



COM-392, System Programming

American University of Central Asia
Software Engineering Department

1 Course Description

This course introduces students to programming single-core, multi-core, and multi-processor systems, emphasizing data and task parallelism across various hardware configurations, from single machines to distributed systems connected via high-performance networks. Key topics include CPU pipelining for improved processing efficiency and SIMD (Single Instruction, Multiple Data) for parallel data processing. Students will also learn widely used shared-memory parallel programming APIs, such as Pthreads, OpenMP, and CUDA, as well as a distributed-memory programming API like Open MPI. By the end of the course, students will be equipped with the skills to develop efficient parallel applications across diverse computing environments.

2 Course Information

Course Materials

<https://github.com/auca/com.392>

Course Code

COM-392

Course ID

4953

Prerequisite

COM-119, Object-oriented Programming

Credits

6

Time and Place

Lecture: Monday 12:45–14:00, Room 410

Lab: Wednesday 12:45–14:00, Room 410

3 Contact Information

Professor

Dmitrii Toksaitov
toksaitov_d@auca.kg

Office

AUCA, room 315

Office Hours

Office hours are available by appointment, either on-site or remotely, during business hours (Monday–Friday). Please contact your professor to schedule an appointment.

4 Topics Covered

- Week 1–2: Flynn’s Taxonomy, Amdahl’s Law (6 hours)
- Week 3–4: CPU Caches and Locality (6 hours)
- Week 5–6: CPU Pipelines and Branch Prediction (6 hours)
- Week 7–8: Data-parallelism With SIMD Instructions (6 hours)
- Week 9–10: Shared Memory Parallel Systems (6 hours)
- Week 11: Synchronization (3 hours)
- Week 12–13: Distributed Memory Systems (6 hours)
- Week 14–16: General-purpose Computing on Graphics Processing Units (9 hours)

5 Learning Outcomes

By the end of this course, students will:

1. **Acquire programming skills to write and test parallel and distributed applications:**
 - Implement and test programs using parallel programming and distributed paradigms and APIs
 - Apply data parallelism using SIMD instructions, shared memory (Pthreads, OpenMP), and distributed memory (MPICH) systems
 - Develop GPU-based parallel applications using CUDA
2. **Develop proficiency in modeling systems with hardware-software interactions:**
 - Represent parallel computing systems using appropriate abstractions

- Apply knowledge of CPU architectures with pipelines, hazard mitigation, and caches
- Design parallel algorithms with consideration for underlying hardware characteristics

3. Evaluate and use appropriate tools and techniques for system programming:

- Use code editors, compilers, build systems, debuggers, and IDEs effectively
- Work with parallel programming APIs
- Utilize profiling tools for parallel applications
- Apply version control and collaborative development practices to system programming projects

4. Design, plan, and critically evaluate software solutions for parallel computing problems:

- Analyze problems to identify opportunities for parallelization
- Assess possible solutions for performance, scalability, and resource utilization
- Apply principles like Flynn's Taxonomy and Amdahl's Law when designing parallel systems

5. Assess parallel systems in terms of quality attributes and trade-offs:

- Evaluate parallel implementations based on performance, scalability, and efficiency
- Measure and analyze performance metrics through empirical testing
- Optimize applications through knowledge of CPU caches, locality, and synchronization
- Consider trade-offs between different parallel programming models and hardware platforms

6. Reflect on and reason about resource allocation and concurrency problems:

- Critically examine techniques for managing shared resources and synchronization
- Consider energy efficiency and sustainable computing practices in system design

7. Experiment with and analyze parallel computing techniques through practical applications:

- Complete projects demonstrating real-world applications of parallel programming
- Communicate technical findings and defend implementation decisions in presentations and documentation

6 Assignments and Exams

6.1 Moodle/e-Course Checkpoints

Students must maintain instructor-provided private GitHub repositories for their assignments. They must periodically commit and push the required number of project solutions, as directed by the course staff. Instructors or teaching assistants will review the work either during lab sessions (on-site) or after the submission deadline (off-site) and award points based on completed assignments.

6.2 Projects

Students will be required to develop up to four course projects that demonstrate real-world applications. In addition to completing checkpoint assignments for these projects, students must defend their work to the instructor during the midterm or final examination period.

7 Course Materials, Recordings and Screencasts

All course materials are available on GitHub at <https://github.com/auca/com.392>. Using GitHub will help students become familiar with the Git version control system and the widely used GitHub platform.

We aim to record each class and upload it to YouTube for accessibility; however, recordings are not guaranteed. Recordings are produced on a best-effort basis as time permits. If you need immediate access, consider recording class sessions on your own computer. Links to YouTube recordings can be found in the course repository at <https://github.com/auca/com.392>.

While recordings provide flexibility, they are not a substitute for attending classes. Active participation is crucial for success in this course. Accumulating three or more unexcused absences may lead to an *X* grade. If overall class attendance is poor, the instructor reserves the right to discontinue class recordings.

