Chapter 2

Course Policies

Please familiarize yourself with the information on this page.

2.1 Course Textbook

The course materials are organized into several books.

- The course Textbook includes the material covered in lectures.
- The Recitation Book includes the materials for the recitations.
- The Book of All Labs includes your assignments.
- The Library Documentation includes the document for the Standard ML library that you will use for you assignments.
- Miscellaneous Documents will provide additional readings, and practice exams.

2.2 Lectures

We would love to see you in class and will reward attendance. So please come, sit close to the front, and interact. Interacting with the course staff is fun and a great way to learn. Also, please note that coming to class is an excellent way to be in the know. If you don't come to class, then you will almost surely miss important things.

Regular lectures will generally occur Monday and Wednesday though we might have make-up lectures. These will be announced in class.

2.3 Grading Policy

- 25% Assignments
- 20% Exam 1
- 20% Exam 2

- 25% Final
- 5% Participation
- 5% Quizzes

The grading policy is designed with the goal of reducing unnecessary stress by providing students with flexibility in managing their deadlines. The exams are graded on a curve, and the assignments are graded on a bucket system described below.

Our philosophy is to use grading as a tool to encourage clarity of explanation, introspection, and self-learning. We hope that both instructors and students can think about grades in as relaxed a manner as possible. We don't want you to stress out about grades, because we don't think they matter as much as what they are made up to be. In return, we expect you to be relaxed about grading. This means that you should focus your energy on learning rather than stressing about a few points that you might have lost on an exam.

There will be weekly assignments (12 total), 2 exams during the semester, and a final. The due dates and exam dates will all be posted on the course schedule. The assignments will be distributed via Diderot. Each assignment will have a coding section, a written section, or both. Coding tasks will be submitted and graded on Diderot. Written tasks will be submitted and graded on Gradescope.

Each assignment will have a total score usually between 100 and 200 points. Somewhat unusually, you don't need to earn all the points to get full credit for each assignment. If you reach 80% of the total available points for an assignment, you get full credit on that assignment. Each of the 12 assignments are equally weighted when calculating your final semester grade.

Participation means attending, and being actively engaged in, recitation and lecture, as well as being respectful to other students and course staff. We actively take attendance in recitation, and lecture sometimes includes informal quizzes or other measurable activities that contribute to your participation grade. As with homework scores, our computation of your participation grade forgives a small but reasonable number of absences; you will receive full participation credit if you regularly attend and participate in lecture and recitation.

Quizzes will be released weekly throughout the semester. These short multiple-choice quizzes are related to the topics recently discussed in the course. Every quiz is due at 11:59 PM US Eastern Time, and no points will be given for late submissions. Quizzes are graded for correctness, but quizzes are graded similarly to homework assignments where you do not have to answer every question correctly in order to receive full points.

In grading, we will reward not only correctness but also clarity and simplicity. To avoid losing points for hard-to-understand solutions, you should make sure your code is either self-explanatory or contains good comments; bad comments make your code harder to read. We reserve the right to manually deduct correctness points for code that passes some test cases but does not show a sincere attempt to complete the assignment.

Grading your assignments is a lot of work for your TA's, especially determining the cost bounds for your code. For each coding assignment, you can help us by providing a statement of what you think your cost bound is and a short description of why you think that this bound holds. Think of this as a safety mechanism to help you avoid losing points; just as we may take off points for unclear code, we may take off points if we don't understand the costs associated with your solution. Thus you want your comments to be helpful! See the 210 style guide for an example.

We may sometimes choose to not grade certain questions or certain parts of each assignment (although we will grade the majority of your work). In such cases, everyone will receive a full score for the ungraded parts.

2.4 Late Assignments

Homeworks are due at 11:59 PM US Eastern Time unless otherwise indicated on the assignment or on Diderot (remember to check your email). You may submit up to two days late on Diderot and/or Gradescope, with a 10% score penalty for each day (for example, if you turn in one day late and score 85 points then you would be deducted 8.5 points).

In terms of grading, the coding and written portions of each assignment are considered separate; you could, for example, submit the coding one day late at a 10% penalty and the written two days late at a 20% penalty.

We grant additional extensions on homeworks only in extraordinary circumstances, almost always involving a family or medical emergency with your academic advisor or the Office of Student Affairs requesting the extension on your behalf. Please discuss exceptional circumstances with your academic advisor or the Office of Student Affairs before requesting an additional extension outside the regular late policy.

2.5 Style Grading

Handed in code will be graded for style and clarity on a pass/fail basis. You should refer to the style guide `http://www.cs.cmu.edu/~15210/resources/style.pdf` for general guidelines.

If your coding style on a particular homework assignment is sufficiently poor, then you will receive a score of 0 for the programming portion of that assignment. In order to restore your grade, you must:

1. review the comments given on Diderot (which will detail why you failed style grading),
2. rewrite your code for the lab (*do not resubmit*), making sure to address every comment, and
3. show your new code to a TA (preferably at office hours) before the regrade deadline and explain your fixes.

