



# CoSc 215 Data Structures and Algorithms

"[T]he challenge remains the same: *How can we get the best performance out of the current technology?* The design and analysis of data structures lay the fundamental groundwork for a scientific understanding of what computers can do efficiently." (Duane A. Bailey)

## Objectives

1. To know the abstraction, specification, implementation, and application of commonly used data structures such as lists, stacks, queues, trees, maps, and graphs.
2. To analyze the efficiency and space/time requirements of algorithms - particularly those algorithms used for sorting and searching,
3. To develop skills in algorithm abstraction, specification, design, implementation, and analysis.
4. To learn more advanced programming techniques, program development methods, and software testing and verification procedures.

## Prerequisites

All students are expected to have completed, with at least a grade of C, the equivalent of CoSc 200 Programming Techniques for 4 credit hours and have completed, or have current enrollment in CoSc 205 Discrete Mathematics. The Java programming language will be used, but a student with a strong knowledge of C++ can pick up the necessary Java rapidly.

It is assumed that students understand the following programming concepts: variables, constants, and declarations; primitive data types and operators; classes and objects; methods, parameters, and return types; conditional and loop control constructs; inheritance and subtyping; console and graphical input and output; file and string processing; iteration and recursion; arrays; and linked lists.

It is assumed that students have the following programming skills: problem solving; problem abstraction and decomposition; class, method, and algorithm development; and coding, compilation, tracing, and execution of applications and applets.

It is assumed that students understand the following mathematical concepts: logic, deduction, inductive reasoning (including proof by induction), series and sequences, elementary number theory, counting and enumeration, set theory, recurrence relations, and the basics of trees and graphs.

## Instructor

Instructor: David Housman  
 Office: SC 117, 535-7405  
 E-mail: [dhousman@goshen.edu](mailto:dhousman@goshen.edu)  
 Course Home Page: <http://www.goshen.edu/~dhousman/cosc215>  
 Office Hours: See Zimbra calendar, stop by, or make an appointment

If you are having difficulties or just a few questions not resolved during class time, please stop by my office, call me, or email me. I want difficulties to be resolved and questions answered before they become immovable obstacles to your success. Please let me know if you find the course material too difficult or too easy, if the pace of the course is too fast or too slow, if you have suggestions for improving the learning environment, or if the course is not what you

expected. We will work out a solution that will make the course much more profitable and enjoyable for you.

## Resources

Class	Come to AD 21on MWF 8:00 - 8:50 a.m and R 8:00 - 9:15 a.m. Prepare beforehand by reading and solving problems. Bring your text. Engage in the activities and discussion. Ask questions. Share your understanding with others. Take notes on your discoveries.
Text	<i>Data Structures and Problem Solving Using Java</i> , 4th edition, by Mark Allen Weiss (ISBN-13: 978-0-321-54140-6). Read carefully. Work on exercises and problems. Write down questions to ask.
Website	Visit Moodle and <a href="http://www.goshen.edu/~dhousman/cosc215">www.goshen.edu/~dhousman/cosc215</a> . Review lecture notes. Obtain laboratory assignments and other handouts. Download software. View software documentation. Link to a variety of resources.
Software	We will use the Java programming language ( <a href="http://www.oracle.com/technetwork/java/javase/downloads/index.html">http://www.oracle.com/technetwork/java/javase/downloads/index.html</a> ), the DrJava integrated development environment ( <a href="http://drjava.org/">http://drjava.org/</a> ), and code provided with the text ( <a href="http://users.cis.fiu.edu/~weiss/dsj4/code/">http://users.cis.fiu.edu/~weiss/dsj4/code/</a> ). These are available free from the listed web sites.
Assignments	Apply the concepts and techniques introduced in class and readings.
Exams	Synthesize and apply the concepts and techniques learned throughout the course. Expect in-class, take-home, and (possibly) oral portions to each exam.
Extra Credit	Attend Science Speakers, conferences, or computer science department events. Write a half-page discussion of some interesting computing or mathematics aspect of the activity. Obtain extra credit.
Classmates	Discuss lectures, labs, readings, and exercises with other students. Ask them questions and try to answer their questions. Schedule a regular time and place for group study. Review together for written exams. Try pair programming.
Instructor	Listen to his remarks during class. Ask questions during class, office hours, and via phone and email. Be prepared to not receive immediate answers. Answer his questions as he tries to guide you toward your own answers.
Time	Schedule sufficiently long periods of study with breaks for other courses, activities, sleep, and so forth. Do not cram for exams. Do not work on programs for too long in a single stretch. Schedule some time for reflection on and synthesis of what you have been learning.
Academic Resource & Writing Center	Goshen College wants to help all students be as academically successful as possible. If you have a disability and require accommodations, please contact Lois Martin, the Director of the Academic Resource & Writing Center early in the semester. In order to receive accommodations, documentation concerning your disability must be on file with the Academic Resource & Writing Center, Good Library 113, x7576, <a href="mailto:lmartin@goshen.edu">lmartin@goshen.edu</a> . All information will be held in the strictest confidence. The Academic Resource & Writing Center offers tutoring and writing assistance for all students. For further information please see <a href="http://www.goshen.edu/studentlife/asc.php">http://www.goshen.edu/studentlife/asc.php</a> .

