Fortgeschrittene funktionale Programmierung

LVA 185.A05, VU 2.0, ECTS 3.0
SS 2021

– Vorbesprechung –

(Stand: 04.03.2021)

Jens Knoop

Technische Universität Wien
Information Systems Engineering
Compilers and Languages

Holding Mode: Online

According to current COVID-19 restrictions and directives of the vice rectorate for academic affairs and the dean's office of academic affairs from January 26 and January 30, 2021, all lecture courses must be held online in the summer term of 2021, and hence the course 185.A04 Advanced Functional Programming.

Advantages and benefits of the directness of face-to-face course meetings shall be preserved to the largest possible extent.

All course and tutorial meetings will thus be held as real-time video conferences.

Advanced Functional Programming in SS 2021

A Motivation

B L^2M: Learning Outcomes, Learning Activities, Major Course Assessments

C Organization, Course Structure

D Resources

A

Motivation

Advanced Functional Programming

...complements and completes the advanced courses on important programming paradigms and styles:

- Object-oriented Programming
  LVA 185.211 Advanced Object-oriented Programming
  VU 2.0 ECTS 3.0

> When the limestone of imperative programming has worn away, the granite of functional programming will be revealed underneath.
>
> Simon Peyton Jones

- **Functional Programming**
  LVA 185.A05 Advanced Functional Programming
  VU 2.0 ECTS 3.0
- Logic-oriented Programming
  LVA 185.A25 Advanced Logical Programming
  VU 2.0 ECTS 3.0

Topics, Objectives of the Course

...advanced principles of functional programming, applications, implementation issues of functional programming languages.

- Functional data structures and algorithms, functional pearls
- Programming with streams, combinators, monads, arrows
- Testing, verification, correctness by construction
  - Automatic testing, automatic test case generation
  - Induction, coinduction, equational reasoning
- Combinator libraries, embedded domain-specific languages
- Parallelism in functional languages
- Applications
  - Monadic parsing, combinator parsing, pretty printing
  - Functional logic programming
  - Functional reactive systems
  - ...
- ...

Central Question: Why is it Good?

Sometimes, the elegant implementation is a function.
Not a method. Not a class. Not a framework.
Just a function.

John Carmack

...a plain, simple answer.

Outline of the Course

- Part I: Motivation
  - Why functional programming matters.
- Part II: Programming Principles
  - Programming with streams, higher-order functions, algorithm patterns, equational reasoning, memoization.
- Part III: Quality Assurance
  - Automatic testing, verification, correctness by construction.
- Part IV: Advanced Language Concepts
  - Functional arrays, abstract data types, monoids, functors, applicatives, monads, arrows.
- Part V: Applications
  - Parsing, logical programming functionally, pretty printing, functional reactive programming.
- Part VI: Extensions, Perspectives
  - Parallelism, 'real world' programming, looking ahead.

For a (Re-) Entry to Functional Programming

...a reminder to two classic articles being a plea for functional programming (even if the title(s) might suggest the opposite):

- Philip Wadler. Why no one uses Functional Languages. ACM SIGPLAN Notices 33(8):23-27, 1998.

  ...there is a tension between building useful systems and extending the frontiers of research ⟨*note: overall, the article is a plea for functional programming*⟩.

- Philip Wadler. An angry half-dozen. ACM SIGPLAN Notices 33(2):25-30, 1998.

  ...You've scrutinized functional languages. You've admired the elegance of lambda calculus, checked the benchmarks from the compilers, noted the security provided by strong typing. Now you want to know if they have been used to serious purpose. Mathematical elegance is well and good, but will it run that mission-critical system? Here are a half-dozen examplars of [serious] functional programs...

