CSC 330 Class Information and Syllabus

Instructor: Stephen R. Tate (Steve)
Lectures: Tues/Thurs 12:30–1:45, Petty 217
Office: Petty 166
Office Hours: Tues/Thurs 10:00–11:30, or by appointment
Phone: 336-256-1033
E-mail: `srtate@uncg.edu`

Class Web Site: `http://www.uncg.edu/cmp/faculty/srtate/330/`

Textbook (required):

> Mark Allen Weiss. *Data Structures and Problem Solving Using Java*, 4th edition, Addison Wesley, ISBN 0–321–54140–5.

Catalog Description: Static and dynamic data structures emphasizing binary trees and graphs. Advanced programming techniques. Advanced sorting and searching algorithms. Hashing techniques. Performance analysis. Methods of developing large applications programs.

Prerequisites: Students should have completed CSC 230 and CSC 250 with a C or better in each. See the end of this handout for detailed information on assumed background.

Student Learning Outcomes: Upon successful completion of this course students will have demonstrated that they

1. understand and use advanced features of Java such as inheritance and generics;
2. can design divide-and-conquer algorithms using three steps and apply to merge sort, quick sort, dynamic programming, and backtracking;
3. understand tree representation and traversals;
4. understand associative containers, red-black trees, and hashing;
5. understand graph representations, traversals, and basic graph algorithms.

Class Structure: Class will meet twice a week for 75 minutes. Class time will include brief overviews of the material, but students are expected to have completed the textbook readings before class so that the majority of class time can be used for discussing the material, answering questions and clarifying topics from the book, and working through examples. If students do not keep up with readings, the instructor may give in-class quizzes or small reading-summary assignments as extra motivation. On some days (which will be announced in advance) we may meet in a computer lab.

Assignments: Assignments will be both written and programming. Programs must be written in Java, and must compile and work correctly in Netbeans version 8.1 with Java 8 (this is the version on the UNCG Computer Lab machines, and is freely available to download and install on your personal computers). The instructor will *not* make any adjustments to a student's code when grading, so if any submitted program does not compile the student will get a zero on the "correctness" portion of the grade (or 50 points off the overall grade), with no exceptions. More information on grading criteria and program submission instructions will be distributed and discussed before the first assignment. Written assignment solutions must be turned in on paper at the beginning of class (solutions can be handwritten or typed).

Late Policy and Makeup Exams: Assignments are due at the beginning of class on the due date, and may be turned in up to 7 calendar days late with a 25% late penalty. Students with planned absences, whether for university events, religious observance, or other reason, are expected to make arrangements with the instructor to turn in assignments or take exams before the scheduled date of the assignment or test. *No assignment will be accepted more than 7 calendar days after the original due date!*

Exam/test dates are on the schedule on the following page — if there are any changes, they will be announced at least two weeks in advance if possible. A missed exam may be made up *only* if it was missed due to an extreme emergency and arrangements are made *before* the exam date. Exams (including the final) may not be taken early or late due to personal travel plans.

Evaluation and Grading: Each student activity will contribute to the final grade in the class according to the following percentages.

Assignments	50 points
Mid-semester exams (15 points each)	30 points
Final exam	20 points

Tentative Schedule:

Date	Topic	Reading
Aug 23	Class Introduction and Simple Review	
Aug 25	CSC 230 Review with Programming Style Guidelines	6.1 – 6.6
Aug 30	More review examples, and iterators	
Sep 1	Sets and Maps	6.7 – 6.8
Sep 6	Trees: Definitions and Traversals	Chapter 18
Sep 8	Trees: Working with trees and traversals	
Sep 13	Binary Search Trees: Basics	19.1 – 19.3
Sep 15	Binary Search Trees: Balancing – concepts and AVL trees	19.4
Sep 20	Binary Search Trees: Red-Black Trees	19.5
Sep 22	Hashing – Part 1	20.1 – 20.3
Sep 27	Hashing – Part 2	20.4 – 20.6
Sep 29	Make-up/slack/review day	
Oct 4	Exam 1	
Oct 6	Priority Queues and Heaps	21.1 – 21.3
Oct 11	Heapsort and other applications	21.5
Oct 13	Algorithm design: Divide and Conquer	7.5, 8.5, 8.6
Oct 18	*Fall break – no class*	
Oct 20	Algorithm design: Dynamic Programming	7.6
Oct 25	Algorithm design: Backtracking	7.7
Oct 27	Algorithm design: Greedy	12.1
Nov 1	Make-up/slack/review day	
Nov 3	Exam 2	
Nov 8	Graphs: Terminology, Representations, and Traversals	14.1–14.2
Nov 10	Graphs: Terminology, Representations, and Traversals – cont'd	
Nov 15	Weighted Graphs and Paths	14.3–14.4
Nov 17	Minimum Spanning Trees	Handouts
Nov 22	Acyclic Graphs and Applications	14.5
Nov 24	*Thanksgiving – no class*	
Nov 29	Advanced topic (tentative: disk-based data structures)	
Dec 1	Class Review	
Dec 8	Final Exam (12:00–3:00PM)	

