CSC 385: Data Structures & Algorithms

Syllabus—Spring 2018

Roger West, Instructor

Topics Covered

This course covers basic algorithm analysis and intermediate-to-advanced data structures, using the Java programming language. We will cover the following topics in roughly the order listed, as time permits:

1. Algorithm complexity, Big-Oh analysis
2. Recursion, backtracking
3. Searching algorithms (e.g., binary search)
4. Sorting algorithms (e.g., insertion sort, quicksort)
5. Lists
6. Iterators
7. Stacks
8. Queues
9. Trees
10. Binary Search Trees (e.g., AVL tree, red-black tree)
11. B Trees
12. Hashtables
13. Sets
14. Maps
15. Graphs
16. Other topics as time permits

We will also cover some of the basics of object-oriented design throughout the course.

Prerequisites

1. The equivalent of 2 semesters fundamental programming in Java (CSC 225 + CSC 275 at UIS, or equivalent from another university)

2. Discrete Mathematics (CSC 302 at UIS, or equivalent from another university)

3. Familiarity with using the Java programming language and the Java 2 Software Development Kit (JDK/J2SE)

The first two prerequisites are required by the Computer Science Department at UIS. It is important that the two semesters of programming be in Java, as opposed to some other language.

Course Materials

Required Text:

Data Structures and Abstractions with Java, **4th edition** (2015) , by Frank M. Carrano & Timothy M. Henry, (ISBN 10: 0-13-374405-1; ISBN 13: 978-0-13-374405-7).

The textbook comes with a scratch-off access code that gives you access to the book's companion website (http://www.pearsonhighered.com/carrano). The code is good for 6 months from the time it is activated. I do not require that you utilize the companion website, so if you prefer to buy a used book, which may no longer have a valid access code, you will not be missing anything required by this course.

Important note: I will be using the textbook to help explain the concepts in the course, but for the most part I will **not** be using the source code the textbook provides. The source code we will be using will be available for download on the Blackboard site. The textbook is organized such that you do not need to use the author's source code to understand the concepts.

Operating System:

Since Java is highly platform-independent, and all course material is accessible via the web, you can probably use just about any operating system that supports Java. You should be in good shape if you are using Windows XP/2000/Vista/7, some flavor of UNIX or Linux, or Mac OS X.

Required Software:

1. The Java 2 Software Development Kit, Standard Edition (J2SE, or JDK) by Oracle. Java SE 1.9 is the current release. I recommend that you install this version, if possible. If you run into a compatibility problem with the IDE you are using, then try installing version 1.8. At the time of writing, Eclipse Oxygen supports Java 1.9, but you have to install an update to get it to work with 1.9.

2. A development environment for writing and compiling Java source code. I personally recommend Eclipse, but you are free to use whatever development environment you wish, or if you prefer, you can simply use a text editor and compile your code at the command prompt. Eclipse is a powerful tool that can be downloaded free of cost. The latest release is code-named Oxygen. Other IDE's students have commonly used in the past include NetBeans and BlueJ.

3. The latest version of Adobe's Flash plug-in (for viewing the animated instructional aids). Go to http://www.adobe.com and click on the link to download the Flash Player.

4. Some application for creating archive files, for submitting multiple documents in a single package. There are many applications out there that will do this, such as WinZip, 7-Zip (free), WinRAR, StuffIt (for Mac OS), etc. Most of these applications can be freely downloaded, but require you to pay a fee (around $30) to **legally** use them beyond the trial period.

5. A word processor capable of reading Rich Text Format (.rtf) files. The feedback for all graded assignments will be sent as .rtf documents.

Recommended Software:

1. A tool for drawing UML class diagrams. There are many free UML class diagram tools out there. Some of them are only free for a trial period, but they should still work for this course. Microsoft Visio will let you create class diagrams, although it lacks support for Java data types. Some examples of UML tools students have used in the past are ArgoUML, UML Lab, Visual Paradigm, and ObjectAid UML Explorer for Eclipse. If you choose an Eclipse plug-in, make sure it is compatible with the version of Eclipse you are using. Since you may choose to use a tool I don't have installed on my computer, it is best to export your diagram as an image file to ensure I will be able to open it.

