Course Information

This handout explains how the course is organized and administered. It describes how you will be graded and what the course staff expects of you. You should reread this information several times during the term. One effective strategy is to reread it before starting on each project.

Contents

1 Immediate action items

- Read this Course Information handout thoroughly and completely.
- Complete Homework 0 by Monday, January 13, at 10:00 P.M.
- Set up lab environment for Homework 1 before our next lecture on Wednesday, January 15. For this Tuesday, January 14, attend the office hour if you have any question for our TA, regarding your lab environment setup in Homework 1.
- We will have a short recitation (half an hour) as the second part of our next lecture on Wednesday, January 15. Do not miss this one, as our TA will do checkoff on lab environment setup.

- Complete Homework 1 by Friday, January 17, at 10:00 P.M. Homework 1 will start to familiarize you with the tools we shall be using in this class. Do not delay to the last minute in starting this homework, as you will not be able to start on the first project without it. The first recitation and office hour will help you on this assignment.

2 General information

Modern computing platforms provide unprecedented amounts of raw computational power. But with great power comes great complexity, to the point that making useful computations exploit even a fraction of the computing platform's potential becomes a substantial challenge. Indeed, obtaining good performance requires a comprehensive understanding of all layers of the underlying platform, deep insight into the computation at hand, and ingenuity and creativity to obtain an effective mapping of the computation onto the machine. The reward for mastering these sophisticated and challenging topics is the ability to make computations process large amounts of data orders of magnitude more quickly and efficiently and to obtain results that are unavailable with standard practice.

This class provides a hands-on, project-based introduction to building scalable and high-performance software systems. Topics include performance analysis, data representation, bit tricks and vectorization, assembly language and compiler intermediate representation, task-parallel programming, work/span analysis, compiler optimization, cache-oblivious algorithms, races and synchronization, nondeterministic programming, and building scalable systems.

All of the projects in the class are open ended. You should not spend more than an average of 12 hours per week on the class. As is true in real life, you can almost always improve a work product by putting in extra time, but there are diminishing returns. Since performance engineering can be addictive, to take this class successfully, you must have the self discipline to determine when enough is enough.

The prerequisites for this class are *C/C++ programming*, *Computer Architecture*, *Algorithms and Data Structures*. **These prerequisites are mandatory and will be enforced!** Because this class entails several joint projects, it is not fair for your partners if they must compensate for your lack of knowledge. If you do not satisfy the prerequisites, you will not be allowed to take the class unless granted an exception by emailing Professor Chen at `cxh@msu.edu`. The course programming language is C augmented with the OpenCilk 2.0 task-parallel extensions.

The lecturer would prefer you to attend class and get to know you, but all lectures will be recorded for your review. Technology does sometimes fail, however. Do not rely on lecture recordings to meet the requirements of the class. If a recording fails and you have a legitimate excuse, the course staff will accommodate, but otherwise, options to catch you up will be limited. We have too many students to cater to the few who seek to consume staff time owing to their own negligence.

3 Staff

Position	*Name*
Lecturer, in charge	Professor Xuhao Chen
Teaching Assistant, recitation leader	Raaghav Ravishankar
Administrative Assistant	Brenda Hodge

4 Administrative infrastructure

Content	*Provider*	*Location*
Course management	website	https://software-performance-engineering.github.io/
Forum, Q&A	Piazza	https://piazza.com/msu/spring2025/cse491
Code repositories	GitHub	https://github.com/CSE491-spring25
Assignment submissions	Gradescope	https://www.gradescope.com/courses/849343
Office hours queue	Whiteboard	To be announced

We will be using the website for assignments, lecture slides, lecture videos, and reading materials. The calendar will list all of the important dates for the class.

The Piazza website provides a wiki-like service for organizing questions and answers regarding course content. The course staff will use Piazza, rather than email, for communications with students. Piazza allows students to submit questions to the course staff, which other students can see. The course staff and other students can respond to questions and provide answers. **All questions about course content and administration should be posed via Piazza, not via email.** If you are shy, you can post your question to Piazza anonymously, but unlike public questions, anonymous questions will not count towards your "class contributions" grade (see Section 8).

