

COM 421, 430: Software Engineering I, II

American University of Central Asia
Software Engineering Department

1 Course Information

Course Codes

COM-421, COM-421.1
COM-430, COM-430.1

Course IDs

3705
3881

Prerequisite

COM-117, Programming II. Intro to Object Oriented Programming
or
COM-119, Object-Oriented Programming

Prerequisite for

COM-430, Software Engineering II

Credits

6

Time and Place

Lecture: Tuesday 14:10–15:25, Room 410
Lab: Thursday 14:10–15:25, Room 410

Course Materials

<https://github.com/auca/com.421-430>

2 Contact Information

Professor

Dmitrii Toksaitov
toksaitov_d@auca.kg

Office

AUCA, room 315

Office Hours

By appointment throughout the work week (write to your professor to make an appointment for any day from Monday to Friday)

3 Course Overview

The course introduces students to the various phases of the software engineering life cycle: requirements elicitation and analysis, design, coding, testing, and software delivery. Beyond these core concepts, students will also explore and practice with essential software engineering tools, including version control systems (VCS), testing tools, continuous integration/continuous delivery (CI/CD), and application lifecycle management (ALM) systems. Throughout the course, students will gain hands-on experience with all stages of software development through two major projects. This two-semester course is designed for students majoring or minoring in Software Engineering.

By the end of the course, students will be equipped to research, analyze, design, develop, and maintain functional software systems aligning with the objectives of the AUCA Software Engineering Department and the 510300 IT competency standard.

4 Topics: Software Engineering I

- Week 1–3: Software Engineering Concepts (9 hours)
- Week 4–8: Software Development Tools and Technologies (15 hours)
- Week 9–11: Software Development Methodologies (9 hours)
- Week 12–14: Agile Software Development (9 hours)
- Week 15–16: Project Management (6 hours)

5 Topics: Software Engineering II

- Week 1–3: Requirements Engineering (9 hours)
- Week 4–6: System Design and Modeling (9 hours)
- Week 7–9: System Implementation (9 hours)
- Week 10–12: Software Testing (9 hours)
- Week 13–14: Software Evolution (6 hours)
- Week 15–16: Dependability and Security (6 hours)

6 Assignments and Exams

6.1 GitHub/Moodle Checkpoints

Students are required to maintain personal, private GitHub repositories provided by the instructor for all their assignments. They must periodically commit and push a specific number of lab and project solutions as directed by the faculty. Either the

professors or teaching assistants will check the work regularly and assign points in Moodle based on the completed assignments.

6.2 Labs and Projects

Throughout the course, students will be assigned several laboratory tasks and must complete course projects. They are expected to present and defend their work to the instructor during the midterm and final examination sessions.

Laboratory tasks are aligned with topics covered in lectures. Some tasks must be completed during class sessions to be eligible for grading.

Each semester, the course will include a project. For these projects, students are tasked with developing a software product for a fictional client. Student teams will address and provide solutions to designated problems, taking them through the complete software development lifecycle—from requirements specification and software design to construction, testing, deployment, and maintenance. This process offers students the opportunity to engage with agile methodologies like Scrum and Kanban, use Version Control Systems (VCS) like Git, interact with project management tools, learn software testing techniques, explore Test-Driven Development (TDD), and gain experience with Continuous Integration (CI) and Continuous Delivery (CD).

7 Course Materials, Recordings and Screencasts

All course materials are available on GitHub at <https://github.com/auca/com.421-430>. 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. YouTube recordings can be located in the course repository at <https://github.com/auca/com.421-430>. 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 via Zoom at <http://com-421-zoom.auca.space>. When joining the Zoom session, students must properly 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.

- IntelliJ IDEA Ultimate: <https://www.jetbrains.com/idea/download>
- Git with SSH: <https://git-scm.com/downloads>

Most of the tools required for the course will be available on the remote course server.

9 Reading

Software Engineering (9th or 10th Edition) by Ian Sommerville (AUCA Library Call Number: QA76.758.S657 2011, ISBN: 978-0137035151 or 978-0133943030)

9.1 Supplemental Reading

1. The Mythical Man-Month: Essays on Software Engineering, Second Edition by Frederick P. Brooks Jr. (ISBN: 858-0001065793)
2. Extreme Programming Explained: Embrace Change, 2nd Edition by Kent Beck, Cynthia Andres (ISBN: 978-0321278654)
3. Essential Scrum: A Practical Guide to the Most Popular Agile Process by Kenneth S. Rubin (ISBN: 007-6092046028)
4. Test Driven Development: By Example by Kent Beck (ISBN: 978-0321146533)
5. Code Complete: A Practical Handbook of Software Construction by Steve McConnell (AUCA Library Call Number: QA76.76.D47M39 2004, ISBN: 079-0145196705)
6. Design Patterns: Elements of Reusable Object-Oriented Software by Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides (AUCA Library Call Number: QA 76.64 D47 1995, ISBN: 978-0201633610)
7. Refactoring: Improving the Design of Existing Code by Martin Fowler, Kent Beck, John Brant, William Opdyke, Don Roberts (AUCA Library Call Number: QA76.76.R42 F695 1999, ISBN: 978-0201485677)

10 Grading

10.1 GitHub/Moodle Checkpoints

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

- Labs (5%)
- Project (40%)

10.2 Exams

- Midterm Exam (25%)
- Final Exam (30%)

10.3 Totals

- 100% is formed from the GitHub/Moodle submissions (45%) and the two exams (55%).

11 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.

12 Rules

Students are required to follow the rules of conduct of the Software Engineering Department and the American University of Central Asia.

12.1 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.

12.2 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.

12.3 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.

12.4 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.

12.5 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.

12.6 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.

12.7 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 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.

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.