LABETTE COMMUNITY COLLEGE BRIEF SYLLABUS

SPECIAL NOTE:
This brief syllabus is not intended to be a legal contract. A full syllabus will be distributed to students at the first class session.

TEXT AND SUPPLEMENTARY MATERIALS USED IN THE COURSE (if any):

Please check with the LCC bookstore [http://www.labette.edu/bookstore](http://www.labette.edu/bookstore) for the required texts for this class.

**COURSE NUMBER:** COMP 212

**COURSE TITLE:** PRINCIPLES OF SOFTWARE DESIGN

**SEMESTER CREDIT HOURS:** 3

**DEPARTMENT:** Computer Science

**DIVISION:** General Education

**PREREQUISITE:** COMP 135 C++ Programming

**COURSE DESCRIPTION:**
Intermediate programming techniques using the C++ language. Topics covered include sorting, object oriented programming, data abstraction, algorithmic design, and basic data structure including linked lists and arrays.

**COURSE OUTCOMES AND COMPETENCIES:**
Students who successfully complete this course will be able to:

1. Write programs with arrays, vector types, records (struct) and Classes.
   - List processing with searching, sorting and binary search
   - Explain the differences between arrays and structs
   - Use built in utilities to implement member functions
   - Be able to create a program to hide information

2. Create and modify programs using; inheritance, composition, pointers, classes, and virtual functions.
   - Redefine (override) member functions of base class
   - Create programs using Object-oriented design (OOD) and (OOP)
   - Create data type and pointer variables
   - Create and modify programs using inheritance, classes, virtual destructors and address of operator.
3. Create programs to monitor overloading and exception handling.
   - Explain and demonstrate the overloading operator
   - Create a customized exception classes

4. Define recursion, create linked list and use stacks and queues in a program
   - Explain the difference between recursion and iteration
   - Create and retrieve data of the first and last node
   - Create a program using stack operators
   - Design a queuing system