CS 2308: Foundations of Computer Science II
Fall 2016

Section 001
Section 002

Instructor: Dr. Jill Seaman
Comal 307G
js236@txstate.edu

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

Office Hours: M, W: 3:30pm – 4:30pm
T, R: 1:30pm – 3:00pm
and by appt.

Meeting Time/Place: Section 001: MW 2:00PM-3:20PM DERR 234 (top hat: 697794)
Section 002: MW 12:30PM-1:50PM HINES 204 (top hat: 073955)

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, 8th Edition, ISBN: 0133769399

List of recommended/required readings:

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

Required In-Class Response system: We will be using the **Top Hat** classroom response system in class. You will be able to submit answers to in-class questions using smartphones, tablets, laptops, or text messaging on a mobile phone. An email invitation has been sent to your school email account. If you didn't receive this email, you can register by visiting: **tophat.com/e/697794** for section 001 and **tophat.com/e/073955** for section 002. Top Hat will require a paid subscription.

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. Develop (Implement) C++ programs that create and use simple linked-lists, including code to insert into, delete from, and traverse a linked list structure.
17. Compare and contrast the costs and benefits of dynamic and static data structure implementations.
18. Describe the principle of the Abstract Data Type (ADT) and, in particular, explain the benefits of separation of interface and implementation.
19. Implement user-defined data structures in a high-level language.
20. Implement the list, stack, and queue ADT using arrays and linked lists.
21. Write programs that use each of the following data structures: linked lists, stacks, and queues.

Grading:

Participation:	5%	Top Hat
Quizzes	10%	6 total, lowest dropped
Programming Assignments:	15%	6 total, lowest dropped
Exam 1:	20%	Oct 17 (Mon)
Exam 2:	20%	Nov 28 (Mon)
Final Exam (comprehensive):	30%	

section 001: Wed, Dec 14, 2:00pm - 4:30pm
section 002: Wed, Dec 14, 11:00am - 1:30pm

Participation: Bring a web-enabled device or a phone with texting capabilities to every class to access the Top Hat system. You will be asked questions in each class. We will also use Top Hat to take attendance. Your participation grade is computed as follows: (2 x Course Average + Attendance) / 3.

Quiz dates: 9/14 (W), 9/28 (W), 10/12 (W), 11/02 (W), 11/21 (M), 12/07 (W).

Makeup Policy: Missed quizzes and programming assignments cannot be made up. Exams may be made up in exceptional circumstances, with 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)
- Resources (code you can use in your programing assignments)

Everything else will be on the class webpage and/or the Top Hat webpage.

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: October 30, 2016.

Classroom Behavior: The main rule is to not disrupt or distract other students during class.

Academic Honesty: You are expected to adhere to the University's Academic Honor Code as described here: http://www.txstate.edu/effective/upps/upps-07-10-01.html.

- You may work together on your programming assignments. If you submit a program that is the result of working with others, you must list the names of all contributors in the file header. Each student must submit their own program, even if it is the same as another students'.

  Note: In order to do well on the quizzes and exams, you must be able to write code on your own, so you must practice this.

- Do not include code obtained from the internet or any other source in your programming assignment (except what is provided by the instructor during the current semester). The penalty for submitting a program that includes code from the internet or any other source outside of the class will be a 0 for that assignment.

Accommodations for students with disability:
Any student with needs requiring special accommodations should contact the office of disability services at the LBJ student center. Students who qualify for extra time for exams must take their test with ATSD and must schedule their test at the same time the test is given in class.