Course Overview / Review

Recall the software life cycle:

Analysis
de
define the problem
develop program specifications

Design

Coding

Testing
black box vs white box
test coverage
test cases: positive, negative, boundary

Deployment (or operation or maintenance)
bug fixes
add new features
Design

Choose classes, methods, data, etc.

Recognize relationships between classes

Goals:
  - maximize cohesion
  - minimize coupling

Describing the design using UML diagrams
  - class diagrams
  - class relationship diagrams
Coding

Edit - compile - run cycle

Variables

Data types: primitive vs reference

Sending messages (calling methods)

Defining classes
  - data members
  - constructors
  - methods
  - visibility modifiers
  - class vs instance

Control-flow constructs
  - selection statements: if, if-else, switch
  - repetition statements: for, while, do-while

Building connections between classes
  - interfaces
  - inheritance – is-a relationships
  - polymorphism

Handling problems
  - building in checks for "bad" input
  - exceptions
    - defining
    - throwing
    - handling using try-catch or by passing down the call stack
  - checked vs unchecked exceptions

Input and output
  - console
  - file – text based
Three Themes

Compatibility

Accessibility

Compile-time vs run-time

_compatability_ - assignment & casting

Primitive types: \( \text{double } f = 7; \)

but not: \( \text{int } x = 3.14; \rightarrow \text{int } x = (\text{int}) 3.14; \)

Reference types

_interfaces_: \( \text{TaskList list = anything that implements TaskList} \)

_inheritance_: \( \text{Super } t = \text{SubClass value} \)

Vehicle \( t = \text{new Car}() \);

_casting_: \( \text{Car } c = (\text{Car}) t; \rightarrow \text{use instanceof before casting to avoid ClassCastException} \)

Accessibility _public vs private_

_static vs non-static of inherited data & methods_

Compile-time vs Runtime

Compile-time errors vs run-time errors

Polymorphism

Compile-time \( \rightarrow \) type of _var_ determines what can be done with the _var_

Run-time \( \rightarrow \) type of _object_ determines which code to execute

Exceptions

Compile-time \( \rightarrow \) must handle checked exceptions

Run-time \( \rightarrow \) how an exception gets handled
What's next?

CS major

need 302, 367, and 240 to declare

CS certificate

6 courses:

- 302, 367
- up to 2 other under -400 level courses
- at least 2 400+ level classes

See http://www.cs.wisc.edu/ugac/ for full details
What's next next?

CS 367: Introduction to Data Structures and Algorithms

Data structures
- stacks
- queues
- priority queues
- linked lists
- trees
- graphs

Algorithms
- recursion
- searching
- sorting
- hashing

Analysis of data structures and algorithms
- complexity

Example: array of addresses
look up an address - how long does it take

unsorted

best case: 1 step
worst case: N steps \( (N = \text{size of array}) \)
avg case: \( N/2 \) steps

sorted

best case: 1 step
worst case: on the order of \( \log_2 N \) steps
avg case:

\[
\text{height of tree} \\ n \log_2 N
\]