Data Structures

Instructor: Shawn Healey
Mail: shealey@sdsu.edu
Lecture Hour: TTh 1730-1845
Lecture Room: EBA-343
Office: GMCS-534
Office Hours: TWTh 1330-1530
Final Date: 18-DEC-2018
Final Time: 1530-1730

Overview

**CS 310**: Representations and operations on basic data structures. Arrays, linked lists, stacks, queues, and recursion; binary search trees and balanced trees; hash tables, dynamic storage management; introduction to graphs. An object oriented programming language will be used.

This course familiarizes computer science students with several classical data structures and algorithms. It builds upon the student’s prerequisite programming experience, which established a familiarity with Java syntax and style, and it prepares students for upper-division computer science coursework by applying these concepts to significantly larger problems. Consequently, this class requires a substantial, ongoing time-commitment by all students.

**Intended Audience**

As a required course for the degree, CS 310 prepares Computer Science majors for the computational tasks expected of them both academically and professionally. Students must understand not only the theory behind the data structures and algorithms we study, but they must also implement several at a low level, directly in Java. Computer Science minors, and computationally minded students from other departments, however, remain welcome in the class.

Should non-majors desire a similar class without the programming depth, and at least partially taught in Python, they should consider CS 305/496: *Data Structures for Scientists and Engineers*. It frequently fulfills the same degree requirements, but it does not require students demonstrate their programming abilities at the same level.
Student Learning Outcomes

Upon conclusion of the course, students will:

1. Successfully apply an appropriate abstract data type (e.g., list, deque, graph, map, priority queue) when solving a computational problem.

2. Analyze the inner-workings of several classical data structures (e.g., linked lists, array lists, hash tables, heaps, binary search trees, adjacency matrix) and identify their limitations.

3. Understand several iconic algorithms (e.g., binary search, sequential search, depth-first search, Dijkstra’s shortest path) and their applications to problems both within data structures and without.

4. Write Java applications capable of interacting with the operating system’s file system.

5. Setup and use industry-standard development tools, including an Integrated Development Environment and version control system.

6. Perform popular software development tasks with a remote server through the terminal window or shell.

7. Produce cleaner code adhering to industry standard formatting, naming convention, and software quality heuristics.

Learning Resources

This class does not require the student to purchase any additional books or pay lab fees. All required material remains freely accessible. It does identify several paid resources available to students, but these remain optional. This section identifies some of the assets available to students during their individual studies.

Class Website

Assignments, links to the class resources, and other notices appear on the instructor’s edoras class website: http://edoras.sdsu.edu/~healey
Video Lectures

In addition to the identified books, this course includes a series of required video lectures for each of our major topics. The course website includes a link to these lectures. The lectures help reinforce the discussion topics from the lecture hour and add depth to the readings.

Material from these lectures will appear on the exams.

Some students report the greatest success with these videos when they watch them prior to attending the lecture on the topic, yet others prefer watching the production after the lecture to reinforce the class discussion.

Optional Texts

Although numerous quality works on the subject of data structures exist (in nearly every programming language) none discuss the material in a universally appealing way. That is, some students gravitate toward one text while others prefer the design approaches of another author. Consequently, this course requires no text books.

<table>
<thead>
<tr>
<th>Description</th>
<th>ISBN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Michael T. Goodrich and Roberto Tamassia,</td>
<td>978-1118771334</td>
</tr>
<tr>
<td><em>Data Structures &amp; Algorithms in Java (6th ed.)</em></td>
<td></td>
</tr>
<tr>
<td>Nell Dale, Daniel T. Joyce, and Chip Weems,</td>
<td>978-1284089097</td>
</tr>
<tr>
<td><em>Object-Oriented Data Structures Using Java</em></td>
<td></td>
</tr>
<tr>
<td>(4th ed.)</td>
<td></td>
</tr>
</tbody>
</table>

None of the worksheets or programming projects given to students during the semester require access to any of the above texts.

Course Reader

Professor Alan Riggins prepared an excellent set of accompanying lecture notes available through Montezuma Publishing. Like the text book, this reading material remains optional. The notes cover much of the content in the class and may provide students with additional insight into the class and its material.

The reader is optional.
Office Hours

The instructor provides regularly scheduled office hours to assist students with programming projects. Office hours end at the specified time, so students may not receive attention if they arrive late in the session on a popular date. Due to overwhelming demand, students should not count on assistance on the programming projects the last week of class.

In addition to the instructor’s weekly office hours, the university supplies multiple tutors. These teaching assistants will help with worksheet problems and provide programming project assistance as they deem necessary.

Computer Lab

This course requires every student’s active participation in the programming assignments. Students may certainly use their personal machines. Additionally, the university provides a open computer laboratory available for student use at designated times.

<table>
<thead>
<tr>
<th>Location</th>
<th>Hours</th>
</tr>
</thead>
<tbody>
<tr>
<td>GMCS-425</td>
<td>MW 1230-1600</td>
</tr>
</tbody>
</table>

Email

In addition to office hours, students may contact the instructor about the class through email. One cannot debug software easily through this channel, however, so students should limit the discussion to logistical and theoretical questions.

Grading

Examinations, in-class exercises, reading assignments, and programming assignments all serve as vital components in each student’s final grade.
Scale

This class is not curved, and the grading scale is defined below.