We will use Piazza for almost all course communication. To communicate directly and privately with course staff, please write your question or note on Piazza and send it to "Instructors" only. Even if you want to communicate with a specific member of the course staff, please post to "Instructors," but put a greeting in the body to address the staff member with whom you wish to communicate. Your question or note will not be viewable by the rest of the class unless you later choose to make it public. **Please do not use email** unless absolutely necessary, as that may delay a response, perhaps by several days.

We will use the Git distributed version control system for code release and submission. We will be using Github to host repositories, so you should create a Github account if you do not already have one.

You will submit your completed assignments and view your grades via Gradescope, which is accessible via the course website.

5 Lectures

Lectures will be held live in 12:40-2:00 MW in 1225 Engineering. History has shown that you will do better in the class if you attend lectures regularly. All material covered in the lectures will be fair game for homeworks and projects. We will post lecture slides and other reading materials on course website after the lecture. We regret that it is impractical for us to provide copies of lecture slides before the lecture.

You should feel free to raise your hand during lecture if you would like to contribute a relevant question or comment.

After the lecture period is over, the lecturers will generally be available for questions and general discussion outside the classroom. You can talk about course content, your projects, performance engineering in general, something you discovered, administrivia, and random other topics. The lecturers can also relate amazing stories when prompted! Please take advantage of this opportunity. In the past, many former students obtained research projects, and Mater/PhD advisors by first establishing a relationship with a lecturer. Sometimes these opportunities came directly from the lecturer, but sometimes an opportunity from a different faculty mentor was discovered by networking with one of the lecturers. Come hang out for a few minutes after class.

6 Recitations

Our TA will hold bi-weekly recitations to help students on homeworks and projects. Recitations are designed as hands-on tutorials covering tools and other practical topics that are explored in the homeworks (see Section 11). Recitations are **mandatory**. **If you miss more than one recitation without an approved excuse, you risk failing the course.** If you must miss a recitation, notify your TA as soon as possible in advance. In particular, you must make arrangements with your TA to complete the recitation check-off portion of that week's homework assignment.

7 Office Hours

Our TA will be available during office hours to help you learn the course material. Office hours will be held regularly on Tuesdays from 3:00 P.M. to 4:00 P.M., location to be announced. To queue for staff help, write your name on the whiteboard. Deviations from the regular schedule may occur due to holidays and other events. We will keep the course calendar up to date with any changes.

8 Grading

Each assignment will describe how you will be graded. The scores you receive on each assignment will be combined to produce your final grade after being weighted approximately as

follows:

Assignment	#	*Grade*
homeworks	5	20%
Project 1	1	35%
Project 2	1	40%
Class contributions	∞	5%

Class contributions include asking questions in lectures , recitations, and Piazza; answering questions on Piazza; contributing scripts or code snippets for public use; etc.

You must complete all of the projects and all of the homeworks. **Failure to turn in a project (or turning in a project that shows an insufficient attempt) is grounds for failing the class! Missing more than one recitation check-off or more than one homework assignment is also grounds for failing the class.** Our TA will be available at recitations and office hours to help you perform to the best of your ability. Please take advantage of these opportunities.

You do not compete with other students in the class for your grades. There is no grading curve. To determine your grade for performance on a project, for example, your project will be compared to staff "reference" implementations, not to other students's projects. Consequently, you should not feel inhibited, subject to the academic honesty guidelines (see Section 15), from contributing to the class learning experience. You will be graded on your class contributions (or lack of them).

The course staff reserves the right to reward, beyond the percentages listed above, any students who outperform in any category.

9 Punctuality and extensions (i.e., late policy)

With the number of students in the class, the course staff cannot afford to make accommodations for students who intentionally schedule a lecture conflict. We have too many students and too many assignments to accommodate more eventualities than necessary. Consequently, you should not take another class that meets at the same time as either the lectures or your recitation. Quality employers respect a student who understands priorities.

