CSE 332

Data Structures and Parallelism

Fall 2022

Instructor: Richard Anderson

Email: anderson@cs.washington.edu

Office Location: CSE2 344

Course Web Page: http://www.cs.washington.edu/332/22au

Description: Covers abstract data types and structures including dictionaries, balanced trees, hash tables, priority queues, and graphs; sorting; asymptotic analysis; fundamental graph algorithms including graph search, shortest path, and minimum spanning trees; multithreading and parallel algorithms; P and NP complexity classes.

Prerequisites: CSE 311

Course Text: (Optional) Data Structures and Algorithm Analysis in Java 3rd Ed., Mark Allen Weiss, Addison Wesley: 2012, ISBN-10: 0132576279. (The 2nd edition is fine, and is much cheaper used.) Our course calendar will list sections of the textbook that are most relevant to the topic discussed in class that day. You may find the textbook useful to clarify topics and find more examples as well as to examine Java implementations of the data structures and algorithms discussed during lecture. We will not be assigning problems from the textbook. We will use a set of free on-line notes for the material on parallelism and concurrency.

Computing Resources: We will use Java for programming assignments. We strongly recommend although will not require that you use the IntelliJ development environment. Links for downloading and installing Java can be found on our course web page.

Communications: The course message board should be your first stop for questions about course content and assignments. Before posting, first check that your question has not already been answered on the message board, and if not, ask it there. If it is not possible to ask your question on the message board without revealing details of your solution, please either use a private post on the message board or send email to cse332-staff at cs.washington.edu, which will go to the instructor and TAs. In general we prefer that you send questions to the cse332-staff list instead of to an individual staff member so that you will get a faster response time and the entire staff can remain aware of questions and issues. You will be automatically subscribed to the course email list if you are registered for the course and will be held responsible for anything posted there (only course staff will post to the course email list).

Assignments: Assignments will be a mix of smaller exercises and larger programming projects:

Exercises: There will be approximately weekly exercises. These will directly test your understanding of topics we are covering and the theory behind them. Some exercises require you to write short bits of code or complete an activity online, while others are more traditional “written” exercises. For “written” exercises, we require these to be turned in electronically. LaTex is one option, but neatly written scanned documents will also be fine as long as they are readable. Exercises are to be done individually. Your two lowest scores on exercises will be discarded.

Programming Projects: There will be three programming projects. Programming projects will be graded on correctness, architecture and design, and analysis. Note that your answers to the analysis questions will be part of the grade. We will not grade you on code style, as long as your code is readable and you follow the guidelines explicitly given in the project handouts. Program design/architecture and analysis are crucial in this course. Projects are designed for individual students (we are NOT allowing pair programming this quarter).

Exams: The midterm (November 4) and final (December 15) will be in-class.

Late Policy: Exercises and Exams may not be turned in late. You will, however, have five late days that may be used to gain an extra 24 hours for a programming project. If you have used up your late days, a penalty of 10% per day will be assessed. Programming projects will not be accepted beyond 72 hours from the original due date.

Late days are intended to help you manage your time and unexpected life circumstances; leftover late days will not contribute to your grade in any way. If there are extenuating circumstances that prevent you from submitting an assignment on time, you should discuss this with the instructor (preferably in advance).

Grading and Evaluation: Grades will be computed approximately as follows (weights may be modified):

- Exercises: 25%
- Programming Projects: 35%
- Midterm: 15%
- Final: 25%

Extra Credit: We will keep track of any extra features you implement for programming projects (the Above and Beyond parts). You won't see these affecting your grades for individual projects, but they will be accumulated over all projects and used to bump up borderline grades at the end of the quarter. The bottom line is that these will only have a small effect on your overall grade (possibly none if you are not on a borderline) and you should be sure you have completed the non-extra credit portions of the homework in perfect form before attempting any extra credit. They are meant to be fun extensions to the assignments.

Collaboration and Academic Integrity: Programming projects will be "single person assignments" in which you will work on an assignment individually. While you should write the code on your own, it is permissible to get help in debugging from other students. You are also permitted to get help from others in resolving "infrastructure issues" or "computer issues." For exercises, we expect all written/programmed work to be your own. You must at least attempt a problem on your own before discussing it in a group–but after first attempting on your own, we do encourage you to brainstorm together! During brainstorming sessions, you may use a whiteboard, but you may not take any typed/written or photographed work outside of the session. After discussion, take a 30-minute break before writing up your solution individually. If you collaborate with anyone in any capacity, you must identify them at the top of your assignment as a collaborator.

Referring to solutions found on the web or solutions from this or other courses from previous quarters is also considered cheating, as is requesting help with an assignment on an outside message board. We plan on running similarity-detection software over all submitted student assignments, including assignments from past quarters.

If you do not follow these rules, you will be considered to have cheated. Cheating is a very serious offense. If you are caught cheating, you can expect a failing grade and initiation of a cheating case in the University system. Cheating is an insult to the instructor, to the department, and most importantly, to you. If you feel that you are having a problem with the material, or don't have time to finish an assignment, or have any number of other reasons to cheat, then talk with the instructor. Copying others' work is not the solution. To avoid creating situations where copying can arise, never e-mail or post your solution files. You can post general questions about interpretation and tools but limit your comments to these categories. If in doubt about what might constitute cheating, send the instructor email describing the situation. For more details see the Academic Misconduct web page.

COVID-19 Safety: If you are sick, do not attend class. Masks may be worn, but they are no longer required.

Disability Resources: The Disability Resources for Students (DRS) is a unit within the Division of Student Life and is dedicated to ensuring access and inclusion for all students with disabilities on the Seattle campus. They offer a wide range of services for students with disabilities that are individually designed and remove the need to reveal sensitive medical information to the course staff.

You can refer to the university policies regarding Disability Accommodations for more information.