SJSU SAN JOSÉ STATE UNIVERSITY

College of Science · Computer Science

Data Structures and Algorithms

CS 146

Spring 2026 Section 02 In Person 3 Unit(s) 01/22/2026 to 05/11/2026 Modified 01/23/2026

Contact Information

Instructor: Dr. David Scot Taylor

- **Email:** david.taylor@sjsu.edu
- **Office:** MacQuarrie Hall 212
- **Phone:** (408) 924-5124
- **Website:** www.cs.sjsu.edu/~taylor

Office Hours

- Monday, 12 PM to 2:00 PM, MacQuarrie Hall 212
- by appointment

MacQuarrie Hall 212 or https://sjsu.zoom.us/j/89560963006 (https://sjsu.zoom.us/j/89560963006)

Course Information

Lecture, Section 2

Monday, Wednesday, 9:00 AM to 10:15 AM, Boccardo Business Center 226

Lecture, Section 3

Monday, Wednesday, 10:30 AM to 11:45 AM, Duncan Hall 450

Course Description and Requisites

Implementations of advanced tree structures, priority queues, heaps, directed and undirected graphs. Advanced searching and sorting techniques (radix sort, heapsort, mergesort, and quicksort). Design and analysis of data structures and algorithms. Divide-and-conquer, greedy, and dynamic programming

algorithm design techniques.

Prerequisite(s): MATH 30, MATH 42, CS 46B, and [(CS 48 or CS 49J) if CS 46B was not in Java], each with a grade of "C-" or better; Computer Science, Applied and Computational Math, Forensic Science: Digital Evidence, Software Engineering, Data Science majors only; or instructor consent.

Letter Graded

* Classroom Protocols

Other than the beginning of the semester (to establish that you are in the class), exams, and in-person individual code review, I will not take attendance. Nevertheless, you are expected to attend class: the class was designed to be taught in a `flipped' style: most new material will be introduced through videos, as part of homework. During class time, we will try to spend our time on interactive activities, rather than me lecturing to you. I have never tried to teach this class to a large section before, so I will have to see how much adjustment the course needs as we go.

Many students feel like this style doesn't teach them as well. Research shows the strong possibility that it is more effective, even if students don't feel that way. Honestly...it feels pretty strange from my side too, and being that "class discussions" are difficult to have with a class of over 150 students, it may be even more awkward this semester than usual.

Class participation and feedback are very important to keep the course interesting. *If I am covering material too slowly or quickly, or if I am not clearly explaining things, you must let me know.* I prefer an interactive learning environment, though some of that will fall by the wayside with such a large section. If you disagree with something I say, speak up. Argue with me in front of the class. It will make the class better, and right or wrong, constructive interaction will not not hurt your grade. If you are correct, clearly my mistake should be corrected. If you are incorrect, probably I have not explained something clearly anyway, and at least half of the class is confused by it. Point it out right then and there. In cases of exceptional participation that seem to benefit the class as a whole, I reserve the right to improve a student's grade by up to 1/3 grade.

Class Modality

This is an *in-person* class, held on the main SJSU campus.

Program Collaboration Policy

You are expected to code your own programs, with at most minor help from others. Talking abstractly, while still trying to figure it out, with somebody else who is in the same situation, is fine. Sharing code is not, and this includes reading their code and retyping it, or having them dictate it to you. Do not look for premade solutions. **Do not copy code.** You should understand what your code does. If I ask you what something does in your code, and you don't understand why it is in your code or what it does? That is unacceptable, as it indicates that you are submitting work which is not your own. If you can get

somebody to explain something to you in detail, to the point that you can understand and code it, that is okay. **Your code will be checked for correctness, but graded on your ability to answer questions about it. It is possible to get credit for code that doesn't work. It is possible to not get credit for working code if you don't seem to understand what it does. This latter case may also be deemed academic dishonesty.**

Although talking with others who are working on the problem is fine, do not directly tell somebody else how to do it if you have already figured out an algorithmic aspect and have yours working. If I think it is appropriate to give algorithmic hints, I will give them, that is my role as the instructor. They should be asking me, not you. **Beyond scheduled office hours, I generally spend over 100 hours this semester answering emails and holding additional office hours. Answering questions, based on the students situation, is a large part of my job as an instructor.** I might not always give as direct an answer as you are hoping for, it depends on how far you have gotten, what you have tried, and what the deadlines are. But, I really do try to give you answers that will push you towards learning the material. Do not get the answers from someone who has already finished the programs.