Students are expected to manage their time and complete each assignment punctually, but we recognize that unforeseen challenges may arise during the term. The course staff will accord you a budget of 3 total late days to use as needed on homework assignments and project write-ups throughout the semester. For project write-ups, the late days will be charged to every team member's budget. Late days are atomic and cannot be subdivided. The beta and final project submissions themselves may not be extended. Moreover, no single assignment may be submitted more than two days after the deadline.

If you need to submit a homework or project write-up after the deadline, please post a private note via Piazza with a tag of `#late_days` before the deadline and tell us how many late days you

intend to use. If you exceed your late-day budget, we will require a note from Student Support Services to accept your late work.

As mentioned, late days may not be used for the beta and final submissions of projects. We unfortunately have no capacity for handling such late submissions because this class has many tight deadlines, and project code will be released immediately after the deadline. For the beta and final submissions, you should submit whatever you have by the deadline, and we will award partial credit as appropriate. If extenuating circumstances force you to miss a deadline, we require a note from Student Support Services to accept your late work. Please speak with your TA immediately if you believe you will have trouble completing an assignment in time. We can often make reasonable accommodations if you tell us early enough.

In the unfortunate situation that you do miss an assignment, please talk with your TA immediately.

10 Weekly status report

Each Monday by 10 P.M. during the semester starting on Monday, Jan 20, you must submit a short (one paragraph is sufficient) personal status report via Gradescope. **Failure to keep current with status reports is grounds for failing the class.** Your status report should serve as your personal summary of your involvement in the course since your last status report. Here are some of the things you might consider including:

- Describe your impressions of the past week's lectures and recitations.
- Talk about your feelings regarding your project partner, study group, or the course in general.
- Reflect on how effectively you feel you are working on class assignments and how much time you're devoting.
- Identify the aspects of the class you found most engaging or frustrating.
- Offer constructive criticism about how the class is being taught.
- Report on any outside events that may have impacted your work on the course.

In addition, you can always provide feedback at any time via a private post on Piazza. The post can be anonymous, but if you want to receive a response, please identify yourself. We will make every effort to preserve your confidentiality.

11 Homework

Homework 0 is due on Monday, January 13, at 10:00 P.M.

In addition to homework 0, there will be 5 homework assignments throughout the semester. These homeworks are intended to give hands-on experience with tools and other practical topics related to the lectures. Each homework will help you prepare for your projects and evaluate your understanding of the course material. Part of each homework will typically need to be completed and checked off by your TA in recitation.

12 Projects

The bulk of your out-of-class time will be spent completing three projects of increasing scope and complexity. The basic premise of each project is straightforward. You are given a correct but inefficient program, and your mission is to make it run as fast as possible. Each project involves a beta submission, a design/code review, and a final submission. Students will work in pairs to complete the projects. You will be assigned a different random classmate as your partner for each of these three projects.

Project 1

The first project addresses the problem of rotating a square bit image. You will learn how to understand the performance of a program using profiling tools and to experiment with word parallelism and vectorization.

Project 2

For the second project, you will begin with a single-threaded physical simulation. You will optimize this simulation and then parallelize it using the OpenCilk implementation of Cilk: a language, compiler, library, and tool chain for developing task-parallel applications. This project will expose you to many of the issues associated with correctness and performance in task-parallel applications.

13 Project structure

Each project consists of a beta submission and a final submission. You must also submit a write-up that describes your implementation for each beta and final submission. The exact grading scheme for the projects will vary. The assignment handout for each project will describe exactly how your implementation will be evaluated.

A large portion of the grade depends on how fast your code runs. The grading scale is fixed and determined by the course staff. **You are not competing with others in the class.** The teams with the fastest correct implementations for the beta and final submissions will receive only the admiration of their peers (and the course staff). After all, being fast is cool!

