ECE122

March 22, 2005
Scope
Encapsulation

- Encapsulation is a feature of object-oriented world.
- We don’t want to put a million lines of code in the main method to solve a large complex problem. It is too complex to comprehend and difficult to delegate to a team of programmers to solve it.
- We want to dissect the real world problem into lots classes/objects, then model a real world problem as object interactions.
Encapsulation

• Each class/object has limited functionality and complexity and can be delegated to a single programmer who can develop easily without knowing the details of other classes/objects.
• Each class/object needs to publish its interface by declaring its public methods.
• Object interacts with each other through calling each other’s public interface.
• The data (variables) within each class/object is the implementation details that should be hidden from other classes/objects.
Encapsulation

• The implementation details of a class/object are the data members and certain methods that are internal to the class/object. They are used to implement the functionalities of the class/object. These information should be hidden from the other classes/objects. We achieve this encapsulation by declare them private.
Encapsulation

• Encapsulation reduces the complexity of solving a large complex real world problems.

• Encapsulation is one of the core concept of object-oriented approach.
Apply Encapsulation Principle in Object Oriented Design

- Each class is designed to be a specific identity or to provide a specific service.
- It exposes certain methods by declaring them public. Other classes/objects can interact with this class/object only through its public methods.
- Declare all data members private. The data are the class/object’s internal implementation details that others don’t care.
- Use public access methods when necessary.
Example of Object-oriented Design

- Consider such a problem. The goal is to create a program that can rank two vehicles by their range with a full fuel tank.
How many classes do I need?

- I need a Vehicle class that will take care of all the details relating Vehicle. After initialization, a vehicle object should be able to calculate its range.
- I need a Ranking class that knows how to do the comparison of ranges. It doesn’t need to know how a vehicle calculates its range, because that’s Vehicle’s implementation details.
- That’s all I need.
Rewrite Vehicle class with encapsulation principle

• Add constructor with parameters list. The vehicle object will be well constructed after operator “new”.
• Declare all variables private.
• Add one public access method.
• Publish a method called “range()”, which will tell the method caller the range of this vehicle.
• Delete methods not used.
• Call this new class Vehicle1
Add a new class RankVehicle1 class

- This class has a method, compareRange(..), which can compare the range of two vehicles.
- This method doesn’t know how to calculate the range of each vehicle. So it will query each Vehicle1 object for its range. It does so by calling “range()” method of each Vehicle1 object.
- It doesn’t care about the private variables of each Vehicle1 object, nor how “range()” method is implemented.
- It interacts with Vehicle1 object through its published interface, public method, “range()”. 
Demo with Eclipse
Assignments

• Practice and understand Constructor.java, Vehicle1.java, RankVehicle1.java
• Read “Head First Java” Chapter 4, Chapter 9, p270-278
• Read “Java2” P195-210