Access lecture screencasts remotely via Zoom at either <http://com-392-zoom.auca.space>. When joining a Zoom session, students must identify themselves using their properly capitalized first and last names in the Latin alphabet. Install and configure the Zoom on your computer during the first week of the semester so you can properly share your camera, microphone, and screen.

Remember that to attend lecture or lab classes remotely, you must request permission from the instructor in advance. Please send a brief email explaining why you need to attend remotely (e.g., illness, personal reasons) and provide appropriate documentation to support your request, either immediately or as a follow-up in the same email thread. If documentation is not provided within a reasonable timeframe, you will be marked absent without excuse and may receive a grade of *X* if you accumulate three or more unexcused absences. Given the substantial number of such requests we receive, do not expect a reply to your email. Regardless, you must still submit all required documentation as soon as possible in the same email thread. AUAF students who are outside the country and can only participate online do not need to request this permission.

8 Software

Students are recommended to install the following software on their machines.

- Git: <https://git-scm.com>

The compilers, assemblers, and debuggers will be available on the remote course server. A *Vagrantfile* will be provided on our public course GitHub repository with information about which software is used on the course server, in case you would like to recreate the environment on your personal computer.

9 Hardware

There are no specific hardware requirements, as you can use the course server where we guarantee you will be able to complete all the tasks of our course. However, we recommend a machine with more memory (rather than less), high-speed disk I/O, and a GPU compatible with current graphics and compute APIs.

10 Reading

1. Computer Architecture: A Quantitative Approach, 6th Edition by David Patterson and John L. Hennessy (AUCA Library Call Number: QA76.9.A73 P377 2019, ISBN: 978-0128119051)
2. Parallel Programming: for Multicore and Cluster Systems, 2nd Edition by Thomas Rauber and Gudula Rünger (AUCA Library Call Number: QA76.642 R38 2010, ISBN: 978-3642438066)

11 Grading

The preliminary distribution of points is outlined below. Please note that the distribution may change throughout the course if tasks are canceled, merged, or made optional (for bonus points). This usually happens for reasons such as software issues, online service outages, or classes canceled due to events outside our control.

Remember that some LMS platforms, such as Moodle, may not calculate final scores and grades correctly until all tasks have been published in the system and properly weighted. Do NOT assume your grade is accurate until all tasks are in the system and your instructor has asked you to review your scores and grades.

One common mistake students make is assuming that attendance points are part of the grade and will be summed at the end—they will not. These points are used only to help us decide on X grades in the middle of the semester and will be removed from the totals in Moodle at the end of the semester.

11.1 Moodle/e-Course Checkpoints

Your instructor will periodically announce reviews of your work. For such checks, you can be awarded up to the specified number of points.

- Project (35%)

11.2 Exams

In the middle and at the end of the course, you will need to pass the Midterm and Final examinations. These exams are significant and will greatly affect your grade.

- Midterm Exam (30%)
- Final Exam (35%)

11.3 Totals

- 100% is formed from the Moodle/e-Course Checkpoints (35%) and the two exams (65%).

11.4 Scale

- [94%–100]%: A
- [90%–94)%: A-
- [87%–90)%: B+
- [83%–87)%: B
- [80%–83)%: B-
- [77%–80)%: C+
- [73%–77)%: C
- [70%–73)%: C-
- [67%–70)%: D+
- [63%–67)%: D
- [60%–63)%: D-
- Less than 60%: F

Please note that requests for a higher grade due to points being marginally close will be ignored. For instance, 93.99 is an A-, NOT an A. Similarly, requests for extra assignments to boost points will also be disregarded.

12 Rules

First and foremost, in addition to all the rules listed in the syllabus document, students are required to follow the Code of Conduct of the American University of Central Asia.

12.1 Participation

Active participation during class may be rewarded with extra points at the instructor's discretion. Poor student performance during class can result in points being deducted from the final grade.

Instructors may conduct random pop-up checks during classes without prior notice. Students MUST be prepared for every class to avoid losing points. Students who are absent from classes with graded work without a valid reason will also lose points, unless their absence is due to force majeure circumstances. Instructors must be notified in advance of the reason for a student's absence in order to avoid a loss of points.

12.2 Questions

A question raised by one student is often relevant to others as well. Therefore, students are encouraged to use the online discussion board of the LMS (Learning Management System) they are enrolled in (e.g., AUCA e-Course System) to post questions publicly so that all students may benefit from the discussion.

Students must not post complete source code for any task on the LMS discussion board. Any such public post will result in a grade of zero for that assignment. In addition, students should refrain from submitting vague or overly general questions such as "Why doesn't my code work?" Instead, they are expected to carefully analyze and debug their code before seeking assistance.

12.3 Late Policy

Late submissions and late exams are not permitted. Exceptions may be granted at the professor's discretion only in cases of force majeure. If you become ill, experience serious personal difficulties, or encounter technical problems with your computer or internet connection, you must notify the instructors at least 24 hours before the deadline. Failure to do so will result in no extension being granted. Requests made on the day of the deadline will be treated as procrastination and will not be considered. No student will be granted more than one emergency extension throughout the course.

