

# EDAN70: Project in Computer Science

Fall 2024, Lund University  
Introduction lecture

# Coordinators and supervisors

**Compiler projects:** Görel Hedin, E:2111a, [gorel.hedin@cs.lth.se](mailto:gorel.hedin@cs.lth.se)

Supervisors:

- Idriss Riouak, E:2163, [idriss.riouak@cs.lth.se](mailto:idriss.riouak@cs.lth.se)
- Anton Risberg Alaküla, E:2163, [anton.risberg\\_alakula@cs.lth.se](mailto:anton.risberg_alakula@cs.lth.se)

**Program analysis projects:** Christoph Reichenbach, E:2112b, [christoph.reichenbach@cs.lth.se](mailto:christoph.reichenbach@cs.lth.se)

**Language Technology projects:** Pierre Nugues, E:4134a, [pierre.nugues@cs.lth.se](mailto:pierre.nugues@cs.lth.se)

**Multicore projects:** Jonas Skeppstedt, E:2190, [jonas.skeppstedt@cs.lth.se](mailto:jonas.skeppstedt@cs.lth.se)

**Autonomous systems projects:** Volker Krueger, E:4133a, [volker.krueger@cs.lth.se](mailto:volker.krueger@cs.lth.se)

# Course structure

## Select project

Preferably today. Otherwise, email your coordinator (For compilers: Görel Hedin) by Wednesday Nov 6 ([gorel.hedin@cs.lth.se](mailto:gorel.hedin@cs.lth.se)).

2 students in each group. Preferably all groups do different projects.

## Project work

Weekly meetings with supervisor.

Decide concrete actions each week.

Work incrementally, both for tool and report.

Implement tool

- Use git repository (your supervisor might give you a repo).

- Demo/release to supervisor each week.

- Open-source using BSD modified license.

Evaluate tool: performance, code size, working examples, test cases, user study, ...

Read and review: Related research papers.

Write research-oriented report.

Present your work at seminar.

# Overall workplan and deadlines

Week	What	Submit by deadline (email supervisor)
1	Discuss project with supervisor. Decide on first overall plan. Start writing on intro and outline of report. Start implementation.	
2	Discuss first version of report with supervisor. (Flesh out introduction with problem description, include outline of rest of report.)	Fri Nov 15: First version of report
3	Demo first version of tool to supervisor.	Fri Nov 22: Updated report
4	Do first release of tool. Discuss evaluation with supervisor.	Fri Nov 29: Info about first release
5	Write about solution in report.	Fri Dec 6: Updated report
6	Discuss presentation slides with supervisor.	Fri Dec 13: Draft of presentation slides
7	Present your work at one of the seminar sessions (Dec 18 or 19). Listen to other presentations.	Fri Dec 20: Final slides
January	Finalize report and release.	Wed Jan 15, 2025: Final report and info about final release.

# Evaluation possibilities

(discuss with your supervisor what is relevant for your project)

## Proof of concept examples

Show that the tool works on interesting examples.

Show that the tool works on examples you did not construct yourself.

“Eat your own dogfood”: Use the tool on the tool itself. Not always possible, but nice if you can!

## Performance

JVM startup performance: measure from command line.

JVM Ahead-of-time-compilation performance: measure from command line.

JVM steady state, after JVM warm up: measure from inside program.

Measure several times. Compute confidence intervals.

Scalability? How does the tool scale with larger input?

## Code size

SLOC: Source Lines of Code.

## Quality

Systematically constructed regression test suite.

## Usability

Can users use your tool? “Think aloud” for improving usability.

## Comparisons

Can you compare your tool to existing tools? On examples, performance, ...

