Project

- graded together with Laboratory (half of effects, so it is mandatory)

Scope (=Topic Group,to choose):

1. Requirements analysis, design of the system (structure, logic, graphical design)
2. All phases of IT project, but limited functionality (requirements, design, implementation, testing). Source code documentation is plus (doxygen, javadoc…). Gitlab repository will be available. Activity distributed through the more semester will be preferred over “projects of last night”
3. Scope 1 or 2 can be performed in agile methodology (user stories, backlog, tasks, report date when task state was changed), as well as RUP methodology
4. Apply design thinking (5 x why?, prototypes, user questionnaire)

Note, that when you do the implementation, the design and tests should take the number of pages at least similar to implementation (implementation chapter - when you describe most interesting places in implementation, show views, work of program)

Deadlines:
23.04 -project description (for teacher acceptance)
11.06 (6LID-A, ASL01), 18.06 (6LID-A, ASL01;4LID-A,ASLInop01), project submission, project defence

Small groups (2 students) possible, but the total complexity 2x bigger

Formal requirements: 2 posts at Blackboard forum (Project_Laboratory_Software Engineering)
Mandatory: documentation, project defense

Documentation (recommended content, more means better, also refer to project scope)
- problem statement
- requirements analysis
- design of structure, activity, architecture (communication interfaces), not only diagrams but there should be explanations
- graphical design (GUI)
- more interesting places of implementation
- testing
- code documentation as an attachment, its complexity is evaluated (not recommended: extraction of methods, but no description)

If the code was produced it has to be attached (*.zip or through repository). If graphic was not attached to documentation it has to attached too.

Minimal requirements:
Topic group 1- 4 diagrams, 3 use cases.
Topic group 2- 2 diagrams, 3 use cases, 1 screen.
Feedbck

Content of laboratories:

1. Requirements specification. Construction and modeling of software components with the use of use case diagram notation.

2. Methods of building module tests with the use of dedicated structural unit testing tools.Verification and validation of the produced software product.
3. Building and modeling of the software components using the class diagram notation.
4. Building and modeling of the software components using the state diagram notation.
5. Building and modeling of the software components using the activity diagram notation.
6. Automation of the software development and software integration process (tool will be chosen). Creating API code documentation (doxygen).
7-8.Design patterns and their implementation.
9 (??) Communication software interfaces.