Software I: Software Components
CSE 2221

Description / Conditions

Transcript Abbreviation:
SW 1: Components

Course Description:
Intellectual foundations of software engineering; design-by-contract principles; mathematical modeling of software functionality; component-based software from client perspective; layered data representation.

Course Levels:
Undergraduate (1000-5000 level)

Designation:
Required
Elective

General Education Course:
(N/A)

Cross-Listing:
(N/A)

Course Detail

Credit Hours (Minimum if “Range” selected):
4.00

Max Credit Hours:
(N/A)
Select if Repeatable:
Off

Maximum Repeatable Credits:
(N/A)

Total Completions Allowed:
(N/A)

Allow Multiple Enrollments in Term:
No

Course Length:
14 weeks (autumn or spring)
12 weeks (summer only)

Off Campus:
Never

Campus Location:
Columbus
Lima
Marion

Instruction Modes:
In Person (75-100% campus; 0-24% online)

Prerequisites and Co-requisites:
Prereq: 1212, 1221, 1222, 1223, 1224, Engr 1221, 1281.01H, 1281.02H, or CSE Placement Level A. Prereq or concur: Math 1151, 1161.01, or 1161.02.

Electronically Enforced:
No

Exclusions:
Not open to students with credit for 5022.

Course Goals and Learning Objectives
**Course Goals / Objectives:**
Reasons it is important that software be "correct", i.e., why "good enough" is not good enough when it comes to software quality
Reasons for designing software to minimize the impact of change, and why it is difficult to achieve this
Be competent with using design-by-contract principles to write software that uses existing software components based on their interface contracts
Be competent with using interface contracts that are described using simple predicate calculus assertions with mathematical integer, string, finite set, and tuple models
Be competent with extending existing software components by layering new operations on top of existing operations
Be competent with layering new software components' data representations on top of existing software components
Be competent with using simple recursion
Be competent with using simple techniques to test application software, layered implementations of extensions, and layered data representations, including developing and carrying out simple specification-based test plans
Be competent with using simple techniques to debug application software, layered implementations of extensions, and layered data representations
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 writing Java programs that use software components similar to (but simplified from) those in the Java collections framework
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 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, e.g., Eclipse

**Check if concurrence sought:**
No

**Contact Hours**
Contact Hours:

<table>
<thead>
<tr>
<th>Topic</th>
<th>LEC</th>
<th>REC out-of-class</th>
<th>REC in-class</th>
<th>Weekly LAB out-of-class</th>
<th>Weekly LAB in-class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction to Java; value types; control structures; basic input/output; introduction to Eclipse</td>
<td>4.0</td>
<td>0.0</td>
<td>0</td>
<td>4.0</td>
<td>0</td>
</tr>
<tr>
<td>Software components; packages; interfaces; design-by-contract; classes; reference types; methods, calls, and parameter passing; equals and toString methods; Text component; Natural component; introduction to UML class diagrams (or similar)</td>
<td>6.0</td>
<td>0.0</td>
<td>0</td>
<td>6.0</td>
<td>0</td>
</tr>
<tr>
<td>Layered implementations of new Text and Natural methods; introduction to recursion; introduction to specification-based testing and JUnit</td>
<td>4.0</td>
<td>0.0</td>
<td>0</td>
<td>4.0</td>
<td>0</td>
</tr>
<tr>
<td>Generics; Sequence component; Queue component; Stack component; List component; layered implementations of new Sequence, Queue, Stack, and List methods; more recursion</td>
<td>4.0</td>
<td>0.0</td>
<td>0</td>
<td>4.0</td>
<td>0</td>
</tr>
<tr>
<td>Set component; Map component; iterators</td>
<td>4.0</td>
<td>0.0</td>
<td>0</td>
<td>4.0</td>
<td>0</td>
</tr>
<tr>
<td>Layered data representation concepts; representation invariants and abstraction functions; Natural representation using a Stack; Sequence/Queue/Stack representation using a List</td>
<td>4.0</td>
<td>0.0</td>
<td>0</td>
<td>4.0</td>
<td>0</td>
</tr>
<tr>
<td>Total</td>
<td>26</td>
<td>0</td>
<td>0</td>
<td>26</td>
<td>0</td>
</tr>
</tbody>
</table>