Academic Integrity: Students are required to sign the Academic Integrity Pledge on any work they do. The pledge is the statement "I have abided by the UNCG Academic Integrity Policy on this assignment." For information on the UNCG Academic Integrity Policy, see `http://academicintegrity.uncg.edu/`.

Assignments in this class are for individual work, unless explicitly stated otherwise. General concepts and material covered in the class may be discussed with other students or in study groups, but specific assignments should not be discussed and any submitted work should be done entirely your own. Programs should always be 100% your own work and represent your independent thinking, and code should never be copied from the Internet or from another student. The instructor uses a program comparison system that compares submissions and highlights programs that are too similar in order to detect cheating.

It is expected that the class textbook will be used as a reference, but if any other reference materials (including web sites) are used in preparing homework solutions they must be clearly cited. Any incidents of academic dishonesty will be handled strictly, resulting in either a zero on the assignment or an F in the class, depending on the severity of the incident, and incidents will be reported to the appropriate UNCG office.

Attendance Policy: Attendance will not be taken in class, and is voluntary; however, all students are responsible for everything done or said in class (this can include changes in assignments, due dates, etc.). The university allows for a limited number of excused absences for religious observances — students who plan to take such an absence should notify the instructor at least two weeks in advance so that accommodations can be made (see the late work policy below). It is the student's responsibility to obtain notes from another student if they miss class.

In-class Behavior: When you are in class you should be focused on the class, and you should act in a professional and mature manner. During class there should be no eating, drinking, e-cigarettes, cellphone use, non-class related laptop use, or anything else that does not pertain to the class activities. Any distracting items may be confiscated at the discretion of the instructor.

ADA Statement: UNCG seeks to comply fully with the Americans with Disabilities Act (ADA). Students requesting accommodations based on a disability must be registered with the Office of Accessibility Resources and Services located in 215 Elliott University Center: (336) 334–5440 (or `http://oars.uncg.edu`).

University Closings: If university facilities are closed due to flu outbreak or other emergencies, it does not mean that classes are canceled. In such an event, please check the class web page and Canvas site for information about if and how the class will proceed.

Necessary Prerequisite Background: To enroll in this class, you must have made a C or better in CSC 130, CSC 230, and CSC 250. Through CSC 130 and CSC 230, you should have a fairly thorough background in basic Java programming (both syntax and design). You should also have seen and used some of the more advanced features of Java, and while you are not expected to be experts in these topics yet, it is your responsibility to review or brush up on these topics as needed. We will use these Java features extensively in the class, and through discussion and practice these concepts should become more and more clear as the class progresses. Some specific advanced topics, with references to material in the CSC 230 book (*Data Structures and Abstractions with Java, 4th edition, by Carrano and Henry*) and the book for this class, are given below:

Topic	***Carrano (CSC 230) Ref***	***Weiss (CSC 330) Ref***
Object-Oriented Design • Composition • Inheritance • Polymorphism • Abstract classes • Interfaces	Appendix D (pp. 869–886) Java Interlude 7 (pp. 499-500)	Sections 4.1-4.6 (pp. 109-149)
Generics	Java Interlude 1 (pp. 53–58) Java Interlude 3 (pp. 235–244)	Section 4.7 (pp. 150–156)
Exceptions	Java Interlude 2 (pp. 95–102) Java Interlude 4 (pp. 293–300)	Section 2.5 (pp. 47–51)
I/O, Scanner, Files	Appendix B	Section 2.6 (pp. 51–58)

You should also be familiar with elementary linear data structures from CSC 230, including arrays, lists (singly and doubly linked), stacks, and queues, as well as recursion and algorithms for searching and sorting. From CSC 230 and 250 you should be good at basic running time analysis using asymptotic notation. Finally, you should know basic discrete mathematics concepts and be able to write simple proofs.