# **CS 2010 :** PROGRAMMING FUNDAMENTALS

| | | |
|---|---|---|
| *Semester Hours:* | 3.0 | *Contact Hours:* 3 |
| *Coordinator:* | Jadwiga A. Carlson | |
| *Text:* | Programming in C++ with zyBooks & zyLabs | |
| *Author:* | VAHID & LYSECKY | |
| *Year:* | 2020 | |

## SPECIFIC COURSE INFORMATION

*Catalog Description:*

Problem solving and algorithm development. Basic programming concepts including elementary data types, arrays, strings, files, control structures, and functions. Searching and sorting algorithms. Testing and debugging strategies. Prerequisite: Math placement score of 32 or MATH 1200 or MATH 99 or higher. Approved for distance education.

Course type: **REQUIRED**

## SPECIFIC COURSE GOALS

- I can explain the fundamental concepts of procedural programming.
- I can use a high-level language to write programs to solve problems.
- I can analyze problem requirements in order to understand what type of data and processes are involved in the system.
- I can design a solution using a modular approach and organize program code to implement the design.
- I can debug programs and verify that the output of a program satisfies the problem requirements.
- I can implement algorithms to search and sort an array.
- I can implement simple recursive functions.

COMPUTER SCIENCE STUDENT OUTCOMES ADDRESSED BY THIS COURSE

- CS 1 Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions
- CS 2 Design, implement, and evaluate a computing-based solution to meet a given set of computing requirements in the context of the program's discipline
- CS 5 Function effectively as a member or leader of a team engaged in activities appropriate to the program's discipline
- CS 6 Apply computer science theory and software development fundamentals to produce computing-based solutions

SOFTWARE ENGINEERING STUDENT OUTCOMES ADDRESSED BY THIS COURSE

- SE 5 An ability to function effectively on a team whose members together provide leadership, create a collaborative and inclusive environment, establish goals, plan tasks, and meet objectives
- SE 7 An ability to acquire and apply new knowledge, as needed, using appropriate learning strategies

LIST OF TOPICS COVERED

- Introduction and Basic Concepts (1 week = 7%)
    - Computers, programs, C++ (compiler, linker, syntax/logic/run-time errors)
    - Problem solving and use of abstraction
    - Developing algorithms (flowcharting, pseudocode)
- Basic concepts (2 weeks = 14%)
    - Primitive data types (int, char, string, double, and bool)
    - Assignment statement
    - Evaluating expressions (order of precedence)
    - Coercion, type casting (type of polymorphism)
    - Testing/debugging (step over/into/out, conditional breakpoints)
    - Documentation standards (self-documenting naming conventions)

- Decision/Control structure (1.5 weeks = 11 %)
    - Relational and logical operators
    - Conditional clauses (if, if/else, if/else if, switch)
- Loops (2 weeks = 14%)
    - Loop structures (while, do while, for)
    - Sentinel value with while & do while
    - Processing a list of numbers (compute average, find largest/smallest, running total)
    - Input Validation
- Introduction to text file I/O (1 week = 7%)
    - Opening, processing, closing of a file, eof()
    - Additional data types (ifstream, ofstream, fstream)
- Functions (2 weeks = 14%)
    - Function definition, prototype, call
    - Parameter passing (by value, by reference)
    - Function overloading (another type of polymorphism)
- Recursion (direct/linear, 1 week = 7%)
    - Students trace recursive functions
- Arrays (2 weeks = 14%)
    - 1D and 2D arrays
    - Arrays as parameters
- Sorting and Searching Arrays (1.5 weeks = 11%)
    - Bubble sort
    - Linear and binary search

COMPUTER SECURITY TOPICS

Faculty who recently offered CS 2010 have discussed and identified a list of topics related to computer security in this course. Below is a list for instructors to incorporate. (*) indicates topics that are mandatory.

| Security Topic | Description | Textbook Reference[1] | Estimated Class Hours |
|---|---|---|---|
| *Read Only | Constant values (const) cannot be changed during execution. | Chapter 2.10 | <1 |
| *Validation | Validating/inspecting input data to determine whether it is acceptable. Bad output will be produced from bad input. | Chapter 2.1<br>Chapter 3.7<br>Chapter 5.2 | 1 |
| *Integer overflow | Integer overflow occurs when assigning a value that is too large (overflow) or too small (underflow) to be held in a variable.<br>Variable contains value that is 'wrapped around' set of possible values to negative. | Chapter 2.17 | <1 |
| *Initialization | C++ local variables are not initialized by default, initialize before use. Static local and global variables are initialized by default. | Chapter 2.2<br>Chapter 7.11 | 1 |

[1]Programming in C++ with zyBooks & zyLabs