I hope to run a code plagiarism check at the end of the semester. Having conversations about copied code with students, and writing reports on academic integrity is one of the few parts of this job that I hate. Please don't put yourself, or me, into this position. There will be an assignment on Canvas to state that you understand the homework and program policy, and that you will not share code, nor use someone else's code.

Recording Lectures or Sharing Course Materials

You can make audio recordings of class for your own personal use. Maybe you want to want to have my dulcet tones lull you to sleep at night instead of only during class, that is fine. Weird, but fine. Perhaps you want to torture your neighbors by blasting it on your porch, that is not fine: aside from possible violations of the Geneva Convention, recordings should not be reproduced, distributed, or publicly broadcasted. If you want to make video recordings, please discuss it with me.

Course material developed by the instructor is the intellectual property of the instructor and cannot be shared publicly without his/her approval. You may not publicly share or upload instructor generated material for this course such as exam questions, lecture notes, or homework solutions without instructor consent.

Program Information

Diversity Statement - At SJSU, it is important to create a safe learning environment where we can explore, learn, and grow together. We strive to build a diverse, equitable, inclusive culture that values, encourages, and supports students from all backgrounds and experiences.

Course Goals

We will examine various ways to represent data used by programs and to compare these representations in terms of their memory requirements and the resulting program execution times.

Time will be spent learning algorithms and data structures, mathematical tools and techniques (recursion, recurrence relations) useful for their design and analysis, and seeing some examples of when they are needed.

Objectives:

- To ensure that students are familiar with ways to implement elementary data structures and their associated algorithms.
- To introduce students to the implementation of more complex data structures and their associated algorithms.
- To acquaint students with advanced sorting techniques.
- To teach students how to determine the time complexity of algorithms.
- To introduce students to algorithm design techniques.

Course Learning Outcomes (CLOs)

Upon successful completion of this course, students should be able to:

- Understand the implementation of lists, stacks, queues, search trees, heaps, union-find ADT, and graphs and be able to use these data structures in programs they design
- Prove basic properties of trees and graphs
- Perform breadth-first search and depth-first search on directed as well as undirected graphs
- Use advanced sorting techniques (heapsort, mergesort, quicksort)
- Determine the running time of an algorithm in terms of asymptotic notation
- Solve recurrence relations representing the running time of an algorithm designed using a divide-and-conquer strategy
- Understand the basic concept of NP-completeness and realize that they may not be able to efficiently solve all problems they encounter in their careers
- Understand algorithms designed using greedy, divide-and-conquer, and dynamic programming techniques

Course Materials

Introduction to Algorithms

Author: Cormen, Leiserson, Rivest, and Stein
Publisher: MIT Press
Edition: 3rd
ISBN: 0262033844
Availability: widely available

The ISBN-10 is given above, the ISBN-13 is 978-0262033848

This textbook is very widely used, and I hope it will come in handy beyond this course. The 3rd edition, for the material we cover, is quite similar to the 2nd edition. (The 2nd edition managed to obfuscate a few issues from the 1st edition while clarifying others.) I think the majority of changes from the 2nd to the 3rd edition are in sections we don't cover, though some of the exercises and readings have changed. I will try to post assignments for the 2nd, 3rd, and 4th editions of the book.

You can find errata (bug reports) for the book http://www.cs.dartmouth.edu/~thc/clrs-bugs/bugs-3e.php, for whichever printing of the book you get.

A 4th edition of the book was released in 2022. I am sure that it would be fine as well, I don't think it differs greatly for the material that we cover, though it might have a few sections that are organized differently. My impression is that students prefer to use the 3rd edition due to costs, if they choose to purchase the physical book.

Computer

You need to have access to a computer with Java and a development environment. Additionally, some days you will be expected to have access to some kind of computer/tablet/phone for communication during some classroom exercises, and to a wireless laptop, with the lockdown browser installed for exams.