Grading and Texts

Grading Plan:
Letter Grade

Course Components:
Lecture
Lab
Grade Roster Component:
Lecture

Credit by Exam (EM):
Yes

Grades Breakdown:

<table>
<thead>
<tr>
<th>Aspect</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Homework and Class Participation</td>
<td>8%</td>
</tr>
<tr>
<td>Closed Labs</td>
<td>12%</td>
</tr>
<tr>
<td>Programming Lab Assignments</td>
<td>30%</td>
</tr>
<tr>
<td>Midterm Exam</td>
<td>20%</td>
</tr>
<tr>
<td>Final Exam</td>
<td>30%</td>
</tr>
</tbody>
</table>

Representative Textbooks and Other Course Materials:

<table>
<thead>
<tr>
<th>Title</th>
<th>Author</th>
<th>Year</th>
</tr>
</thead>
<tbody>
<tr>
<td>On-line reference materials</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

ABET Student Learning Outcomes
**ABET-CAC Criterion 3 Outcomes:**

<table>
<thead>
<tr>
<th>Contribution Level</th>
<th>Result</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Significant (7+) hours</td>
<td>1</td>
<td>Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions.</td>
</tr>
<tr>
<td>Significant (7+) hours</td>
<td>2</td>
<td>Design, implement, and evaluate a computing-based solution to meet a given set of computing requirements in the context of the program’s discipline.</td>
</tr>
<tr>
<td>Some (1-2 hours)</td>
<td>3</td>
<td>Communicate effectively in a variety of professional contexts.</td>
</tr>
<tr>
<td>Some (1-2 hours)</td>
<td>4</td>
<td>Recognize professional responsibilities and make informed judgments in computing practice based on legal and ethical principles</td>
</tr>
<tr>
<td>Substantial (3-6 hours)</td>
<td>6</td>
<td>Apply computer science theory and software development fundamentals to produce computing-based solutions.</td>
</tr>
</tbody>
</table>

**ABET-ETAC Criterion 3 Outcomes:**

(N/A)

**ABET-EAC Criterion 3 Outcomes:**

<table>
<thead>
<tr>
<th>Contribution Level</th>
<th>Result</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Significant (7+) hours</td>
<td>1</td>
<td>an ability to identify, formulate, and solve complex engineering problems by applying principles of engineering, science, and mathematics</td>
</tr>
<tr>
<td>Significant (7+) hours</td>
<td>2</td>
<td>an ability to apply engineering design to produce solutions that meet specified needs with consideration of public health, safety, and welfare, as well as global, cultural, social, environmental, and economic factors</td>
</tr>
<tr>
<td>Some (1-2 hours)</td>
<td>3</td>
<td>an ability to communicate effectively with a range of audiences - pre-2019 EAC SLO (g)</td>
</tr>
<tr>
<td>Some (1-2 hours)</td>
<td>4</td>
<td>an ability to recognize ethical and professional responsibilities in engineering situations and make informed judgments, which must consider the impact of engineering solutions in global, economic, environmental, and societal contexts</td>
</tr>
<tr>
<td>Some (1-2 hours)</td>
<td>6</td>
<td>an ability to develop and conduct appropriate experimentation, analyze and interpret data, and use engineering judgment to draw conclusions</td>
</tr>
<tr>
<td>Some (1-2 hours)</td>
<td>7</td>
<td>an ability to acquire and apply new knowledge as needed, using appropriate learning strategies</td>
</tr>
</tbody>
</table>
Embedded Literacies (UG courses only)

Embedded Literacies Info:

Attachments / Additional Notes or Comments

Attachments:
(N/A)

Additional Notes or Comments:
(N/A)