Microsoft Software:

As computer science students at UIS, you should be able to download certain Microsoft products for free, through the UIS DreamSpark website (https://e5.onthehub.com/WebStore/ProductsByMajorVersionList.aspx?ws=8b6aa06c-1c70-e011-971f-0030487d8897&vsro=8&JSEnabled=1). You should be able to access this site using your UIS NetID and password. Let me know if you have any trouble accessing the site or downloading the software.

Organization of the Course

Since this is an online course there will be no physical class meetings, but for the purposes of maintaining some degree of regularity I have divided the course into a series of modules. One module roughly corresponds to one week of the semester. Each module will include all reading and assignments for that week. New modules will be posted on Mondays.

The textbook provides the core source material for the course. However, I am also providing written online lectures that are designed to supplement the textbook. I think it is important that you get another perspective on the concepts besides the one presented by the textbook, and there are some details the textbook leaves out that I feel are important. The lectures are written, rather than prerecorded.

I will also providing several interactive labs throughout the course to help illustrate some of the concepts we will be covering. The interactive labs will either be Java code that you can download and execute, or Flash animations that you will be able to view in your browser using the Flash Player plug-in from Adobe. The Flash animations allow you to step through certain algorithms one statement at a time, allowing you to see exactly what the algorithms are doing and how they are doing it. I think you will find the animations very enlightening.

Organization of the Blackboard Site

The material for the Blackboard site used for the course is organized into the following categories, all of which can be accessed using the buttons in the left panel of the main page:

1. Faculty Info - my info (name, email, phone, etc.)
2. Syllabus - this syllabus
3. Announcements - check the announcements frequently; I will post all important announcements here
4. Calendar - shows the tentative schedule; **all deadlines will be posted here**.
5. External Links - contains links to software download sites and other external sites relevant to the course
6. Communication - lets you send email, etc.
7. Discussion Board - There are no group discussion assignments in this course, but I may use this section to post frequently asked questions about the material, homeworks, etc., as they arise

Each module will have its own button, labeled "Module 1", "Module 2", etc. Clicking on the button for a module will take you to a page that will have the objectives, lecture, reading, and homework assignments for that module.

email

These days there is a plethora of communications options available. **The best way to reach me is through my UIS email account (rwest2@uis.edu)**, as I check my email frequently during the day, and I usually check it seven days a week. In general, I strive to answer all emails no later than one day after I receive them. Please do **not** send me messages through Blackboard's "Messages" feature in the Communications section—I never check for messages here. I do frequently check the discussion boards, even when there is not an ongoing discussion assignment, but the discussion boards are not appropriate if you have an urgent question. I have found that >99% of problems can be resolved through email, but in the event you need to reach me by phone, that can be arranged. Please email me first, though, to schedule a good time for a phone conversation; that way, we can avoid problems with missed phone messages and phone tag.

When you send me an email, please include the course number (CSC 385) in the subject line of the email.

Exams and Quizzes

Each module will have a short, open book/notes quiz that will be administered through Blackboard. Each quiz will be worth 10 points. There will be 12 quizzes over the course of the semester. Please bear in mind that while the quizzes will allow multiple attempts, only the first attempt will be graded. This allows you to go back and review the quiz questions, and practice them again if you wish.

There will also be a timed final exam at the end of the semester. The exam will be open book/notes, and it will be administered on Blackboard. You will **not** be required to obtain a proctor for this exam. I will provide a study guide for the exam during the week before final exam week.

