

**CS 1511: Computer Science I (5)****Catalog Description:**

Introduction to the discipline of computer science. Emphasis on problem analysis, design, and development using event-driven programming in a graphical user interface environment. Programming concepts include control structures, arrays, recursion, pointers, classes and introduction to the object-oriented approach.

**Textbook:** Allert, J. (2009) *Programming with Visual C++: Concepts and Projects*. Boston, Course Technology/Cengage Learning.

**Course Goals:**

This course covers all of the elementary concepts of computer science as well the basics of program design, analysis and the C programming language. Students will study the fundamentals of computer systems, ethical computing, the history of computing, structured program design techniques (algorithms and traces), elementary C programming statements, functions, control structures (sequential, selection, repetition), complex expression rules and truth tables, counting and summation, the array data structure,  $n^2$  sorting methods, sequential and binary searching, analysis, strings, pointers, object-oriented programming, software engineering tools and recursion.

**Prerequisites by Course & Topic:**

3½ years of high school math

**Major Topics Covered in the Course:**

- The Fundamentals of Computer Systems
- The History of Computing
- Structured Program Design Techniques
- Elementary C programming Statements
- Functions, Parameter Passing and Variable Scope
- Operator Precedence and Complex Expressions
- Control Structures
- Counting and Summation
- One and Two-dimensional Arrays
- Sorting and Searching Algorithms
- Program Analysis and Big-O
- Strings and String Functions
- Pointers and Pointer Arithmetic
- Dynamic Memory Allocation
- Object-Oriented Programming
- Recursion

**Class/Laboratory Schedule:** Lecture: 3 hours per week, Discussion: 1, Laboratory: 1

**Course Outcomes:**

1. Students have an understanding of the fundamental concepts of computer science.
  - a. Students know basic hardware and software concepts.
  - b. Students are familiar with key persons and advances in the history of computing.
2. Students understand issues related to software development.
  - a. Students understand structured program design.
  - b. Students develop efficient and effective algorithms.
  - c. Students verify the correctness of algorithms.
3. Students understand key programming constructs.
  - a. Students master the essential programming constructs including program structure, variable declaration and scope, data types, operator precedence, input and output, assignment.

- b. Students are familiar with complex program flow control including the implementation of a variety of selection and repetition structures as well as nesting.
- c. Students understand value and reference parameters.
- 4. Students understand the concept of program complexity.
  - a. Students understand how the complexity and efficiency of an algorithm is investigated and determined.
  - b. Students understand the Big-O concept and families of functions related to sorting, searching and other algorithms presented in class.
  - c. Students gain proficiency in n<sup>2</sup> sorting techniques.
  - d. Students understand principles of recursion.
- 5. Students understand basic data structure.
  - a. Students become proficient with strings, one and two-dimensional arrays and pointer-based structures.
  - b. Students understand dynamic memory allocation.
- 6. Students gain experience with object-oriented programming.
  - a. Students understand how to construct UML class diagrams.
  - b. Students understand the principles of object-oriented programming including instantiation, encapsulation, data hiding and the use of const.
  - c. Students are proficient with constructors (including copy constructors), destructors and operator overloading.

**Relationship to Program Outcomes**

This course is the first course in computer science. Students do not have to have prior programming experience. This course contributes to meeting the following program outcomes:

2. *Students can design, develop, and analyze significant software systems.*

This course introduces students to the basics of software design and analysis. Students learn a variety of data structures, the algorithms associated with them, and how to analyze algorithms. Course outcomes 1, 2, 3, 4, and 6 map to this program outcome.

3. *Students understand the fundamentals of computer organization and architecture, data structures and related algorithms, and programming languages.*

Students write C programs using a variety of data structures and implement a variety of searching and sorting algorithms. Students acquire a basic understanding of the components of a computing system and how these components interact. Course outcomes 1, 3, 4, 5, and 6 map to this program outcome.

4. *Students can apply computer science principles and practices to a variety of problems.*

Students design and implement solutions to a wide range of problems requiring familiarity with the broad spectrum of programming techniques and tools covered in this course. Course outcomes 2, 3, 4, and 6 map to this program outcome.

**Assessment Plan for Course:**

This course is assessed every other year by the instructor and a course assessment document covering all of the course outcomes and their effect on the program outcomes is prepared.

**Estimate CSAB Category Content**

	CORE	ADVANCED		CORE	ADVANCED
Data Structures	1		Computer Organization and Architecture		
Algorithms	1		Concept of Programming Languages	2	
Software Design	1				

**Coordinator/Prepared by:** J. Allert