CS 260: Data Structures

Course Syllabus 2017

Professor: Richard Croft, Ph.D. Badgely Hall 109: 962–3695
e-mail: rcroft@eou.edu
web: cs.eou.edu/rcroft

Office Hours: MT RF 11:00—11:50
and by appointment or drop-in.

Note that I may be in BH 123 or LH 235 during office hours.

Catalog Description

This class introduces the analysis of algorithms and data structures commonly used in computer science, and the selection and design of data structures for the solution of specific problems.

Prerequisites

CS 162, Foundations of Computer Science II.

Outcomes

Upon completion of this course, students will:

1 Define fundamental abstract data types (ADTs);

2 Describe several applications for each fundamental ADT;

3 Describe several different structures to implement ADTs;

4 Select data structures for specific situations;

5 Write object classes to implement basic data structures and programs to use them; and

6 Describe the performance of algorithms that operate on data structures, particularly best-case and worst-case performance.

Textbook

Karumanchi, N. (2015). *Data structures and algorithms made easy in java*. Bombay, India: Careermonk Publications. ISBN: 9781468101271. (Required)

Means of Assessment

Quizzes will serve as formative assessment of student mastery of conceptual outcomes (outcomes 1-4, and 7) and a final examination will be used to assess student mastery of these outcomes. A paper developed over the course of the term will provide further in-depth assessment of conceptual outcomes. Programming assignments will be used for assessment of application of course concepts (outcome 5).

Course Activities

Lectures will provide a less formal perspective on readings from the text. Written exercises and programming assignments provide opportunities to apply what they have learned. Quizzes will indicate how well students are learning the material. A final exam will indicate students' overall understanding of course concepts. A research log

developed throughout the course will help students gain a better understanding of problem solving methods and the applications of data structures.

UWR Writing Intensive Outcomes:

- Students will produce at least 3,000 words (including drafts, in-class writing, informal papers, and polished papers); 1,000 words of this total should be in polished papers which students have revised after receiving feedback and criticism.
- Students will practice the forms of writing and reflect upon the nature of the writing used by graduates and professionals in the discipline the course represents.
- Students will write at least one paper integrating information from at least one source, employing the appropriate documentation style for the discipline represented by the course.
- Students will draft, revise, and edit their formal written work.
- Students will seek assistance from a Writing Tutor in the Writing Lab when needed and when referred by the instructor.

Note that you must receive a C- or better to receive UWR credit.

The requirements for some assignments in this course may exceed the minimum requirements for UWR credit listed above.

Course Policies

Attendance in this class is important. The concepts covered in the text will be further explained in class, and revisions to assignments may be announced in class. Your contributions to class discussions will help you and your classmates better understand material.

Note-Taking is important in any class, but computer science classes details are critical. Many concepts will be introduced in lecture that are not explained in the text. Your translation of class lectures will be essential for your success. Take thorough notes, and review them (preferably with another class member) frequently.

Assignments: The link for assignment sheet for each project will be accompaniedf by its due date. Programs are due at the beginning of class on the due date. Late work will be penalized 20 percent for each class day it is late and may not be turned in at all once a sample solution has been posted (typically two days after the due date).

Except for code provided to all class members as part of an assignment, **all** work must be your own. **No** code may be "borrowed" from other sources, including sample solutions posted for previous terms. Do NOT browse the internet for inspiration. Failure to heed this rule will be treated as a violation of EOU's rules concerning academic misconduct (see below).

Save all returned assignments.

Programming Style , Documentation & Formatting: Your programs should adhere to good software engineering principles. Document appropriately, choose sensible identifiers, format code for legibility, and divide code into logical procedures. A set of guidelines for programming style, documentation, and formatting are posted on the class web page. Programs that do not adhere to the standards established at that time will suffer a penalty of up to fifty percent.

Occasional **quizzes** over lectures, homework problems, and readings will provide feedback to let you determine if you are assimilating enough detail in course topics. You may drop one quiz grade. Quizzes may be made up only if I deem the documented excuse valid.

If you have any questions, comments, concerns, or suggestions, please feel free to write them on a slip of paper and leave it on the lectern (or hand it to me) when the class breaks. Your feedback may help improve the course.

Academic Misconduct
Eastern Oregon University places a high value upon the integrity of its student scholars. Any student found guilty of an act of academic misconduct (including, but not limited to, cheating, plagiarism, or theft of an examination or supplies) may be subject to having his or her grade reduced in the course in question, being placed on probation or suspended from the university, or being expelled from the university—or a combination of these. Please see Student Handbook at: Academic Honesty

Statement on Americans with Disabilities:
If you have a documented disability or suspect that you have a learning problem and need reasonable accommodations, please contact the Disability Services Program in Loso Hall 234 (telephone 962-3081) **before** the end of the second week of classes.

Assignments

Some assignments are low-risk or risk-free opportunities to experiment with ideas and see how well you understand course topics. Lab activities, exercises, and in-class group programming problems all fall into this category. These assignments are not graded but they are required—skipping these assignments will make graded work more difficult.

Programming assignments are graded, and each program will count eight or nine percent of your grade. In other words, programs are high-stakes activities. Programming, including developing problem solutions and designing the programs to implement them, is the kernel of computer science and your programs must 1) be your own work, 2) function according to specification, and 3) be well-documented *in English*. Labs and exercises will give you a solid start on programming assignments.

As we examine different abstract data types, you will write a *Data Structure Research Log* detailing the definition of the type and different ways of implementing it, as well as at least one example use for each implementation. Your log will refer to one or more sources to support your discussion. At midterm, you will submit your in-progress log for review, and at the end of the term a polished, complete copy will be due. The log is worth 25% of your course grade and if properly written will satisfy EOU's requirement for one lower-division University Writing Requirement (UWR) class. Precise requirements for the research log will be made available during the first two weeks of class.

Grading

Your final grade for this course will depend on your completion of the assigned homework and programs, quizzes, and a midterm and final exams. All activities will measure your ability to apply the concepts introduced in the text and class lectures. Distribution of credit is as follows:

Programming Problems: *Including documentation*	35 percent
Quizzes	15 percent
Data Structure Log	25 percent
Final Exam:	25 percent

Grade cutoffs will be no higher than 92 for A, 84 for B, 75 for C and 65 for D, but *may be lower* if statistical analysis of the distribution of scores indicates they should be.

Course Outline (Tentative*)

Week	Topics	Reading**
1.	Course intro; Java Review; General information about data structures; Abstract data types	Syllabus; 1.1–1.9, 1.14
2	Data and operations; Algorithm Analysis; ADT List; List implementations	3.1–3.6
3	List implementations, continued	3.7
4	ADT Stack and implementations	4.1–4.7
5	ADT Queue and implementations	5.1–5.7
6	Recursion; Trees (overview); General Trees	Chapter 26.1, 6.2,6.5
7.	Binary Trees; ADT Binary Search Tree	6.3, 6.4, 6.9
8	AVL Trees;	6.10, 6.11
9	Heaps & Priority Queues	7.1–7.6
10	No class Monday (Memorial Day); Special topics, course review	
11	FINAL EXAM: Wednesday 10:00–12:00	

*Schedule is subject to change based on class dynamics.