Fortgeschrittene funktionale Programmierung

LVA 185.A05, VU 2.0, ECTS 3.0
SS 2020

– Vorbesprechung –

(Stand: 05.03.2020)

Jens Knoop

Technische Universität Wien
Information Systems Engineering
Compilers and Languages

Advanced Functional Programming in SS 2020

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 and 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?

...a plain and simple answer:

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

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.

As a Reminder

...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-) Start

- 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^{2}M: Learning Outcomes, Learning Activities, Major Course Assessments

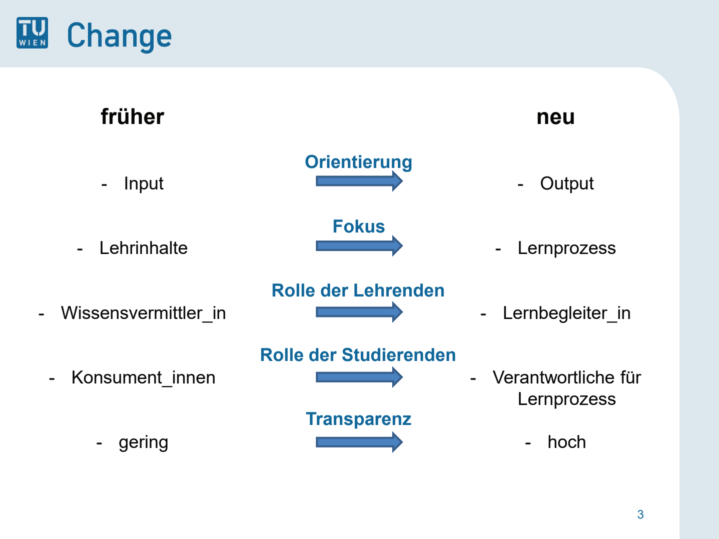
Once Upon a Time: Teach'g & Learn'g Targets

...providing you with the munition for your knowledge and skill acquirement in theory and practice encompassing an/a

- overview, insight, and profound understanding of advanced principles and concepts of functional programming and their application in practice
- profound understanding of the theoretical, especially mathematical foundations of functional programming, and their relevance for verifying and validating the correctness of functional programs
- in-depth insight in advantages and limitations of a purely functional programming style

empowering you to solve programming tasks of medium size and complexity applying advanced concepts and principles of functional programming appropriately and professionally.

From a Broadcaster of Knowledge to an Attendant of Learning

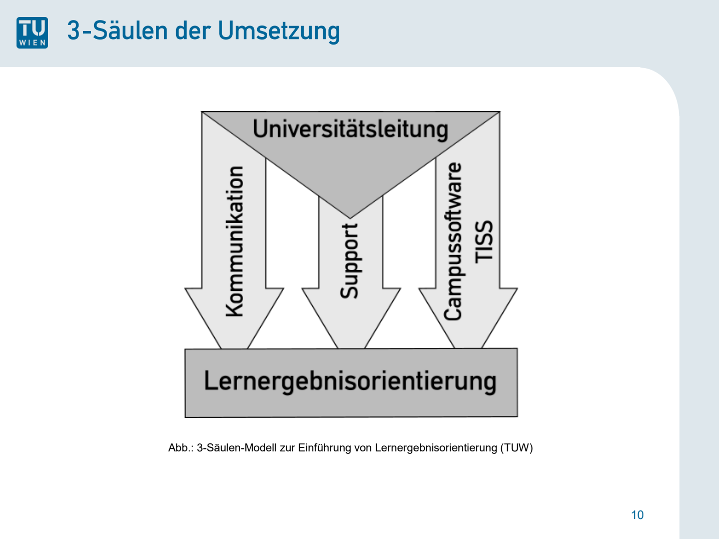


Change

früher		neu
- Input	Orientierung →	- Output
- Lehrinhalte	Fokus →	- Lernprozess
- Wissensvermittler_in	Rolle der Lehrenden →	- Lernbegleiter_in
- Konsument_innen	Rolle der Studierenden →	- Verantwortliche für Lernprozess
- gering	Transparenz →	- hoch

3

...at/with the Highest Inducement and Support



3-Säulen der Umsetzung

Universitätsleitung

Kommunikation

Support

Campussoftware TISS

Lernergebnisorientierung

Abb.: 3-Säulen-Modell zur Einführung von Lernergebnisorientierung (TUW)

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, especially mathematical 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 purely functional programming style.
4. solve programming tasks of medium size and complexity using advanced concepts and principles of functional programming appropriately and professionally.

