CSC 112 - Fundamentals of Computer Science

Course Description: (4h) Lecture and laboratory. Problem solving and program construction using top-down design, data abstraction, and object-oriented programming. Linear data structures, recursion, and software development tools are introduced.
Lab: One hour, fifteen minutes per week.
Prerequisites: Computer Science 111 or permission of instructor.

Professor: Dr. William Turkett - West 240, 758-4427, email: turketwh@wfu.edu
Office Hours: (shared with CSC 111 class) MR 5:30-7:00pm, T 4:15-6:00pm, W 1:00-2:00pm, and by appointment

Teaching Assistant: Nicholas Mertaugh (nmertaugh@gmail.com), Edison Munoz (munoef9@wfu.edu)
Office Hours: Nicholas Mertaugh – 10:00 am to 11:00 am, MWF, Manchester 336

Meeting Time: Lecture: MWF 12:00 – 12:50pm, Manchester 241, *Lab:* M 4:00 - 5:15pm, Manchester 17

Webpage: http://turkett112.tumblr.com

Textbook: **Absolute C++,** 4th Edition (2009/2010) by Walter Savitch; Addison-Wesley; ISBN 0-13-608381-1 (Available online or at the Wake Forest University Bookstore)

Grading:
- 3 Tests and Cumulative Final - 60% (15% each)
- Lab Assignments - 30%
- Homeworks - 10%

Because of the intent and nature of this course, failure of the lab portion of the course (below a 60 average on the lab grades) will lead to failure of the course as a whole.

Expected Grading Scale:

A+	97-100	A	92-96	A-	90-91
B+	87-89	B	82-86	B-	80-81
C+	77-79	C	72-76	C-	70-71
D+	67-69	D	62-66	D-	60-61
F	0-59				

Attendance: Regular attendance in class and at labs is expected.

Tests and Final Exam: There will be three tests during the semester to judge the student's progress in the course. These tests may include material from the readings, lectures, and labs. The final exam will be cumulative over the material for the entire semester. All tests and exams will be closed book. Make up tests will be allowed only if the absence is excused by the University.

Homework:
Homework will be assigned approximately once a week. Homework should be turned in at the start of the lecture on the due date. No late homeworks will be accepted.

Programming Lab:
Programming Lab begins on August 31st and will be held in Manchester 17 (the octagon room in basement of Manchester Hall). Please bring your laptop, textbook, class notebook, lab assignment, and any pre-lab materials to each lab. Labs should be completed by the due date printed on the lab, which is currently planned to be Friday morning of the week of the lab.

Academic Integrity: All work should be done independently by each student. Copying of partial or complete work will be referred to the University Judicial System. You should keep evidence when possible to demonstrate your own work. Should a question of authorship arise you will be expected to produce documents that trace the development of your work. Algorithmic and electronic means of detecting copying may be used by the instructor on submitted assignments.

Learning Assistance: If you have a disability that may require an accommodation for taking this course, please contact the Learning Assistance Center (758-5929) within the first two weeks of the semester.

Course Calendar:
Wednesday, August 26 – First day of class
Monday, September 28 – 1st test
Wednesday, September 30 – Last to drop with a W
Friday, October 16 – Fall Break holiday
Sunday, October 18 – Midterm grades posted
Friday, October 23 – 2nd test
Friday, November 20 – 3rd test
Wednesday, November 25th – Sunday, November 29th – Thanksgiving holiday
Friday, December 4 – Last day of class
Saturday, December 12 – Final exam, 2:00pm

University Closure: In the event that the University closes due to pandemic or other disaster, you will be provided with my home address, phone number, and a *CSC 112 Lecture Plan* document. You are requested to read the textbook material denoted within that document. Lecture and lab materials, in the form of Powerpoint slides and/or videos; lab programming exercises; homeworks; and examination materials will be distributed electronically via the web, email, or via postal mail during the closure period. If the Internet is available, you should send electronic versions of your answers to the homeworks and programming exercises to either my WFU email address or turketwh@gmail.com. Tests should be taken closed book, without access to papers, persons, or other resources, and submitted via postal mail. A return date for the examinations will be specified in the mailing.

Course Outcomes:

By the end of this course, students will:

- Explain the fundamental components found in modern imperative programming languages and indicate the corollary implementing structures in the C++ language
- Design and implement an efficient procedural algorithm for a fairly complex problem expressed in words
- Demonstrate an understanding of the definition of, advantages, and disadvantages of a linked list data-structure
- Design an OOP model of a real-world scenario requiring associations between classes
- Design an OOP model of a real-world scenario requiring inheritance between classes
- Design and implement programs requiring intermediate level mastery of the fundamental concepts of dynamic memory management and the mechanisms implementing these concepts in the C++ language.
- Implement intermediate level programs from a given OOP design, using the mechanisms implementing OOP concepts in the C++ language
- Implement simple C++ programs that employ concepts of inheritance, virtual functions, and polymorphism
- Design and implement a simple recursive algorithm
- Understand the notion of computational complexity, with respect to time and space, at a basic level
- Employ basic unit testing procedures during program development
- Write and employ makefiles for managing compilation processes
- Demonstrate the ability to employ a debugger to aid in program development
- Be able to demonstrate successful navigation through, programming within, and command line management of a Linux-based operating system.

Schedule:

This schedule reflects the general ordering of topics to be covered in class. It should be considered as tentative and open to modification as the class progresses.

Topic	**Related Textbook Material**
Unix introduction & usage	
Review of variables, control structures, simple I/O, C++ syntax for these	Chapters 1 and 2
Review of functions, scope, C++ syntax	Chapter 3
Review of arrays, C++ syntax	Chapter 5
Pointers and references	Chapter 10
Addressing, dynamic memory allocation	Chapter 10
Arrays of pointers	Chapter 10
Parameter passing, command line arguments	Chapter 4

C++ input and output - advanced	Chapter 12
C strings vs char*	Chapter 9
Multi-dimensional arrays	*Test 1*, Section 5.4
Makefiles	Chapter 11
Recursion	Chapter 13
Sorting, searching, and performance evaluation	Chapter 13
Debugging	Section 4.3
Structs	Chapter 6
Classes, constructors, destructors	Chapters 6 and 7
Class members, class method overloading, operator overloading	Chapters 7 and 8
Pointers to classes, classes with pointers	*Test 2*, Section 10.3
Linked lists – conceptual	Chapter 17
Linked lists – implementations	Chapter 17
Iterators	Chapter 17
Inheritance, polymorphism, virtual functions	Chapters 14 and 15
Templates	*Test 3*, Chapter 16
Abstract classes	Chapters 14 and 15
Stack and Queue data structures	Section 17.2