

Introduction to CSE 331

Software Design & Implementation

Winter 2011

Course staff

- Lecturer:
 - Michael Ernst
- TAs:
 - Brian Burg
 - Jacob Nicholson
 - William Pitts

Ask us for help!

Main topic: Managing complexity

- Abstraction and specification
 - Procedural, data, control flow
 - Why they are useful and how to use them
- Writing, understanding, and reasoning about code
 - The examples are in Java, but the issues are more general
 - Object-oriented programming
- Program design and documentation
 - What makes a design good or bad (example: modularity)
 - The process of design and design tools
- Pragmatic considerations
 - Testing
 - Debugging and defensive programming
 - Managing software projects

The goal of system building

- To create a correctly functioning artifact!
- All other matters are secondary
 - Many of them are ***essential*** to producing a correct system
- We insist that you learn to create correct systems
 - This is hard (but fun and rewarding!)

Why is building good software hard?

- Large software systems are enormously complex
 - Millions of “moving parts”
- People expect software to be malleable
 - After all, it’s “only software”
 - Software mitigates the deficiencies of other components
- We are always trying to do new things with software
 - Relevant experience often missing
- Software engineering is about:
 - Managing complexity
 - Managing change
 - Coping with potential defects
 - Customers, developers, environment, software

Programming is hard

- It is surprisingly difficult to specify, design, implement, test, debug, and maintain even a simple program
- CSE 331 will challenge you
- If you are having trouble, *think* before you act
 - Then, look for help
- We strive to create assignments that are reasonable if you apply the techniques taught in lecture
 - ... but hard to do in a brute-force manner

Prerequisites

- Knowing Java is a prerequisite
 - We assume you have mastered 142 and 143

Examples:

- Sharing:
 - Distinction between `==` and `equals()`
 - Aliasing (multiple references to the same object)
- Subtyping
 - Varieties: classes, interfaces
 - Inheritance and overriding
- Object-oriented dispatch:
 - Expressions have a compile-time type
 - Objects/values have a run-time type

Logistics

- Website: <http://www.cs.washington.edu/cse331>
 - See the website for all administrative details
 - Read (all) the handouts!
 - There are required texts
- Run student-setup by 8pm tonight
 - Problem Set 0 is due on Wednesday morning
- Collaboration policy:
 - Discussion is permitted
 - Carrying materials from discussion is not permitted
 - Everything you turn in must be your own work
 - You may not view others' work
 - If you have a question, ask