Course Requirements and Assignments

The following will be regularly assigned for time outside of class:

- Video lectures
- Rote homework problems or video quizzes given in Canvas.
- Readings from textbook or handouts
- Canvas Quizzes covering other material (this has replaced written exercises of the past, but perhaps sometimes there will still be written questions for submission)
- 4 Programming assignments
- 2 Practice Exams

In addition to these, there will be three days of exams during the semester (each taking most or all of a class meeting), a final exam, and several meetings scheduled for in-person code review.

During the introduction of new material, homework is our chance to learn by making mistakes. It is expected that you will make an effort in all of the above for the sake of learning, and to give yourself feedback about your understanding the material.

The purpose of the rote homework is to give you enough practice working on problems to either understand how to solve the problems, or at least to learn from solutions to those problems.

Note: all homework due before drop date will be available until Friday of that wee, while the class enrollment is still being worked out. After that, homework will only be available until it is due. If you submit the homework, you will still be able to see it afterwards, but if you don't get around to looking at your

homework before its due date, you won't even be able to see it later for studying purposes.

Success in this course is based on the expectation that students will spend, for each unit of credit, a minimum of 45 hours over the length of the course (normally three hours per unit per week) for instruction, preparation/studying, or course related activities, including but not limited to internships, labs, and clinical practica. Other course structures will have equivalent workload expectations as described in the syllabus.

Videos and Readings: New Material

Almost all lectures on actual algorithms, data structures, and analysis techniques for them are covered on video, that you will watch at home. Readings in the textbook will also accompany these videos.

The videos are generally more dense than a class lecture would be, and if you just watch the video once, in real time, you will likely not absorb it all. I encourage you to watch each video closely and carefully, pausing and taking notes if needed, before attempting other homework based on that video. Some students prefer to watch the video through quickly once, to get the gist of it, and then to watch it again at a slower pace to try to catch details.

Canvas Quizzes: Rote Questions on Lectures

Most videos are accompanied by a quiz in Canvas. **The purpose of these quizzes is to give you feedback on whether or not you are watching the videos closely enough.** If you don't do well on your first attempt at these quizzes, it indicates that you have not watched the video closely enough.

Canvas quizzes on the videos can (usually) be taken up to 5 times, **but it is really your first attempt that will give you the most feedback on how well you understand the material.** Using what would be a terrible homework assignment to give an example of the difference between the first attempt and later attempts: imagine that you are told to study the first 15 digits of pi. After doing that, if you were asked "What digit is in the is the 10 millionths position?" and you could answer that, it would give some indication that you had memorized the requested digits. On the other hand, if you got it wrong, and then looked at the digits of pi before taking a second attempt to answer the question correctly after memorizing the single digit requested, it would give little confidence that you knew any other digits of pi. Hopefully, your homework will be much more interesting than that example.

If, even **after a second attempt** at a Canvas homework, you still cannot get the correct answers, you can ask a classmate or somebody else to help you. **You should not simply ask them for the answers. Getting the correct answers with no understanding doesn't help you in any way, as these homework assignments do not count towards your course grade.** You can, on the other hand, have them walk you through the problem, step by step, until you get to an understanding of the answer. (For this course, that counts as diligence. You should understand your submitted answers, otherwise you need to spend more time on them.) Because the homework has a time limit for submission, you might want to

take your 3rd attempt, if needed, to record the questions, so that you can do them with more time on your 4th attempt. You really shouldn't need a 5th attempt. The reason attempts are limited is to stop anybody from making blind submissions until they happen to get the correct answers.

Generally, these quizzes may involve calculations, but if you understand the material presented in the videos and readings, they should not be conceptually different. They are made to give you feedback into whether or not you understand the factual material presented in the videos and readings.

Other Canvas Quizzes: Replacement for Textbook Questions

Once upon a time, I assigned questions out of the textbook. During the pandemic, with remote teaching, the logistics of this were untenable, so I tried to convert these questions to Canvas format. Some converted well, others were a bit more awkward.

