# Syllabus - COMP 120 – Section 2 (Spring 2021)

## Programming abstractions and methodologies

## Course Instructor

**Dr. Tom Yeh**

- **Office**: Gather Town
- **Email**[1]: tyeh at sandiego dot edu

## Zoom Links

Since we are remote again this spring, our meetings will be conducted via zoom. The links for lecture, lab, and office hours are all posted on blackboard, under the "Zoom links" tab. Note that you will have to join our zoom meetings by logging in to zoom through your USD account.

## Class Meeting Times

| | |
|---|---|
| **Lecture:** | Tuesday, Thursday<br>9:15 am - 10:35 am |
| **Lab** | Thursday<br>2:30pm - 3:50pm |

## Office Hours

**Location**: Gather Town

| | |
|---|---|
| **Tuesday** | 10:35 am - 11:00 am<br>1:40 pm - 2:15 pm |
| **Thursday** | 10:35 am - 11:00 am<br>3:50 pm – 4:25pm |

[1] I highly prefer to communicate via Campuswire. You are much more likely to quickly get an answer if you send questions through Campuswire. Read onward for details about Campuswire!

## Course Description

A continued exploration of computational problem solving, with a focus on using abstraction to manage program complexity. Students will learn to use both functional and data abstractions, analyze the time and space complexity of algorithms, and utilize functional, object-oriented, and event-driven paradigms within their programs. (3 hours lecture, 1.5 hours lab per week.)

## Course Learning Outcomes

At the end of this course you should be able to do the following.

1. Use abstraction to build programs to solve more complex problems (abstract data types, higher order functions, use of existing libraries)
2. Design and implement computer programs that use object-oriented, functional, and event-driven paradigms.
3. Evaluate tradeoffs between different algorithms for solving a problem (space and time tradeoffs, big O notation, searching and sorting algorithms as examples).
4. Design and implement an abstract data type.
5. Use modern tools for software development and maintenance (IDE, debugger, source code management).

## Computer Science Student Outcomes Addressed by this course

| | **Student Outcome** | **Addressed by Course Outcome(s)** | **PO Assessment** |
|---|---|---|---|
| 1 | Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. | 1-4 | |
| 2 | Design, implement, and evaluate a computing-based solution to meet a given set of computing requirements in the context of the program's discipline. | 1-4 | |
| 3 | Communicate effectively in a variety of professional contexts. | | |
| 4 | Recognize professional responsibilities and make informed judgments in computing practice based on legal and ethical principles. | | |

| | | | |
|---|---|---|---|
| 5 | Function effectively as a member or leader of a team engaged in activities appropriate to the program's discipline. | | |

## Tentative Schedule of Course Topics

Below is a list of topics that I would like to cover during the course. It is tentative and subject to shortening based on how we progress throughout the semester.

- Basic Python review
- Python exception handling
- Basic abstract data types: lists, stacks, and queues
- Basic algorithmic analysis
- Modeling Hierarchy with Trees
- Recursive Backtracking
- Implementing a Dictionary with Search Trees
- Implementing a Dictionary with a Hash Table
- GUI / Event-driven programming
- Functional Programming with Lists (Map / Filter / Fold)
- Discrete Event Simulation with Priority Queues
- Searching and sorting algorithms
- Implementing Backtracking with Stacks

## Online Course Components

**Course Website**: https://ole.sandiego.edu/ (Blackboard Learn)

We will use Blackboard for distributing course information, assignment submissions, and keeping track of grades. Log in using your MySanDiego login name and password.

**Course Shared Google Folder:**

**https://drive.google.com/drive/folders/1Whc_zzKtAWrofZD7TxpVySK-8720ccRY?usp=sharing**

We will use the shared folder to distribute assignments, handouts, lecture notes, lecture code, etc. The syllabus and tool installation directions are also here.

**Textbook Website:**

This course features a digital, interactive textbook, which is described in the section below on required materials for the course.

**Discussion Site**: https://campuswire.com/p/GBDE54C8D (Campuswire). (You will need the code 2659 to sign up for this class.)

Campuswire enables everyone in class to quickly ask and answer questions. Please make Campuswire your first stop for asking all class-related questions: it is much faster than waiting for me to reply to your e-mails and there is no better feeling than having the instructor endorse one of your answers to another student's question.