Also Worth a Look at for a (Re-) Entry

- Leo A. Meyerovich, Ariel S. Rabkin. Empirical Analysis of Programming Language Adoption. In Proceedings of the 2013 ACM SIGPLAN International Conference on Object-Oriented Programming Systems Languages and Applications (SPLASH 2013), 1-18, 2013.
- Paul Hudak, Mark P. Jones. Haskell vs. Ada vs. C++ vs... An Experiment in Software Prototyping Productivity. Research Report YALEU/DCS/RR-1049, Yale University, 1994. `www.cs.yale.edu/publications/techreports.html#1994`
- Neal Ford. Functional Thinking: Why Functional Programming is on the Rise. IBM developerWorks, 11 pages, 2013. `www.ibm.com/developerworks/java/library/j-ft20/j-ft20-pdf.pdf`

Also lautet der Beschluss,
dass der Mensch was lernen muss.

Max und Moritz
Wilhelm Busch (1832-1908)
dt. Schriftsteller, satirischer Zeichner und Maler

B

L^2M: Learning Outcomes, Learning Activities, Major Course Assessments

Learning Outcomes

After successful completion of the course, students are (among others) able to

1. describe, explain, and demonstrate advanced principles and concepts of functional programming and their application in practice.
2. explicate the theoretical and practical foundations of functional programming, and to exploit them for verifying and validating the correctness of functional programs.
3. identify, discuss, and demonstrate in detail advantages and limitations of a pure functional programming style.
4. solve programming tasks of medium size and complexity using advanced concepts and principles of functional programming adequately and professionally.

Learning Activities: Process, Methods

...for acquiring the learning outcomes:

1. Guided self-dependent learning and practicing: Guided by lecture and flipped classroom sessions, the self-dependent learning and practicing of the competencies described in the learning outcomes by means of lecture notes, theoretical and practical exercises, and further self-reliantly chosen material from text books, tutorials, and scientific articles suggested for further reading.
2. Role model and feedback-oriented learning: Presenting, explaining, comparing, contrasting, and assessing own and others' solutions of assignments in tutor-led tutorials.
3. Self-assessment tests: Tests supporting the regular self-assessment and self-reflection of one's own progress and success of learning; additionally, central and control questions.

Major Course Assessments

Die Freude flieht auf allen Wegen,
der Ärger kommt uns gern entgegen.

Balduin Bählamm, der verhinderte Dichter
Wilhelm Busch (1832-1908)
dt. Schriftsteller, satirischer Zeichner und Maler

Major Course Assessments

1. Four plus one rated programming and project assignments
   - 1.1 Four (smaller) programming assignments.
     (Score per assignment: $\frac{1}{2} * \sum_{i=1}^{2} Score\_of\_Submission_i$)
   - 1.2 One rated (larger) project assignment presented and defended in a 30min demo session (online only)
2. One rated oral examination (per team) about course and tutorial notes and topics (online only) (consensually on-site, if possible wrt to then valid COVID-19 restrictions)

There are no other course assessments.

Required technical equipment: Stable Internet connection, Internet connectable device with audio/video receiver and transmitter.

Assessment Scheme, Overall Grade, Certificates

Assessment Scheme:

Score as per centage of the maximum score	(Partial) Grade
≥ 87.5	1
≥ 75.0	2
≥ 62.5	3
≥ 50.0	4
< 50.0	5

Overall Grade: Weighted assessment (ratio 1 to 1) of the assessments of the

1. programming (25%) and project (25%) assignments
2. oral exam (50%)

if all three are positive; otherwise not sufficient.

Issuance of Certificates: After project demo and oral exam.

ECTS Break Down

Guided Learning Activities (online only)	
– Lecture (7 units $*$ 1.0h)	7.0h
– Flipped classroom (7 units $*$ 0.5h)	3.5h
– Tutorial (7 units $*$ 1.0h)	7.0h
Independent Learning Activities (home universitying)	
– Self-dependent gaining of learning outcomes (Guide value: P I/1.0h, P II/6.0h, P III/6.0h, P IV/7.0h, P V/7.0h, P VI/1.0h)	28.0h
– Specially: Solving assignments, project (Guide value: 4 assignm. $*$ 3.0h, 1 project $*$ 15.5h)	27.5h
– Project demo and oral exam preparation	1.0h
Project Demo (online only, video mtg.)	0.5h
Oral Exam (online only, face-to-face video mtg.)	0.5h
Grand Total	75.0h

Note: The notions P I,...,P VI refer to Part I,...,Part VI of the course notes.

