CS 2308: Foundations of Computer Science II
Spring 2014

Section 256

Instructor: Dr. Jill Seaman
Nueces 221
js236@txstate.edu

Course Webpage: http://www.cs.txstate.edu/~js236/cs2308

Office Hours:
M: 12:00pm – 1:00pm
T: 2:30pm – 3:30pm
W: 11:00am – 12:00pm
R: 2:30pm – 4:30pm
and by appt.

Meeting Time/Place: Section 256: MW 2:00PM-3:20PM DERR 241

Open Labs: DERR 231: Linux Lab
MCS 590: Windows Lab
MCS 594: Lab tutors

Text: Tony Gaddis, Starting out with C++: From Control Structures through Objects, 7th Edition, ISBN: 0132576252

List of recommended/required readings:

Chapters 1-7 (review of CS 1428)	(recommended)
Chapters 8,9,10,11,13,14,17,18	(required)

Prerequisites: C or higher in CS 1428: Foundations of Computer Science I

Course Description: Fundamentals of object-oriented programming. Introduction to abstract data types (ADTs) including lists, stacks, and queues. Searching and sorting. Pointers and dynamic memory allocation. A continuation of CS 1428.

Course Objectives:

At the end of the course, the students should be able to:

1. Describe and demonstrate at least two different algorithms for searching and at least two different algorithms for sorting.
2. Implement a divide-and-conquer algorithm to solve an appropriate problem (binary search).
3. State the time/space efficiency of various algorithms (using one of 6 categories of mathematical functions).
4. List the 6 categories of mathematical functions used in analyzing algorithms in order

from slowest to fastest growing.
5. Read and write C++ code that uses pointer variables and memory operations (new, &, *, delete), including pointers to arrays, structures, and objects and the -> operator.
6. Write C++ code that resizes an array using dynamic memory allocation.
7. Write C++ code that deletes dynamically allocated memory to avoid memory leaks.
8. Describe the basic concepts of object-oriented programming.
9. Design, implement, test, and debug simple programs (using objects) in an object-oriented programming language (C++).
10. Describe how the class mechanism supports encapsulation and information hiding.
11. Develop (implement) programs using multiple classes and arrays of objects
12. Develop and use appropriate algorithms, especially for processing lists (insert, remove, search, sort, etc.)
13. Describe structured programming in terms of modules and functions.
14. Develop (implement) programs with source code separated into multiple files, including header (.h) files
15. Create, compile, and run a C++ program in a unix style command-line environment
16. Write programs that use each of the following data structures: arrays, structures, strings, and linked lists.
17. Develop (Implement) C++ programs that create and use simple linked-lists, including code to insert into, delete from, and traverse a linked list structure.
18. Compare and contrast the costs and benefits of dynamic and static data structure implementations.
19. Describe the principle of the Abstract Data Type (ADT) and, in particular, explain the benefits of separation of interface and implementation.
20. Implement user-defined data structures in a high-level language.
21. Describe concepts and demonstrate operations of Stacks and Queues.

Grading:

Attendance:	required	
Quizzes:	5%	7 total
Programming Assignments:	25%	7 total
Exam I:	20%	Feb 24 (M)
Exam II:	20%	Apr 9 (W)
Final Exam (comprehensive):	30%	Mon, May 5, 2:00PM to 4:30PM

Attendance: I record attendance every day and I expect you to be in class every day. However, it is not part of the calculation of your final grade.

Quizzes: Quizzes are announced during the previous class and will count for 5 points each.

Makeup Policy: Missed quizzes and programming assignments cannot be made up. Exams may be made up in exceptional circumstances, with documentation and/or approval from the instructor.

Late policy for programming assignments: see the class webpage.

Notifications from the instructor: Notifications related to this class will be sent to your Texas State e-mail account. Be sure to check it regularly.

TRACS: We will use the TRACS website for the following:

- Grades (Gradebook2 tool)
- Programming assignment submissions (Assignments tool)

Everything else will be on the class webpage (including lecture presentations)

Withdrawals/drops: You must follow the withdrawal and drop policy set up by the University and the College of Science. You are responsible for making sure that the drop process is complete.
http://www.registrar.txstate.edu/registration/drop-a-class.html

Last day to drop: March 20, 2014.

Classroom Behavior: The main rule is to not disrupt or distract other students during class. Please do not arrive late or leave early (without notifying the instructor).

Academic Honesty: You are expected to adhere to both the University's Academic Honor Code as described here: http://www.txstate.edu/effective/upps/upps-07-10-01.html, as well as the Computer Science Department Honor Code, described here: 2013 0426 HonestyPolicy CSPPS.doc.

- **All assignments are to be done individually!** You may discuss general strategies for attacking assignment problems with other students in the class but **you must write your own code**.

- Do not include code obtained from the internet in your programming assignment (except what is provided by the instructor).

- **Do not email your program to anyone (except the instructor)!**

The penalty for submitting a program that has been derived from a common ancestor of another students' program or from the internet or any other source will be a 0 for that assignment. Violators will be reported to the Texas State Honor Code Council (http://www.txstate.edu/honorcodecouncil/).

Your submitted programs may be run through an internet service designed for detecting plagiarism in software code.

Accommodations for students with disability:
Any student with a special needs requiring special accommodations should inform me during the first two weeks of classes. The student should also contact the office of disability services at the LBJ student center.