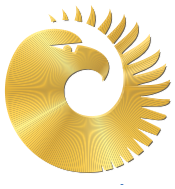

COM-392, System Programming

American University of Central Asia
Software Engineering Department

1 Course Description

The course introduces students to programming multi-core, multi-processor systems, emphasizing data-parallel facilities across various hardware on single machines or distributed systems connected to high-performance networks. Key topics include CPU pipelining to enhance processing efficiency and SIMD (Single Instruction, Multiple Data) for parallel data processing. Students will also learn popular shared-memory parallel programming APIs such as Pthreads and CUDA, alongside a distributed-memory programming API like MPICH.

At the end of the course student should be able to research, analyze, design, develop, and maintain functioning parallel software systems in alignment with the objectives of the AUCA Software Engineering Department and the 510300 IT competency standard.

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 15:35–16:50, Room 410
Lab: Wednesday 14:10–15:25, Room 410

3 Contact Information

Professor
Dmitrii Toksaitov
toksaitov_d@auca.kg

Office
AUCA, room 315

Office Hours
Available by appointment, either on-site or remotely, during work hours (Monday to Friday). Please contact your professor or TA to schedule a meeting.

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 be able to:

1. **Understand and apply fundamental concepts of system programming**:
   - CPU architecture, including caches, locality, pipelines, and branch prediction
   - Data-parallelism using shared and distributed memory systems
2. **Develop skills in parallel programming**:
   - Optimize through data-parallelism using SIMD instructions
   - Write shared-memory parallel programs using APIs like Pthreads
   - Implement distributed-memory programs with MPICH
   - Utilize CUDA for GPU programming
3. **Optimize and debug parallel software systems**:
   - Optimize performance using efficient memory usage

- Debug and profile parallel applications on multi-core and multi-processor systems

4. **Perform effective software development practices**:
    - Research, analyze, and design functional software systems
    - Maintain code repositories using version control and application life-cycle management systems like Git and GitHub
5. **Engage in practical learning through projects and labs**:
    - Complete lab assignments and course projects that demonstrate real-world applications
    - Defend work during final examinations
6. **Collaborate and communicate effectively**:
    - Participate in class discussions and online forums
    - Attend WARC consultation sessions for additional learning and support

6 Assignments and Exams

6.1 Moodle/e-Course Checkpoints

Students are required to maintain private GitHub repositories provided by the instructor for their assignments. They must periodically commit and push a specific number of lab solutions as directed by the faculty. Professors or teaching assistants will review the work either during the lab (on-site) or after the submission deadline (off-site) and assign points based on the completed assignments.

6.2 Labs and Projects

Throughout the course, students will be assigned several projects. Students are expected to defend their work to the instructor during both the Midterm and Final Examination periods.

7 Course Materials, Recordings and Screencasts

All course materials are available on GitHub at `https://github.com/auca/com.392`. By using GitHub, students will gain familiarity with the Git version control system and the widely-used GitHub service among developers.

Every class will be screencasted and uploaded to YouTube for student accessibility, though it's important to note that we do not guarantee every class will be recorded. Recordings will be done on a best-effort basis as time permits. Consider recording the class videos on your own computer if you need them to be available promptly. YouTube recordings can be located in the course repository at `https://github.com/auca/com.392`. While recordings provide flexibility, they should not be a substitute for attending classes. Active participation is crucial for success in this course. Each unexcused absence will result in a one-point deduction from your grade. Accumulating

five or more unexcused absences may lead to an X grade. If overall attendance is poor, the instructor reserves the right to discontinue class recordings.

Access the course lectures remotely via Zoom at `http://com-392-zoom.auca.space`. When joining the Zoom session, students must identify themselves by providing their first and last names in Latin characters, properly capitalized.

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.

9 Reading

1. An Introduction to Parallel Programming by Peter Pacheco (ISBN: 978-0123742605)
2. Parallel Programming, 2nd Edition by Thomas Rauber and Gudula Rünger (ISBN: 978-3642048173)

9.1 Supplemental 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)

10 Grading

10.1 GitHub Checkpoints

Your instructor will announce a periodic review of your work. You will be awarded up to the following number of points for such checks.

- Project (35%)

10.2 Exams

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

10.3 Totals

- 100% is formed from the GitHub submissions (35%) and the two exams (65%).

10.4 Scale

- [92%–100]%: A
- [85%–92)%: A-
- [80%–85)%: B+
- [75%–80)%: B
- [70%–75)%: B-
- [65%–70)%: C+
- [60%–65)%: C
- [55%–60)%: C-
- [50%–55)%: D+
- [45%–50)%: D
- [40%–45)%: D-
- Less than 40%: F

Please note that requests for a higher grade due to points being marginally close will be ignored. For instance, 91.99 is an A-, NOT an A. Similarly, requests for extra assignments to boost points will also be disregarded.

11 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.

11.1 Participation

Active work during the class may be awarded with extra points at the instructor's discretion. Poor student performance during a class can lead to points being deducted from the final grade.

Instructors may conduct pop-checks during classes at random without prior notice. Students MUST be ready for every class in order not to lose points. Students absent without a good reason from such classes with graded work will also lose points unless it is force-majeure circumstances. Instructors must be notified in advance about why a student is absent not to lose points.