Competency Fields (addressed, not explicitly instructed)

Functional and methodical competencies

- Functional C.: Knowledge about the basics, the foundation, realization, and application of the functional programming style.
- Methodical C.: Proper and professional application of the principles and concepts of the functional programming style.

Cognitive and practical competencies

- Cognitive C.: Abstract and mathematical-logical thinking, analysis, comprehension and problem solving ability.
- Practical C.: Appropriate application of Haskell to concrete programming tasks of medium size and complexity.

Competency Fields (addressed, not explicitly instructed)

Social competencies and self-competencies

- Social C.: Communication, presentation, reasoning skills, team and group aptitude.
- Self-c.: Self-responsibility, self-organization, self-reflection, eagerness for knowledge, hardheadedness, endurance,...

C

Organization, Course Structure

Signing on for the Course

Signing on

- Via TISS on or before March 12, 2021 (12:00 am), in teams of 2 students (in exceptional cases of 1 or 3 students)
- *Signing off the course:* Via TISS on or before March 26, 2021 (12:00 am).

Accounts

- Each student/team of students gets an account for the machine `g0.complang.tuwien.ac.at`
- Account information and initial password are sent to every team member via email to the generic standard address `e<Matr.Nr>@student.tuwien.ac.at`

Further information

- Available at the homepage of the course: `www.complang.tuwien.ac.at/knoop/ffp185A05_ss2021.html`

Course Structure by Learning Activities

Six main learning activities (as the basis of further self-dependent learning activities):

1. Guided Learning Activities
   - 1.1 Lecture (7 units; weekly/biweekly, Thu, 4.15 - 5.15 pm).
   - 1.2 Flipped Classroom (7 units; weekly/biweekly, Thu, 5.30 - 6.00 pm).
   - 1.3 Tutorial (7 units, 90min. each; weekly/biweekly, time to be announced).
2. Central&Control Questions (1 per lecture unit; not rated)
3. Assignments (4 plus 1 assignments, weekly/biweekly).
   - 3.1 Programming exercises (4), project (1) (rated).
   - 3.2 Non-programming exercises (occasionally; not rated).
4. Self-assessment Tests as part of the tutorials (7 tests, 1 test per tutorial; not rated).
5. Project Demo (30min, during 3 afternoons mtgs; rated).
6. Oral Exam (30min, indiv. appointments per team; rated).

Tutorials

1. Issueance of Programming Assignments: Weekly/biweekly on Mondays, down-loadable at the course homepage; in total: 4 assignments, starting Monday, 15 March 2021; 1 project assignment in April.
2. Submission of Programming Assignment Solutions:
   - 2.1 Submission: One week after issueing until 12.00 am; automatic copying from the group home directories (top-level, not in subdirectories; checked with GHCi!).
   - 2.2 Re-submission: After bug fixing/improving the original submission (regarding the deadline, please, check-out the document 'Allgemeine Hinweise zum Übungsablauf' at the homepage of the course).
   - 2.3 Score: The half of the sum of the scores of the original submission and the re-submission of each assignment.
3. Non-programming Exercises: No submission; presentation and discussion of solution proposals of tutorial participants by the tutorial participants.

Tutorials (cont'd)

4. Self-assessment Tests: One 15 minutes test per tutorial meeting; presentation and discussion of solution proposals of tutorial participants by the tutorial participants.
5. Tutorials' Mode of Operation: Presenting, explaining, reasoning, comparing, contrasting, and rating own and others' solution proposals of programming and non-programming exercises from the perspective of adequacy and professionalism.

