COM 299, Game Development

American University of Central Asia
Software Engineering Department

1 Course Information

Course Code
COM-299

Course ID
3956

Prerequisite
COM-119, Object Oriented Programming

Credits
6

Time and Place
Lecture: Monday 15:35–16:50, Room 410
Lab: Wednesday 15:35–16:50, Lab G30

Course Materials
`https://github.com/auca/com.299`

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 or TA to make an appointment during business hours for any day from Monday to Friday)

3 Course Overview

The course introduces students to the topic of game development, covering both the theory and practice of video game production. It explores fields such as computer graphics, computational physics, artificial intelligence, and gameplay design. During the course, students will have the opportunity to build two market-ready games for desktop, web, or mobile platforms. They will learn not only how to create their own lightweight graphics, physics, and gameplay engines but also how to use heavyweight third-party solutions such as Unity 3D, Unreal Engine, or Godot.

4 Topics Covered

- Week 1: Introduction, History, Industry Overview (3 hours)
- Week 1–2: The Unity Engine (6 hours)
- Week 1–7: C# Scripting (21 hours)
- Week 2: Vectors (3 hours)
- Week 3: Matrices (3 hours)
- Week 4: Space Transformation (3 hours)
- Week 5–9: The Unity OOP Model (15 hours)
- Week 10–11: The Graphics, Physics and UI Subsystems (6 hours)
- Week 12–13: The AI Subsystems (6 hours)
- Week 14–16: Game Engine Internals (9 hours)

5 Learning Outcomes

By the end of this course, students will

- acquire basic programming skills to write and test games
- develop proficiency in modeling object-oriented game systems
- evaluate and use appropriate tools and techniques for game software development
- design, plan, and critically evaluate game software solutions to problems
- evaluate game systems in terms of quality attributes and trade-offs

6 Assignments and Exams

6.1 Moodle/e-Course Checkpoints

Students are required to maintain a personal, private GitHub repository provided by the instructor for all their assignments. They must periodically commit and push a specific number of 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.

6.2 Projects

Throughout the course, students will be required to create two games independently. The first game will be a recreation of a classic game to study development environments. The second game will be a clone of a currently popular product on the market. Students are expected to present and defend their work to the instructor during the midterm and final examination sessions.

6.3 Presentation

Students will have the opportunity to present about a market game of their choice. The presentation should focus on the game's internals, its development or production process, and the tools or techniques used in its creation.

7 Course Materials, Recordings and Screencasts

All course materials are available on GitHub at `https://github.com/auca/com.299`. 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.299`. 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-299-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 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`
- Unity: `https://unity.com`
- JetBrains Rider: `https://www.jetbrains.com/rider`

9 Hardware

We have taken great care in selecting a game engine that should work on a wide range of machines, from high-end personal computers to low-powered notebooks. However, each year we encounter students whose computers have outdated GPUs or buggy drivers that are unable to run the software. Unfortunately, this means you may need to invest time in finding a compatible environment for our primary course software tool. Please be aware that we cannot offer extensions or preferential treatment if your machine is unable to handle it.

10 Reading

1. 3D Math Primer for Graphics and Game Development, Second Edition by Fletcher Done and Ian Parberry (ISBN: 978-1439869819)

10.1 Supplemental Reading

1. Mathematics for 3D Game Programming and Computer Graphics, Third Edition by Eric Lengyel (AUCA Library Call Number: QA76.76. C672 L46 2012, ISBN: 978-1435458864)
2. Foundations of Game Engine Development, Volume 1: Mathematics by Eric Lengyel (AUCA Library Call Number: QA76.76. C672 L46 2012, ISBN: 978-0985811747)
3. Foundations of Game Engine Development, Volume 2: Rendering by Eric Lengyel (AUCA Library Call Number: QA76.76. C672 L462 2019, ISBN: 978-0985811754)
4. Game Programming Patterns by Robert Nystrom (ISBN: 978-0990582908)

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.

- Project #1 (15%)
- Project #2 (20%)
- Presentation (10%)

11.2 Exams

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

11.3 Totals

- 100% is formed from the Moodle/e-Course Checkpoints (45%) and the two exams (55%).

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

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.

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.