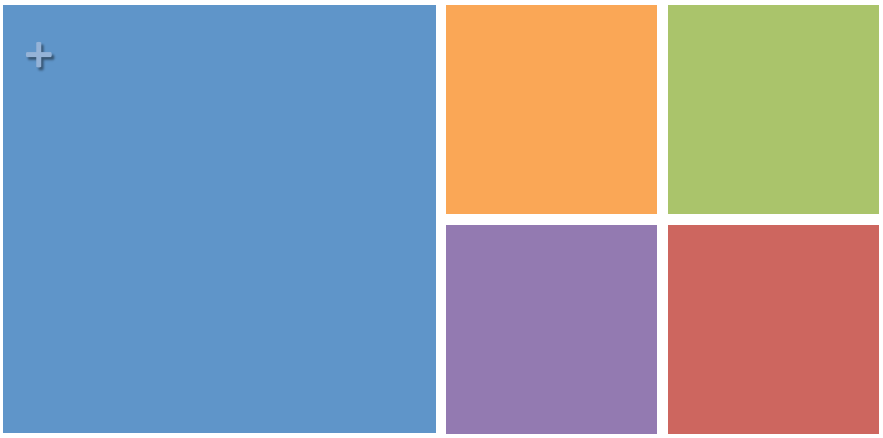

# CS 62: Data Structures

## Today

- Reading
  - Java Structures (JS) Ch. 0, Ch. 1, B.3.2
- Objectives:
  - Course Overview
  - Review of Object-oriented design
  - Review of Java

## Prerequisite

- One of:
  - CS 51 at Pomona or CMC
  - AP CS A exam with score of 4 or 5
  - Fluent in Java and object-oriented programming & permission of instructors
- Come see faculty if any questions
- Assume comfortable with classes & objects, recursion, multi-dimensional arrays, etc. in Java

## Course Goals

- Goal 1
  - How to implement data structures (and algorithms) in Java and C++
- Goal 2
  - Object-oriented design
- Goal 3
  - Complete your knowledge of Java
  - Complete your knowledge of useful tools (e.g. Javadoc, debuggers)

## Course Information

- Prof. America Chambers
  - Office hours: MF 2-3:30pm, Tues 10:30-12:00pm
- TAs: Jane Kang, Matt Sloane, Sarah Jundt, Aloke Desai, Ryan Luo
  - Thurs 8-10pm, Sun 2-4pm and 8-11pm in Edmunds 229
- Course web page:
  - http://www.cs.pomona.edu/classes/cs062
- Piazza invitation

## Assignments

- Reading
  - Come to class with reading already done!
- Lab work:
  - Learn tools & prep work for weekly assignments
  - Lab attendance mandatory!
- Weekly assignment is separate
  - Programs generally due on Sunday nights.
  - See late policy on syllabus.
- Daily homework
  - Not collected but often on **regular Friday quizzes**
  - *No quiz this Friday*!

## Texts

- Java Structures, √7 edition, by Duane Bailey
  - available on-line for free
- C++ for Java Programmers, by Mark Allen Weiss
  - highly recommended for last 1/3 of course

## Slides

- Will *generally* be available before class
  - with code, where applicable
- Designed for class presentation, not for complete notes
  - This is why the reading is necessary!
- Will need to take notes (perhaps on slides)
- No laptops in class
  - If that is problem, come see me.

## Your Responsibilities

- This class is a *lot* of work (8+ hrs/week)
  - Do the reading in advance of lecture.
  - Do the daily problems in advance of lecture
  - Review lecture notes, sample code, and text until well understood.
  - Come to labs prepared.
  - Don't remain confused. Ask faculty or TAs.

## Academic Integrity Guidelines

- Read the academic integrity guidelines on the course webpage.
  - It is your responsibility to be aware of the policy (i.e. what constitutes cheating)
- Consequence: You will fail the course. Period.
- Bad (but common) mindsets that lead to cheating:
  - "I won't get caught"…
  - Panic: "The deadline is in an hour!"

See syllabus for other important information!

## Object-Oriented Design

- Objects are building blocks.
- A program is a collection of interacting objects.
- Objects cooperate to compute solution.
- Objects communicate via sending messages.

## Objects

- Model the physical and conceptual world as well as processes.
- Objects have:
  - Properties, e.g. color, size, manufacturer, …
  - Capabilities, e.g. drive, stop, admit passenger
- Objects responsible for knowing how to perform actions.
  - Commands: change state
  - Queries: response based on properties

## More Objects

- Properties typically implemented as fields or instance variables
  - Affect how object reacts to messages
  - Can be
    - Attributes, e.g., color
    - Components, e.g., doors
    - Associations, e.g., driver
- Capabilities as methods
  - Invoked by sending messages

## Java Review: Interfaces and Classes

- Interfaces
- Classes

## Java Review: Interfaces & Classes

- Interfaces
  - Provide info on publicly available methods and constants
- Classes
  - Templates for objects
    - Constructors generate new distinct objects
      - new Car( "Toyota" ,...)
    - Specify all fields and methods – public and non-public
    - May be used as basis for more refined classes via inheritance

## Java Keywords

- Abstract class
- Information hiding qualifiers:
  - public
  - private
  - protected
- Static
- Final

## Java Keywords

- Abstract class: can't be instantiated
  - usually some methods missing
- Information hiding qualifiers:
  - public
  - private
  - protected
- Static: copy associated with class, not objects
- Final: only assigned to once
  - in its declaration or constructor

## All Classes Specialize "Object"

- Object class has methods:
  - public boolean equals(Object other)
    - Default behavior returns true only if same object
  - public String toString( )
    - Returns string representation of object - default is hexadecimal
    - Typically want to override to be more useful
  - public int hashCode( )
    - Unique identifier defined s.t. if a.equals(b) then a, b have same hashCode.
    - Cover in later chapter of text.

## Enum Types

- Example:
  - enum Suit {CLUBS, DIAMONDS, HEARTS, SPADES};
- Operations:
  - int compareTo(Suit other)
  - String toString()
  - int ordinal() *starts with 0, not 1*
  - static Suit valueOf(String name)
  - static Suit[] values() *returns array of all values*

## Interfaces & Inheritance

- Class implements interface if supports all methods defined in interface
- Interface can extend another by adding methods
  - If A extends B and x has type A, then also has type B
- One class can extend another
  - inherits fields and methods
  - can override existing methods, add new ones
- instanceof & casts
  - Ex: in Ratio class later

## Card Deck Examples

- CardInterface -- interface
- AbsCard
  - abstract class, implements CardInterface
- Card extends AbsCard
- OtherCard extends AbsCard

Alternate implementations

- Deck
  - Class using cards

## To Do

- Read through CS 62 course webpage
- Java Structures (JS) Reading
  - Today's lecture: JS Ch 0, Ch 1, B.3.2
  - Friday's lecture: JS Ch 2
- Daily problems (see website)
- Accept Piazza invitation