CS 3358: Data Structures
Spring 2015
Section 253

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

Course Webpage: http://www.cs.txstate.edu/~js236/cs3358

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

Meeting Time/Place: MW 2:00PM–3:20PM DERR 235

Prerequisites:
C or higher in CS 2308: Foundations of Computer Science II
C or higher in MATH 2358: Discrete Mathematics I

Course Description: A course covering classic data structures and analysis of elementary algorithms, with an introduction to recursion.

Course Objectives:

Before of the course, the students should be able to:

i. Develop (implement) C++ programs including multiple classes and arrays of objects.
ii. Read and write C++ code that uses pointer variables and memory operations (new, &, *, delete), including pointers to arrays, structures, and objects and the -> operator.
iii. Create, compile, and run a C++ program (in a unix style command-line environment).
iv. Develop (implement) C++ programs with the source code separated into multiple files, using header (.h) files.

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

1. State the definition of an Abstract Data Type.
2. Explain the value of ADTs in computer science.
3. Describe the semantics (values and operations) of the following ADTs :
   a. Lists
   b. Stacks and Queues
   c. Trees (binary trees, self-balancing trees, heaps)
   d. Hash Tables
   e. Graphs
4. Perform (demonstrate) each of the ADT operations given an instance of the ADT, including each of the 3 tree traversals on binary search trees.
5. Evaluate which ADT should be used to solve a given problem (and defend the choice).

6. Summarize the advantages and disadvantages of static vs dynamic implementations of ADTs.
7. Implement the ADTs in C++
   a. using both dynamic and static underlying types, (including singly and doubly linked lists).
   b. using header files and implementation files.
   c. using C++ templates (generic programming).
8. Write modular programs in C++ that use the ADTs, including using ADTs from the C++ Standard Template Library
9. Describe elementary algorithms for sorting, searching, and hashing, including linear/ binary search, selection/insertion/bubble/merge and quicksort
10. Perform (demonstrate) each of the elementary algorithms listed above.
11. List (in order of increasing growth rate) the 6 categories of mathematical functions used in analyzing algorithms.
12. Analyze algorithms, including sorting, searching, and implementations of ADT operations, for efficiency
    a. Compute the formula to describe the number of steps a given algorithm requires in terms of the amount of data
    b. Classify the formula according to the 6 classifications of mathematical functions
13. State a definition of recursion
14. List and describe common applications of recursion.
15. Read and write recursive functions in C++.
16. Write programs that are well designed and organized.

Text: Data Structures and Problem Solving Using C++, Weiss, 2nd Ed.
ISBN 0-201-61250-X

List of required readings: (Reading schedule will be announced in class)
Selected sections from chapters 1-3, 6-9, 16-21

Grading:			
	Quizzes:	5%	5-7 total
	Programming Assignments:	25%	6-7 total
	Exam 1:	20%	Mar 2 (Mon)
	Exam 2:	20%	Apr 20 (Mon)
	Final Exam (comprehensive):	30%	May 11 (Mon) 2:00pm-4:30pm

Quizzes: 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 approval from the instructor.

Attendance: I record attendance every day and expect you to be in class every 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 (Gradebook2 tool)
- Programming assignment submissions and feedback (Assignments tool)
- Resources (code you can use in your programing assignments)

Everything else will be on the class webpage (including lecture presentations and demo code).

Campus Labs: You may use DERR 231 (the Linux Lab) or MCS 590 to work on your programming assignments. You may also use your own computer. You may use any IDE that supports C++ programming.

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

Last day to drop: March 26, 2015.

Classroom Behavior: Do not disrupt other students during class. Be respectful.

Academic Honesty: You are expected to adhere to both the University's Academic Honor Code as described here, as well as the Computer Science Department Honor Code, described here: 2013 0426 HonestyPolicy CSPPS.doc.

Unless otherwise stated, all assignments are to be done individually. You may discuss general strategies for solving 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 or otherwise provide an electronic copy of your program to another student.

Accommodations for students with disability:

Any student with a special needs requiring special accommodations should inform me during the first week of classes. The student should also contact the Office of Disability Services at the LBJ Student Center.