CSE 143 Computer Programming II

Credits
5.0 (3 hrs lecture, 2 hrs section)

Lead Instructor
Stuart Reges

Textbook
Building Java Programs: A Back to Basics Approach, Reges & Stepp

Course Description
Continuation of CSE 142. Concepts of data abstraction and encapsulation including stacks, queues, linked lists, binary trees, recursion, instruction to complexity and use of predefined collection classes.

Prerequisites
CSE 142.

CE Major Status
Required

Course Objectives
Students learn about data abstraction and the basic design principles of object oriented programming. Students will become familiar with standard data abstractions (lists, maps, sets, stacks, queues) as well as a variety of implementation techniques (arrays, linked lists, binary trees). Students will learn to program using recursion and recursive backtracking. Students will learn the object-oriented constructs that support code reuse (encapsulation, interfaces, inheritance, abstract classes) as well as learning how to make use of off-the-shelf components from libraries like the Java Collections Framework.

ABET Outcomes
(1) an ability to identify, formulate, and solve complex engineering problems by applying principles of engineering, science, and mathematics (H)
(2) 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, , and economic factors (H)
(7) an ability to acquire and apply new knowledge as needed, using appropriate learning strategies (H)

Course Topics
Abstract data types: stacks, queues, lists, maps, sets

Implementing linked structures (linked lists, binary trees)
Recursion and recursive backtracking
Using off-the-shelf components (e.g., Java Collections Framework)
Use of inheritance for additive change and factoring out common code into abstract classes
Class design: encapsulation, documentation, throwing exceptions, appropriate choice of fields
Thorough testing and debugging
Time and space complexity
Efficient sorting and searching algorithms (binary search, mergesort)
Iterator use and implementation