## Collaboration

Learning is often enhanced when students study together; however, learning does not occur when one student simply provides to another student large portions of computer code or written solutions to assignments. Similarly, it is often helpful to make use of a variety of textual resources in addition to the text book; however, learning is not aided by wholesale copying from a textual resource. I encourage beneficial collaboration and discourage detrimental copying

with the following policies.

**Assignments.** You may *not* obtain someone else's solutions or code. You may look at and discuss another student's solutions or code, but you may *not* create or receive any written record during the discussion. You should give written acknowledgement to people with whom you have had discussions (even if you primarily gave, rather than received information) and to written materials that were helpful.

**Pair Programming.** Pair programming consists of two programmers working side-by-side, collaborating on the same design, algorithm, code, or test. One programmer, the driver, has control of the keyboard/mouse and actively implements the program. The other programmer, the observer, continuously observes the work of the driver to identify tactical (syntactic, spelling, etc.) defects and also thinks strategically about the direction of the work. On demand, the two programmers can brainstorm any challenging problem. Because the two programmers periodically switch roles, they work together as equals to develop software. (Laurie Williams, North Carolina State University Computer Science, [williams@csc.ncsu.edu](mailto:williams@csc.ncsu.edu), obtained from [www.pairprogramming.com](http://www.pairprogramming.com)). You may complete labs using the pair programming alternative. To do so, you must (1) be engaged in the pair programming during any time you work on the lab, (2) include both student names in all source files, and (3) submit all files to the drop box of a single student of the pair.

**Exams.** You may *not* use any resources unless a specific exception is stated by the instructor. You may converse about the exam with *no* one other than the instructor.

**Penalties.** Failure to observe the above rules will result in a penalty ranging from a zero on the assignment, lab, or exam to immediate failure of the course. Any violation of academic integrity will be reported to the Academic Dean.

**Rewards.** Observation of the above rules will help you learn the material well and give you the satisfaction of knowing that you have earned your grade

## Grading

Your grade will be based upon your performance on assignments (70%), the midterm exam (15%), and the final exam (15%). A semester average of at least 90%, 80%, 70%, and 60% will typically earn a grade of A, B, C, and D, respectively. Some upward or downward adjustment may be made based upon class participation, effort, and progress. In order to pass this course the weighted average of your midterm and final exams must be greater than or equal to 60%. The grade you receive for assignments and labs cannot raise your weighted exam grade by more than one letter grade.

**Due Date Policy.** Except in extraordinary circumstances (such as an extended illness or injury, or personal or family crisis), assignments, labs, and exams should be submitted by the specified deadline. For most assignments, there will be a resubmission opportunity about one week after the due date. Resubmissions (or late first submissions) will be penalized 30% of the total possible grade.

**Submission Standards.** All work submitted must contain your name and an acknowledgement statement. Files should be placed in the appropriate drop box folder. Written work should be submitted on paper that is standard size (8.5" by 11") and free from ragged edges. Multiple pages should be stapled, paper clipped, or placed in a folder. Work should be typed or written neatly. Resubmissions must include the original graded work in addition to the revisions.

**Grade Needed for Upper-Level Courses.** You must earn a grade of C or better in this course in order to take subsequent computer science courses for which this course is a prerequisite. If your grade is lower than C and you want to take upper-level computer science courses, you must repeat this course until you attain a grade of C or better.

**Incomplete Grades.** As a general rule, "incomplete" grades are not given in this course. Exceptions may be granted for documented prolonged medical or extenuating circumstances. However, an incomplete grade will not be granted because a student missed assignments, got behind in coursework, or had difficulty keeping up with the pace of the course. It is your responsibility to be current in your coursework and to ask for assistance or tutoring as needed.

## **Roles of Instructor and Student**

You and I share the responsibility to ensure that tasks and feedback actually facilitate learning and that evaluations are accurate and fair. Based upon my experience and training, I should establish course goals that are important and realistic, assign tasks that should facilitate learning, ensure that necessary resources for your learning are made available, and accept and respond to your assessments of the value of tasks assigned and resources provided. You should ensure that course goals are compatible with your personal goals, make a good-faith effort to complete assigned tasks, utilize available learning resources, and assess the value of tasks assigned and resources provided.