If your new code is acceptable, the TA will change your style grade for the assignment to a "pass". Your original Autograded score will then be restored. It is very important that you do not attempt to resubmit any fixed code to Diderot; a resubmission after the deadline would be considered late, and will be graded with a penalty.

2.6 2 Week Cut-off

All issues regarding a given assignment or exam must be resolved within two weeks of the grades being released and announced (on Diderot/Gradescope). After the two week deadline, we will throw away unclaimed written feedback, and set the grades in stone.

This means that if you need to contest a grade, fix a failed style grade, or have any other problems regarding an assignment, you must come to the course staff within two weeks.

To resolve any grading issues, you should talk directly to the TAs assigned to your recitation section. You may speak with them in recitation, at their office hours, or by email. Please do not send an email to the entire 15210 staff mailing list.

2.7 Exams

For extra time and other special accommodations on exams (or otherwise) students should submit an accommodation letter to one of the instructors. Please arrange your exams through the disability services office.

We give make-up exams only in extraordinary circumstances, e.g., involving a medical condition, or an official, university-related athletic competition. If you believe you need a make-up exam for a medical reason, please ask your academic advisor to request the make-up exam on your behalf. If you believe you need a make-up exam for an official athletics-related activity, please email the course staff mailing list. We reserve the right to adjust your overall grade computation (based on your other exam and homework scores) to exclude a missed exam, rather than give you a make-up exam.

2.8 Communication: Diderot, Email

Diderot makes it more efficient for the course staff and instructors to handle course related communications, also while reasonably protecting your privacy. We therefore strongly encourage the students to use Diderot over email. Your questions and feedback will likely receive better attention and faster response via Diderot than via email.

For all routine help and questions for the course staff, please use Diderot.

For more general concerns and feedback, e.g., you are unhappy about something in the course, please reach out to us as quickly as possible. You can do so by creating a Feedback post on Diderot. Feedback posts are anonymous and therefore you can expect a degree of privacy (note, however, that the instructors can reveal author of all posts but do so only under very specific circumstances such as inappropriate or objectionable language). This is probably the most efficient method for many concerns.

For more specific and personal situations, you can email the instructors.

2.8.1 Diderot

Please familiarize yourself with Diderot.

1. Diderot is organized around *atoms*, blocks of text that are "actionable." You can ask questions or more generally make posts about atoms, bookmark them, take (private) notes on them, etc. To take or see you notes, press on the "pen and pencil" icon on the top right corner of a page.

2. The lecture notes are divided into chapters. Each chapter is a collection of atoms organized into sections. Some atoms *expand* to reveal more information, such as examples, that might be particularly helpful in initial reads of the material. You can

expand an atom by clicking on the triple bar on its left corner. You can also expand whole chapters by clicking on the triple bar on the right corner of the page.

3. You can make different kinds of posts on Diderot, including feedback, question, and social. Please use the right kind of post so that important questions can be answered as promptly as possible. For example, if you have some feedback about an atom (e.g., typos), then use the "feedback" kind. If you want to share a meme that you have made, please use the "social" kind. For private and public questions, use the "Question" kind.

4. Diderot is an educational platform. Please be kind, respectful, and civil. Any inappropriate or offensive language is not acceptable. Even if an offending post is anonymous, we reserve the right to identify the author. As a student, if you encounter inappropriate language, then please let us know.

2.8.2 When to email staff

Please remember that this is a large class and refrain from unnecessarily emailing course staff. We recommend that you avoid emailing course staff for any of the following reasons:

1. *Waitlists.* Our department administrators make a sincere effort to accommodate our entire waitlist as promptly as possible. Course staff (including the instructors) do not accommodate special requests and do not have additional information about your likelihood of being enrolled. Please be patient and let the department administrators work efficiently to enroll as many people as possible.

2. *Grades, exam scores etc.* Rather than email course staff about your grades, please see a course staff member in person if you want feedback about how you could have improved your score.

3. *Attendance.* Please do not email us if you plan to be absent from recitation or a regular lecture. We expect that you will review course materials and discuss them with course staff in office hours, as needed, to keep up with what you missed. Our computation of your participation score assumes it is reasonable to miss a small number of course events; you will receive full participation credit even if you miss some recitations and lectures, as long as you regularly attend and participate.

4. *Questions.* Please use Diderot for questions about assignments and lecture material.

We will post announcements, clarifications, corrections, hints, etc. to Diderot—please check it on a regular basis.

2.9 Use Standard and Parallel ML

The lecture notes and the book use a pseudo-code language to specify the algorithms, and the labs in the course nearly all involve programming in the closely related Standard ML (SML) and Parallel ML languages. If you are a CMU undergraduate, then you should have little or no difficulty with SML because it is heavily used in 15-150, one of our course prerequisites. If you don't know SML or another functional language, then you might

have significant difficulties in the course; we recommend that you become familiar with SML before attempting this course.

