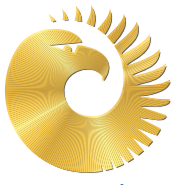

COM 122, 119: Introduction to Programming and Object-oriented Programming

American University of Central Asia
Software Engineering Department

1 Course Description

This two-semester required course provides a foundational introduction to programming, focusing on structured and object-oriented principles using the C++ language. Offered by AUCA's Software Engineering Department, it serves as a prerequisite for most advanced courses and is also suitable for students seeking a general introduction to programming.

Students will learn essential concepts, including memory management, flow control, functions, procedural decomposition, objects, classes, encapsulation, inheritance, and polymorphism. The course emphasizes hands-on practice in developing C++ applications while reinforcing fundamental development tools such as code editors (or IDEs), compilers, build systems, package managers, debuggers, and version control systems. By the end of the course, students will be equipped to write efficient, well-structured programs and apply good programming practices consistently.

2 Course Details

Course Materials

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

Course Codes

COM-122
COM-119

Course IDs

5682
4357

Prerequisites

COM-122:

None

COM-119:
COM-122, Introduction to Programming

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

COM-122:
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 G30

Anatoliy Ignatev
Lab: Monday 14:10–15:25, Lab G31
Lab: Monday 15:35–16:50, Lab G31

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

3 Contact Information

Professors

Dmitrii Toksaitov
toksaitov_d@auca.kg

Anatoliy Ignatev
ignatiev_a@auca.kg

Vyacheslav Muravev
muravev_v@auca.kg

TAs

Luna Maltseva (Teacher Assistant)
md12366@auca.kg

Sofia Kan (Administrative Assistant)
ks12246@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

4.1 Introduction to Programming

- Weeks 1–2: Introduction to the Process of Software Development (6 hours)
- Weeks 3–6: Selections (12 hours)
- Weeks 7–10: Loops (12 hours)
- Weeks 11–13: Functions (9 hours)
- Weeks 14–16: Single- and Multidimensional Arrays (9 hours)

4.2 Object-oriented Programming

- Weeks 1–3: Objects and Classes (9 hours)
- Weeks 4–5: Exception Handling (6 hours)
- Weeks 6–8: Encapsulation, Inheritance, and Polymorphism (9 hours)
- Weeks 9–10: Abstract Classes and Interfaces (6 hours)
- Weeks 11–12: GUI and Computer Graphics Basics (6 hours)
- Weeks 13–14: Generics and Container Classes (6 hours)
- Weeks 15–16: Working with I/O (6 hours)

5 Learning Outcomes

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

1. **Acquire basic programming skills to write and test programs**:
   - Write programs using fundamental C++ language constructs
   - Apply debugging and testing techniques to ensure correctness
2. **Develop proficiency in modeling object-oriented systems**:
   - Represent software designs using object-oriented principles
   - Use appropriate modeling techniques (e.g., UML) to describe classes, objects, and relationships

3. **Evaluate and use appropriate tools and techniques for software development**:
    - Use code editors, compilers, build systems, debuggers, IDEs, and other development tools effectively
    - Employ best practices in version control and collaborative development
4. **Design, plan, and critically evaluate software solutions to problems**:
    - Work with user requirements and constraints
    - Assess possible solutions for correctness and maintainability
5. **Assess systems in terms of quality attributes and trade-offs**:
    - Use automated testing and code analysis tools to evaluate software quality
    - Weigh design decisions against project constraints and overall goals
6. **Reflect on and reason about information handling problems**:
    - Critically examine data structures and algorithms for effective information management
    - Consider privacy, data protection, and ethical implications of software solutions
7. **Understand and apply ethical principles and professional standards**:
    - Demonstrate awareness of legal and ethical issues in computing
    - Practice professional conduct and responsibility in the design and implementation of software

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 OJ Problems and Projects

Throughout the course, students may be assigned problems from an Online Judge (OJ) platform, such as Codeforces, to enhance their problem-solving skills and prepare for programming contests or job interviews. Additionally, they will be required to develop one or two course projects that demonstrate real-world applications. Students must present and 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.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. 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.

8 Software

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

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

9 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)

9.1 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)

10 Grading

10.1 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 and Class Work (16%)
- Online Judge Problems (16%, COM-122 only)
- Project #1 (13%)
- Project #2 (16%, COM-119 only)

10.2 Exams

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

10.3 Bonus Opportunities

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

10.4 Totals

- 100% is formed from the Moodle/e-Course Checkpoints (45%) and the exam sessions (55%). You can earn up to 5% as a bonus by attending WARC or Extra Lecture Sessions conducted by your instructors.

10.5 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.

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.

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.

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.

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.

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.

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.

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.