For exams I usually use a combination of problem-solving and non problem-solving questions. I may have you write a small amount of code, but you will **not** be required to memorize source code from the course materials. You will, however, need to understand how the algorithms and data structures work. By the end of this course you should be capable of implementing the more basic data structures (e.g., linked list) from scratch. I will not likely ask you to implement an entire data structure on the final exam, but I may, for example, ask you to write code for adding an item to a data structure. I may also ask you to write code to sort a list of items, but again you should understand these concepts well enough that you can write your own code without memorizing the code I provide.

Homework Assignments and the Semester Project

Homework assignments will comprise slightly over 50% of your course grade. Most of the assignments will involve writing code, but a few will focus on problem solving and program design.

There will be a substantial semester project that will require you to write a complete program from scratch. The purpose of this project is to give you some experience in building a moderately sized software project from the ground up, and to expose you to the notion of software design (as opposed to immediately hacking out code). The project will be sufficiently difficult that you most likely will not be able to just sit down and write it in a weekend, or even a week. The idea is for you to follow the basic steps of the software life cycle:

1. understand the problem and the requirements
2. design the project, without writing code
3. based on your design, write the code
4. test and debug your code

Deadlines

All deadlines for quizzes and homework assignments will be posted on the calendar available by clicking the "Calendar" button in the control panel. Ideally, I would like to receive all work by the posted deadline, but I understand there may be circumstances that sometimes prevent timely submissions. If you think you will need more time for an assignment, please let me know via email. My philosophy is that it is better for a student to turn in an assignment late, but complete, as opposed to on time, but incomplete, since more learning comes from a completed assignment than from an incomplete one. When I post solutions to assignments, they are triggered to become visible only when you have received a grade, so there is no opportunity to see a solution before you have submitted the assignment. Some important things to note about this policy are:

1. You may turn in an assignment as many times as you wish. My policy is to only grade your most recent submission. However, for those assignments where a solution or discussion is posted, since these will become visible once I have graded your assignment, there will be no further opportunities for you to submit that assignment.

2. My ability to grade in a timely fashion depends heavily on being able to spread out the grading of all assignments over the semester, so if too many assignments are turned in late, as a class, it can cause a severe backlog in grading. In the event of such a backlog, I reserve the right to modify the deadline policy to ensure I am able to complete all the grading.

Source Code

The overall goal of this course is to teach you concepts of intermediate level data structures and algorithms. Throughout the course, keep in mind that there are always two parts to every data structure and algorithm:

1. the logical part that defines how it should work (i.e., the algorithms)

2. the implementation that is used to build it (i.e., the code)

Do not expect to understand the concepts in this course by simply memorizing the code. Use the code to help you understand how the algorithm works, but make sure you understand the logic underpinning each algorithm. Memorizing code will limit your understanding, since many code variations can produce correct results. Also, a programming construct that works in one programming language may not work in all programming languages.

Feedback & Grading

I do the best I can to grade assignments as they come in, but sometimes things get backlogged and I fall behind. When I have graded an assignment I will post your grade on Blackboard and send you an email containing your grade and a brief explanation of what you missed, if anything.

For final course grades I use the standard University of Illinois grading scheme shown on the next page. How well you do on the semester project and the final exam can positively affect your grade. If you are very close to the borderline between two grades and you do very well on either the semester project or the final exam, or both, I will be more likely to give you the higher of the two grades.

% of Total Points	Letter Grade
93 1/3% to 100%	A
90% to <93 1/3%	A-
86 2/3% to <90%	B+
83 1/3% to <86 2/3%	B
80% to <83 1/3%	B-
76 2/3% to <80%	C+
73 1/3% to <76 2/3%	C
70% to <73 1/3%	C-
66 2/3% to <70%	D+
63 1/3% to <66 2/3%	D
60% to <63 1/3%	D-
<60%	F

Participation

Many online courses place a very heavy emphasis on group discussion. While that may be ideal for many subjects, I don't feel it would be appropriate to do that for this course. I may post a question or two for group discussion, but primarily you will be doing your assignments individually.

If you don't understand something, or you discover an error (they unfortunately happen from time to time), please don't hesitate to contact me via email.

