Return to Module Overview Page (https://onlinelearning.berkeley.edu/courses/665040/pages/module-four)

4.1: Prologue

At long last, the time has come for us to talk about these mysterious creatures called objects, in all their glory.

As you will soon see, objects are far more powerful than their poor cousins, the C struct and Visual Basic TYPE. That's because objects contain not only data but also code to manipulate the data. In fact, in Java *all* the code is inside of classes, as you've already seen. Dividing code up among classes in this way is the ultimate version of structured programming, called *object-oriented programming*. You put each group of related functions together inside a class, along with the data that those functions manipulate. Doing this divides a program into logically distinct modules, each relatively easy to maintain and understand.

Furthermore, in Java, objects can protect their data from meddling by code in other parts of a program, in much the same way that local variables are protected. This makes Java objects more like physical objects in the real world. For example, you normally can't modify the insides (the "data") of a locked safe; you can only interact with the interface that the safe presents to you: a dial, a handle, a door. A Java object is much the same. It contains some (hidden) data, which you can interact with only by calling the functions in the visible interface of the object. An important consequence is that in Java you can "lock up" data and be sure that no code elsewhere in the program is changing it behind your back.

One note: the code in this course note is going to "evolve" as we proceed. The early examples are *not* examples of good Java style; during this module we're going to develop good style.