Specialization Seminar

Cezary Kaliszyk

In a nutshell

- Step 1: Choose a topic (today)
- Step 2: Find literature and plan experiments
- ... Discuss literature and outline with supervisor (April 3)
- Step 3: Prepare the presentation
- ... Show planned slides to supervisor **in advance** and try before!
- Step 4: 20 minute presentation (unless exception given demo!) + 5 minutes of questions
- ... May 24: Present!
- ... Be active: Ask questions to others (bonus points)!
- Step 5: Prepare a final report. Send one by June 2 to me (cezary.kaliszyk@uibk..)
- Step 6: Grade!
    - Combination of your research, slides, presentation, answers, report, participation.

Outline

- **Step 1: Topics**
- **Step 2: Literature Research**
- **Step 3: Presentation**
- **Step 4: Report**

Step 1: Topics

- Older:
  - ~~fortran~~
  - algol 60
  - ~~cobol~~
  - ~~modula-2~~
  - intercal
- Data:
  - ~~apl~~
  - ~~SQL~~
  - ~~Octave~~
- Scripts:
  - ~~csh/bash~~
  - XSLT
  - AWK
  - autohotkey
- Presentation:
  - ~~vvvv (visual prog.)~~
  - delphi
  - postscript
- Functional:
  - clojure
  - ~~OCaml~~
  - emacs lisp
  - ~~unlambda~~
- More:
  - ~~logo~~
  - Turing
  - lustre
  - ~~ADA~~
  - Forth
  - ECLiPSe Prolog
  - Datalog
- ...
  - ~~Quantum Programming~~
  - ~~Typescript~~
  - ~~Lean~~
  - ~~Elixir~~
  - ~~Erlang~~

Outline

- Step 1: Topics
- **Step 2: Literature Research**
- Step 3: Presentation
- Step 4: Report

Step 2: Literature Research

Scientific Resources

- Best: papers that include language features or standards
- Less good: documentation / websites / books

Analyze

- limitation of languages
- distinguished features of languages
- the power of esoteric languages
- languages for dedicated purposes

Brief history and vision

Actual Use

Differences to common / known languages

- C, Haskell, Python, OpenGL, TEX

Send found literature and planned outline to supervisor

Example Outline

```
Abstract                                                                    0.5
Introduction
  Motivation                                                                0.5
    Usage of network changes, more and more is available remotely
      without install: WebMails, Calendars, Games etc.
    Provers and programming languages remain same
      often complicated to install
      with language/version dependencies
  What we want                                                              0.5
    Installation simplicity (nothing to do)
    Lightweightness (thin-client)
    Independent of updates of software on server (done by maintaner)
    Remote saving (access from any inet-cafe)
    Potentially: cooperation in proof development (wiki instead of cvs)
    Potentially: prover/programming lang independant (with pg arch)
  What was done (Related works)                                             0.5
    Omega/Web (requires install of mozart, which is only linux/unix)
    LogiCoq (still transmitting whole buffer)
    Java/Flash remote interfaces
    AJAX web interfaces, active web pages, web-edits
    Other proving related web techs: HELM / Mowgli
  Why this is different and innovative                                      0.25
    Never done before, allows working remotely with full speed,
    requires no install, is platform independant even with platform
    dep provers
What is AJAX                                                                1
  Name, Invention, History, AJAX vs Classical Web model, Examples of usage
  Browsers that are supported, maybe 3 line comparison
...
```

Bibliography

```
@inproceedings{proofgeneral,
  author    = {David Aspinall},
  title     = {Proof {G}eneral: A Generic Tool for Proof Development.},
  booktitle = {TACAS},
  year      = {2000},
  pages     = {38-42},
  ee        = {http://link.springer.de/link/service/series/0558/bibs/1785/17850038.htm},
  crossref  = {DBLP:conf/tacas/2000},
  bibsource = {DBLP, http://dblp.uni-trier.de}
  note="{\\URL: \url{http://homepages.inf.ed.ac.uk/da/papers/pgoutline/}}",
}

@inproceedings{pizza,
  author    = {Christian Urban},
  title     = {Implementation of Proof Search in the Imperative Programming
               Language Pizza.},
  booktitle = {TABLEAUX},
  year      = {1998},
  pages     = {313-319},
  ee        = {http://springerlink.metapress.com/openurl.asp?genre=article{\&}issn=0302-9743{\&}volume=1397{\&}spage=0313},
  crossref  = {DBLP:conf/tableaux/1998},
  bibsource = {DBLP, http://dblp.uni-trier.de}
}

@Misc{logicoq,
  author =    {Lo\"ic Pottier},
  title =     {{L}ogi{C}oq},
  year =      {1999},
  note =      {\\{URL}: \url{http://wims.unice.fr/wims/wims.cgi?module=U3/logic/logicoq}}
}
```

Outline

- **Step 1: Topics**
- **Step 2: Literature Research**
- **Step 3: Presentation**
- **Step 4: Report**

Step 3: Prepare a presentation (1/2)

- Proper plan: introduction, related work, citations
- Level: other students (not for experts in the field)
- Lively!: Include demo / examples
- Try before! And measure your time. If needed repeat!
- Think of questions for each talk. Backup slide?
- Optional: Try with another student
- You may think if it can fit a bachelor project already
- Is it also a good impression of you - as defence/interview?

Step 3: Prepare a presentation (2/2)

- Plan: Outline on paper before making the slides. **Story!**
- If theory: Intuition before tens of definitions
- Use illustrations / diagrams if possible!
- Examples to explain new concepts. Examples simple!
- Number your slides
- Test your laptops before! Plan share with other students for safety.
- Don’t be late! (Presence!)
- Do not make too long bullet lists (like this one)

Outline

- Step 1: Topics
- Step 2: Literature Research
- Step 3: Presentation
- **Step 4: Report**

Step 4: Prepare a report (1/2)

Features

- Scientific document, as discussed in EWA
- Focus: (non-standard) features of the languages in detail
- Examples of use, simple introduction
- Did these features make sense, where they used?
- Did features of these language become standard in the meantime? (functional programming was considered esoteric when it was first considered)
- How hard is it to program in this language?

Step 4: Prepare a report (2/2)

Dedicated Purposes

- Necessity of creation
- What would programmers use, if the language would disappear?
- Give (a set of) languages that could be used instead, explain your answer
- Has the goal been achieved?

Grade

Literature research (20%)

- Bibliography + Outlines (April 3)
- Start Early; Contact if issues

Presentations (40%)

- Slides, presenting, answers, questions
- May 24, 20min+5min, in English

Seminar Report (40%)

- June 2, 10-15 pages, in English