Your comments and suggestions (both positive and negative) are greatly encouraged. If you have any questions about any of the material, please do not hesitate to email me. I want everyone to enjoy the course and come away with a good understanding of the material, and I will do the best I can to help you. Your feedback will help me to correct any problems in the course that may arise.

Academic Integrity

The UIS community of faculty, staff, students, and alumni are committed to academic excellence, which thrives on honesty, trust, and mutual respect. Academic integrity is at the heart of this commitment. Academic violations include plagiarism, cheating, misrepresentation, academic interference, unauthorized access, and facilitation. Violations of the Academic Integrity Policy may result in sanctions including failing the assignment, failing the course, transcript notation, or referral for Academic Hearing.
Students are responsible for being aware of the UIS Academic Integrity Policy (available at http://www.uis.edu/academicintegrity/policy/) and for demonstrating behavior that is honest and ethical in their academic work. Computer Science students are also responsible for being aware of the Computer Science Department's policy on academic integrity (available at http://csc.uis.edu/#Honesty). Basically, the Computer Science Department supports and upholds the UIS Academic Integrity Policy, but extends the definitions of cheating and plagiarism to include examples that are pertinent only to the field of computer science, according to the following working guidelines:

- Plagiarism is suspected if an assignment calling for independent design and implementation results in two or more solutions that differ only by simple mechanical transformations. An example of a simple mechanical transformation would be for on student S to take student T's code, globally change the variable names and submit this code as the work of student S.

- Cheating is suspected if an assignment calling for independent design and implementation results in a solution that the student cannot explain to the instructor in terms of either general method or specific techniques.

It is impossible to provide an exhaustive list of examples that constitute cheating or plagiarism. The examples presented on the CSC Department website have been extended to include the purchasing code from another student or a vendor on the Internet and submitting this code as your own. This includes purchasing or freely downloading code from another student or a vendor on the Internet, and submitting that code as your own.

The use of third-party libraries such as Google's Guava library does not constitute academic dishonesty, although the use of such libraries may be disallowed for certain assignments.

Please ask me if you have questions regarding course expectations that pertain to academic integrity, or if you are unsure whether what you are about to do will be a violation of the academic integrity policy.

Disabilities

If you are a student with a documented temporary or ongoing disability in need of academic accommodations, please contact the Office of Disability Services at 217-206-6666.

Disabilities may include, but are not limited to: Psychological, Health, Learning, Sensory, Mobility, ADHD, TBI and Asperger's syndrome. In some cases, accommodations are also available for shorter term disabling conditions such as severe medical situations. Accommodations are based upon underlying medical and cognitive conditions and may include, but are not limited to: extended time for tests and quizzes, distraction free environment for tests and quizzes, a note taker, interpreter and FM devices.

Students who have made a request for an academic accommodation that has been reviewed and approved by the ODS will receive an accommodation letter which should be provided by the student to the instructor as soon as possible, preferably in the first week of class.

For assistance in seeking academic accommodations, please contact the UIS Office of Disability Services (ODS) in the Human Resources Building, Room 80, phone number 217-206-6666.

Guidelines for Submitting Assignments

1. When you click on an assignment to view/download it, there will also be a link that will allow you to upload your completed assignment. Please reserve submitting assignments via email as a last resort.

2. For programming assignments, make sure you submit **only** your .java source files. Do **not** submit any project-related files that are generated by your IDE, compiled `.class` files, or `.jar` files. I cannot give you credit for programs that are already compiled.

3. I will not accept Java source code submitted as Word documents, .txt files, or any file format other than .java.

4. If you need to submit multiple files for a given assignment, please place all the files into a single archived file, such as a .zip file, and submit the single archive file.

Point Distribution (tentative, depending on how the semester progresses)

Homework assignments	(180 points)
Semester programming project	(100 points)
Quizzes	(120 points)
Final Exam	(100 points)
Total points for the course:	500