The more that everyone uses Campuswire, the more useful it becomes so please use it regularly. You may be intimidated to post to online forums, especially if you are afraid that your question is "silly." Have no fear though: you won't be judged as there are no silly questions. If you are still nervous, Campuswire has you covered: you can post without your classmates knowing who posted (I can still tell).

Campuswire makes it much easier for me to keep track of "open problems" and therefore I highly prefer all course communication to go through there instead of e-mail. Please don't be offended if I redirect your e-mail questions to Campuswire: most questions are of interest to your classmates as well. If you don't want anyone but me to see your question, Campuswire allows you to make questions/notes as private so that only I can see them.

Follow the above link to create a Campuswire account and join the class.

Neither I nor your classmates will tolerate abusive online behavior so please keep your discussion civil.

## Required Textbook

This course features a digital, interactive textbook from Pearson's Revel platform. The title is "Revel Introduction to Python Programming and Data Structures," by Y. Daniel Liang, ISBN 9780135187753. Here is the link to join the class: https://console.pearson.com/enrollment/cbxo3v. This link is specific to our class, and you must join this class to get credit for some of the programming exercises that I assign from the book.

You will have to pay for access to the book, but you can start off by getting 14-day temporary access without paying. Then, you can get an access card from USD's bookstore, or (more likely given that we are remote this semester), you can also buy access through Pearson.

Pearson tells me that when you pay for online access to the book, you will be given the option of purchasing a loose leaf hardcopy of the book for $24.99. Also, Liang has a book "Introduction to Programming Using Python" that you might see on Amazon for example, but this is not the same as the book we are using, so don't buy it. It does not have the chapters on data structures that we will be covering.

Here is a link to the Revel Student Getting Started Guide.

## Grading

The university forces me to give out grades at the end of the semester. I would like to make the grading process as painless as possible and as such I have tried to develop a grading system that is both fair and transparent.

Your final grade will be a weighted average of the components listed below. Each of these components will be described in more detail throughout the rest of this document. Please make sure you read them carefully. If anything is unclear, please feel free to ask me.

- **Problem Solving (programming) Assignments**: 30%
- **Final Exam**: 25%
- **Midterm Exam**: 25%
- **Revel (digital textbook) programming exercises**: 10%
- **In class quizzes**: 10%

As much as I hate to admit it I do occasionally make grading mistakes. If you happen upon such an error (e.g. incorrect or missing grade), let me know in writing within a week of posting and I will fix the error. After one week the grade will become *immutable*.

## Revel Reading Assignments

At least once a week I will make reading assignments in the textbook. The textbook is interactive, and so you will have opportunities to trace the execution of code, fill in missing code, and answer multiple choice and short answer questions. The due dates for the reading assignments are such that you should have read the textbook material prior to our discussion of it in class. In this way you should come to class with a basic understanding of the material so that we can focus on harder concepts in class. I expect you to have completed them prior to their due date. You will get much more out of the class if you come to class having completed the reading assignments. Note that you are not graded on completing the reading assignments, but I can see how much time you have spent on each section of the textbook.

## Revel Programming Exercises

The programming exercises in the textbook tend to be shorter than the larger PSAs that I will assign to you, and they are targeted to the material that we have just gone over in class. These are individual programming assignments that you will work on on your own, without a programming partner. To get credit for the Revel programming exercises, you must do them while logged into the Revel course for this class. The due dates for these exercises will be after we have covered the material in class. To get credit for a programming exercise, the code you submit must pass the automated tests that Revel performs on your code. The nice thing is that you have an unlimited number of attempts to get each program correct.

## In-class Quizzes

We will have in-class written quizzes approximately once per week. You will take these quizzes on Blackboard, and they may include multiple choice, true/false, short answer, and code writing questions. The quizzes will be announced one class meeting before they occur. The quizzes will be over material that we have covered up to and including the previous class meeting.

At the end of the semester I will drop your two lowest quiz scores. Each quiz will count equally toward your grade, although the quizzes may have differing numbers of points on them.

## Problem Solving Assignments (PSAs)

You will be given one programming assignment every week or two, which will be posted on Blackboard. You will do your programming with a pair programming partner, using the pair programming software development technique (see next section). For each assignment, each team will submit just one program, and it must be submitted by the due date and time. If you submit after the due date/time, there will be a 10% of possible points penalty per class that it is late, up to five classes late. After this, you will receive no points for the assignment. Each team must follow the pair programming development technique. This means that you must meet with your partner to do the assignments. When assignments are due, you should find at least 6 hours per week when you can meet and work together. (So, do not split up the work of an assignment, or trade off doing assignments.)

