Good Code is Not a Puzzle

What can be done?

Either
1. **choose function names that make it obvious** which arguments might change value (good luck!), or
2. **mark arguments that might change** in the code (as with **C**).

Can We Do Better for Class Instances?

What about arguments that don't change?

Most of your data is class instances.

One rarely wants instances copied onto the stack.

And using "&" everywhere is a little clunky.

Use Pointers for Modifiable Arguments

Solution:
- **use `const` with reference arguments!**
- **`const`** was introduced for **C++**, then back-propagated into **C**.

Avoid using non-`const` reference arguments.

If an argument can be modified, use a pointer.

Named Return Value Optimization Reduces Overhead

One last implementation aspect:
named return value optimization
In practice,
- most C++ compilers
- **transform** a **returned instance**
- **into** an implicit **instance pointer**
- as a new first argument,
- returning either **`void`** or the instance pointer.