<table>
<thead>
<tr>
<th>Weighted Percentage</th>
<th>Grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>&gt;= 93</td>
<td>A</td>
</tr>
<tr>
<td>90-92</td>
<td>A-</td>
</tr>
<tr>
<td>87-89</td>
<td>B+</td>
</tr>
<tr>
<td>83-86</td>
<td>B</td>
</tr>
<tr>
<td>80-82</td>
<td>B-</td>
</tr>
<tr>
<td>77-79</td>
<td>C+</td>
</tr>
<tr>
<td>73-76</td>
<td>C</td>
</tr>
<tr>
<td>70-72</td>
<td>C-</td>
</tr>
<tr>
<td>67-69</td>
<td>D+</td>
</tr>
<tr>
<td>63-66</td>
<td>D</td>
</tr>
<tr>
<td>60-62</td>
<td>D-</td>
</tr>
<tr>
<td>&lt; 60</td>
<td>F</td>
</tr>
</tbody>
</table>

Weighting

Students earn points toward their grade several ways. Each component contributes a portion to the final grade. The weighting for each category is as such:

<table>
<thead>
<tr>
<th>Category</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Worksheets:</td>
<td>20</td>
</tr>
<tr>
<td>Programming Projects:</td>
<td>30</td>
</tr>
<tr>
<td>Midterm Exams:</td>
<td>25</td>
</tr>
<tr>
<td>Final Exam:</td>
<td>25</td>
</tr>
</tbody>
</table>

Worksheets

Students must work individually, but may seek the instructor or teaching assistants’ help, on a series of regularly assigned homework worksheets. The instructor shall post these assignments to the class website with their corresponding due dates at least one week in advance. Every worksheet requires a *physical* submission due at the start of class on the due date.
Programming Projects
In addition to individual studies, students must work with a partner on the
design, implementation, and testing of multiple programming projects of
significant complexity. They include the implementation of several classical
abstract data types. During these paired exercises, both partners must
participate in the coding activity.

Midterm Exams
Two examinations during the semester help evaluate the student’s progress
and assist with grade assessment. These tests remain fairly encapsulated
and typically stick to the current class material. Although not designed
as cumulative tests, one must understand the concepts addressed in earlier
classes in order to succeed on the later exams.

Final Exam
All students must take the final exam at the date and time specified by the
Registrar.

Class Policies

Attendance
The material covered during class discussion appears on the exams. The
lecture hours provide students with an opportunity to ask questions about
the material in-class and outside of office hours.
Although the instructor does not take attendance during lecture hours,
the lecture material is mandatory.

Special Assistance
If you are a student with a disability and believe you will need accommoda-
tions for this class, it is your responsibility to contact Student Disability
Services at (619) 594-6473. To avoid any delay in the receipt of your ac-
ccommodations, you should contact Student Disability Services as soon as
possible. Please note that accommodations are not retroactive, and that
accommodations based upon disability cannot be provided until you have
presented your instructor with an accommodation letter from Student Dis-
ability Services. Your cooperation is appreciated.
Late Work

The instructor will not accept worksheets submitted after their required due date. Failure to deliver the worksheet on time results in zero points on that assignment. Students may deliver worksheets early to the instructor’s office in the event of a foreseen absence.

Students may deliver programming projects up to 72 hours after the required delivery day. Doing so results in a 10% reduction per day.

Makeup Exams

Exams shall take place on the specified dates and times. This class does not offer makeup exams. Failure to take an exam results in a zero on the exam.

Academic Integrity

University on academic integrity clearly defines both Cheating and Plagiarism (Student Affairs: Cheating and Plagiarism).

In this class, students may only collaborate with their chosen partner. Collaboration on programming assignments, between groups, remains forbidden. Exchanging and communicating ideas with other classmates remains acceptable, but never share source code. Doing so constitutes cheating by both parties.

You may not submit code you found on the Internet.

One will fail as a computer professional if one cannot design, write, debug and deploy computer code without copying it from the Internet or another person. Anyone caught cheating, either on a programming assignment or on a test, will receive an F and will be turned in to Judicial Procedures.

Extra Credit

Struggling students frequently seek extra credit near the semester’s end to help improve an undesirable grade. If the tests, assignments, or lectures leave one uncertain or uneasy, contact the instructor or teaching assistant as soon as possible. This section, like most classes in the department at this level, offers zero individually-directed extra credit. Extra credit points emerge, if they emerge, in the form of additional examination questions.

Computing Environment

Building upon the prerequisite programming experience, this course makes extensive use of the Java programming language. Transfer students who
took equivalent prerequisites in a different, object-oriented language must allocate additional time to learning Java’s non-trivial nuances and difficulties. Students lacking the required programming coursework will, with near total certainty, fail the course.

The unit tests and supporting Java source code students use during the semester require Java Development Kit 8. Class accounts, detailed below, arrive with Java at the correct version level by default.

This class also makes use of the industry-standard build-manager Maven. Specifically, projects submitted must adhere to its directory structure. Projects produced through the recommended IDE provide this automatically. Students using an alternative development environment remain responsible for understanding the maven directory layout and correctly submitting the contents of the src folder for each of the required projects.

Class Accounts

The instructor shall distribute temporary accounts for Edoras, and their initial passwords, in class during the first week of the semester. All current students may apply for a personal account on Edoras, but unlike those, the course instructor may step into the class account at any time. Thus, class accounts lack the privacy a personal account affords, so students shall avoid using it for work unrelated to the course. Moreover, University policy prohibits the use of SDSU computing resources for commercial purposes.

Several files in the class account serve important functions (e.g., .login, .cshrc, and .ssh), so students should avoid arbitrarily modifying their contents, for it may impact the ability to connect to the impacted account. Additionally, the .ssh and .rhosts files remain essential to the grading process, and, as such, students must not modify, rename, move, or delete these files. Doing so opens up the possibility of a zero on the programming assignment.

To connect to the university server, Linux and Mac users will connect through the terminal window. Windows users may use a third-party ssh application, like PuTTy, but the instructor overwhelmingly encourages windows 10 users to install the Ubuntu shell (via the Windows store) and connect through this window.