To receive any credit for your program, it must pass all of the automated tests which I provide with the assignment. (Not all assignments will have an automated test, in which case this does not apply.) If it fails to pass all tests, you will not receive any credit for the program. But since you are provided with the tests, there is no reason for you to submit a program that does not pass them.

If your program does pass all tests, then points can be deducted for failing to follow the programming guidelines that I will provide to you.

Each programming team is required to work on the programming assignments independently of any outside sources. You can read books, surf the web, or talk to your friends and other computer science students to get help understanding the concepts you need to know to solve your programming assignment problems. However, you can only work with your assigned partner on programming assignments. You must work together on every assignment, not dividing up the work, but working jointly and each understanding what solution you have produced and how and why it works.

The objectives of this course are about skills as much as they are about knowledge. Future instructors, employers, and colleagues will expect that you will have attained a certain proficiency in problem solving and programming. Using unauthorized aids in doing your work will prevent you from attaining the proficiencies that others will expect. Meeting or not meeting these expectations relate directly to getting internships, retaining employment, and success in your future studies.

In this course, using program code that someone else has written (unless it was explicitly provided as part of the assignment), or providing program code to someone else, or turning in code that you have written with someone else other than your partner, is considered

cheating. To ensure you don't have a problem with this, here are some rules to follow (in this discussion I am not talking about your assigned partner):

1. Don't even look at or discuss another student's code for a programming assignment you are working on, and don't let another student look at your code.
2. Don't start with someone else's code and make changes to it, or in any way share code with other students.
3. If you are talking to another student about the assignment, don't take notes, and wait an hour afterward before you write any code.
4. It is possible that someone from outside the class will provide help to you on your programming assignments (a tutor or a friend). That help should be limited to helping you and your partner understand the concepts underlying the assignment, and general guidance in how to approach developing your program. They should not write any code for you.
5. If you have a program, but there are compiler or runtime errors in it, it is ok for a tutor to look at your code for purposes of identifying the problems, but they should not change your code for you. They should just point you to parts of your code that are causing the problems.

I will assign pair programming partners.

## Pair Programming

Creating software programs is a complex task that is typically performed by teams of developers. Different methodologies have been devised to manage the software development process. One such technique is called pair programming.
In pair programming, as the name suggests, two programmers work together at a single computer. One of them types in code while the other reviews each line of code as it is entered. The programmer doing the typing is called the driver, and the programmer doing the reviewing is called the navigator. The two programmers trade roles periodically (for you, around every 15 minutes). Besides reviewing the work of the driver, the navigator is responsible for the overall design of the program being written, and looks for ways to improve it, and watches for potential problems. This allows the driver to concentrate on getting the current piece of code correct, knowing that the navigator is watching for problems and guiding his/her work.

Here is a somewhat cheesy video that provides an overview on pair programming, including some tips for making it work effectively for you.

Teamwork is a critical skill on which your future employers will evaluate you. You will greatly benefit if you work hard now on learning how to become a good teammate. Being a good teammate includes always showing up on time for planned meetings and adequately preparing for your meetings. Working with others can be challenging; if an issue arises, you should first bring up the issue with your partner. If that fails to resolve the problem, please contact the instructor. Don't put off addressing these issues; often a heartfelt talk with your partner as soon as the issue arises can avert many headaches later.

## Midterm Exam

There will be one midterm exam, on Thursday, March 11, during the regular class meeting. It will cover all the assigned reading, lecture material, and assignments up to that point in the class. It will be closed book and closed note, no electronics (including cell phones and calculators) allowed, and you cannot receive any kind of external help on a test. You will take the midterm exam on Blackboard using the Respondus lockdown browser and monitor. (More on this closer to the exam date.)

A serious illness, a death in the family, and other traumatic events are unfortunately sometimes a part of life. If you contact me within 24 hours with acceptable documentation, I will re-weight the final exam component of your grade to compensate for the missed exam. Otherwise, the midterm exam must be taken at its scheduled day and time.

## Final Exam

The final exam will cover all the assigned reading, the lecture materials, and the assignments from the semester. You must take the final during the scheduled time: Tuesday, May 18, from 11 AM – 1 PM.

