

FAQ: CS/Math 220 policies & structures

View this document on Google Docs at <https://bit.ly/33l1yaD>

How will the end of the semester go?

Is there any final exam or other summative assessment?

There is no final exam or other summative assessment - except for an optional reflective essay. Instead, you will have the time you would have spent studying and taking the exam to complete Learning Targets and/or Challenge Problems.

What work do I need to complete to get a particular grade?

See "[How will final grades be assigned?](#)" I will aim to update each of you on where you stand around the last day of classes.

What's the final deadline to submit work for this course?

All work is due by the end of day Saturday, December 5 (the end of the semester). I do not promise feedback on work submitted after 11 am Wednesday, December 2 (the end of our officially scheduled final exam period), unless you make particular arrangements with me.

What help will be available?

Join me on Zoom for co-working / drop-in help:

- During my usual office hour on the last day of class (2-3:30 pm, Tuesday, November 24);
- Any time on the day before Thanksgiving (9-5 pm, Wednesday, November 25);
- Any time on Reading Day (9-5 pm, Monday, November 30);
- During our final exam period (9-11 am, Wednesday, December 2).

If I'm not on Zoom during any of those scheduled times, send me a quick private message on Slack!

I will continue to be available for appointments through Friday, November 4.
Our class mentor Nick may also schedule additional office hours, TBA.

How will you teach online?

Will the course be synchronous or asynchronous?

I will meet with small groups of 4-5 students three times a week, mostly during our regularly scheduled class time, so that everyone has the opportunity to work closely with me on solving problems and understanding concepts.

While you will have the remainder of class time available to work with your group, I'll ask you about your availability to meet with your group outside of class time. I will also ask you about your preferences for working on your own versus with others, so that I can group those with similar preferences.

So that our time together can focus on collaborative work and discussion, I will use recorded videos for "sneak previews," mini-lectures, guided readings, and sample problem solutions.

When will work be due?

Practice problems should be completed, or at least attempted, before our regularly scheduled group meetings during class time. All summative assessments will be due on Fridays. All work will be due on Wednesday, December 2, the date scheduled for our final exam.

Will I need to use my computer to write code?

Not really. Programming in this class will be in the academic functional programming language SML/NJ, which is taught by our textbook. Although you can [install SML/NJ](#) on your own computer if you want, there's no particular need to.

You'll write your code in a text editor of your choice. Then you have three options for how to run your code in the SML interpreter:

1. Use this [online SML/NJ interpreter provided by TutorialsPoint](#).
2. Log in to a department workstation using ssh (see the [lab manual](#) for instructions) or remote desktop software TBD;
3. Install SML on your own computer.

More details are on Canvas.

How will we do problems together?

If you have an iPad, Microsoft Surface, or other tablet, you can use it for this class. That's what I'll be using. I'll be exploring the whiteboard feature in Zoom, and other possible apps, to make a recommendation for collaborative group work.

If you don't have a tablet computer, that's fine! Prof Bares has had success with students writing on a small portable whiteboard and holding it up to the camera in Zoom. Plain paper could also work, but you need to write with a marker so that the lines are thick enough to be easily seen.

Here is the small portable whiteboard that Prof Bares most recommends:

- Fold-out whiteboard, 26" x 22" folding down to 13" x 11", dry erase with 1-inch grid lines on one side and 1-inch hexagons on the other side - [\\$15 from Amazon.com](#)

What will we learn in this course?

Would you be able to send out an old or in process syllabus to get a better idea of the material?

The Canvas course is published. And here is a [tentative plan for the semester](#).

What textbook will we use? Is it available online?

The textbook is [Discrete Mathematics and Functional Programming](#) by Thomas VanDrunen of Wheaton College (incidentally, one of Prof Gray's former colleagues). You can [order an eBook or hardcopy through the College Bookstore](#).

This is a first-edition textbook. The [author's web site](#) provides a [list of errata](#). I encourage you to go through the book and mark these errata before we get too far: the first two errata - both reported by Whitman students - are on page 48.; I will collect these errata and forward them to the author.

The author also provides [source code for textbook examples and exercises](#).

Readings fit into five categories:

Read carefully means I will not cover the material in class at all. It's background for what we really want to work on. You solely are responsible for it.

Read means I will highlight and review the main ideas but not lecture on them. I will assume you have already seen these ideas in the reading. Usually, I will assign a few small exercises from these sections; we will review the exercises and work through further problems in class.

Skim means this is difficult material that most students will need to see at least twice to understand. Familiarize yourself with the reading before class, and then

we will work through the reading together in class. Note that **skim** does not mean skip.

Review means we have already discussed this material, but you will benefit from revisiting it.

Optionally read means just that: A suggestion to read material that you might find interesting, but is not required. We generally won't discuss this material in class.

How will you assess our learning?

What's the overall approach to grading?

My new grading policies will be guided by Joe Feldman's (2019) [Grading for Equity](#). The premise of the approach is that traditional points-based grading schemes are inaccurate, unfair, and demotivating. Some key ideas for this course:

- Your grades will depend only on summative assessments designed to evaluate what you have already learned. Your practice work will not be graded.
- If “summative assessments” sound scary, please don't be scared! Summative assessments will consist of small, regular assignments (see below). You will have the opportunity to redo/retake/revise every summative assessment except the final one.
- Grades will be based only on mastery of course content. Attendance will not count towards your grade, and there will be no extra credit. Although there will be deadlines for assignments, there will be no late penalties. You will suffer the natural consequences of embarrassment, nagging, and still having the work left to do. If you fall more than a week behind, we'll have a conversation about it. And if you disappear, I'll alert the Student Affairs office so that someone can check on your safety and well-being.