Feedback on Progress and Success of Learning

Four main kinds:

1. Programming and Non-programming Exercises, Project
   - Presentation and discussion of (selected) solution proposals of tutorial participants during the tutorial meetings.
   - Direct and indirect feedback on presentations and active participation in the tutorial meetings.
   - Feedback on submitted programming assignments by the semi-automatic test system.
   - Capacity-dependent: Comments by tutors of selected assignments, exercises (no impact on the overall grade).
2. Central and Control Questions
   - Discussion in the flipped classroom, tutorial meetings.
3. Self-assessment Tests
   - Presentation, discussion in the tutorial meetings.
   - Self-reflection.
4. Project Demo, Oral Exam
   - Reconsidering, discussing the course of demo and exam.

Tentative Dates f. Lectures, Flipped Classroom

Lecture, Flipped Classroom	Topic Lecture	Topic Flipped Classr.
Thu, 03/04/2021, 4.15-6.00 pm	Part I, Chap. 1 Part II, Chap. 2	n.a. / Prel. Mtg.
Thu, 03/11/2021, 4.15-6.00 pm	Part IV, Chap. 7, 8 Part II, Chap. 3	Part I, Chap. 1 Part II, Chap. 2
Thu, 03/25/2021, 4.15-6.00 pm	Part II, Chap. 4 Part IV, Chap. 9–12	Part IV, Chap. 7, 8 Part II, Chap. 3
Thu, 04/15/2021, 4.15-6.00 pm	Part IV, Chap. 13, 14	Part II, Chap. 4 Part IV, Chap. 9–12
Thu, 04/22/2021, 4.15-6.00 pm	Part III, Chap. 5, 6	Part IV, Chap. 13, 14
Thu, 04/29/2021, 4.15-6.00 pm	Part V, Chap. 15, 16	Part III, Chap. 5, 6
Thu, 05/20/2021, 4.15-6.00 pm	Part V, Chap. 17, 18 Part VI, Chap. 19, 20	Part V, Chap. 15, 16

The lecture and flipped classroom times include a 15min break.

Tentative Dates for Assignments, Tutorials

Programming, Project Assignments, Project Demos

Assignm.	Issueance	(Fst) Delivery	Score	(Corresp.) Tut. Mtg.
1	Mo, 03/15/21	Mo, 03/22/21	100	CW 15: 4/12-16/21
2	Mo, 03/22/21	Mo, 04/12/21	100	CW 16: 04/19-23/21
3	Mo, 04/12/21	Mo, 04/19/21	100	CW 17: 04/26-30/21
4	Mo, 04/19/21	Mo, 04/26/21	100	CW 19: 05/10-14/21
Project	Mo, 04/26/21	Fr, 06/11/21	300	CW 20: 05/17-21/21 CW 21: 05/24-28/21 CW 22: 05/31-06/04/21
Demo			100	06/14-06/25/2021*)

*) Three 4h afternoon meetings between June 14 and 25, 2021.

D

Resources

Textbooks, Scientific Articles, Haskell On-line

...detailed hints and pointers to

- Textbooks, scientific articles
  - See lecture notes, detailed reading recommendations at the end of each chapter.
- Haskell On-line
  - Haskell Homepage: `www.haskell.org/`
  - Haskell Wiki: `wiki.haskell.org/Haskell/`
  - Haskell Tutorial: `www.haskell.org/tutorial/`
  - GHC/GHCi: `www.haskell.org/downloads/`
  - Hugs Interpreter: `www.haskell.org/hugs`
- Functional Programming on youtube
  - John Hughes. Why Functional Programming Matters. Invited Keynote Presentation, Bangalore, 2016. `https://www.youtube.com/watch?v=XrNdvWqxBvA`

    cp. Chapter 1 'Why Functional Programming Matters'

Important Scientific Journals and Conferences

...for publicizing research results in the field of functional programming and Haskell are particularly:

