CONTACT INFORMATION:

Instructor: Breno de Medeiros,
           Assistant Professor
           269 Love Bldg
           Office hours: Mon./Tue. 1:00-3:00pm
           Phone: (850) 645-2356 / Fax: (850) 644-0058
           breno@cs.fsu.edu
           http://www.cs.fsu.edu/~breno

Teaching Assistant: Ling Toh
                   Doctorate Student
                   Office hours location: Majors Lab
                   Office hours: Thu. 2:00-4:00pm
                                  Fri. 9:30-11:30am
                   toh@cs.fsu.edu

COURSE MATERIALS:

Textbook:
Data Structures and Algorithm Analysis in C++ (3rd edition), by M. A. Weiss.

Suggested texts:
C++ How to Program (5th Edition), by (Harvey & Paul) Deitel & Associates.

Other Materials:
Posted class slides & programming assignments.

Class URL:
http://www.cs.fsu.edu/~breno/COP-4530/

COURSE DESCRIPTION:

Topics

• Data structures:
Abstract data types (ADTs), vector, list, deque, stack, queue, graph, digraph, table, map (associative array), priority queue, sets, trees. Etc.

- **Algorithms:**

  Efficient program design requires good matching of data structures (which determine how the data can be easily accessed and manipulated) and algorithms (strategies for processing the data to achieve the desired program goals). Algorithm design, complexity analysis and correctness proof form important components in study of algorithms.

- **Generic programming:**

  Generic programming is a software engineering philosophy to create programs that use data structures from their high-level functionality for data access and manipulation, without looking at the details of how that data structure is implemented as a program. Generic programming permits component re-use and leads to more maintainable code. This course teaches coding for re-use of both data structures and algorithms in C++. Provide a description and rationale for the course indicating where it fits into the overall intellectual area.

**COURSE OBJECTIVES:**

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 & 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.

**COURSE POLICIES:**

*Class attendance is mandatory.* While roll-call will not be used, the instructor will present solutions in class for the types of problems that will be argued in exams: Each class will include about 15 minutes of exam preparation, and there will be no specific class devoted to midterm review.
**Attendance of recitation sessions is also required.** Pop-quizzes given in recitation sessions will account for 10% of the final grade.

**Lateness policy on assignments:** No deadline extensions are allowed without a medical reason and proper written documentation provided by the Thagard Student Health Center, as per University policies. Assignments are due at 11:59pm on Fridays. Assignments late for 1 minute up to less than 24 hours will receive a 10% penalty. Assignments late for less than 48 hours (but for more than 24 hours) will be penalized 20%. Assignments late for more than 48 hours will be assigned a grade of 0, but still reviewed for correctness, with feedback provided.

**Make-up policy on exams:** No make-up exams will be given without a medical reason and proper written documentation provided by the Thagard Student Health Center, as per University policy (see Student Handbook). Accommodations for missed exams for other reasons are possible within circumstances foreseen by University policies. Examples of situations where such accommodations can be made are for conflicts with religious observance practices and/or in cases involving bereavement due to the loss of a close relative. Students dealing with such a personal loss are urged to seek support from the University counseling center, and to contact all their instructors to prepare a comprehensive coping strategy (Individual Counseling Student Counseling Center, 2nd Floor, Student Life Building, 644-2003.).

**Policy on violations of the Academic Honor code:** All programming assignments and exams are individual assignments. No cheating or plagiarism will be tolerated. A tool that detects plagiarism will be employed which is very effective at detecting improper collaboration/sharing of code. Assignments where evidence of cheating is conclusive will be given a grade of 0, irrespective of correctness. In addition, students who actively seek to hurt the performance of other students by using the department/academic computing resources in any manner that is incompatible with the stated policies will be assigned a final grade of F for the course.

**Note of compliance with American with Disabilities Act:** Students with disabilities who may require special accommodation with classes or exams should register with the Student Disability Resource Center (SDRC), and bring a letter from the SDRC to the instructor indicating their needs. This should be done within the first week of class. The instructor is personally committed to provide opportunities for students with disabilities to achieve their potential, and such students are encouraged to provide suggestions on how their learning may be facilitated.

**GRADING/EVALUATION:**

- **A:** 91-100
- **A-:** 89, 90
- **B:** 79 – 88 (B+ and B- are assigned within this range based on both grades and student participation/attendance)
- **C:** 70 – 78 (C+ and C- are assigned within this range based on both grades
and student participation/attendance)  
D: 59 – 69  
F: 0 – 58

**Additional requirement for C or better grade:** Students must score a minimum of **300 points** on both the programming assignments (1—5) and exams (midterms and final) to get a grade of C or better, in addition to 700 total points in all assignments.

