COP 4530 Data Structures, Algorithms, and Generic Programming

Spring 2010

Schedule:

MCH 0201
10:10AM - 11:00AM, MWF

Course Description:

Making efficient use of computational resources is one of the important tasks of any computer scientists. In this course we will explore different ways of organizing data to facilitate such sufficient use. This course covers the following topics:

- **Data structures:** Abstract data types (ADTs), vector, deque, list, queue, stack, graph, digraph, table, map (associative array), priority queue, set, and tree, etc.
- **Algorithms:** Algorithms are formalizations of processes that result in predictable and desirable outcomes. They are used in a variety of contexts. Particularly, data structures are made usable by implementing algorithms for searching, sorting, and indexing the structures. Algorithm design, complexity analysis and correctness proof form important components in study of algorithms.
- **Generic programming:** Generic programming is the science of component re-use. We will explore coding for re-use of both data structures and algorithms in C++. *Coding for re-use* and *re-use of code* are important aspects of software engineering.

We will also have several substantial programming projects that involve the implementation and use of data structures, algorithms, and generic programming.

Course Objective:

The objective of the course is to teach students how to design, write, and analyze the performance of C/C++ programs that handle structured data and perform more complex tasks, typical of larger software projects. Students should acquire skills in using generic principles for data representation and manipulation with a view for efficiency, maintainability, and code-reuse. Successful students will, at the end of the course, be able to demonstrate analytical comprehension of concepts such as abstract data types (vectors, lists, deques, trees, etc.), generic programming techniques (containers, adaptors, accessing data through interface, iterators, etc.), algorithms (sorting, using stacks and queues, tree exploration algorithms, etc.), and efficiency analysis (which data structures allow efficient interfaces to particular forms of data access, such as random vs. sequential data access or insertion). The students should be able to demonstrate similar skills in related implementation tasks in the C/C++ language, including extensive use of templates to allow for modularity and re-usability of code.

Prerequisites:

- **COP 3330: Object-Oriented Programming**,
- **MAD 2104: Discrete Mathematics**.
- CDA 3100: Computer Organization I (co-requisite)
- This course requires that the student be proficient with C++ and object oriented programming concepts.
- Student also need to have a user-level knowledge of Unix, and be comfortable working in a Unix environment.
- The pre-requisites will not be waived.
- If you have doubts whether you satisfy the course pre-requisites, please contact the instructor.

Textbooks

- Required textbook
  - "Data Structure and Algorithm Analysis in C++", by Mark Allen Weiss. Publisher: Addison Wesley, 3rd Edition, 2006.

- Recommended optional textbooks
  - "Data Structures with C++ using STL", by William Ford and William Topp. Publisher: Prentice Hall.
  - "Absolute C++", by Walter Savitch, 3rd Edition. Publisher: Addison Wesley
  - "The C++ Standard Library: A Tutorial and Reference", by Nicolai M. Josuttis. Addison-Wesley.
  - "C++ How to Program", by H. M. Deitel, and P. J. Deitel. Publisher: Prentice Hall.
  - "Introduction to Algorithms", by Cormen, Leiserson, and Rivest, Publisher: MIT press and McGraw-Hill Book Company

Course Contents

The course outline will closely follow the material presented in book by Mark Allen Weiss. We will cover chapters 1--9 in detail and chapters 10--12 in any remaining extra time.

Chapter 1 - Introduction

11.1 An Unrelated Puzzle
11.2 Binomial Queues
11.3 Skew Heaps
11.4 Fibonacci Heaps
11.5 Splay Trees

Chapter 12 - Advanced Data Structures and Implementation
12.1 Top-Down Splay Trees
12.2 Red-Black Trees
12.3 Deterministic Skip Lists
12.4 AA-Trees
12.5 Treaps
12.6 k-d Trees
12.7 Pairing Heaps

Workloads and Grading:

There will be one final exam, two midterm exam, five (programming) home assignments, and several in-class quizzes.

1. Five home Assignments (45%) - 9% each
2. Three exams (50%)
    - Midterm Exam 1 - 12%
    - Midterm Exam 2 - 13%
    - Final Exam - 25%
3. Quizzes - 5 %

Final letter grades

In order to obtain a course grade of C- or better, your course performance must satisfy the following two requirements.

- You must earn at least 60% in each (1) programming assignments and (2) exams.
- You must obtain a grade of C- or better for certain components of two designated projects (which will be specified when we announce the projects). COP4530 is selected as one of the capstone courses by the Department of Computer Science for assessment of the following expected outcomes for its degree programs, as required by our accreditation agencies, the University, and the State of Florida: (1) Recursive Algorithm Use and (2) Data Structure Knowledge. Departmental policy does not permit a final grade of C- or better to be assigned unless the student has at least earned a grade of C- or better on each of these components, regardless of performance on other work in the course.