11.2 Questions

We believe that a question from one student is most likely a question that other students are also interested in. That is why we encourage students to use the online discussion board of the LMS (Learning Management System) that you use (e.g., AUCA e-Course System) to ask questions in public that other students can see and answer. We discourage students from asking questions through E-mail. If it is a private matter, write direct messages to your instructor through the LMS system too. We will not be answering most E-mail messages this semester (unless it is a severe emergency) to consolidate all the course correspondence in one place.

Do not post the complete source code for any task on the LMS discussion board. You will get zero for that work for any such public post. Do not ask generic questions about your code to know why it does not work. Please spend some time thinking about your code, debugging it.

11.3 Late Policy

Late submissions and late exams are not allowed. Exceptions may be made at the professor's discretion only in force-majeure circumstances. If you got ill, got severe personal issues, got problems with your computer or the Internet, you MUST notify instructors at least 24 hours in advance. Otherwise, we will not give you an extension. We will consider that you were procrastinating until the very last day. We will also not be giving more than one emergency extension throughout the course.

Six hours before the deadline for any work on the course, instructors will go into a silent mode. No questions will be answered about the work that has to be submitted, no requests to have office hours will be considered. However, at any other work time before the deadline, we will try our best to answer your questions and help you through Zoom or in our office.

11.4 Exam and Task Submission Ceremonies

Students MUST follow exam and task submission ceremonies. It means they MUST strictly follow all the rules specified by the instructors in written or verbal form. Failure to do so will result in lost points. Throughout your career, you will have to work with various supporting documents (contracts, timesheets, etc.). It is a good idea to start learning to work with such documents accurately early. We will remove points for not following these rules or even refuse to accept your exam defense or tasks submitted to us. We will also give zero for not following deadlines or the strict exam timing rules.

11.5 Administrative Drop

Instructors have a right to drop a student from the course for non-attendance. If you have five classes or more missed without an excuse, the faculty may consider dropping you by giving you the X grade.

11.6 Incomplete Grade

Similar to the policy for late exams, the grade I may be awarded only in highly exceptional circumstances. Students MUST initiate a discussion about receiving an I grade with the instructors well in advance and NOT during the last week before final exams.

11.7 Academic Honesty

Plagiarism is the act of copying or stealing someone else's words or ideas and presenting them as one's own. This definition encompasses various task elements, including but not limited to program code, comments, software documentation, abstracts, reports, diagrams, and statistical tables.

The following are examples of plagiarism in the context of a Software Engineering course:

- Presenting code written by others or AI as your own
- Purchasing code, software, or any project-related content from online platforms or other sources and submitting it as your own creation
- Using algorithms, patterns, or architectural designs without acknowledging the source
- Incorporating code snippets, sentences, design patterns, or any intellectual content from sources, published or unpublished, without proper citation
- Modifying someone else's code or design (e.g., changing variable names, changing the structure of the code) and claiming it as original
- Utilizing graphics, data sets, audio, video, or other elements from external works without proper acknowledgment.

Engaging in plagiarism is not only unethical but also undermines the educational process. The consequences for plagiarism in this course for all parties involved are as follows:

- First instance: The students will receive a grade of zero for the plagiarized work, and a report will be filed with the Registrar's Office.
- Second instance: The students will receive an F grade for the entire course.

It's important to note that both parties involved in plagiarism—the one who plagiarizes and the one whose work was copied—will face equal consequences. This underscores the imperative for honest students to exercise caution in ensuring the security of their work. It is the student's responsibility to guarantee that their assignments, code, or any related content can only be accessed by them and the course instructors. Sharing, unintentionally exposing, or not securely storing one's work can lead to unintended consequences and sanctions.

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.

Students are advised against rote memorization of code for examinations. Relying solely on memorization is an ineffective learning strategy in programming. Examinations in this course may contain open-ended questions targeting the student's analytical and design skills, and memorization may lead to answers that are off-target and of subpar quality.

In addition to the rules outlined in this syllabus, we abide by all global university policies concerning plagiarism. Should the global university rules evolve to be more consequential or stringent than what is stipulated here, those university-wide regulations will take precedence over our course-specific rules.

11.8 Access and Support Services

In this course, we are committed to providing an inclusive learning environment that accommodates the diverse needs of all students. If you have a disability or require specific accommodations to participate fully in this course, please contact us as early as possible to discuss your needs. We will work together to ensure that appropriate adjustments are made to support your learning experience.

If you need guidance on improving your time management, presentation, writing skills, and study skills beyond the scope of Software Engineering parlance, we encourage you to connect with the Advising Office. The Advising Office offers a wide range of practical and creative workshops, and peer advisors can help you navigate the academic environment.

If you feel more comfortable working with other students, you can make an appointment with a student tutor from the WARC (Writing and Academic Resource Center). You can meet with them face-to-face or online. Visit the WARC webpage for more resources. Remember, team/group work with other students in this course is not allowed.

If you feel stressed, overwhelmed, find it difficult to manage your emotions, socialize, or have concerns related to your mental health, please consult the AUCA Counseling Service. There are excellent professionals working there; your visits are completely confidential and free.