# Example resources for evaluation

(discuss with your supervisor what is relevant for your project)

## Performance on the JVM

Andy Georges Dries Buytaert Lieven Eeckhout: Statistically Rigorous Java Performance Evaluation. OOPSLA '07. <https://doi.org/10.1145/1297105.1297033>

S. M. Blackburn et al. Wake up and smell the coffee: evaluation methodology for the 21st century. Communications of the ACM, 51(8):83–89, 2008. <https://doi.org/10.1145/1378704.1378723>

## Tool for SLOC code size (handles several languages)

cloc <https://github.com/AIDanial/cloc>

## Usability

Xavier Ferré, Natalia Juristo, Helmut Windl, Larry Constantine: Usability Basics for Software Developers. IEEE SOFTWARE January/February 2001. <https://doi.org/10.1109/52.903160>

Jakob Nielsen: 10 Usability Heuristics for User Interface Design. <https://www.nngroup.com/articles/ten-usability-heuristics/>

# Doing a release

- Should be **tagged** in the repository.
- Should contain a **README** file explaining how to build, test, and, run.
- There should be a **build script** for building and testing, e.g., in gradle. All dependent libraries that are not part of the repository should be automatically installed by the build script.
- There should be **automated tests**, if relevant.
- There should be **simple examples**, showing use of the software. It should be explained in the README how to run these.
- Any **tools needed** for running the software should be documented in the README. Document in the README which versions of these tools you have used successfully.
- There should be a **LICENSE** text file, declaring the software license used (modified BSD).
- The code should be **platform independent**, if possible. If this is not possible, the requirements should be stated in the README.

## Before mailing your supervisor/uploading info about the release:

- Do a **clean checkout** of your release, verify that your tool can be installed and built according to the instructions in your README.

# Giving the seminar

## Form:

- 15 minutes talk, followed by 5 minutes questions.
- Around 10 slides.

## Advice on the presentation:

- Simon Peyton Jones: [How to give a great research talk.](#)

## Mandatory participation

- Give your own talk
- Attend at least your own session (2 hours), and pose questions



# Writing the report

## Form:

- Double column SIGPLAN format. LaTeX/BibTeX is recommended. Alternatively, use Word/OpenOffice/Pages.
- 3-6 pages for single author. 4-8 pages for two authors.
- The report must not be longer or shorter than these limits.

## Advice on structuring the contents:

- Look at the research papers you read as part of the course. Your report should be written in a similar style, with abstract, introduction with motivation and problem description, sections about your work, evaluation, related work, conclusion, and references. (IPSERC rather than IMRaD)
- Discuss the structure with your supervisor. Different research areas use different structures.
- Simon Peyton Jones: [How to write a great research paper.](#)
- Generative AI tools not allowed for producing report text. (But please do use spell-checkers and grammar-checkers.)

# Structure of research papers. IPSERC vs IMRaD

**IPSERC (for new ideas/methods/solutions)**

**I - Introduction.** Introduces a **Problem** and why it is important to solve. Lists contributions.

**S - Solution.** Explains the new solution.

**E - Evaluation.** Evaluates the new solution.

**R - Related work.** Compares the new solution to previous work.

**C - Conclusion** and future work

**IMRaD (for new knowledge about existing things)**

**I - Introduction.** Why the new knowledge is important to know. Formulates research questions and hypotheses.

**M - Method.** Explains what research methods are used to prove/disprove hypotheses, or answering research questions.

**R - Results.** What answers were found.

**a - and**

**D - Discussion.** What are the implications of the answers.

# IMRaD can be used to structure the evaluation section in an IPSERC paper

## IPSERC (for new ideas/methods/solutions)

**I - Introduction.** Introduces a **Problem** and why it is important to solve. Lists contributions.

**S - Solution.** Explains the new solution.

**E - Evaluation.** Evaluates the new solution.

**R - Related work.** Compares the new solution to previous work.

**C - Conclusion** and future work

## IMRaD (for new knowledge about existing things)

**I - Introduction.** Why the new knowledge is important to know. Formulates research questions and hypotheses.

**M - Method.** Explains what research methods are used to prove/disprove hypotheses, or answering research questions.

**R - Results.** What answers were found.

**a - and**

**D - Discussion.** What are the implications of the answers.

# More detailed advice

**Detailed week-by-week advice, [see course web for compiler projects](#)**

(Check with your coordinator/supervisor if you are doing a project in another area than compilers).