Why Object-Oriented Programming?

Why has object-oriented programming become a major programming paradigm?

So far, you've built two kinds of C++ programs:

- Procedural: The program moves through a linear series of instructions.
- Functional: The program moves from one function to another.

But there is another very common way to structure C++ code: object-oriented programming.

Let's consider a physical object: a light bulb. A light (usually) has two possible states: on and off. It also has functionality that allows you to change its state: you can turn it on and you turn it off. Thankfully, you don't need to know electrical engineering to use the light! You only need to know how to interact with it.

Object-oriented programming (OOP) is a programming paradigm that allows you to package together data states and functionality to modify those data states, while keeping the details hidden away (like with the lightbulb). As a result, code with OOP design is flexible, modular, and abstract. This makes it particularly useful when you create larger programs.

In C++, you can apply OOP in your code with classes and objects. And the C++ objects you create will have states and functionality.

There are four major benefits to object-oriented programming:

- **Encapsulation:** in OOP, you bundle code into a single unit where you can determine the scope of each piece of data.
- **Abstraction:** by using classes, you are able to generalize your object types, simplifying your program.
- **Inheritance:** because a class can inherit attributes and behaviors from another class, you are able to reuse more code.
- **Polymorphism:** one class can be used to create many objects, all from the same flexible piece of code.