As with the midterm, it will be closed book and closed note, no electronics (including cell phones and calculators) allowed, and you cannot receive any kind of external help on a test. Also as with the midterm, if you have a serious illness, a death in the family, or other traumatic event, contact me within 24 hours with documentation, I will make an arrangement to make up for your missed final. Note that conflicting travel plans are not an excuse for taking the final at another time. Put the final on your calendar now.

## Grading Scale

Your final grade will be based upon a weighted average of the individual course grading components (i.e. participation, quizzes, exams, etc.). I do not grade on a strict scale, but you will do no worse than the scale shown below for your final, weighted average. One important exception is that you must pass the final exam (i.e. get 55% or higher) in order to get a C- or better in the class.

| | |
|---|---|
| [93 - 100] | A |
| [90 - 93) | A- |
| [87 - 90) | B+ |
| [83 - 87) | B |
| [80 - 83) | B- |
| [76 - 80) | C+ |
| [69 - 76) | C |
| [65 - 69) | C- |

[55 - 65) D
[0 - 55) F

Note that the upper end of each range is non-inclusive. For example, 90% would be considered an A-, not a B+.

If you are taking the pass/fail option, you must receive at least a C- to pass.

## Academic Integrity

The Computer Science Department strongly promotes academic integrity. There have been cases in the past of students copying someone else's work in programming courses or using code from the web or other books. This is considered plagiarism and is a major violation of academic integrity.

When working on an assignment, you are allowed to discuss the problem and the possible ways of solving it, but what you submit for grading must be the result of your (and your partner's) own work and based on your understanding. No discussion with anyone other than me is allowed on quizzes or exams. The possible consequences of academic integrity violation include, but are not limited to: a score of 0 on the given assignment and lowering the course grade or assigning an F in the course.

Please don't assume you can cheat the system by using someone else's code then modifying it so that it *appears* different. Every submission is run through sophisticated plagiarism detection software that catches even the most subtle forms of code sharing.

I'll end on a personal note. It's no fun when I have to report academic integrity violations—and I do report them—so please do us both a favor and don't cheat.

## Asynchronous Class Attendance

I realize that some of you may have circumstances that do not allow you to attend class (lecture or lab) via Zoom during its scheduled time (e.g. intermittent Internet connection or residing in radically different time zone). If you do have circumstances that prevent you to attend synchronously, you should contact me at the start of the semester to explain your situation, and you will be allowed to attend asynchronously (and so take the quizzes and midterm at other times).

To facilitate asynchronous attendance I will record our class meetings and make the recordings available on Blackboard. I will record only that part of our class meetings where I am lecturing, or there is a class wide discussion. If, for example, during lab students are in Zoom breakout rooms working together, I will not record that.

## Using Zoom

1. Leave your microphone off unless you are speaking.
2. Turn on your video, so that I can tell you are "present" in class. (Zoom allows you to set up a background image so that your surroundings are not seen by the rest of us.) That said, if you have just medium fast Internet, then you can turn off your video and watch the screencast; and if you have slow Internet you can turn off your video, use your phone for audio, and watch the screencast; and if you have unreliable Internet, you can attend class asynchronously (see section above).

## Disabilities and Learning Disabilities

Many students have disabilities or learning differences. It is my goal to make sure those people feel fully supported in this class. If you need special accommodations because of one of these, please reach out the Disabilities and Learning Resource Center at the beginning of the semester. They have many resources available to help manage your disability and/or learning difference. Most notably they can provide official documentation of your needs so that I can provide the appropriate resources to help you succeed in this class.

## Additional Information

The last day to select the pass/fail option is Wednesday, April 7. The last day to withdraw from the course *without* a W is Wednesday, February 3. The last day to withdraw from the course is Friday, April 9.

A grade of incomplete will be assigned only if there is a serious, documented reason that prevents you from completing the requirements of the course. Getting a low grade or falling behind is not a sufficient reason.

The only exceptions to the rules regarding no late assignments are extended absences (one week or more) due to verifiable extraordinary circumstances, and absences due to official USD activity travel. In the case of absences due to a USD activity travel, you must give me as soon as it is available a list of your travel dates.

This syllabus is subject to change during the semester. The instructor will make announcements about any non-trivial changes within a day of the changes.