CMSC 132:
Object-Oriented Programming II

Object-Oriented Programming Intro

Department of Computer Science
University of Maryland, College Park
Object-Oriented Programming (OOP)

- Approach to improving software
  - View software as a collection of objects (entities)

- Motivated by software engineering concerns
  - To be discussed later in the semester

- OOP takes advantage of two techniques
  - Abstraction
  - Encapsulation
Techniques – Abstraction

Abstraction
- Provide high-level model of activity or data

Procedural abstraction
- Specify what actions should be performed
- Hide algorithms

Data abstraction
- Specify data objects for problem
- Hide representation
Techniques – Encapsulation

- Encapsulation
  - Confine information so it is only visible / accessible through an associated external interface

- Approach
  - For some entity $X$ in program
    - Abstract data in $X$
    - Abstract actions on data in $X$
    - Collect data & actions on $X$ in same location
  - Protects and hides $X$

- Extension of abstraction
Abstraction & Encapsulation Example

- Abstraction of a Roster
  - Data
    - List of student names
  - Actions
    - Create roster
    - Add student
    - Remove student
    - Print roster
  - Encapsulation
    - Only these actions can access names in roster

<table>
<thead>
<tr>
<th>ROSTER</th>
</tr>
</thead>
<tbody>
<tr>
<td>List of names</td>
</tr>
<tr>
<td>create()</td>
</tr>
<tr>
<td>addStudent()</td>
</tr>
<tr>
<td>removeStudent()</td>
</tr>
<tr>
<td>print()</td>
</tr>
</tbody>
</table>
Java Programming Language

Language constructs designed to support OOP

- Example
  - Interface – specifies a contract
  - Class – implements/defines contracts, supports encapsulation of implementation

Class libraries designed using OOP principles

- Example
  - Java Collections Framework
  - Java Swing
Java Interface

- An Interface defines a contract
  - Collection of
    - Constants
    - Abstract methods; no implementations
  - Can not be instantiated
- Classes can implement interfaces
  - Must implement all methods in interface
  - Example
    class Foo implements Bar { … }
- Similar to abstract class
  - But class can “inherit” from multiple interfaces
Java Collections Framework

Collection
- Object that groups multiple elements into one unit
- Also called container
- Example: ArrayList

Collection framework consists of
- Interfaces
  - Abstract data type
- Implementations
  - Reusable data structures
- Algorithms
  - Reusable functionality

Collection – Java Interface
- See Java API entry for Collection
- Example (CollectionExample.java)
Homework #1

Let’s go over the check out process and the submit server information

Regarding the quality of your student tests