Classes

CSCI 201
Principles of Software Development

Jeffrey Miller, Ph.D.
jeffrey.miller@usc.edu

USC Viterbi
School of Engineering

Outline

- Classes vs Objects
- Program

Object Analogy

- To drive a car, you need the accelerator pedal, the brake pedal, the steering wheel, and the ignition (among other things)
- But what actually happens when you turn the key in the ignition?
  - Most drivers do not know, but they do not need to know to be able to drive the car
- This is the idea behind objects
  - The code that uses the object knows *what* the method will do, but it does not need to know *how* it does it

Object vs Class

- An object is a high-level term for the idea behind a class
  - We need something to represent a "Car"
- A class is the actual implementation of an object

```
public class Car {
   ...
}
```

- Once a class is created, you can create instances of the class (which will be variables) – this is often referred to as an object as well

```
Car myCar = new Car();
```

Object Description

- Objects (and in turn, classes) can be described by two key features
  - Data
  - Methods
- The methods of an object operate on the data that is part of the object or data that is passed into the method (and usually on both)
  - If a method of a class does not operate on data that is part of the class, why is the method part of the class?

BankAccount Object

- What is the data that is associated with a bank account?
  - Name
  - Account Number
  - Balance
  - PIN
- What are the functions (methods) that can be performed on a bank account?
  - Deposit
  - Withdrawal
  - Transfer
  - Check Balance
  - Change PIN

BankAccount Class

```
1  public class BankAccount {
2    private int accountNumber;
3    private int pin;
4    private double balance;
5
6    public void setBalance(double bal) {
7       balance = bal;
8    }
9    public double getBalance () {
10      return balance;
11   }
12   public void withdraw(double amount) {
13      balance -= amount;
14   }
15   public static void main(String [] args) {
16      BankAccount ba = new BankAccount();
17      ba.setBalance(2000.00);
18      System.out.println("balance = " + ba.getBalance());
19      ba.withdraw(100.00);
21      System.out.println("balance = " + ba.getBalance());
22   }
23 }
```

Access Modifiers

- There are four access modifiers in Java
- If a variable or method is declared…
  - `public`, any other piece of code can access the variable or method
  - `private`, only the class in which that variable or method is declared can access it
  - `<package>` (meaning that there is no identifier), any class in the same package can access it
  - `protected`, any class declared in the same package or who inherits from the class can access it
- Rule of Thumb – Hide the data and expose the necessary methods

Encapsulation

- A class is considered *encapsulated* if all of the data is declared private
- A class is considered *fully encapsulated* if all of the data is declared private AND you provide an accessor (getter) and manipulator (setter) method for each piece of data
  - Manipulators are also called mutators

Outline

- Classes vs Objects
- Program

Program

- Write a program that reads data from a user that represents employee names and hourly rates. Calculate the annual salary of the employee assuming 50 weeks at 40 hours/week are worked each year. Use two classes and methods where applicable. Here is a sample execution with user input bolded:

```
c:\>java com.uscInc.Salary
How many employees? 2
Please enter name of employee #1: Mary
Please enter hourly rate of Mary: 10.50
Please enter name of employee #2: Tony
Please enter hourly rate of Tony: 20.00
Mary's annual salary is $21000.00
Tony's annual salary is $40000.00
c:\>
```