**CSE 331** Software Design and Implementation

## Credits
4.0 (3 hrs lecture, 1 hr section)

## Lead Instructor
Michael Ernst

## Textbook

- *Effective Java*, Bloch
- *Pragmatic Programmer*, Hunt & Thomas

## Course Description
Explores concepts and techniques for design and construction of reliable and maintainable software systems in modern high-level languages; program structure and design; program-correctness approaches, including testing; and event-driven programming (e.g., graphical user interface). Includes substantial project and software-team experience.

## Prerequisites
CSE 143.

## CE Major Status
Selected Elective

## Course Objectives

There is a level of programming maturity beyond introductory programming that comes from building larger systems and understanding how to specify them precisely, manage their complexity, and verify that they work as expected. After completing this course successfully students should be able to:

- Successfully build medium-scale software projects in principled ways
- Understand the role of specifications and abstractions and how to verify that an implementation is correct, including effective testing and verification strategies and use of formal reasoning
- Analyze a software development problem and be able to design effective program structures to solve it, including appropriate modularity, separation of abstraction and implementation concerns, use of standard design patterns to solve recurring design problems, and use of standard libraries

- Use modern programming languages effectively, including understanding type systems, objects and classes, modularity, notions of identity and equality, and proper use of exceptions and assertions
- Gain experience with contemporary software tools, including integrated development environments, test frameworks, debuggers, version control, and documentation processing tools

To gain experience we will use Java and associated tools like Eclipse, JUnit, JavaDoc, and Subversion, but the goal is to understand the underlying ideas and concepts that are widely applicable to software construction.

## ABET Outcomes

(a) an ability to apply knowledge of mathematics, science, and engineering
(c) an ability to design a system, component, or process to meet desired needs within realistic constraints such as economic, environmental, social, political, ethical, health and safety, manufacturability, and sustainability
(e) an ability to identify, formulate, and solve engineering problems
(k) an ability to use the techniques, skills, and modern engineering tools necessary for engineering practice

## Course Topics

- Reasoning about programs: pre- and post-conditions, invariants, and correctness
- Abstract data types, specification, implementation, abstraction functions, representation invariants, notions of equality
- Java language issues: subclasses and subtypes, generics, exceptions, assertions, etc.
- Tools: Eclipse IDE, version control, svn
- Code quality, style, comments, documentation, JavaDoc
- Testing, test coverage, black- and white-box testing, test-first development, regression testing, JUnit
- Debugging strategies and tools
- Design: modular design, coupling, cohesion; design patterns; basic UML as a design notion
- User interfaces, callbacks, separation of model from view/control