CSE 2221 Syllabus

CSE 2221: Software I: Software Components

Description

Intellectual foundations of software engineering; design-by-contract principles; mathematical modeling of software functionality; component-based software from client perspective.

Level and Credits

- U 4 (two one-hour lectures, two one-hour labs)

Prerequisites

- CSE 1211 or CSE 1212 or CSE 1221 or ENGR 1221 or CSE 1222 or CSE 1223 or CSE 201 or CSE 202 or CSE 203 or CSE 204 or CSE 205 or EG 167 or CSE Placement Level A; co-req: Math 1151 or Math 1161

General Information, Exclusions, etc.

- Java is taught and used

Course Goals (Intended Learning Outcomes)

- Be familiar with the reasons it is important that software be "correct", i.e., why "good enough" is not good enough when it comes to software quality
- Be familiar with the reasons for designing software to minimize the impact of change, and why it is difficult to achieve this
- Be familiar with using design-by-contract principles to write software that uses existing software components based on their interface contracts
- Be familiar with using interface contracts that are described using simple predicate calculus assertions with mathematical integer, string, finite set, and tuple models
- Be familiar with extending existing software components by layering new operations on top of existing operations
- Be familiar with using simple recursion
- Be familiar with using simple techniques to test application software and layered implementations of extensions, including developing and carrying out simple specification-based test plans
- Be familiar with using simple techniques to debug application software and layered implementations of extensions
- Be exposed to using basic algorithm analysis techniques and notations to analyze and express execution times of operations whose implementations involve

straight-line code and simple loops
- Be competent with writing Java programs in a procedural style using the basic control structures, primitive value types, character strings, and input/output
- Be familiar with writing Java programs using core language features including interfaces, classes, inheritance, and assertions
- Be familiar with using an understanding of the difference between value types and reference types to trace the execution of simple Java code in situations involving both flavors of types, including their use as parameters to method calls
- Be familiar with writing Java programs that use software components similar to (but simplified from) those in the Java collections framework
- Be exposed to writing Java simple programs with graphical user interfaces
- Be familiar with testing using JUnit
- Be familiar with illustrating key dependencies between software components using UML class diagrams (or similar)
- Be familiar with using the most important features of a modern IDE such as Eclipse

Texts

- All course materials are provided on-line for free, including the recommended textbook used as a Java reference:

  C.S. Horstmann, *Java for Everyone*, John Wiley and Sons, 2013

Course Topics

- Introduction to Java; introduction to Eclipse; basic input/output; value types; control structures; methods, calls, and parameter passing
- Software components; design-by-contract; packages; interfaces; classes; reference types; equals and toString methods; XMLTree components; NaturalNumber components; introduction to UML class diagrams (or similar)
- Layered implementations of new NaturalNumber methods; introduction to recursion; introduction to specification-based testing and JUnit
- Generics; Sequence components; Queue components; Stack components
- Iterators; Set components; Map components
- Graphical user interfaces (GUIs) using Swing; model-view-controller (MVC) design pattern
- Reasoning about software correctness; loop invariants

Grading Plan

- **Note: A passing score on the final exam is *required* in order to receive a passing grade for the course.**

Homework Assignments (many)	6%
Project Assignments (several)	30%
Midterm Exams (2 @ 15% each)	30%

Final Exam	30%
Participation	4%

BS CSE Program Outcomes

Course Contribution		Program Outcome
***	a	an ability to apply knowledge of computing, mathematics including discrete mathematics as well as probability and statistics, science, and engineering;
*	b	an ability to design and conduct experiments, as well as to analyze and interpret data;
***	c	an ability to design, implement, and evaluate a software or a software/hardware system, component, or process to meet desired needs within realistic constraints such as memory, runtime efficiency, as well as appropriate constraints related to economic, environmental, social, political, ethical, health and safety, manufacturability, and sustainability considerations;
	d	an ability to function on multi-disciplinary teams;
**	e	an ability to identify, formulate, and solve engineering problems;
	f	an understanding of professional, ethical, legal, security and social issues and responsibilities;
*	g	an ability to communicate effectively with a range of audiences;
	h	an ability to analyze the local and global impact of computing on individuals, organizations, and society;
*	i	a recognition of the need for, and an ability to engage in life-long learning and continuing professional development;
	j	a knowledge of contemporary issues;
***	k	an ability to use the techniques, skills, and modern engineering tools necessary for practice as a CSE professional;
**	l	an ability to analyze a problem, and identify and define the computing requirements appropriate to its solution;
*	m	an ability to apply mathematical foundations, algorithmic principles, and computer science theory in the modeling and design of computer-based systems in a way that demonstrates comprehension of the tradeoffs involved in design choices;

***	n	an ability to apply design and development principles in the construction of software systems of varying complexity.