ILLINOIS VALLEY COMMUNITY COLLEGE

Course Syllabus

Division: Occupational Technologies
Course: CSI 2007 – Advanced C++

Date:	July 5, 2004
Semester Hours:	3
Lecture hours per week:	2
Labs hours per week:	2
Seminar hours per week:	
Other hours:	
Prerequisite:	CSI 1007
Semester Offered:	Fall
Instructor(s):	Tannahill

I. CATALOG DESCRIPTION

As the second class in the C++ programming language sequence, this class builds on the previous programming experience to design and implement large-scale programs for real world problems. Topics to be covered include: data structures, program verification, object-oriented design, text processing, recursion, abstract data types, and user defined types.

II. EXPECTED LEARNING OUTCOMES AND RELATED COMPETENCIES:

Upon completion of the course, the student will be able to:

1. design and implement programs to solve real life problems
2. use data types
3. use data structures
4. use functions
5. use process strings (text)
6. use sequence control structures
7. use structures
8. use object-oriented programming

Outcome 1 – Students will continue a disciplined approach to problem solving, which will include algorithms, pseudocode, and flowcharts

Competency 1.1 –Students will create flowcharts, pseudocode, and algorithms to represent logic flow in the problem solutions

Competency 1.2 - Students will use comments to add clarity to the code, and include documentation acceptable in a professional program

Competency 1.3 - Students will employ good programming style

Competency 1.4 - Students will use test data to verify and validate programs, and understand their complexity
Competency 1.5 - Students will understand the dynamic concepts associated with the use of different programming options available, memory, scope, ect.

Outcome 2 - Students will use data types
Competency 2.1 - Students will declare valid user defined terms and appropriate data types
Competency 2.2 - Students will store user defined data types using linked lists
Competency 2.3 - Students will use arrays as a data type
Competency 2.4 - Students will abstract data, base classes, and classes
Competency 2.5 - Students will understand variables, casting, conversions, unsigned, copying, memory, and scope rules

Outcome 3 - Students will use data structures
Competency 3.1 - Students will use the STL (standard template library)
Competency 3.2 - Students will use sequential containers for vectors, lists, and deques
Competency 3.3 - Students will use files, sets, multisets, stacks, lists, trees, graphs, and queues
Competency 3.4 - Students will use pointers, iterators, file pointers and understand their memory management
Competency 3.5 - Students will create linked lists through class templates
Competency 3.6 - Students will use the algorithms find, count, sort, search, merge, and add_if
Competency 3.7 - Students will use exception syntax and exception handler, multiple exceptions, and exceptions with arguments

Outcome 4 - Students will use functions
Competency 4.1 - Students will create programs with modules
Competency 4.2 - Students will code user defined functions
Competency 4.3 - Students will pass parameters by value, pointer, and reference variables
Competency 4.4 - Students will use recursive, virtual, friend, and static functions
Competency 4.5 - Students will understand function side effects
Competency 4.6 - Students will use simple function templates and function templates with multiple arguments

Outcome 5 - Students will process strings (text)
Competency 5.1 - Students will understand the stream class hierarchy
Competency 5.2 - Students will understand ios and error handling
Competency 5.3 - Students will create programs using objects and disk files
Competency 5.4 - Students will overload directives
Competency 5.5 - Students will use command line arguments
Competency 5.6 - Students will direct output to a printer
Competency 5.7 - Students will use arrays to process text

Outcome 6 - Students will use sequence control structures
Competency 6.1 - Students will use simple, nested, complex, and negated if statements
Competency 6.2 - Students will use switch (case) statements
Competency 6.3 - Students will use pretest and post test repetition structures
Competency 6.3 - Students will use goto, break, and continue statements

Outcome 7- Students will use structures
Competency 7.1 - Students will understand structure declarations and definitions
Competency 7.2 - Students will create programs that use structures, and nested structures
Competency 7.3 - Students will use structures as objects and data types
Competency 7.4 - Students will use enumerations

Outcome 8- Students will use object oriented programming
Competency 8.1 - Students will create programs with public and private objects and classes
Competency 8.2 - Students will create programs with members, and containers
Competency 8.3 - Students will understand inheritance, multiple inheritance, base and derived classes
Competency 8.4 - Students will understand polymorphism
Competency 8.5 - Students will understand object oriented design, generalization, and reusability of code
Competency 8.6 - Students will know when to use objects, constructors, and destructors

III. COURSE CONTENT:

Introduction and requirements
Loops and decisions
Structures
Functions
Objects and classes
Arrays and strings
Operator overloading
Inheritance
Pointers
Vitual functions
Streams and files
Multifile programs
Templates and exceptions
The standard template library
Object-Oriented design

IV. INSTRUCTIONAL METHOD:

Lecture
Lab - hands-on training
Testing
Programming assignments
Teacher demonstration
Group work

V. INSTRUCTIONAL MATERIALS:

Computer overhead projection system

Computer lab

Object-Oriented programming in C++, Robert Lafore

VI. STUDENT REQUIREMENTS AND METHODS OF EVALUATION:

Develop an understanding and/or a comprehensive knowledge of the items listed as course content.

Flowchart, code, compile, test, and document computer programming assignments and individual projects.

1. Read required material on the topic
2. Attend class on current topic
3. Complete all tests and homework
4. Ask questions about any misunderstood area either in class, during office hours, or of the tutor
5. Join in discussions

Grading Scale

A	90-100%
B	80-89%
C	70-79%
D	60-69%

There will be 3 tests given worth 100 pts each. 8-12 programming assignments will be completed worth 40 pts each. Pop quizzes will be given at the instructor's discretion worth 10 pts each.