After each subchapter of the textbook, there are some questions related directly to that subchapter. I have created similar questions, sometimes of questionable quality, but in Canvas format. This will allow you to get instant grading/feedback from Canvas, instead of waiting for limited, delayed feedback from me. These questions may be a bit more challenging than the questions directly on the videos or readings, not only for content, but also just because the format of the answer might be a bit awkward.

The goal of these problems is never "to get the right answer". It is to learn the material. We learn by working on the problem. That being the case, you should not simply copy solutions, nor look for solutions (on the web or elsewhere). If needed, after trying (really trying, not just glancing) to do a problem, you can have somebody explain a problem to you in full, until you understand the solution.

Understand the material and get the correct answer? Fantastic. Think that you understand the material, but can't get the right answer? Let's discuss it in class to figure out what is going wrong. Frequently, working on a problem is what actually exposes some hole in our understanding, perhaps a small one that still stops us from working out solutions. Don't understand the material enough to even try the problem? That indicates that you should spend more time watching the video, reading, or maybe coming to office hours.

Programs

There will be 4 programming assignments for the semester, worth 2, 5, 4, and 2 points respectively, for 13 points total. **Different parts of different programs maybe have different deadlines.** Because of that, you cannot just leave the program until the final deadline, as it may be too late to earn points for prior parts of the program.

You are expected to meet with me, individually, for 5 separate code reviews during the semester. The first three reviews will be scheduled for approximately 15 minutes each, followed by a 10 and 5 minute review. For each of those 5 scheduled code reviews, if you schedule, and show up, and have

something to review (working or not), you will get...a point added to one of your in semester exams. (Those exams will probably be worth 80 points each, so if you get all 5 of these points, it is like getting a 6.25% bonus on one in-semester exam.)

You are expected to work on all programs, to submit them, and to be able to explain them to me. At your individual code review, we will discuss your code. For the semester, four regular class meetings will actually be canceled, to make more time for code review. During the week of that cancelled class, you will schedule an individual code review with me, as well as scheduling one last, shorter code review (without any cancelled class for that last one).

I do understand that meeting me for individual code review might not be convenient, time-wise, for some of you. However, you are replacing four 75 minute lectures with 5 code reviews totaling one hour in time, so I don't think many students will be "losing time" in this deal. It looks different from my perspective: I will be cancelling 10 hours of lectures (4 lectures for each of 2 sections), and replacing those 10 hours with over 90 hours of time reserved for individual meetings. Those meetings are...not convenient, but I have found them to be valuable enough that I think it is worth the time.

For some programs, if you are able to program enough of the work, and explain it to me, you might still get full or near-full credit for the program, even if it does not answer every test case. A program that works perfectly that you cannot explain will not get credit, and likely indicates a violation of academic honesty policies. (Please see the Program Collaboration Policy above.)

Don't have LLMs or AI write your code for you. Maybe I will catch it, maybe I won't. If you meet me to talk about your code, and don't understand what you have written? It will frequently indicate to me that you didn't write your code, and I will need to write an academic integrity report. Last semester, for one of the programs, LLMs were able to give a solution...but that solution was pretty strange, in a particular way, and unless they have improved...I will recognize it when if I see it again this semester. In the end...don't you want to be able to understand how to program? **At some point soon, in your career as a programmer, AI will likely be a useful tool to make you more productive at your job. In this class, it isn't your job to write working programs, it is your job to learn how to write working programs. With the programs that I assign, AI will hurt that goal, not help it.**

✔ Grading Information

Criteria

For your exam grade, I will take **the maximum** of either your combined exams, or your final exam, to be your exam score. Because of that, until the final is done, it is never too late to pass the course...the final can always be used as a lifeline.

That grade will then be modified by your programs, which make up one letter grade of the course.

