

Lesson 2:
Intro to OOP

SEARCH

RESOURCES

CONCEPTS

1. Classes and OOP

2. Bjarne On Classes In C++

3. Jupyter Notebooks

4. Structures

5. Member Initialization

6. Access Specifiers

7. Classes

8. Encapsulation and Abstraction

9. Bjarne on Encapsulation

10. Constructors

11. Scope Resolution

12._INITIALIZER Lists

13. Initializing Constant Members

14. Encapsulation

15. Accessor Functions

16. Mutator Functions

17. Quiz: Classes In C++

18. Exercise: Pyramid Class

19. Exercise: Student Class

20. Encapsulation in C++

21. Bjarne On Abstraction

22. Abstraction

23. Exercise: Sphere Class

24. Exercise: Private Method

25. Exercise: Static Members

26. Exercise: Static Methods

27. Bjarne On Solving Problems

Exercise: Private Method

SEND FEEDBACK

In 1 | 1: #include <assert>
#include <cmath>
#include <stdexcept>

class Sphere {
public:
 Sphere(int radius) : radius_(radius), volume_(pi_ * 4 / 3 * pow(radius_, 3)) {
 if (radius <= 0) throw std::invalid_argument("radius must be positive");
 }

 int Radius() const { return radius_; }
 int Volume() const { return volume_; }

 // TODO: mutators
 void Radius(int radius) {
 if (radius <= 0) throw std::invalid_argument("radius must be positive");
 radius_ = radius;
 volume_ = pi_ * 4 / 3 * pow(radius_, 3);
 }

private:
 float const pi_{3.14159};
 int const radius_;
 float const volume_;
};

// Test
int main(void) {
 Sphere sphere(5);
 assert(sphere.Radius() == 5);
 assert(abs(sphere.Volume() - 523.6) < 1);

 sphere.Radius(3);
 assert(sphere.Radius() == 3);
 assert(abs(sphere.Volume() - 113.1) < 1);

 bool caught{false};
 try {
 sphere.Radius(-1);
 } catch (...) {
 caught = true;
 }
 assert(caught);
}

Compile & Run

Explain

Loading terminal (id_2q0hq6), please wait...

Menu

Shrink

NEXT