Team contract

A team contract is an agreement between you and your teammate(s) to establish a set of conventions about how your team will operate. Ideally, it ensures a smooth team experience — think back to good or bad aspects of past team projects when negotiating it. Your contract should answer questions such as the following:

- How will you communicate outside of meetings (Zoom, phone, email, Slack, etc.)?
- If someone on the team decides to drop the class, what obligations do they have to their teammates?
- How will work be divided among team members?
- How will you review each others' code? How will you manage your code branches?
- What expectations regarding team grade do team members have? If there are differences, is it acceptable for some team members to do more or less work than the others?

If your team has problems working together, and you have little in your team contract, it will be more difficult for the course staff to help you. Your TA will review your team contracts and let you know of any concerns we see.

Beta submission

For the beta, you should focus on creating a correct implementation and developing a good set of tests. We will announce a performance target for the beta submission corresponding approximately to the B/C line for the performance part of the grade. Projects that do not meet or exceed the performance target will receive a grade based on how close they come to it. Better performance than the performance target guarantees you at least a grade of B for performance, but you should try to make your implementation faster in the time available if you want an A.

Immediately after the beta-submission deadline, all submitted codes will be anonymized and posted online so that everyone can learn from what others have done. Please do not put team members' names or other personal details that could be used to identify you or them into your code or comments. In addition, performance results will be posted, so that you can see how your submission compares to the rest of the class. (Once again, you are *not* competing against each other, but it is helpful to see what others have done.)

Please look at others' submissions to learn from them. You may borrow ideas for your final submission, but you may not steal code outright from your classmates or any other sources. We will be checking code for any such instances of cheating. The work you submit must be your own, but it is fine to seek inspiration from your classmates' beta submissions after they are posted. A good strategy to avoid copying code inadvertently is to leave a significant interval of time (e.g., an hour) between when you look at someone else's code and when you write your own version.

Final submission

The final submission will typically be due a week or more after the beta and will be provided explicitly on the project handout. Try to make your implementation as fast as possible. In the real world, no one can tell you the performance you can hope to achieve for a given piece of software. Likewise, in this class, you will not know what performance can be achieved. We expect your final submission to incorporate feedback from your Deputy and demonstrate improvement in code quality and documentation. Your final submission will be anonymized and posted online for classmates to see and learn from.

Write-ups

The day after each beta or final submission, you must submit a short write-up describing your code. Describe the structure of your code, how you diagnosed the bottlenecks, the optimizations you implemented, and what kinds of speedup the various optimizations produced. Feel free also to describe things that did not work and what you learned from the experiments. The first project handout will suggest an organization for write-ups.

Team dynamics

With the exception of the code-review part of the project grade, team members will generally receive the same grade on their joint project. You must make a substantial contribution to each project. If a team member does not make a substantial contribution to the project, the course staff will adjust the student's individual grade accordingly.

To help ensure that all team members contribute, we will be reviewing the Git commit logs to assess the dynamics of your team. **Please ensure that commits are well balanced among your team members' user names.** We understand that with pair programming, commits from one team member may represent work by others, but it is up to the pair to ensure that commits are balanced.

If you feel that a team member is not pulling his or her weight, please contact the course staff as soon as possible via a private post on Piazza.

14 Class contributions

You and your classmates are not competing for grades. View students in the class as a learning team, helping each other to master the material and become good performance engineers. Contributing to the learning team will improve your grade. You can contribute by asking or answering questions on Piazza. (Anonymous Piazza posts do not count towards your class-contributions grade, however.) You can contribute by writing and sharing a software-productivity tool or script to solve a common problem, but announce your software via Piazza. There are many other ways.

Be creative. Please cite any classmates' contributions and other references in project write-ups and Piazza following academic norms for honesty and according credit where credit is due. And please do not be shy in letting us know about your own contributions during the term.

15 Academic honesty

University guidelines relating to academic honesty require that we inform you of our expectations regarding permissible academic conduct. It is your responsibility to satisfy both the letter and the spirit of these rules. If any part of this policy is unclear, or if you have any questions or concerns, please ask a member of the course staff for clarification.