**ASSIGNMENTS/RESPONSIBILITIES:**

Previous knowledge:

- Required pre-requisites: COP 3330 (Object-oriented programming) and MAD 2104 (Discrete mathematics).
- Familiarity with objected-oriented programming concepts and proficiency in C/C++ programming.

Programming Assignments – 45% of total grade

1 and 2 - 15% (7.5% each = 75 points)  
3, 4, and 5 - 30% (10% each = 100 points)

Exams – 45% of total grade

Midterm Exam 1 - 10% (100 points)  
September 27, Wednesday. 3:35 to 4:50pm, in regular classroom.

Midterm Exam 2 - 10% (100 points)  
October 30, Monday. 3:35 to 4:50pm, in regular classroom.

Final Exam - 25% (250 points)  
December 11, Monday. 5:30pm to 7:30pm, in regular classroom.

Recitation Quizzes – 10% (100 points), at random dates during recitation (Thursday) lessons.

**COURSE CONTENT AND OUTLINE:**

The course outline will closely follow the material presented in the textbook. Chapters 1-7 and 9 will be covered in detail. Some material from chapters 10-12 may be covered in any remaining extra time.

Chapter 1 - Introduction

1.1 What’s the Book About?  
1.2 Mathematics Review  
1.3 A Brief Introduction to Recursion  
1.4 C++ Classes
1.5 C++ Details
1.6 Templates
1.7 Using Matrices

Chapter 2 - Algorithm Analysis
2.1 Mathematical Background
2.2 Model
2.3 What to Analyze
2.4 Running Time Calculations

Chapter 3 - Lists, Stacks, and Queues
3.1 Abstract Data Types (ADTs)
3.2 The List ADT
3.3 vector and list in the STL
3.4 Implementation of vector
3.5 Implementation of list
3.6 The Stack ADT
3.7 The Queue ADT

Chapter 4 - Trees
4.1 Preliminaries
4.2 Binary Trees
4.3 The Search Tree ADT—Binary Search Trees
4.4 AVL Trees
4.5 Splay Trees
4.6 Tree Traversals (Revisited)
4.7 B-Trees
4.8 Sets and Maps in the Standard Library

Chapter 5 - Hashing
5.1 General Idea
5.2 Hash Function
5.3 Separate Chaining
5.4 Hash Tables Without Linked Lists
5.5 Rehashing
5.6 Hash Tables in the Standard Library
5.7 Extendible Hashing

Chapter 6 - Priority Queues (Heaps)
6.1 Model
6.2 Simple Implementations
6.3 Binary Heap
6.4 Applications of Priority Queues
6.5 d-Heaps
6.6 Leftist Heaps
6.7 Skew Heaps
6.8 Binomial Queues
6.9 Priority Queues in the Standard Library

Chapter 7 - Sorting
7.1 Preliminaries
7.2 Insertion Sort
7.3 A Lower Bound for Simple Sorting Algorithms
7.4 Shellsort
7.5 Heapsort
7.6 Mergesort
7.7 Quicksort
7.8 Indirect Sorting
7.9 A General Lower Bound for Sorting
7.10 Bucket Sort
7.11 External Sorting

Chapter 9 - Graph Algorithms
9.1 Definitions
9.2 Topological Sort
9.3 Shortest-Path Algorithms
9.4 Network Flow Problems
9.5 Minimum Spanning Tree
9.6 Applications of Depth-First Search
9.7 Introduction to NP-Completeness

Chapter 10 - Algorithm Design Techniques
10.1 Greedy Algorithms
10.2 Divide and Conquer
10.3 Dynamic Programming
10.4 Randomized Algorithms
10.5 Backtracking Algorithms

Chapter 11 - Amortized Analysis
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

**FLORIDA STATE 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://www.fsu.edu/~dof/honorpolicy.htm](http://www.fsu.edu/~dof/honorpolicy.htm).)

**AMERICANS WITH DISABILITIES ACT:**

Students with disabilities needing academic accommodation should: (1) register with and provide documentation to the Student Disability Resource Center; (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
97 Woodward Avenue, South
Florida State University
Tallahassee, FL 32306-4167
(850) 644-9566 (voice)
(850) 644-8504 (TDD)
sdrc@admin.fsu.edu
[http://www.fsu.edu/~staffair/dean/StudentDisability/](http://www.fsu.edu/~staffair/dean/StudentDisability/)

**SYLLABUS CHANGE POLICY:**

Except for changes that substantially affect implementation of the evaluation (grading) statement, this syllabus is a guide for the course and is subject to change with advanced notice. The course website contains up-to-date information about the course, including posted class slides and assignments.