Lecture 0

Introduction

The Goal of CS 1130

- Acquire competency in basic Java
  - Leverage previous programming experience
  - Focus on the aspects that (might be) new
- Acquire competency in OO programming
  - The concepts extend beyond Java
  - Lots of OO languages (Python, Objective-C…)
- This course is for students who took old 1112
  - Freshmen do not need to take this course

Course Structure

- Hands on labs every Wednesday
  - Designed for quick feedback on your progress
  - Go to any lab you want or none at all
  - But you must do the lab and show it to someone
  - Can submit during Consultant hours if you want
- Three assignments
  - Two programming, one written
  - Keep revising assignments until you pass
- No final exam!

Outside of Class

- Course Web page
  - http://www.cs.cornell.edu/courses/cs1130
  - All assignments and labs are posted
  - Welcome to finish them all early
- Course Management System
  - Where to submit assignments, receive feedback
    - http://cms.csuglab.cornell.edu
  - Not on CMS? E-mail ccf27@cornell.edu

Outside of Class

- Sign up for Piazza
  - Online discussion forum for students
  - Questions can be answered by anyone
  - Faster than waiting for an e-mail response
- Consultant Hours
  - Sunday-Thursday 4:30-9:30 in ACCEL Labs
  - There to help CS 1110 AND CS 1130
  - Some extra hours near CS 1110 deadlines
  - Can turn in your labs at this time
DrJava: An IDE for Java

- **IDE**: Integrated Development Environment
  - Makes programming easier
  - Other IDEs: Eclipse, NetBeans
- **Analogy**: Web Design Tools
  - Could just write pure HTML
  - But design tools make easier
- **DrJava**: Interactions pane
  - Works like a calculator
  - Allows us to get started quickly
  - But you still have to understand types

Java is a Strongly Typed Language

- **Type**: A set of values and the operations on them.
  - Examples of operations: +, -, /, *
  - The meaning of these depends on the type
- **Type int**
  - Values: ..., –3, –2, –1, 0, 1, 2, 3, 4, 5, ...
  - Operations: +, –, *, /, unary –
  - **Principal**: operations on int values must yield an int
  - **Example**: 1 / 2 rounds result down to 0

Java is a Strongly Typed Language

- **Type double**
  - Values: Numbers in scientific notation, e.g. 2.0, 22.3, 22.51E–6 (same as 0.00002251)
  - Operations: +, –, *, /, unary –
  - **An approximation** to the real numbers
    - Again, Java cannot represent them all
    - Double.MIN_VALUE: 4.9E–324
    - Double.MAX_VALUE: 1.7976931348623157E308

Casting: Converting Value Types

- **Basic form**: `(type)value`
  - `(double) 2` casts 2 to type `double`. Value is 2.0
  - **Widening cast**: Java does it automatically if needed
  - `(int) 2.56` casts 2.56 to type `int`. Value is 2
  - **Narrowing cast**: Java never does it automatically because it might lose information.
  - **Narrow to wide**: `int ⇒ long ⇒ float ⇒ double`
  - **Other examples**:
    - `(double)(int) 2.56` Value is 2.0
    - `(double) 2.56` Value is 2.56