CS601 : Software Development for Scientific Computing

Autumn 2021

Lectures Tuesdays (11:30AM - 12:30PM), Wednesdays (9:30AM to 10:30AM), and Thursdays (11:30AM to 12:30PM)

Course web page https://hegden.github.io/cs601/

Instructor Nikhil Hegde (nikhilh@iitdh.ac.in)
Office Hours (doubt clearing sessions) : decided based on individual's / groups' availability online.

TAs Gayatri Rayar
Office Hours (doubt clearing sessions) : decided based on individual's / groups' availability online.

Prerequisites Exposure to Data Structures and Algorithms, C / C++ / Java / Matlab

Textbooks / References

1. Stroustrup C++ Language Reference https://www.stroustrup.com/4th.html
2. Suely Oliveira, David Steward: Writing Scientific Software: A Guide to Good Style. Cambridge University Press, 2006
3. Steve McConnell, Code Complete: A Practical Handbook of Software Construction, 2nd Edition
4. The landscape of parallel computing research https://www2.eecs.berkeley.edu/Pubs/TechRpts/2006/EECS-2006-183.html
5. CS294-73

We will primarily be referring to Textbook 2. A very similar course to this one is reference 5. While I highly encourage you to purchase the Textbook 1, it is not required. All lecture notes and supplements will be posted online.

Course outcomes A student who successfully fulfills the course requirements will have demonstrated the ability to:

1. Develop skills to use and understand tools that cross most disciplines and apply the skills towards larger software development goals.
2. Explain dominant algorithmic patterns and data structures as they arise in scientific computing
3. Analyze asymptotic behavior of algorithms to understand their complexity
4. Develop, debug, and profile programs in C++, a widely used language in scientific computing

more specifically, after taking this course, you will be able to:

- Explain and use tools such as Make, CMake, Git, GDB, Doxygen, and GCC tool chain. Use libraries from sources such as Netlib (BLAS) or OpenBLAS
- Describe an overview of frequently occuring algorithms in scientific computing and implement some of them: FFTs, dense and sparse linear algebra, structured and unstructured grid methods, particle methods (N-body, Particle-Particle, Particle-in-cell, Particle-in-a-mesh), PDEs.
- Explain the algorithmic complexity with the help of asymptotic analysis.
- Fine tune code with the help of profiler tools such as valgrind. Factor in clarity, flexibility, re-use, and efficiency measures while developing software.

Course assessment The achievement of course objectives will be assessed through a combination of tests (2), in-class short quizzes, and a series of individual programming assignments (5). Tests, quizzes, and class activities will test your understanding of the concepts covered in the class, while the programming assignments will test your ability to put those concepts into practice.

Course grading Grades will be assigned as follows:

Score	Assessment	Comments
65%	Exams	Mid-sem (25%) and End-sem (40%). There will be two components to each of these exams: written and programming. Written component is paper-based. Programming component is a take-home, 24-hour exam.
12%	Short Quiz	Multiple-choice based questions, Roughly one per class. Best 12 taken.
5%	Class participation (live and discussion forum)	Asking a technical question (1), Answering a technical question (4). Monthly accrual.
30%	Programming Assignments (5)	6 points per assignment. Of the 5 assignments, two will be part of the exams.

Given your final numerical score, your course grade will be determined according to the following scale:

If your numerical score is at least:	Your course grade will be at least:
90	AA
80	AB
70	BB
60	BC
50	CC
45	CD
40	DD

Programming assignments We will have programming assignments once in 3 weeks. The assignments may involve writing a substantial amount of code depending upon the language you choose for implementation (e.g. C instead of C++). Matlab and Python implementations are not accepted. Programming assignments must follow the below guidelines:

- All programs should compile and run correctly. It is your responsibility to make sure that your code works in the execution environment made available to you (details about the execution environment will be communicated later).
- Unless otherwise specified, all assignments are due at 11:59 PM on the deadline date.
- You may use C/C++/Java for implementing the programming assignments.

Programming assignments will be submitted via GitHub Classroom (https://classroom.github.com). You are required to have a GitHub account. These can be obtained for free at https://github.com. *You must send your GitHub username to the Instructor latest by Friday, August 6th, 2021. You must provide the details in a Google Form, the link for which will be emailed to you.*

Prerequisite Software To develop the programs on your local environment (laptop / desktop), you must have access to Linux environment. We may provide a little support if you are running virtual Linux environments on Windows Laptops. It is also strongly advised that you have `Git` software available on your local environment. E.g. Windows users may find this helpful.

Late submission policy except for medical and family emergencies (accompanied by verification), there will be no individual extensions granted for programming assignments. Late submissions will be scaled according to lateness, docking 10% from your score per day late, up to a maximum of 50%. Submissions more than 5 days late will be assigned a score of 0.

Exams we will have two exams for CS601. The exams (written component) are open book and open notes. You may bring the course Textbook as well as any written/printed notes/programs from lectures or otherwise. The programming component is an open, take-home exam.

Exam	Topic	Week
Midsem	Scientific computing and frequently occuring patterns, Selected C++ language features and data structures, Tools	TBD
Endsem	Asymptotic analysis, Debugging, Documentation, and Profiling	TBD

Exam topics and tentative dates Note that this is a tentative assessment scheme. The academic office will communicate the exact scheme at a later date. If you need a change in either the test times or environment due to approved accommodation from Dean AP's office, it is your responsibility to contact the Professor two weeks prior to the exam so alternate arrangements can be made.

Course discussion this term we will mostly be using the `Discussion` feature of GitHub Teams for class discussion. If you have questions about the course or the project, we encourage you to post them on the discussion page in GitHub Teams. It's a shared discussion forum, where your question can be answered by the instructors, TAs, or your fellow students!

Students who are active participants, asking (or answering!) questions are eligible for class participation points that are accumulated monthly.

Email Questions about course material or programming assignments should be posted to the discussion page or raised during lecture or office hours. *The Professor or TAs will not answer programming questions via email.* This is to allow other students who might have similar questions to benefit from our answers. Of course, if you have questions of a personal or confidential nature, we welcome your email.

Course announcements Course Webpage is the primary source of information for course-related material and important announcements. Homework assignment links will be distributed via `Discussion` page/email. Other course announcements, including changes in due dates, course topics, programming assignment details, etc., will be communicated in one or more of the following ways:

1. Updates to the course webpage.
2. Announcement posts GitHub Teams.
3. Email announcements.

Academic honesty unless explicitly allowed, you are expected to complete all assignments by yourself or as a team when teams are allowed. However, you are allowed to discuss general issues with other students (programming techniques, clearing up confusion about requirements, etc.). You may discuss particular algorithmic issues on the Discussion forum (but do not copy-paste your code!). *We will be using software designed to catch plagiarism in programming assignments, and all students found sharing solutions will be reported to the Dean.*

Punishments for academic dishonesty are severe, including receiving an FR in the course or being expelled from the University. By departmental rules, all instances of cheating will be reported to the Dean. On the first instance of cheating, students will receive a 0 on the assignment; the second instance of cheating will result in a failure of the course.