A	[90-100)
A-	[88-90)
B+	[85-88)
B	[80-85)
B-	[78-80)
C+	[75-78)
C	[70-75)
C-	[68-70)
D	[60-68)
F	<60

Course Policies:

Attendance Policy:

The university requires attendance in all classes, and it is also important to your learning. The attendance record may be provided to deans who request it. If your grade is just a little below the cutoff for a higher grade, your attendance will be one of the factors that we consider, in deciding whether to "bump" you up to the higher grade. Missing three or fewer lectures will be considered good attendance. In rare cases, such as medical needs or jury duty, absences may be excused with appropriate documentation. You should let me know in advance, when possible, and submit the documentation I seek. You should make up for any materials missed due to absences.

Missed exam Policy:

A missed exam will be recorded as a grade of zero. We will follow the university rules regarding missed final exams (see http://registrar.fsu.edu/dir%5Fclass/spring/exam_schedule.htm), for all the exams, including the final exam.

Late Assignment Policy:

In order to enable us to provide timely solutions to assignments, we have the following policy regarding submission of late assignments.

- An assignment that is turned in no more than 24 hours late will be scored with a 10% penalty.
- An assignment that is turned in more than 24 and no less than 48 hours late will be scored with a 20% penalty.

- An assignment that is turned in more than 48 hours late will receive the score of zero, though we will review it and comment on it.

Incomplete Grade (Grade of 'I') Policy:

The grade of 'I' will be assigned only under the following exceptional circumstances:

- The final exam is missed with an accepted excuse for the absence. In this case, the final exam must be made up during the first two weeks of the following semester.

ACADEMIC HONOR POLICY:
The Florida State University Academic Honor Policy outlines the University's expectations for the integrity of students' academic work, the procedures for resolving alleged violations of those expectations, and the rights and responsibilities of students and faculty members throughout the process. Students are responsible for reading the Academic Honor Policy and for living up to their pledge to . . . be honest and truthful and . . . [to] strive for personal and institutional integrity at Florida State University. (Florida State University Academic Honor Policy, found at http://dof.fsu.edu/honorpolicy.htm.)

AMERICANS WITH DISABILITIES ACT (ADA):

Students with disabilities needing academic accommodation should:
(1) register with and provide documentation to the Student Disability Resource Center; and
(2) bring a letter to the instructor indicating the need for accommodation and what type. This should be done during the first week of class.

This syllabus and other class materials are available in alternative format upon request.

For more information about services available to FSU students with disabilities, contact the:

Student Disability Resource Center
874 Traditions Way
108 Student Services Building
Florida State University
Tallahassee, FL 32306-4167
(850) 644-9566 (voice)
(850) 644-8504 (TDD)
(850) 644-7164
sdrc@admin.fsu.edu
http://www.disabilitycenter.fsu.edu/

Academic Integrity:

Remember that the goal of programming assignments and homework is to enhance your analysis, reasoning, and programming skills. Indulging in academic dishonesty defeats this purpose apart from being unfair to other students. In case you have any questions about whether an act of collaboration may be construed as academic dishonesty, **please clarify the issue with the instructor before you collaborate.**

All students should follow FSU Academic Honor Code. You might be assigned a grade of '**F**', if you are found to have indulged in academic dishonesty.

- It is understandable that discussing a problem with other people may lead to more insight into the issues involved. **Thus discussing a problem in assignments/homeworks with other people is fine. However, discussing the solutions to the problem is NOT acceptable.**
- Every student must write his/her own code and homework. **Showing your code or homework to members of other teams, giving it to them, or making it accessible to them (e.g., by making the files world-readable) is academic dishonesty.**
- You are responsible for ensuring that your code/documentation/results are adequately protected and not accessible to other teams. Change permissions of your working directory to 0700 ('chmod 0700 <directory>).
- Consulting code/material from a textbook, or from the Internet, in order to understand specific aspects of your assignment is fine. However, **copying such code/material will be considered academic dishonesty**. If you borrow small parts of code/material from these sources, you must acknowledge this in your submission and additionally you must clearly understand and be able to explain the borrowed code/material.
- Plagiarism detection tools, such as Moss (A system for detecting software plagiarism), will be used in this course.

Syllabus Changes

This syllabus is a guide for the course and is subject to change with advance notice.