המרכז האקדמי לב

Department of Computer Science
Academic Syllabus

Course Number:	**151055**
Course Name:	**Mini project in Introd. to Software Engineering**
Course components:	**Lab: 2 hrs**
Credits:	**1.5 credits**

Overview: This course is one of a kind introductory class to software engineering, design and architecture. The course is a lab class for the course 151050 – Introduction to Software Engineering. This is not an easy course. It requires an understanding of abstract concepts and the implementation of vast amount of code. However, we will approach the end goal carefully and gradually, allowing you to slowly adapt to the course level, and gain as much insights as possible from the course material. During the course, you will practice the main responsibility of a software engineer - to create a coherent and well-designed software architecture toward the implementation of a large project. This responsibility require knowledge in both software and problem domains. You will practice the art of learning a new discipline, to extract an abstract model of the problem and to design an ever-evolving solution that realizes it. The course is based on designing and modeling a virtual 3-dimensional graphical scene, with all the physics involved (light source, rays, reflections, refractions, color, occlusions etc.). We will spend some time understanding the problem domain and invest a lot of thought in designing and implementing a solution. You will constantly practice refactoring of design and code, as it is usually the case in the world of software engineering. No matter how hard it will get, the result in the end will be undoubtedly breath-taking.

Course Objective:

- To gain hands-on practice in acquiring proficiency in a new problem domain and to develop abstract description of it
- To develop understanding and appreciation of complex software projects
- To understand the basic mechanisms of object-oriented modelling and abstraction
- To understand the flow from an idea to code
- To understand and apply principles of class design
- To develop an intuition of basic design patterns to address design challenges
- To understand and apply techniques supporting Agile Development: Refactoring, Testfirst programming and Pair programming.

Course Description:

Presence in all the lessons is mandatory for successful completing of this course (in accordance to the "Takanon").

The **preliminary assignment** must be performed **individually by each student** and it is a **mandatory condition for continuing the course** (students that don't complete the assignment properly and individually before the second lab are not allowed to continue).

All the following assignments are progressive stages of the final mini-project – there is no possibility to start any phase without full and proper completing of all the previous phases. The work is done by **teams of two students** (it is not allowed for more than two students to work together in a team). All teams must be in-group (it is not allowed for two students from different groups to form a team). If there is odd number of students in a group, the teacher will decide who will work alone in this course (only one student in the group!). The work of the teams is done in full accordance with Extreme Programming model (including Pair Programming). **The projects must be maintained in a cloud git site** (e.g. github or gitlab).

Week 1. ***Preliminary assignment 0***: Basic Java OOP exercise. ***Grade***: Pass

Week 2. ***Project phase 1***: Implementation of primitives and geometries. ***Grade***: 10pt.

Week 3. ***Project phase 2***: Implementation of Vector operations tests. **Geometry** interface and implementation of body's normal vector calculation and tests for it. ***Grade***: 10pt. **Bonus**: 1pt. for implementing Finite Cylinder normal.

Week 4. ***Project phase 3***: Implementation of Vector operations unit tests. Refactoring of Geometry interface by introducing **Intersectable** interface. Implementation of class **Geometries** and implementation of composite pattern for finding Ray/Intersectable intersections operation. Implementation of unit tests for intersections. ***Grade***: 10pt (**Bonus**: 1pt.+1pt. for Infinite & Finite Cylinder intersections with ray)

Week 5. ***Project phase 4***: Implementation of Camera class and building rays from Camera through View Plane, including unit tests. Integration tests of ray construction and ray intersections with geometries. ***Grade***: 10pt.

Week 6. ***Project phase 5***: Implementation Color wrapper , AmbientLight, Scene and Render classes. Integration of Image Writer class. Testing of building basic picture with Ambient Light. ***Grade***: 10pt. (**Bonus**: 2pt for full XML support)

Week 7-8. ***Project phase 6***: Refactoring ray-geometry intersections and color calculation. Implementing of direct/point/spot light sources, multiple light sources and simple Phong model. Advanced pictures for testing lighting. ***Grade***: 10pt.

Week 9-10. ***Project phase 7***: Basic Shadowing implementation. Recursive implementation of ray reflections and refractions. Refactoring of shadowing. Advanced pictures for testing basic shadows, reflections, refractions and a picture of integrated testing of all these effects. ***Grade***: 10pt. ***Bonus***: 1pt. for completion of this phase by week 10. In this case - starting the next phase in week 10.

Week 11. ***Mini-Project 1***: Implementing of one of the picture improvement algorithms. ***Grade***: 10pt. (This part will have some updates after Passover 5780)
There are 4 levels of projects, each one appropriate allowing entry level for Mini-Project 2 level. The level is decided by the teacher according to the student's achievements in the previous phases. NB: Choosing level which is not adjusted to a specific student may not allow enough time of one or both mini-projects completion in time. The mini-project grading is based on making appropriate architecture decisions as well as on the quality of the results. It is mandatory requirement to implement highly loaded picture(s) for this mini-project – including dozens of different bodies.

- ***Level 1:*** Super-sampling (allows Mini-Project 2 max level 3)
- ***Level 2:*** Soft shadows (allows Mini- Project 2 max level 4)
- ***Level 3:*** Depth of field (allows Mini- Project 2 max level 5)
- ***Level 4:*** Glossy surface and Diffuse Glass (allows Mini- Project 2 max level 5)

Week 12-13. ***Mini-Project 2***: Implementing of one of the optimization algorithms. ***Grade***: according to project level up to 20pt. (This part will have some updates after Passover 5780)
There are 5 levels of projects, each one appropriate allowing entry level for Mini-Project 2 level. The level is decided by the teacher according to the student's achievements in the previous phases. NB: Choosing level which is not adjusted to a specific student may not allow enough time of one or both mini-projects completion in time.

- ***Level 1:*** Multithreading with thread pools (***Grade***: 5pt. if alone)
  NB: this task is a 3pt. part for all other levels. (this 3pt. is included in the below grades)
- ***Level 2:*** Conservative Boundary Volume (***Grade***: 7+3pt.)
- ***Level 3:*** Boundary Volume Hierarchy (manual hierarchy) (***Grade***: 9+3pt.)
- ***Level 4:*** Adaptive Super-sampling (above the algorithm implemented in mini-project 1) (***Grade***: 15+3pt.)
- ***Level 5:*** Boundary Volume Hierarchy (automatic generation of optimized hierarchy) (***Grade***: 17+3pt.)
- ***Level 5:*** Regular Grids and 3DDDA (***Grade***: 17+3pt.)

Evaluation criteria: Work evaluation 100%

Prerequisite Courses:

- Linear Algebra
- Introduction to Computer Science
- C++ Workshop or Mini-Project in Windows System
- Introduction to Software Engineering (possible to do in parallel)