Chapter 3
Three kinds of classes

- *Instantiable classes* model program-domain entities
- *Application classes* contain a main method, create and manipulate instances
- *Tester classes* exercise the public interfaces of instantiable classes
  - You’ll want at least one tester class for each instantiable class.
How to design a class

• First: decide on behavior
  • what does this kind of object need to do?
  • what does it need to know?
  • how will other classes interact with it?

• Later: consider implementation
  • why in this order?
Abstraction

• What does abstraction mean?
  • Capturing essential properties
  • Taking “high-level” view
  • Insulation from “accidental” properties

• How does the process of abstraction work?
“Black-box”

• We can treat an abstract object like a “black box”
  • We may know how to interact with it, but can’t see inside it
• Why is this desirable?
Abstraction examples

• What abstractions do each of the following people deal with:
  • a computer user?
  • a computer programmer?
  • a systems or compiler programmer?
  • a chip designer?
  • an electronic engineer?
Object-oriented programming

- When we *abstract* program-domain entities, our goal is to find *objects* that interact
- OOP provides a way of thinking about complex problems
- *Encapsulation* (data hiding) forces object users to deal with abstractions and not implementations
How does Java enforce encapsulation?
public class SomeClass {
    private int x;
    private String y;
    public int getX() {
        return x;
    }
    /* ... */
}

/* ... */
Why OOP?

• Isolates individual components for ease of program understanding, maintenance.
• Provides “direct” way to model program behavior.
• Allows code reuse.
Designing a program

- Decide on the concepts involved in the problem (find classes)
- Decide on the responsibilities of each class
- Decide how classes will interact
Example: Coffee maker

- Concepts: water pot, coffee pot, heater, ground storage
- What does each need to know?
- What are the interactions?
Exercise

- Design the classes for a vending machine application
  - Tracks inventory, interacts with customer, takes money, and dispenses products
- What are the concepts involved? What are the responsibilities of each class? How do they interact?