Beginning six hours before any assignment deadline, instructors will enter "silent mode." During this period, no assignment-related questions will be answered, and no requests for office hours will be considered. At all other times, instructors will make every effort to respond to questions and provide assistance, whether via Zoom or in person during office hours.

12.4 Exam and Task Submission Ceremonies

Students must follow all exam and task submission protocols. This means strictly adhering to all rules specified by instructors, whether in written or verbal form. Failure to do so will result in a loss of points.

Throughout your career, you will work with various supporting documents such as contracts and timesheets. Beyond programming, you will also need to use project management systems, application life cycle management tools, version control, and continuous integration and deployment pipelines. You will manage complex configurations, collaborate with colleagues by following specific methodologies, and share code through pull requests or other means. In all these processes, proper etiquette and attention to detail are essential.

In our courses, we aim to help you build these skills. We will deduct points for not following submission rules or protocols, and we may even refuse to accept your

exam defense or submitted tasks. We will also award zero points for missing task deadlines or violating strict exam timing rules, such as arriving late or failing to submit exam documents on time.

12.5 Administrative Drop

Instructors reserve the right to drop a student from the course for non-attendance. If a student misses three or more classes without an acceptable excuse, the faculty may assign an *X* grade and remove the student from the course.

12.6 Incomplete Grade

Consistent with the late exam policy, a grade of *I* (Incomplete) will be granted only under highly exceptional circumstances. Students must initiate the request for an *I* grade well in advance of the final exam; requests made the day before the final exam or later will not be considered.

12.7 Academic Honesty

Plagiarism is the act of copying or appropriating someone else's words or ideas and presenting them as one's own. In the context of this course, plagiarism may occur in many forms, including but not limited to program code, comments, software documentation, design specifications, requirement documents, project reports, and technical analyses.

More specifically, examples of plagiarism in a Software Engineering course include:

- Submitting code written by others or generated by AI as your own
- Modifying existing code or designs (e.g., changing variable names or restructuring code) and claiming originality
- Incorporating code snippets, sentences, design patterns, or intellectual content from any source without citation
- Using algorithms, patterns, or architectural designs without proper acknowledgment
- Using graphics, datasets, audio, video, or other materials from external works without attribution
- Purchasing or otherwise acquiring code, software, or project content and presenting it as original work

Plagiarism is unethical and undermines the educational process. The consequences are as follows:

- First instance: A grade of zero for the plagiarized work, and a report filed with the Registrar's Office
- Second instance: A failing grade (*F*) for the entire course

Both parties, the student who plagiarizes and the student whose work was copied, will face equal consequences, as every student is responsible for safeguarding their assignments, code, and related materials to prevent unauthorized use. This policy reflects real-world expectations in software engineering, where failing to protect intellectual property can result in termination, reputational damage, compromise the safety of customers and users, and even lead to lawsuits in a professional environment.

All work in this course must be completed individually. Group work, collaboration with peers, or seeking assistance from other students is strictly prohibited. At the introductory level, relying on classmates often leads to the spread of misunderstandings and increases the risk of academic dishonesty. Students may seek help only from instructors, teaching assistants (TAs), or WARC tutors, ensuring that guidance comes from qualified sources and that each student develops their own skills and understanding.

The use of artificial intelligence, including but not limited to generative AI tools, to complete any assignments, projects, or exams, either off-site or on-site, is strictly prohibited. If there is suspicion that a student has used AI assistance in their work, the student will be required to perform a similar task and answer relevant questions in a supervised setting with the course instructors. Failure to satisfactorily complete this task or to answer questions convincingly will lead to the conclusion that the work was AI-generated. In such cases, the Academic Honesty policies outlined previously will be enforced, including potential disciplinary actions. Note that for some tasks, the instructor may allow the use of AI tools, but you must obtain explicit written permission in the requirements to use them.

Finally, this course adheres to all global AUCA university policies on academic honesty. If university-wide regulations are stricter than those outlined here, the stricter rules will take precedence.

12.8 Access and Support Services

This course is committed to fostering an inclusive learning environment that supports the diverse needs of all students. If you have a disability or require specific accommodations to participate fully, please contact the instructors as early as possible. We will work with you to ensure that appropriate adjustments are made to support your learning experience.

For guidance on time management, presentation skills, writing, or study strategies beyond the scope of Software Engineering, we encourage you to connect with the AUCA Advising Office. The Advising Office offers practical workshops and peer advising to help you navigate academic challenges.

If you prefer working with peers, you may schedule an appointment with a student tutor through the AUCA Writing and Academic Resource Center (WARC). Tutoring is available both in person and online. Additional resources can be found on the WARC webpage. Please note, however, that team or group work with other students is not permitted for assignments in this course.

If you are feeling stressed, overwhelmed, or struggling with emotions, relationships, or other mental health concerns, we encourage you to seek support from the AUCA Counseling Service. The Counseling Service provides confidential and free professional assistance to students.