There are several reasons for why we use ML. The primary reason is that the course is based on the idea of designing algorithms with a clear distinction between specification and implementation, and using cost models to bridge the gap. Within the domain of functional programming, this leaves only a few languages, namely the ML family (including SML and OCAML) and languages such as Scala. We choose the ML family, because they are smaller languages and more suitable for teaching. From within the ML family, we use SML because we have developed a compiler, called MPL (read "maple") for this language that can generate parallel executable programs.

The course material can in principle be taught in a non-functional lower-level language such as C/C++. But with such a language, it would be difficult (probably impossible) to cover the material covered by our labs. Writing correct parallel programs in imperative languages can be very tedious, e.g., even a simple bug can require many hours of low-level debugging. It can also be rewarding, however, because it is sometimes possible to write faster programs, for example, by controlling memory layout. We encourage the students to learn about these techniques in more focused courses, also taught at Carnegie Mellon.

Using a strict functional programming language such as ML can lead to some loss of performance, let say, out of generosity towards non-functional languages, 3-fold, but this usually comes with a big reduction in time needed to write correct parallel codes, let's say 5-fold. In an algorithms course, we don't worry about even a 5-10x slowdown in performance; it is absorbed by the asymptotic notation. But, a 5-fold increase in programming effort is untenable. Functional programming shields us from the complexities of concurrency and the realities of hardware, allowing us to focus our efforts on algorithmic concerns, which is the topic of this course.

2.10 Word of Caution

You will likely find this course to be difficult. There are several reasons why. First, the material covered in the course, especially parallelism part, will be new to many of you. Second, the way we design our algorithms and implement them, with emphasis on higher-order programming (where functions are first-class values), can be difficult to grasp quickly, though over time you will likely not be able to imagine thinking without them. Third, this course will require generating your own algorithms in addition to understanding existing algorithms. If you don't know Standard ML, then there will also be the additional overhead of learning a new programming language, and you should start learning it immediately. There are many online resources for doing so.

It is thus important for you to mentally prepare yourself for a difficult course. If you do your work, we are confident that you will finish this class with a satisfactory grade. As you will discover throughout the semester, we have an excellent set of teaching assistants that can help great assistance. That said, you should keep in mind that there is no substitute for doing your own work.

2.11 Academic Integrity

All students are expected to be familiar with, and to comply with, the University Policy on Cheating and Plagiarism .

Any work submitted as a homework assignment or examination must be entirely your own and may not be derived from the work of others, whether a published or unpublished source, the worldwide web, another student, other textbooks, materials from another course (including prior semesters of this course), or any other person or program. You may not copy, examine, or alter anyone else's homework assignment or computer program, or use a computer program to transcribe or otherwise modify or copy anyone else's files. It is not acceptable to look at exams from prior semesters.

To facilitate cooperative learning, it is permissible to discuss a homework assignment with other students, provided that the following *whiteboard policy* is respected. A discussion may take place at the whiteboard (or using scrap paper, etc.), but no one is allowed to take notes or record the discussion of what is written on the board, and you must allow two hours to lapse after any discussion before working on the assignment. The fact that you can recreate the solution from memory is taken as proof that you actually understood it.

It is not acceptable to share your solutions or give hints to your friends for a lab after you have already discovered the correct idea. You are not helping your friends by doing so. The right thing to do is to not talk about the lab after you have a solution. Anyone struggling with the homework should visit office hours to talk to an instructor or TA.

We run automatic code comparison programs (such as MOSS) on student solutions. These programs are very good at detecting similarity between code, even code that has been purposefully obfuscated. Such programs can compare a submitted assignment against all other submitted assignments, against all known previous solutions of a problem, etc. The signal-to-noise ratio of such comparisons is usually very distinctive, making it very clear what code is a student's original creative work and what code is merely transcribed from some other source. Cheating is simply not worth the risk.

One final note: receiving credit for an assignment or exam is not an indication that we did not catch you cheating. Because dealing with cheating cases is a lot of work for the TAs and the instructors, we often delay enforcement until well into the second half of the semester and take action all at once, after we identified a number of cases. This usually leads to unfavorable outcomes for the students involved.

The minimum penalty for cheating (including plagiarism) will be a zero grade for the whole assignment; a typical penalty will be -100% on the assignment. Dishonesty while discussing an academic integrity issue (i.e. lying to course staff) usually results in an 'R' in the course. All violations of this collaboration policy will be referred to the appropriate University disciplinary board, with possible additional disciplinary action. For more information, see the University Policy on Academic Integrity.

There is no statute of limitations for violations of the collaboration policy; penalties may be assessed (and referred to the university disciplinary board) after you have completed the course, and some requirements of the collaboration policy (such as restrictions on you sharing your solutions) extend beyond your completion of the course.

2.12 Well-Being and Happiness

We care about you and your happiness and are always available to talk. Also be aware that Conseling and Psychological Services (CaPS) offers many services including therapy and crisis management. When experiencing difficulties, please keep in mind that your problems may appear smaller to you than they actually are, and reaching out to talk with somebody can make a huge difference.

2.13 Safety

Do not hesitate to call CMU police in emergency situations or for other services that they provide. CMU police can also be reached at 412-268-2323.