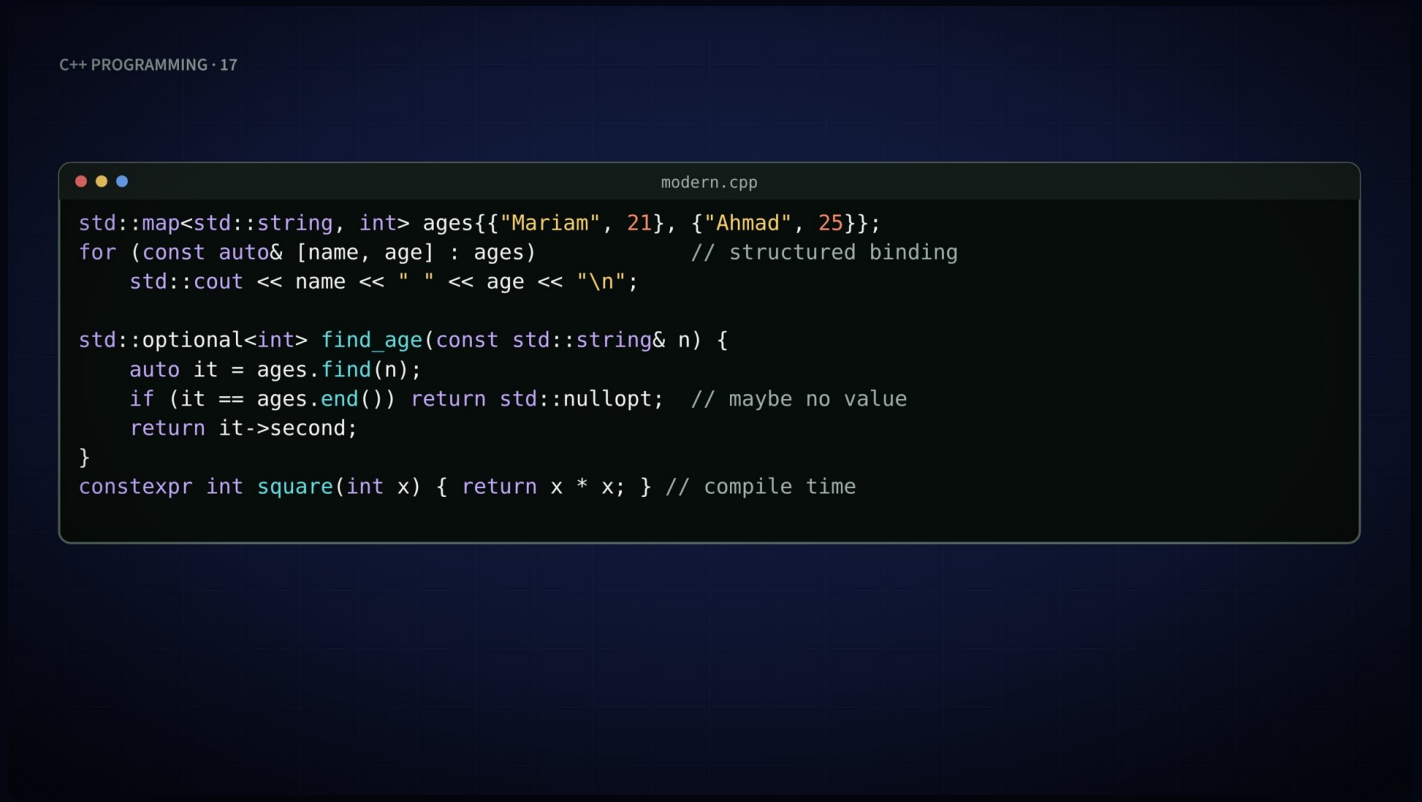


۱۷.۰۱

جعبه‌ابزار C++ مدرن

C++ مدرن به شما auto، حلقه‌های رنج for، lambdaها، اشاره‌گرهای هوشمند، constexpr ...

C++20 concepts، رنج‌ها و ماژول‌ها را اضافه می‌کند. با این‌ها کود بنویسید، و ...



