CSCE 1040 Computer Science II

Instructor: David Keathly
Office: NTDP F202

Semester: Fall 2015
Times: Section 001 TTh 11:30 – 12:20 NTDP B140
Section 002 TTh 2:30 pm – 3:20 pm NTDP B155
Section 003 TTh 4:00 – 4:50 NTDP D201

Office Hours: TTh 10:00 am – 11:00 am
W 10:00 am – 3:00 pm

Lab is in NTDP F270, F218 or F222

Phone: 940-565-4801
Email: david.keathly@unt.edu

Course Catalog Description

CSCE 1040, the second course in the introductory sequence, focuses on more advanced C programming, designing and implementing larger software projects, introduction to dynamic data structures, and a beginning exploration of Object Oriented paradigms using C++ . The main focus is on developing students' software development skills.

Course Outcomes

Course outcomes are measurable achievements to be accomplished by the completion of a course. These outcomes are evaluated as part of our ABET accreditation process.

1. Write readable, efficient, and correct C++ programs for all programming constructs defined for Programming Fundamentals I plus dynamic memory allocation, bit manipulation operators, exceptions, classes and inheritance.
2. Design and implement recursive algorithms in C/C++.
3. Use common data structures and techniques such as stacks, queues, linked lists, trees and hashing
4. Create programs using the Standard Template Library.
5. Use a symbolic debugger to find and fix runtime and logical errors in C software.
6. Using a software process model, design and implement a significant software application in C++. Significant software in this context means a software application with at least five files, ten functions and a make file.
7. Implement, compile and run C++ programs that includes classes, inheritance, virtual functions, function overloading and overriding, as well as other aspects of Polymorphism.

Textbook:

Walter Savitch, Problem Solving with C++, 9th Edition, Addison-Wesley
ISBN-10: 0133591743 **ISBN-13:** 978-0133591743

Prerequisites

Completion of CSCE 1030 with a grade of C or better.

Course Requirements:

Attendance: Highly recommended as student is responsible for all materials covered in lecture and class discussion
Exams: Two
Assignments: There will be some larger programming assignments, quizzes, exams and laboratory exercises to complete

For More information

Faculty Webpage: www.cse.unt.edu/~dkeathly
Class Web Page: http://learn.unt.edu

Course Plan

My plan this semester is that beginning in Week 3 we will be "Fipping the Classroom" at least one lecture period each week. This means that many lectures will be recorded videos that you will watch as homework. Class time will be spent working trough example programming problems as a group, in smaller groups and working on your homework assignments with assistance from Graders, Peer Mentors and the instructor. This means you should be able to begin working on homework early and get most of it done during class time. You can also get additional help during office hours and during lab after you complete the lab assignments. If your classes does not meet in a room with computers, you will need to bring a laptop to class to maximize the effectiveness of these class periods for you.

This means it is imperative that you watch the weekly lessons before you come to class so that you can ask questions and apply the lessons. A typical class period, beginning in week 3, will look like this:

Announcements and Questions	10 minutes
Review of assignments	10 minutes
Work on group example or homework	30 minutes
Total	50 minutes

More details will be provided in class.

Course Calendar (subject to change)

Week	Topics	Readings and Materials
Week 1	Class Overview Programming Review C and C++ I/O Structures and Unions (review) Lab 0 (optional)	Chapter 2, 3, 4.1,4.2, 4.3, 10.1
Week 2	File I/O, Command Line Args (review) Pointers (review) Storage Classes Lab 1 (structures)	Chapter 4.5 and lecture notes Chapter 9.1 and lecture notes
Week 3	Recursion, Hash Tables Stacks and Queues Makefiles Lab 2 (pointers) Hwk 1 due	Chapter 13,14
Week 4	Lists and Trees Bit/Byte Manipulation Lab 3 (recursion and hashing)	Chapter 13 and lecture notes
Week 5	Developing Large Programs Debugging Lab 4 (Lists and Bit/Byte Manip) Hwk 2 due	
Week 6	Thinking in a new Paradigm Objects and Classes (OOAD) Lab 5 (Debugging/Develop Lg Pgms) Hwk 3 due	Chapter 1, 5.5, 12
Week 7	What's new in C++ (non-OO) OO Design and UML Lab 6 (Lab Exam) Hwk 4 due	Chapter 10
Week 8	Implementing Classes	

	Exam Review Exam 1	
Week 9	Working in C++ Design patterns and Design Practices Lab 7 (OO Design) Hwk 5 due	Chapter 8,9
Week 10	Inheritance and Polymorphism Public/Private/Protected and Friends Lab 8 (Function Overloading)	Chapter 9
Week 11	Inheritance and Polymorphism continued Lab 9 (C++ I/O) Hwk 6 due	Chapter 13,, Internet Resources and notes
Week 12	Deep Copying and Copy Constructors Lab 10 (Classes and Objects)	Chapter 11
Week 13	Additional OO /C++ Topics Lab 11 (Dynamic Memory and encapsulation) Hw 7 due	Chapter 15, Internet Resources and notes
Week 14	Additional OO /C++ Topics Lab 12 (inheritance)	Chapter 15, Internet Resources and notes
Week 15	Additional OO /C++ Topics Hwk 8 due	Chapter 16, 17, 18, Internet Resources and notes
Week 16	Final Exams (Exam 2)	

Grading Policy

The various components of your grade are weighted as follows:

Lab Programs (12 drop 1)	30%
Quizzes (6-9 drop 1) in class unannouced	10%
Larger Programming Assignments (7-8 drop 1)	40%
Exams (2, 10% each)	20%

Course Policies:

- On programs do your own work. Do NOT work with other students on shared program

solutions. Do NOT get help with algorithms or coding from anyone other than your instructor or the TAs. Do NOT use even partial program solutions from the internet. Failure to adhere to these strict standards will be cause for disciplinary action that could be as severe as expulsion from the university.