What will our assignments be like?

There will be ungraded Practice Problems assigned for every group meeting. They will be about $\frac{2}{3}$ math problems and $\frac{1}{3}$ programming problems over the course of the semester. To prepare for the Practice Problems, you will be assigned textbook sections to read and sometimes short videos to watch. During group meetings, we will review or solve Practice Problems together and discuss any conceptual difficulties.

There will be four forms of summative assessment, informed by [Robert Talbert's work on Specifications Grading](#):

1. Learning Targets will address basic skills and knowledge. They will be assessed through short quizzes or by submitting code or written problems on Canvas. Learning Targets will be assessed on a binary scale: *Pass* or *Redo*. You can redo Learning Targets as many times as you like. If you find you need a different format to demonstrate what you have

learned, you may schedule up to two, 15-minute appointments per week for oral assessment of Learning Targets.

2. Challenge Problems will provide opportunities to apply and integrate the basic skills and knowledge assessed in the Learning Targets. I'll assign between 1 and 3 Challenge Problems each week, depending on the topics of that week. You may revise until you are satisfied with your work, but you need not complete every one (even to earn an A). You will present your work on Challenge Problems in writing, but you may also be called upon to make an appointment to discuss your work. Challenge Problems will be graded on a scale of *Exemplary*, *Meets expectations*, *Revision needed*, *Not assessable*.
3. The Programming Project will let you apply and integrate the basic programming skills assessed in the Learning Targets. The Programming Project will address a subject of your choice; it will be completed in stages throughout the semester. Revisions and assessment will be similar to the Challenge Problems.
4. There will be an optional Final Reflection in which you will synthesize what you have learned. You will not be permitted to revise the Final Reflection. It will be graded as *Exemplary* or *Not exemplary*.

How will final grades be assigned?

Here is the conceptual framework for assigning final grades:

Base Grade	Mathematical skills	Programming skills
A	Write more complex proofs using multiple techniques and new definitions. (Challenge problems)	Solve complex problems with SML. (Challenge problems)
B	Demonstrate facility with basic proof techniques used in isolation or straightforward combinations. (DM3)	Use let to define efficient recursive helper functions. Implement and apply higher-order functions. (FP3)
C	Demonstrate mastery of Boolean logic and formal reasoning. (DM2)	Build on fundamentals to model mathematical concepts in SML. (FP2)
D	Demonstrate mastery of set notation, operations, and concepts. (DM1)	Define functions over fundamental data types, including recursive functions over integers and lists. (FP1)

And here is an operational schema:

Base Grade	
A	Meet the standards for a B, and complete an additional 8 Challenge Problems for a

	total of 16.
B	Complete 31 Learning Targets covering [DM1-3,FP1-3], and at least 8 Challenge Problems.
C	Complete 19 Learning Targets covering [DM1-2,FP1-2], and at least 4 Challenge Problems.
D	Complete 11 Learning Targets covering DM1 and FP1 learning outcomes.

“Plus” grades may be earned at my discretion through Exemplary work on the Project, Challenge Problems, or Final Reflection.

“Minus” grades may be earned by a “Plus” at the previous grade level plus partial achievement of the requirements for the corresponding base grade.

For example:

- You can earn a C by demonstrating mastery of set notation and operations, Boolean logic, and formal reasoning, working with models of mathematical concepts in SML, and completing 4 Challenge Problems.
- If some of this work is exemplary, or if you write an exemplary final reflection, that C would be bumped up to a C+.
- If you have mastered some proof techniques and shown how to implement a higher-order function, that C+ could be bumped up to a B-.

What are your expectations for academic integrity?

I expect you to attempt each Practice Problem on your own before each study group meeting. Then you will bring your solutions, work in progress, questions, and challenges for the group to consider together.

You may discuss Learning Targets, Challenge Problems, the Programming Project, and the Final Reflection with me, your classmates, or our class mentor. To help you present your ideas clearly in your Final Reflection, you may also consult a COWS tutor or a friend who has not taken CS/Math 220. However, you should write up your work after completing your discussion, and you must not share complete solutions to Learning Targets or Challenge Problems. I expect you to submit your own work and provide attribution of any help you receive.

I encourage you to publish your Programming Project as part of your professional portfolio, for example on GitHub. However, out of respect to others teaching and learning from our textbook, do not publish solutions to Practice Problems, Learning Targets, or Challenge Problems.

For additional context, please review the [Academic Integrity Guidelines for Computer Science](#) at Whitman College.

How can I get help?

How can I get help outside of our regularly scheduled meetings?

There are many options for getting help:

- Ask questions of general interest in the #cs_220 channel on Slack;
- Consult with your assigned study group;
- Join a class mentor session on Zoom;
- Join my office hour on Zoom;
- Make an appointment using [this online tool](#);
- Send me an email, or preferably a message on Canvas, to address private matters such as questions about your standing in the course.

How can I get accommodations for a disability?

More or less as usual as usual. If you are a student with a disability who will need accommodations in this course, please contact [Antonia Keithahn](#), Assistant Director of Disability Support Services, for assistance in developing a plan to address your academic needs. All information about disabilities is considered private; if I receive notification from Ms. Keithahn that you are eligible to receive an accommodation due to a verified disability, I will provide it in as [discreet](#) a manner as possible.

License



This work is licensed under a [Creative Commons Attribution 4.0 International License](#).