Type	Weight	Topic	Notes

Type	Weight	Topic	Notes
Homework	0		Homework is for learning material, and for giving you feedback. Grading homework in the traditional sense encourages students to focus on grades more than learning the material, which is not considered to be the best way to learn. Because of this, for this class, an individual's homework scores will not have a direct effect on their grade as an individual. They will at most have a small, indirect effect, as described below. Of course, proper effort on homework by an individual will likely result in higher exam scores for that individual. Instead, homework scores may be used, slightly, to set the curve for the exams. For instance, if the bottom half of the class (as measured by the in-semester exams) is generally doing their homework, and I have two different reasonable values that I can use as a cut-off between different grades, it will push me towards the more generous cut-off. It will not greatly shift any cut-off. I do **not** believe in assigning homework for the sake of assigning homework, but I do assign a lot of homework. If you understand the material without doing the homework, and the homework seems like a waste of time to you? In that case, your advanced understanding should be reflected by your performance on the exams. That is why I mention only checking the bottom half of the class in the prior paragraph: if you understand the material and do well on the exams, I am not so concerned about whether or not you did the homework, so your lack of homework won't make the curve more difficult for anybody else. You are encouraged to help your classmates to learn the material, teaching it to them will help you to understand it even more. Note, **this policy should highly discourage students from cheating on homework:** you should not be copying homework assignments from anyone else, as it is unlikely they will help your grade: your homework is one of many used to set the curve. We can have open discussions about homework (even working towards solutions, if posted with spoiler tags) before they are due, and full solutions after the due date. Due to the way that Canvas homework (multiple attempts) and written homework (graded on effort) are graded, the students making a real effort on homework should, as a whole, expect to get credit for a high percentage (90%?) of homework problems. Why should you do the homework? To learn the material. **Do not interpret homework to be optional. It is mandatory. I have read plenty of student evaluations of me that complain that it is my fault that the student didn't do the homework, because it was optional, and that is why they did poorly on the exams.** Just because the homework doesn't happen to be part of your weighted course grade, that doesn't mean that is it optional. Musicians need to practice when they aren't performing, athletes need to train when they aren't competing, and you need to study this material even when it isn't graded.

Type	Weight	Topic	Notes
In Semester Exams	20	worth 20% each, for each of 3 exams	There will be **three** in-class tests during the semester, worth 20% of your exam score each. Exam dates can be found on the schedule below. **You are expected to have a laptop computer, with the lockdown browser installed, that you can use for the exams.** (I don't yet know if they will be on computer or on paper.) You will get one letter grade for the 3 combined exam scores. Although the grade cut-offs may change as the exams change, the grade cut-offs should be no more stringent than: 85%+ A plus (entered as 96.67) 80-85% A (entered as 93.33) 75-80% A minus (entered as 90) 70-75% B plus (entered as 86.67) 65-70% B (entered as 83.33) 60-65% B minus (entered as 80) 55-60% C plus (entered as 76.67) 50-55% C (entered as 73.33) 45-50% C minus (entered as 70) 40-45% D plus (entered as 66.67) 35-40% D (entered as 63.33) 30-35% D minus (entered as 60) -30% F (entered as 50) (Grades on the cut-off get the next higher grade.)

Type	Weight	Topic	Notes
Final Exam	40%		Your final exam will be comprehensive, and worth 40% of your exam score. **If your final exam score is higher than your weighted exam score, your final exam will be used as your entire exam score.** So, you will get the maximum of a weighted exam average, or just your final exam. **You are expected to have a laptop computer, with the lockdown browser installed, that you can use for the final.** Although the grade cut-offs may change as the exams change, the grade cut-offs should be no more stringent than: 75%+ A plus (entered as 96.67) 70-75% A (entered as 93.33) 65-70% A minus (entered as 90) 60-65% B plus (entered as 86.67) 55-60% B (entered as 83.33) 50-55% B minus (entered as 80) 45-50% C plus (entered as 76.67) 40-45% C (entered as 73.33) 35-40% C minus (entered as 70) 30-35% D plus (entered as 66.67) 25-30% D (entered as 63.33) 20-25% D minus (entered as 60) -20% F (entered as 50)

