CS 2308: Foundations of Computer Science II
Spring 2019

Section 255

Instructor: Dr. Jill Seaman
Comal 210D
js236@txstate.edu

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

Office Hours: M,W: 2:00pm – 3:00pm
T, R: 1:30pm – 3:00pm
and by appt.

Meeting Time/Place: TR 11:00AM-12:20PM Ingram Hall 03104

Open Labs: DERR 231 and MCS 590
Lab tutors are in DERR 231

Text: Tony Gaddis, Starting out with C++: From Control Structures through Objects, 9th Edition, ISBN: 0134544846 (8th edition is allowed)

List of required recommended/readings:

Chapters 1-7, 11.1-11.8 (review of CS 1428)	(recommended)
Chapters 8,9,11,13,18,19	(required)

See course website for a daily schedule.

Required In-Class Response system: We will be using the **Squarecap** classroom response system in class. You will be able to submit answers to in-class questions using a smartphone, tablet, or laptop. To sign up, visit **www.squarecap.com** on your web browser (Google Chrome is the preferred browser). Squarecap requires a paid subscription.

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.

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

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:	15%	Squarecap & Attendance
	Coding Quizzes	10%	lowest of 7 is dropped
	Programming Assignments:	20%	lowest of 6 is dropped
	Tests:	30%	3 total (10% each)
	Final Exam (comprehensive):	25%	May 9 (Thurs) 11:00am - 1:30pm

Test dates: 2/21 (Thurs), 3/28 (Thurs), 4/25 (Thurs).

Participation: Bring a web-enabled device to every class to access the Squarecap system. You will be asked questions each day in class (2 points each). We will also track attendance (2 points per day). Your participation grade is computed as follows:
avg = (your squarecap points + your attendance points) / total possible points
participation = the minimum of: **avg***100 / 80 and 100%.
This means you can miss up to 20% of the points and still get 100%.

Coding Quizzes: Beginning on Thursday, Jan 31, there will be a quiz nearly every Thursday. The quiz will require writing C++ code, or correcting errors in C++ code, or filling in blanks in provided code to complete a C++ function/class/program.

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

Reading assignments: There will be assigned reading from the book for each lecture class. A reading schedule will be posted on the class website. You should come to class each day prepared to answer Squarecap questions over the assigned reading for that day.

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 (Gradebook tool)
- Programming assignment submissions (Assignments tool)
- Resources (code you can use in your programing assignments)
- Attendance (use in class to report your attendance)

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

HELP: Other than the instructor's office hours, you may obtain assistance here:

DERR 231: Computer Science Department Lab Tutors are available to help with your programming assignments.
CLC (Collaborative Learning Center): Free walk–in tutoring provided by students in the H–LSAMP Scholars Program. Ingram Hall #3202.
SLAC (Student Learning Assistance Center): walk-in tutoring lab and **Supplemental Instruction**: a nontraditional form of tutoring that focuses on collaboration, group study, and interaction for assisting students undertaking "traditionally difficult" courses. A trained peer who has successfully negotiated the course (the SI Leader) will facilitate 3 one-hour study sessions per week for group study.

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

Last day to drop with automatic "W": April 2, 2019.

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. Note: you must submit your request with ATSD at least 2 business days before the exam date!

Classroom Behavior: The main rule is to not disrupt or distract other students during class! Additionally please treat other students and the instructors with respect.

Academic Honesty: You are expected to adhere to the University's Academic Honor Code as described here:

https://www.txstate.edu/honorcodecouncil/Academic-Integrity.html

- You may work together on your programming assignments. If you submit a program that is the result of group work, 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'. The penalty for not citing collaborators will be -30 points for that assignment.

- 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). Do not post your solution anywhere on the internet. The penalty for either of these violations will be a 0 for that assignment. **Submitting work done by others as if it were your own is an act of dishonesty.**

Course Content: There are 6 main topics or units:

Unit 1: Functions, Arrays, & Structs
Unit 2: Searching, Sorting, & Analysis
Unit 3: Pointers & Dynamic Memory Allocation
Unit 4: Intro to Classes
Unit 5: Linked Lists
Unit 6: Stacks & Queues

For each unit I will provide the following (posted on the class website):

- Reading assignments from the book.
- Lecture slides (for your reference)
- A Programming Assignment

Use of Squarecap in this class:

- 2 points per question (1 for correctness, 1 for answering)
- Goal: 3 questions per day (none on test days)

Exam coverage:

Test 1 covers Units 1-2.
Test 2 covers Units 3-4.
Test 3 covers Units 5-6.
Final Exam covers Units 1-6

Each Test will have about 12 multiple choice questions and 2-3 programming questions.