Chair of Computational Modelling and Simulation
Department of Civil, Geo and Environmental Engineering
Technical University of Munich

Software Lab 2022 - Guidelines

1. Software lab

This course is a master seminar over two semesters. For this seminar, the students work together in groups of three on a specific research topic, supervised by research/teaching assistants. In many cases, the topics arise from engineering practice and are executed and managed in cooperation with the industry.
The core idea of this course is to let the students implement a piece of software that is part of typical engineering applications to confront them with specific questions from computational problem solving to real-world problems.
Three presentations/reviews are equally distributed over the working period to show progress and problems. In addition to a documented code, a scientific report of the software developed during the seminar must be submitted.

2. Selection process

Every student that wants to participate must:

1) Be registered for the course BV030004 in TUMOnline (from the 15th of March).
2) Choose eight topics on the main web page of the Chair for Computational Modelling and Simulation (https://www.cms.bgu.tum.de/de/lehrveranstaltungen/master/softwarelab):
    - 2 Gold topics
    - 2 Silver topics
    - 2 Bronze topics
    - 2 No-go topics

At the end of the selection period, groups of 3 (maximum 4) students will be formed. The topics are distributed according to the most desired ones for a student. The "No-go" topics won't be assigned.
Requests to have specific team members will not be considered when distributing the groups.

3. Tasks

In the first arranged meeting, the supervisors will explain the detailed content of the tasks together with the specified evaluation criteria. All necessary information, literature, and project tools will be provided.

4. General Workflow

- *Meetings* with supervisor(s):
    - Regular meetings are highly encouraged (at least once per two weeks).
    - A meeting protocol should document every meeting. This protocol should be submitted to the supervisor at the end of the course.
    - A meeting protocol can be done using an issue tracker (like JIRA), paper, or Excel.
- *Code:* Write clean code and document sufficiently (preferably using proper documentation tools such as Readthedocs, DOXYGEN, Javadoc, …).
    - Usage of a version control system is mandatory. We suggest LRZ GitLab, which is free for university students.
    - Individual contributions will be valued in the final grade.
- *Report:* Write a scientific report explaining what has been done and which results were achieved, conclusions, and possible further development options

- The following are typical sections of a scientific report: Introduction*, Related Work, Methodology*, Experiments, Results*, Conclusions* and Future Work* (sections marked with "*" are mandatory).

5. Evaluations

There will be three preliminary evaluations in the form of presentations during the course. The students must hand in each presentation to the course organizer and their supervisors.

- Before each evaluation, the deadline for handing in the presentations (and the poster for the third evaluation) will be announced by the course organizers.
- All students are highly encouraged to contribute to discussions during the evaluation by asking questions. This has a positive effect on the partial grade.

5.1. Penalties

- Violating the deadline or requesting changes in the slides afterward will result in a grade penalty of 0.3.
- A presentation will be interrupted if it extends the given timeframe.
- If a student does not attend a presentation without a valid excuse, their contribution to the presentation will be graded 5.0 (i.e., with the worst grade).
- Before each presentation, supervisors should be consulted to practice the talk and review the slides. The same holds for the review of the poster before the last presentation.
- Not creating a meeting protocol will harm your final grade.
- Absence in the regular meetings without a valid excuse harms the contribution and might result in a worse grade.

6. Completion

By the end of the course, the students should hand in the completed documented code, and the requested scientific report (as well as the meeting protocol) to their supervisor(s), written in proper English.

Evaluation criteria

1. Final grade

The supervisor will address possible advanced questions and tasks to improve the grade. The final grade will be composed of the following parts:

- 3 Presentations (10% each)
- Organization (10%)
- Report + Poster (10%)
- Code + Documentation (50%)

2. Evaluation criterion

A more detailed explanation of every evaluation criterion is provided in this section.

2.1. Presentations

For the **presentations,** the grade will be given by three supervisors that follow more in detail the group's progress (50%) and by all supervisors auditing the presentations (the other 50%).
The rest of the grades (Organization, Report, and documented code) are given only by the corresponding group supervisor(s).

The following questions will be taken into consideration for the evaluation of the presentations:

- Presentation Quality (50 %): How was the general impression of the presentation? Were the contributions of all group members equally distributed? Was the presentation easy to follow (**structure**)? How is the language? Did it seem well prepared (**talk**)? Was it finished in **time**? How are the **slides** structured/designed? Was the presentation material complete and precise (**bibliography**)?

- Technical Quality (50 %): Was the **information** relevant and accurate? Was the **problem** clear? How did the students describe it? Was the **methodological approach** concise and clear? Was **future work** identified?

More specifically, the following rubric will be used to evaluate each presentation.