If you violate this policy, you will be referred to the Committee on Discipline to face the possibility of expulsion and other punitive actions. We take academic dishonesty extremely seriously. Please do not put us in a position where we must deal with it.

You may not copy or transcribe a solution from any source. The work you submit must be your own. Be aware that generative-AI systems such as Copilot sometimes copy from sources. Consequently, you would be wiser to take inspiration from such tools than to use generated code directly.

The course staff will use technological and other creative means to detect cheating. If one party illegally shares material with another, we treat both the giver and receiver as equally guilty of academic dishonesty.

When working in a project team, you may (of course) share ideas, code, and anything else that may be appropriate within the group, but be sure that you are making a fair contribution to your team and abiding by the team contract. In the written material you submit with each project, please briefly describe the contributions of each group member. If you have not made a fair contribution to that work, putting your name on your team's work is considered academically dishonest.

If you wish to share ideas, code, and anything else outside your project group, you must share it as a public post on Piazza. We encourage you to share things you learn or discover, and it will go towards your "class contribution" grade. You should not, however, share code that is more than a few lines. After the beta submissions are published, you are free to review them and be inspired by your classmates' approaches to the project. You may not directly copy code, however.

While you are working on a project, you may not permit anyone besides the staff and the members of your project team to access your project source code, your compiled binaries, or your written documentation. Wait until the course staff posts them, typically shortly after the submission deadline. If you wish to share a few lines of code with somebody else before the project deadline, however, you may post it publicly on Piazza.

You may collaborate on homeworks, but you must write up your solutions by yourself. If you cannot explain something you submit to a staff member, we will be forced to view it as cheating. Keep homework write-ups private until they are graded.

You may use general conceptual material, such as what you might obtain from a textbook, regardless of its source. For instance, *Introduction to Algorithms*, 4th edition, by Cormen, Leiserson, Rivest, and Stein (The MIT Press, 2022) is an excellent resource for algorithms, including task-parallel algorithms. You may use materials published on the World Wide Web, which has many resources on performance engineering, but be aware that some are good, and some are bad. If you do use any material from an external source, including material from generative AI tools, please cite it briefly and clearly in your documentation. Be aware that the generative-AI program Copilot may copy from sources.

You *may* share code snippets, tools, compiler settings, relevant websites, and other generally useful information via Piazza. Doing so contributes to cooperative learning. (Remember, you are not competing with one another for grades.) You may not choose with whom you share, however. If you choose to share, you must share with the entire class. Students who contribute meaningfully to the class's knowledge pool will receive credit towards their class contribution grade.

In summary, make sure that you are turning in your own work! If you have any questions or concerns, talk to the course staff. If you feel that you may have violated this policy, it will go far better for you if you report your possible transgression to us than if we find out by other means.

16 If you need help

The course staff wants you to do well in this class. We stand ready to help, but you, not we, are fundamentally responsible for your own academic performance. Before seeking help, *please make a reasonable attempt on your own*. It is wise to get started on assignments early. Programming assignments often take longer than you expect, even after taking into account that programming assignments often take longer than you expect. If you find you need help, or you just want a boost of confidence, come to office hours. We devote a significant amount of staff resources to holding office hours. Please take advantage of them. If things are more serious, communicate with us via a private Piazza post (not email!). Although we want you to give it the old college try before seeking staff help, it is still better to err on the side of coming to us too early than too late. If you wait until the last minute, you may not receive help in time for it to be actionable.

17 Acknowledgment

This course was originally designed and developed by Prof. Charles E. Leiserson, Prof. Saman Amarasinghe, and the 6.106 course staff at MIT. It is now adapted and taught by Xuhao Chen at Michigan State University. We extend our gratitude to the MIT 6.106 staff for generously sharing the course materials and for their invaluable support.


SPEED
LIMIT
∞
PER ORDER OF 6.106
Enjoy the Class!
And never do anything half fast!