

## **CSE116: INTRODUCTION TO COMPUTER SCIENCE FOR MAJORS II (4 Credits, Required)**

### **Catalog Description**

Continuation of CSE115. Heavily emphasizes abstract data types (ADTs) and object-oriented methodology, and expects students not only to understand ADTs but also to design and implement robust ADTs using a modern object-oriented programming language. Further emphasizes object-oriented techniques, which support sound software engineering, such as encapsulation, polymorphism and inheritance as well as the use of more complex design patterns. Essential topics integrated in this framework include the use of recursion; linked data structures, including lists, stacks, queues, binary trees, and other advanced data structures; and algorithms for searching and sorting; exceptions and exception handling, event-driven programming. Introduces the analysis of algorithm complexity (O-notation).

### **Prerequisites**

Prerequisite: CSE115 or permission of the instructor.

Corequisites: None

### **Textbooks(s) and/or other required material**

Frank M. Carrano. (2007). Data Structures and Abstractions with Java (2nd ed), Pearson Education (Prentice Hall). (ISBN-13: 978-0-13-237045-5).

Cay Horstmann. (2006). Object-Oriented Design and Patterns (2nd ed), John Wiley & Sons, Inc. (ISBN-13: 978-0-471-74487-0)

Kent Beck with Cynthia Andres. (2005). Extreme Programming Explained (2nd ed), Pearson Education (Addison-Wesley). (ISBN-13: 978-0-321-27865-4).

### **Course Objectives**

The main objectives of this course are to introduce students to data structure design and algorithm analysis as well as the fundamental principles of software design.

### **Topics Covered**

Abstract data types (linear, such as lists and arrays, and non-linear, such as binary trees, including variants which maintain certain properties, such as an order property in a binary search tree)

Analysis of algorithms (big-oh notation)

Design Patterns (e.g. Observer, Visitor, Iterator)

$O(N^2)$  and  $O(N \log N)$  sorting algorithms, implementation and analysis (e.g. selection sort, merge sort)

Team software development (processes, team development)

### **Class / Lab Schedule**

Three 50-minute lectures per week

One 110-minute recitation per week

**Contribution of course to professional component/criterion 5**

Engineering Topics: 4 credits

**Relationship of course to program outcomes**

This course is required of all computer engineering students and has a significant relationship with the following program objectives for computer engineering:

- (h) The broad education necessary to understand the impact of engineering solutions in a global, economic, environmental, and societal context

This course has a strong relationship with the following program objectives for computer engineering:

- (k) An ability to use the techniques, skills, and modern hardware and software engineering tools necessary for computer engineering practice.

**Persons who prepared this description and date of preparation**

Carl Alphonse, last updated June 2008