CSC221 Data Structures & Algorithms I Fall 2015

Professor: Torgersen **E-mail**: *torgerse@wfu.edu* **Office Phone**: 758-5536

Office Hours: Mon. and Wed. 3:00 to 4:30, Thu. at 2:00 to 3:30 and by appointment.

Text: Data Structures & Algorithms in C++, Second Edition, by Michael T. Goodrich, Roberto Tamassia, and David Mount. ISBN: 9780470383278, Publisher: Wiley, 2011.

Facilities: gottlieb.cs.wfu.edu. Xubuntu (Linux), XFCE Desktop, X2Go remote desktop software, Undergraduate Computing Lab, WFU issued laptops, your personal computers (e.g., Macintosh users), "The Bridge" in Z. Smith Reynolds Library, WFU loaner laptops, menehune web server, and the WFU C.S. Center.

Web page: http://menehune.opt.wfu.edu/csc221

TA: Scott Seal, Office hours: TBA.

Innovations: Programing partnerships and debugging teams.

Goals:

1. Introduction to UNIX (on-line tutorial + quiz)
2. Review of Dynamically Allocated Multi-dimensional Arrays
3. Asymptotic complexity measures: big **O**, little **o**, big Θ, big Ω.
4. Review of Linear Data Structures
   (a) Linked lists, variants, doubly linked lists
   (b) Stacks: array-based implementation vs linked implementation
   (c) Queues: circular array implementation vs linked implementation
   (d) Operations on linear data structures: search, insert, delete, list traversal
   (e) Application: buddy system memory management
5. Trees
   (a) Binary search trees
   (b) depth-first and breadth-first search, insert, delete, and traversals
   (c) Height balanced (AVL) trees
   (d) Heaps
   (e) B-Trees
   (f) Application: Huffman codes
6. Hash tables
   (a) Modular arithmetic and hash functions
   (b) Chaining
   (c) Open addressing: clustering, linear probing, pseudo-random probing, quadratic probing, and double hashing
   (d) Operations on hash tables: search, insert, delete
7. Sets:
   (a) bit vectors, efficient implementation (bit packing)
   (b) membership union, intersection, and set-difference

8. Graphs
    (a) Adjacency matrix representation
    (b) Adjacency list representation
    (c) Some basic graph algorithms: depth-first search, minimal spanning tree, shortest paths
9. Introduction to the Analysis of Algorithms
10. Binary search
11. Efficient sorting methods
    (a) Mergesort
    (b) Quicksort
    (c) Heapsort
12. Finding k^{th} largest element of an ordered list, and its relation to quicksort.
13. Proficiency in using Unix, the Unix development environment(s), integrated development environments. (E.g., emacs, g++, make, gdb, ddd, geany)
14. Advanced C++ Topics (if time allows)
    (a) Pointer coercion
    (b) Standard template library
    (c) Template functions and template classes
    (d) Inheritance
    (e) Virtual functions, virtual base classes
15. GUI programming using call-backs (if time allows).

Expectations:

1. Class participation; communicate if things get complicated.
2. Use of good coding practices and some basic coding standards in programming projects.
3. Your best effort.

Grading:
Three exams (65%), programming assignments and take home problem sets (35%). Programming assignment(s) **must** be submitted ready to compile and run under Linux. There are grading penalties for late programming assignments.

Disability Notice:
If you have a disability that may require an accommodation for taking this course, then please contact the Learning Assistance Center (758-5929) within the first two weeks of the semester.

Pandemic Planning Notice:
The University has requested that faculty collect personal contact information as part of emergency planning and preparation. The information you provide is strictly confidential.