۱۷.۰۲

پروژه‌های چندفایلی و CMake

account.h — اعلان کلاس

account.cpp — تابع‌های عضو

main.cpp — از کلاس استفاده می‌کند



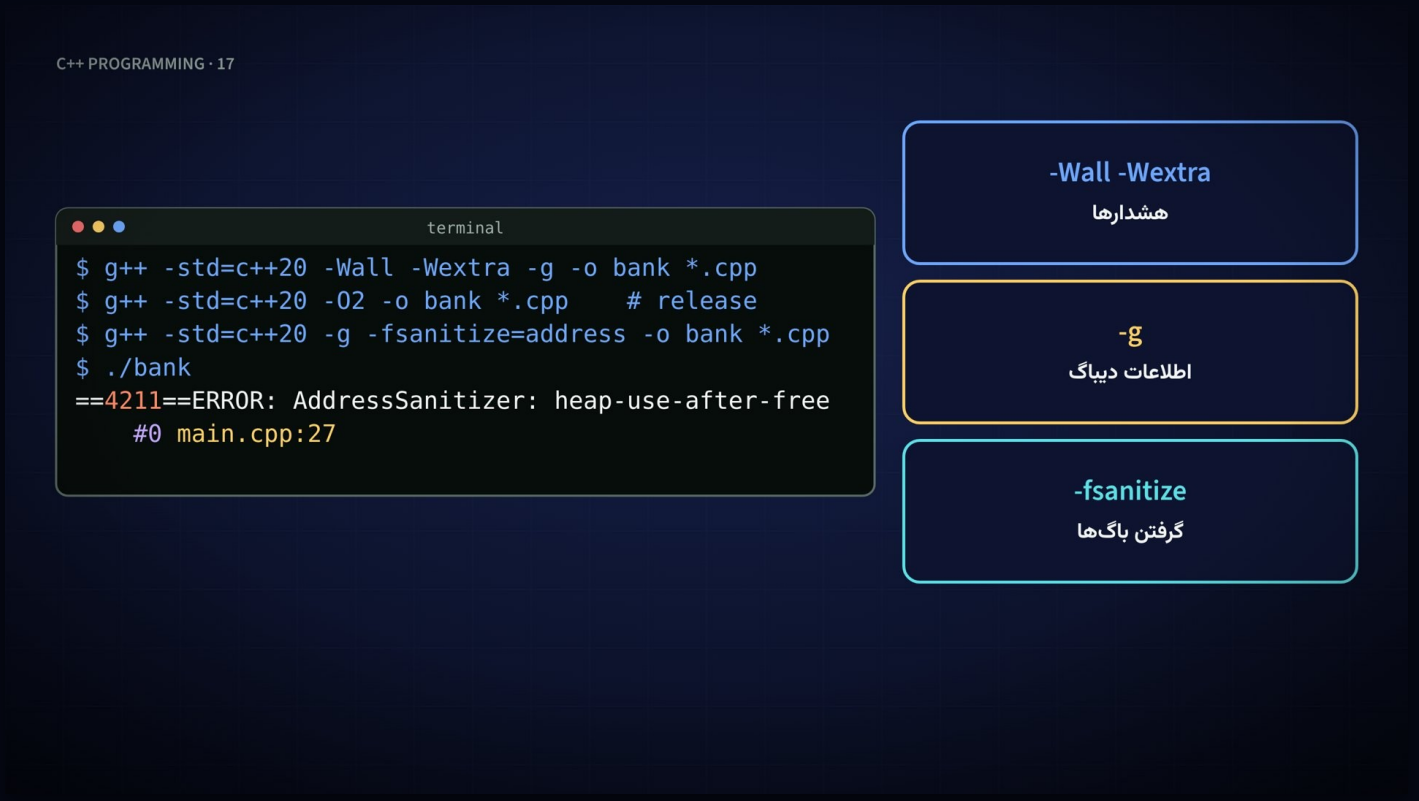
۱۷.۰۳

فلگ‌های کمپایلر و sanitizers

-Wall -Wextra — هشدارها

-g — اطلاعات دیباگ

-fsanitize — گرفتن باگ‌ها



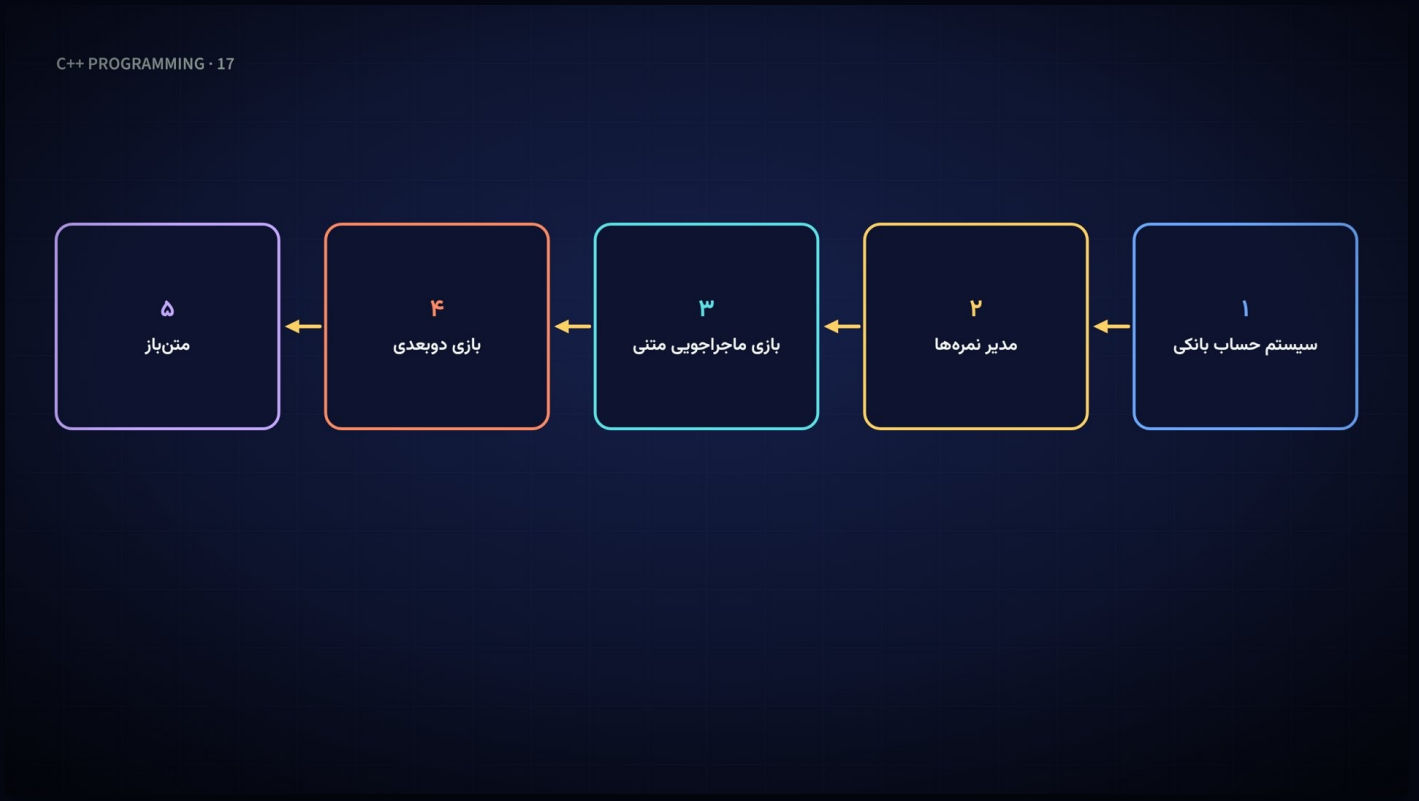
۱۷.۰۴

پروژه‌ها و مسیر کاری

۱ — سیستم حساب بانکی

۲ — مدیر نمره‌ها

۳ — بازی ماجراجویی متنی



۱۷.۰۵

منابع و مطالعهٔ بیشتر (۱/۱)

1. Bjarne Stroustrup, The C++ Programming Language, 4th ed., and A Tour of C++, 3rd ed.
2. ISO/IEC 14882 — the C++ standard (C++20, C++23)
3. cppreference.com — C++ language and library reference
4. C++ Core Guidelines — isocpp.github.io/CppCoreGuidelines
5. MyWebUniversity — free C++ and Linux courses

تشکر از توجه شما

MyWebUniversity.com

@MyWebUniversity-Dari همه را یاد بگیرید

آموزش رایگان تکنالوژی از سال ۲۰۰۰ — انگلیسی و دری