Level of Achievement	Presentation Quality (50 %)	Technical Quality (50 %)
Excellent **5 Points**	• **Structure**: rigorously argued, logical, easy to follow. • **Talk**: Speaks with good pacing. Use of proper language. • **Slides**: Very little text. Figures and images are explained and described well. • **Bibliography** provided for the audience in an appropriate format. • Uses **time** wisely.	• **Information**: detailed, accurate, relevant; key points highlighted. • **Problem**: Identifies the research question or problem. Integrates research findings/significance to the broader context. • **Methodological approach** is clear. • **Next steps**/future avenues of investigation are identified. • **Algorithms** are explained with diagrams, flow charts, videos, or animations.
Good **4 Points**	• **Structure**: generally clearly argued and logical. • **Talk**: Use of proper language. • **Slides**: Figures and images explained and described well. • **Bibliography** provided for the audience.	• **Information**: detailed, accurate, relevant. • **Problem**: Identifies the research question or problem. • **Methodological approach** is clear. • **Next steps** are identified.
Acceptable **3 Points**	• **Structure**: not always clear or logical. • **Talk** a bit disorganized/challenging to follow. Shows some effort to use proper language. • **Slides**: Blocks of text on handouts in the slides. Some figures are not well explained/described. • **Bibliography** is not well-formatted.	• **Information**: generally accurate and relevant, but perhaps some gaps and/or irrelevant material. • **Problem:** Research question/ Significance a bit unclear. • **Methodology:** The description of the methodological approach is a bit confusing. Too much detail. • **Next steps** are a bit unclear. • **Algorithms** are shown as pseudo-code.
Unacceptable **2 or fewer points**	• **Structure**: confused, incoherent. • **Talk:** too fast/too slow: very difficult to follow. • **Slides:** A lot of blocks of text. • **Bibliography** is not provided.	• **Information**: minimal, with many errors and gaps. • **Problem**/ Research question unclear. • **Methodology** unclear. • **Next steps** are not identified. • **Algorithms** are shown as extracts of written code.

Contributions to the discussions during the presentations will positively affect the final total presentation grade.

The next list indicates what is expected for each presentation.

1. The first presentation (First week of **May**):
   a. Create a code repository in GitLab.
   b. Decide on the programming language(s). e.g., C++, Python, or Matlab.
   c. Present some libraries/datasets that are suitable + Literature Review (you can use Google scholar).
   d. Define the methodological approach that will be followed throughout the project.
   e. Timeline for the project.
2. The second presentation (First week of **July**):
   a. First results regarding the coding.
   b. What worked, what did not work.
   c. Next Steps.
3. Third and final presentation (First two weeks of **November**):
   a. Present the final results.
   b. A working prototype that solves or attempts to solve the problem.

c. Prepare a poster that sums up the results.
d. Good: Present also the cases when the method does not work.
e. Limitations / Possible Future work.

After the final presentation, the students must hand in the documented code and the report to the supervisor. The deadline for that submission has to be packed with the supervisor; specific information will be given from the software lab supervisors.

2.2. Organization

The **Organization** will be evaluated according to the following points:

- Did the student organize autonomously (and the supervisor did not need to assign specific tasks)?
- Did the student try to solve problems autonomously instead of contacting the supervisor for every single issue immediately?
- Did the student contribute to creating a meeting protocol and keep it updated?
- Did the student try to couple his part of the development with the rest of the group?
- Did the student meet agreed milestones? (Time management)

2.3. Report and Poster

The following criteria will evaluate the scientific **report** and the **poster**:

- Are the report and the poster correctly formatted and adequately illustrated?
- Does the **report**:
  - follow a coherent structure?
  - have at least the following sections: Introduction, Methodology, Results, Conclusions, and Future Work?
  - serve to understand and use the code?
  - provide enough scientific references?
- Does the **poster**:
  - motivate the topic?
  - Illustrate the implemented methodology?
  - cite relevant sources?

2.4. Documented Code

The documented **code** will be evaluated as follows:

- Does the code solve the assigned problems?
- Does the code run without compiler warnings, issues, or errors?
- Was the code written entirely by the students (without extensive help from the supervisors)?
- Was the code written in a clean and structured manner?
- Were the functions of the code well documented (with its functionality and each of the input and output parameters)?

3. Penalties

- Each violation of a deadline degrades the respective grade by 0.3.
- In case of extreme delays, the associated category is graded with 5.0.
- Changes in any deliverable (presentations, poster, code, or documentation) after the deadline count as a deadline violation.

4. A student does not pass if:

- The final grade is worse than 4.0

OR

- One category (individual presentations, reports, posters, code, and team contribution) is graded 5.0.

Unforeseen issues will be handled internally by the group's supervisors and the software lab organizers.