- Journal Series:
  - The Journal of Functional Programming. Matthias Felleisen, Jeremy Gibbons (Eds.), Cambridge, UK, since 1991.
    `https://www.cambridge.org/jfp`
- Conference and Symposium Series:
  - ACM SIGPLAN International Conference Series on Functional Programming (ICFP), yearly since 1996.
    `http://www.sigplan.org/Conferences/ICFP`
  - ACM SIGPLAN International Haskell Symposium Series, yearly since 2008 (2002-2007 as ACM SIGPLAN Haskell Workshop Series).
    `https://wwww.haskell.org/haskell-symposium`

Usage of Computers, Working at Lab/at Home

- Server: `g0.complang.tuwien.ac.at`
- Lab and terminals for course usage: Institutsgebäude EA, Argentinierstr. 8, ground floor (closed in SS 21!)
- Usage of other computers: recommended (e.g., at home)
- Solutions of assignments: must be up-loaded to the server `g0.complang.tuwien.ac.at`
- Required software: GHC/GHCi (available free of charge)

Place(s) to Go

...in case of questions and problems:

- Term-independent homepage of the course:

  `www.complang.tuwien.ac.at/knoop/ffp185A05.html`
- Flipped classroom sessions following the lecture sessions
- Tutorial meetings

Course Notes, Assignments, Appointed Times

Denn was man schwarz auf weiß besitzt,
kann man getrost nach Hause tragen.

Faust. Eine Tragödie.
Johann Wolfgang von Goethe (1749-1832)
dt. Dichter und Naturforscher

- Homepage of the course:

  `www.complang.tuwien.ac.at/knoop/ffp185A05_ss2021.html`

Teaching Assistants (Tutoren)

- Samuel Pilz, BSc
- Hannes Siebenhandl, BSc

Two Perfect Add-Ons this Term (1)

- LVA 127.008 VU 2.0, ECTS 3.0
  Haskell-Praxis: Programmieren mit der Funktionalen Programmiersprache Haskell
  Tue, 3.00 pm to 4.30 pm
  Seminar room 127, 3rd floor, Gußhausstr. 25-29.

  (The course will be held in English.)

  o.Univ.Prof.(em.) Dr. Andreas Frank
  Institut für Geoinformation und Kartographie

...double-check, if this course will be held this term!

Two Perfect Add-Ons this Term (2)

- LVA 183.653 VU 2.0, ECTS 3.0
  Methodisches, industrielles Software-Engineering mit funktionalen Sprachen am Fallbeispiel von Haskell
  Mon, 5.30 pm to 7.00 pm
  Hörsaal 14, Main Building, Stair case 3, 3rd floor, Karlsplatz 13.

  ao.Prof. Dr. Thomas Grechenig
  Institut für Information Systems Engineering

...double-check, if this course will be held this term!

Interested in Studying Abroad?

The Erasmus Programme of the European Union offers a plenty of exciting opportunities, e.g.

- Linköping University, Sweden
- Aalto University, Finland
- The University of Copenhagen, Denmark
- Universität Halle-Wittenberg, Germany
- Universität Paderborn, Germany
- Universidad Politècnica de València, Spain
- ...

For further information, please, visit:
`http://www.complang.tuwien.ac.at/knoop/erasmus`

Zum Vorbesprechungsabschluss

Dabei sein ist
80 Prozent des Erfolges.

Woody Allen (* 1935)
amerik. Schauspieler und Regisseur

...wir, die FFP-Teammitglieder, wünschen Ihnen viel (Lern-) Erfolg für diese Lehrveranstaltung und dass Sie von ihr profitieren, auch langfristig!

Nicht zuletzt:

Die Veranstaltung lebt mit Ihnen! Ihre Rückmeldungen, Anregungen, Verbesserungsvorschläge sind willkommen! Natürlich auch, wenn Ihnen etwas gut gefallen hat!