- It IS permissible to obtain help from whoever you wish to fix syntax errors. We will be discussing in class the different types of errors that occur in programs so there will be ample opportunity for you to learn the difference between syntax and other errors. But remember, for anything but syntax errors, getting programming assistance from any source other than your instructor or the Class TAs will be considered cheating and dealt with harshly.

- You need to do your own work on quizzes and exams as well. Here there should be no ambiguity at all.

- In case the above description, and in-class discussion of my views on appropriate and inappropriate collaboration does not answer all of your questions, please look at the university Student Rights and Responsibilities web page.

- There will be no make-up exams, quizzes, or programs given in this class. However, for documented *excused absences** or *emergencies** additional time for homework or an alternate lab date may be granted. Exam makeups or substitutions may be granted as well depending on the situation. Note these exceptions are only in the case of documented excused absences or emergencies. In most cases you should contact the instructor before the absence to make proper arrangements.

- You are responsible for the information covered in class, whether you attend class or not. Individualized lectures will not be given. Please check with other class members for any notes that might have been missed during an absence. Except for the start of the term, attendance will not be taken in lecture. However, your attendance is strongly recommended to improve your opportunity to meet course objectives.

- Weekly quizzes will be completed online via the class webpage.

- Students should expect an "in-lab" program each week in lab. The program will be submitted before that lab session is complete. You must make arrangements in advance if you are going to miss your assigned lab section. All labs must be completed within the calendar week they are assigned. All work will be completed in lab unless otherwise instructed by your lab TA.

- There is no curve grading in this class. However the instructor does maintain a “fuzzy borders” policy at the end of the semester for students who complete every lab, homework assignment, quiz and exam. This means that grades that are close to a border (e.g. 87.5 – 89.4) might round up to the next higher grade if students have completed all assignments and have maintained good performance on homework and labs, but perhaps fallen a bit on quizzes or exams. Details of this policy will be discussed during the first class period.

- All non-lab programming assignments are due at 11:59pm on the due date. **Programming assignments will be accepted up to 24 hours late and late programming assignments will be assessed a 50% grade reduction penalty. After 24 (exactly!) hours, late programming assignments will receive a grade of zero.** Partial credit will be given for programs which compile but which are not complete. Starting early on programming projects is strongly encouraged. Students typically have great difficulty completing their projects in one night the day before they are due. Students are allowed to discuss program design and other high level issues with each other. Students are also allowed to help each other understand specific compiler or run time error messages. Copying all or part of another person's program is strictly prohibited and will result in a grade of zero. Supplying printed or electronic copies of your homework to other classmates will also result in a grade of zero. All programs will be submitted through the class website.

- The instructor, peer mentors and TAs require a current copy of the program when a student is asking a question about a program.

- All pertinent information about the class (assignments, exam reviews, sample code, written topic discussions, and day-to-day event schedule) is available via the class webpage. If there is ever a question as to when something is due, or an additional copy of a course document is needed, ALWAYS check the class webpage. If you feel there is incorrect or there is missing information on the class website, email the instructor about the problem immediately. Electronic mail (email) will be a major means of communication with the instructor outside of actual classroom discussions.

- Please keep this information sheet handy during the semester and always periodically check the class homepage for any course information, including scheduling of programming assignments, exams, and exam reviews.

* Excused Absences:Students are expected to schedule routine appointments and activities so as not to conflict with attending class. However, some absences cannot be prevented. In the event of a medical emergency or family death, students must request an excused absence as quickly as feasible following the emergency. Use common sense. Students must provide documentation that verifies an emergency arose.

* Emergencies: By definition, emergencies cannot be planned for. Your instructor attempts to make accommodations in these instances that allow for making up missed work and completion of the course in a timely manner. Among these emergencies are:

- A death in your immediate family.
- An accident or illness requiring immediate medical treatment and where a doctor has indicated attending class is impossible or inadvisable.
- Employees who are on call 24/7 fall in this category but must document that they were called during a scheduled class.

Student Evaluation of Teaching Effectiveness (SETE)

The Student Evaluation of Teaching Effectiveness (SETE) is a requirement for all organized classes at UNT. This short survey will be made available to you at the end of the semester, providing you a chance to comment on how this class is taught. I am very interested in the feedback I get from students, as I work to continually improve my teaching. I consider the SETE to be an important part of your participation in this class

ADA:

UNT complies with all federal and state laws and regulations regarding discrimination including the Americans with Disability Act of 1990 (ADA). If you have a disability and need a reasonable accommodation for equal access to education or services please contact the Office of Disability Accommodation.