Type	Weight	Topic	Notes
Programs	1 letter grade	bonuses for scheduled code reviews	Your programs make up one letter grade of the course...if you skip all programs, your course grade will be one letter grade lower than your exam scores. In Canvas, it will look a bit strange: the 4 programs will be worth 13 points in all (2, 5, 4, and 2 points respectively), but will be scored out of 0 on Canvas. For your programming grade, you will start with -10 points. If you earn at least 10 out of the 13 programming points, this will get you up **to** a score of 0, which won't change your test grade. (And, if you score all 13 points, you will still not move up to the next grade.) If you score less than 10, but at least 6.67 out of 13 points, it will cost you 1/3 of a letter grade from your test grade. For example, if you score 7/13 points, that ends up being -3 points added to your test score, which will lower it, for instance from a B to a B-. Similarly, if you score at least 3.34 points, but fewer than 6.67, it will lower your test grade by 2/3 of a letter grade, and if you fail to get 3.33 points or fewer, it will lower your test grade by 1 letter grade. Additionally, for each of the 5 scheduled code reviews: if you schedule, and show up, and have something to review (working or not), you will get 1 point added to your in-semester test scores, so doing all 5 reviews will get you 5 points which is probably worth 6.25% of one exam.

Breakdown

For each student, I will calculate an exam score, based on the maximum of their weighted exams (3 exams with program bonus points for reviews performed and a final) or just their final. That score will be modified by programs, resulting in a loss of up to one grade for incomplete program work.

University Policies

Per University Policy S16-9 (PDF) (http://www.sjsu.edu/senate/docs/S16-9.pdf), relevant university policy concerning all courses, such as student responsibilities, academic integrity, accommodations, dropping and adding, consent for recording of class, etc. and available student services (e.g. learning assistance, counseling, and other resources) are listed on the Syllabus Information (https://www.sjsu.edu/curriculum/courses/syllabus-info.php) web page. Make sure to visit this page to review and be aware of these university policies and resources.

Course Schedule

The schedule below is fairly firm, but may change if needed. Those changes will be announced.

Date Subject to change	Planned Topic	Notes (Some more detail will be filled in here after material is covered.)
January 26	Introductions and Administrivia	Maybe a lot of slides. Sorry.
January 28	2 Warm Up problems, One Harder Problem	**Program 1 part 1 out.** This is earlier than it should be within the semester, but if someone is in the class who doesn't understand Java, it is best to discover that as early as possible.
February 2	**Program 1: Code Review**	**No class today:** individual code review by appointment (through canvas) for everyone some time this week (Monday to Friday).
February 4	Defining Problems and Loop Invariants	
February 9	Asymptotic Notation Exercise	Program 1 part 2 out
February 11	Recurrence Relations, Recursion Exercise	
February 16	Master Theorem or Sorting Questions? Recurrence Exercise	
February 18	Quicksort/select Questions? Counting Exercise, Unbounded search? Sky?	Program 2 out

February 23	**Program 2: 1st Code Review**	**No class today:** individual code review by appointment for everyone in the class, some time this week, the earlier the better.
February 25	lower bounds by reduction Exercise (Heels? Sky?)	
March 2	**Exam 1** sorting (insertion, merge, quick, heap, linear, lower bounds), invariants, heaps, recurrences	
March 4	**Program 2: 2nd Code Review**	**No class today:** individual code review by appointment for everyone in the class, some time this week, the earlier the better.
March 9	Exam return, lower bounds by reduction	
March 11	Graph Exercise	
March 16	Graph Exercise	
March 18	Graph Exercises	
March 23	Graph Exercise + my 2nd program?	
March 25	Exercise (Z1) Efficiency Exercise	
April 6	**Program 3: Code Review**	**No class today:** individual code review by appointment for everyone in the class, some time this week, the earlier the better.

April 8	**Exam 2** (Graphs, 23-Trees, lower bounds)	
April 13	Exams Returned	
April 15	Exercise (Z2)	
April 20	Graph Counting Exercise	Program 4 out (NP)
April 22	NP decision vs. Opt questions? SS is NP. Efficiency Exercise	
April 27	TBD	Sometime this week, schedule a short code review. (No class cancellation.)
April 29	Exercise	
May 4	Advanced Topics?	
May 6	**Exam 3** DP + NP + FW + ?	
May 11	Exams returned. Review.	
Friday, **May 15,** **8:30-10:30**	**Section 2:** **Final Exam**	

Wednesday, May 13, 10:45-12:45	Section 3: Final Exam	