

CS115 – Object-Oriented Programming I

Last Updated - 1/29/04

Course Manager - Matthew Bauer, Senior Lecturer

2 credit hours; required for CS & CPE (or CS201); 100 min. lecture & 50 min. lab each week

Catalog Description - Introduces the use of a high-level object-oriented programming language as a problem-solving tool – including basic data structures and algorithms, object-oriented programming techniques, and software documentation. Designed for students who have had little or no prior experience with computer programming. For students in CS and CS related degree programs. (2-1-2)

Course Goals - Students should be able to:

- Analyze and explain the behavior of simple programs involving the following fundamental programming constructs: assignment, I/O (including file I/O), selection, iteration, functions
- Write a program that uses each of the following fundamental programming constructs: assignment, I/O (including file I/O), selection, iteration, functions
- Break a problem into logical pieces that can be solved (programmed) independently.
- Develop, and analyze, algorithms for solving simple problems.
- Use a suitable programming language, and development environment, to implement, test, and debug algorithms for solving simple problems.
- Write programs that use each of the following data structures (and describe how they are represented in memory): strings, arrays, structures, and class libraries including strings and vectors
- Explain and apply object-oriented design and testing involving the following concepts: data abstraction, encapsulation, information hiding
- Use a development environment to design, code, test, and debug simple programs, including multi-file source projects, in an object-oriented programming language.

Major Topics Covered in Course

1. Fundamental data storage and manipulation (types and variables, statements and expressions)	
2. Functions	
3. Classes (classes and objects, instance variables and instance methods, and encapsulation).	
4. Flow of control (Boolean expressions, conditional statements, and loops).	
5. Vectors	
6. Problem Solving approaches (This section is dispersed appropriately throughout the semester to illustrate the above techniques.)	
7. Software Engineering – design, testing, debugging (This section is dispersed appropriately throughout the semester to illustrate the above techniques.)	
Exams	
Final Exam	