COM 122, 119: Introduction to Programming, Object-oriented Programming

American University of Central Asia
Software Engineering Department

1 Course Information

Course Codes

COM-122
COM-119

Course IDs

5682
4357

Prerequisites for

The list below depends on the student's year of admission.

COM-118:

COM-119, Object-oriented Programming
COM-123, Principles of Computing Systems

COM-119:

COM-213, Database
COM-223, Algorithms and Data Structures
COM-229, Data structures
COM-421, Software Engineering I
COM-431, Senior Thesis I
MAT-407, Numerical Methods
Various elective courses

Credits

6

Professors, Time, Place

Dmitrii Toksaitov

Lecture: Monday 10:50–12:05, CH 440
Lab: Wednesday 10:50–12:05, Lab 432

Anatoliy Ignatev

Lab: Wednesday 12:45–14:00, Lab 433
Lab: Wednesday 14:10–15:25, Lab 433

Viacheslav Muravev
Lab: Friday 10:50–12:05, Lab G31

TBD
Lab: Thursday 10:50–12:05, Lab G30
Lab: Thursday 12:45–14:00, Lab G30

Course Materials
`https://github.com/auca/com.122-119`

2 Contact Information

Professors
Dmitrii Toksaitov
toksaitov_d@auca.kg
Anatoly Ignatiev
ignatev_a@auca.kg
Viacheslav Muravev
muravev_v@auca.kg

TAs
Luna Maltseva
md12366@auca.kg
Sofia Kan
ks12246@auca.kg

Office
AUCA, Room 315

Office Hours
By appointment throughout the work week (write to your professor or TA to make an appointment during business hours for any day from Monday to Friday)

3 Course Overview

This course aims to equip students with the essential skills required for structured and object-oriented programming. Upon completion of the course, students should understand key programming concepts such as memory management, flow control, functions, procedural decomposition, objects, classes, inheritance, and polymorphism. They should also be able to write simple applications using the basic features of the C++ programming language and consistently apply principles of good programming practices.

4 Topics: Introduction to Programming

- Week 1–2: Introduction to the Process of Software Development (6 hours)

- Week 3–6: Selections (12 hours)
- Week 7–10: Loops (12 hours)
- Week 11–13: Methods (9 hours)
- Week 14–16: Single- and Multidimensional Arrays (9 hours)

5 Topics: Object-oriented Programming

- Week 1–3: Objects and Classes (9 hours)
- Week 4–6: Inheritance and Polymorphism (9 hours)
- Week 7–8: Abstract Classes and Interfaces (6 hours)
- Week 9–10: Exception Handling (6 hours)
- Week 11–12: GUI and Computer Graphics Basics (6 hours)
- Week 13–14: Generics and Container Classes (6 hours)
- Week 15–16: Working with I/O (6 hours)

6 Learning Outcomes

By the end of this course, students will be able to:

1. **Discover basic programming concepts**:
    - Research basics of memory management, flow control in structured programming, functions, and procedural decomposition
2. **Utilize and evaluate appropriate tools and techniques**
    - Utilize code editors, IDEs, build systems, version control systems, standard and 3'rd party libraries
3. **Demonstrate proficiency in Object-oriented development**:
    - Define and use objects and classes
    - Implement inheritance and polymorphism
    - Utilize abstract classes and interfaces
4. **Assess fundamental data structures and algorithms**:
    - Create and manipulate single- and multidimensional arrays
    - Develop programs utilizing generics and container classes
5. **Design and develop graphical user interfaces and graphical applications**:

- Understand the basics of GUI, GUI libraries, and computer graphics
- Integrate graphical elements into software applications

6. **Develop robust and error-free code**:
    - Implement and use exception handling mechanisms
    - Test and debug applications to discover bugs
7. **Adapt effective software development practices**:
    - Apply principles of good programming practices
    - Follow best software development methodologies
    - Maintain code repositories using version control and application life-cycle management systems like Git and GitHub
8. **Engage in project-based learning**:
    - Complete lab assignments and course projects that demonstrate real-world applications
    - Defend their work in person during final examinations
9. **Collaborate and communicate effectively**:
    - Participate in class discussions and online forums
    - Attend WARC consultation sessions for additional learning and support

7 Assignments and Exams

7.1 Moodle/e-Course Checkpoints

Students are required to maintain 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 based on the completed assignments.

7.2 Labs and Projects

Throughout the course, students will be assigned several laboratory tasks. Additionally, they are required to develop one or two course projects that demonstrate real-world applications. Students are expected to defend their work to the instructor during both the Midterm and Final Examination periods.

8 Course Materials, Recordings and Screencasts

All course materials are available on GitHub at `https://github.com/auca/com.122-119`. 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.122-119`. 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-122-zoom.auca.space` or `http://com-119-zoom.auca.space`. When joining the Zoom session, students must identify themselves by providing their first and last names in Latin characters, properly capitalized.

You lab instructor may use some other tools to work remotely. Consult your teacher for get more information.

9 Software

Students are advised to install the following software on their machines at the start of the course. Additional installations may be required later.

- CLion: `https://www.jetbrains.com/clion/download`
- Git: `https://git-scm.com/downloads`

10 Reading

1. Introduction to Programming with C++, 3rd edition by Y. Daniel Liang (AUCA Library Call Number: QA76.73.C153 L53 2014, ISBN: 978-0133252811)

10.1 Supplemental Reading

1. The C++ Programming Language, Fourth Edition by Bjarne Stroustrup (AUCA Library Call Number: QA76.73.C153 S77 2013, ISBN: 978-0275967307)
2. A Tour of C++, Third Edition by Bjarne Stroustrup (AUCA Library Call Number: QA76.73.C153 2023, ISBN: 978-0136816485)

11 Grading

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.

- Lab Submissions (20%)
- Online Judge Problems (10%, COM-122 only)
- Class Work (10%)
- Project #1 (10%)
- Project #2 (10%, COM-119 only)

11.2 Exams

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

11.3 Bonus Opportunities

- Bonus Points for Attending WARC or Extra Lecture Sessions (5%)

11.4 Totals

- 100% is formed from the Moodle/e-Course Checkpoints (50%) and the exam sessions (50%). You can earn up to 5% as a bonus by attending WARC or Extra Lecture Sessions conducted by your instructors.

11.5 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

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.

You can attend WARC consultation sessions conducted by students or Extra Lectures Sessions led by instructors. For each session you attend, you will earn 1 extra point (but no more than 5 in total). We will collect your WARC attendance reports at the end of the course and add these to your course points to calculate your final grade.

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.

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.

Students must ensure that the code they submit can be compiled and run without errors on most modern platforms, including the latest versions of Windows, macOS, and GNU/Linux. The argument 'it works on my machine' will be ignored. Submissions should be thoroughly tested across different environments to guarantee compatibility.

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.