Learning Process: Methods, Learning Activities

...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 utilizing lecture notes, theoretical and practical exercises, and further self-reliantly chosen material from text books, tutorials, and scientific articles proposed for further reading.
2. Role model and feedback-oriented learning: Presenting, explaining, comparing, contrasting and rating own and others solutions of assignments in tutor-guided tutorials.
3. Self-assessment tests: Tests supporting the regular self-assessment and self-reflection of one's own previous progress and success of learning.

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,...

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

- Seven rated submissions of programming assignments.
  (Score per submission: $\frac{1}{2} * \sum_{i=1}^{2} Score\_of\_Submission_i$)
- One rated 30 minute oral examination (per team) about course and tutorial topics and documents.

(There are no other course assessments.)

Assessment Scheme, Overall Grade, Certificates

Assessment Scheme for Assignments:

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

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

- submissions of the programming assignments.
- oral examination.

if both are positive; otherwise not sufficient.

Issuance of Certificates: After the oral examination.

ECTS Break Down

Guided Learning Activities (21.0h)	
– Lecture (7 units ∗ 1.0h)	7.0h
– Flipped classroom (7 units ∗ 0.5h)	3.5h
– Tutorial (7 units ∗ 1.5h)	10.5h
Independent Learning Activities (53.5h)	
– Self-dependent gaining of learning outcomes (Proposal: I/1.0h, II/6.0h, III/6.0h, IV/7.0h, V/7.0h, VI/1.0h)	28.0h
– Specially: Solving assignments (Proposal: 7 assignments ∗ 3.5h)	24.5h
– Oral exam preparation	1.0h
Oral Examination (0.5h)	0.5h
Grand Total	75.0h

Note: The descriptions I to VI refer to Part I to Part VI of the course notes.

C

Organization, Course Structure

Enrolment for the Course

Enrolment

- Via TISS on or before March 12, 2020 (12:00 am), in teams of 2 students (in exceptional cases of 1 or 3 students)
- *Opting out of the course:* Via TISS on or before March 31, 2020 (12:00 am).

Accounts

- Each student/team of students receives an account/group directory 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_ss2020.html`

Course Structure by Learning Activities

Four main learning activities (in addition to further self-dependent learning activities):

1. Guided Learning Activities
   1.1 Lecture (7 units; weekly/biweekly, Thursday, 4.15 - 5:15 pm).
   1.2 Flipped Classroom (7 units; weekly/biweekly, Thursday, 5.15 - 5.45 pm).
   1.3 Tutorial (7 units, 90min. each; weekly/biweekly, time to be announced).
2. Assignments (7 assignments, weekly/biweekly).
   2.1 programming exercises (rated).
   2.2 Non-programming exercises (occasionally; not rated).
3. Self-assessment Tests as part of the tutorials (7 tests, 1 test per tutorial; not rated).
4. Oral examination (30min., individual appointments per team; rated).

Tutorials

1. Issueance of Programming Assignments: Weekly/biweekly on Mondays, down-loadable at the course homepage; in total 7 assignments, starting on Monday, 16 March 2020.
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 minute test per tutorial; 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 appropriateness and professionalism.

Feedback on Learning Progress and Success

Three main kinds:

1. Programming and Non-programming Exercises
   - Presentation and discussion of selected solution proposals of tutorial participants during the tutorial sessions.
   - Direct and indirect feedback on the active participation presentation in the tutorial sessions.
   - Feedback on the submitted programming assignments by the semi-automatic test system.
   - Capacity-dependent: Comments by tutors of selected assignments and exercises (no relevance for the overall grade).
2. Self-assessment Tests
   - Presentation and discussion of solution proposals in the tutorial sessions.
   - Self-reflection.
3. Oral Examination
   - Reconsidering and discussing the course of the exam.

Tentative Dates for Lectures, Flipped Classr.

Lecture, Flipped Classroom	Topic Lecture	Topic Flipped Classr.
Thu, 03/05/2020, 4.15-5.45 pm	Part I, Chap. 1 Part II, Chap. 2	n.a. / Prel. Mtg.
Thu, 03/12/2020, 4.15-5.45 pm	Part IV, Chap. 7, 8 Part II, Chap. 3	Part I, Chap. 1 Part II, Chap. 2
Thu, 03/26/2020, 4.15-5.45 pm	Part II, Chapt. 4 Part IV, Chap. 9–12	Part IV, Chap. 7, 8 Part II, Chap. 3
Thu, 04/23/2020, 4.15-5.45 pm	Part IV, Chap. 13, 14	Part II, Chapt. 4 Part IV, Chap. 9–12
Thu, 04/30/2020, 4.15-5.45 pm	Part III, Chap. 5, 6	Part IV, Chap. 13, 14
Thu, 05/14/2020, 4.15-5.45 pm	Part V, Chap. 15, 16	Part III, Chap. 5, 6
Thu, 05/28/2020, 4.15-5.45 pm	Part V, Chap. 17, 18 Part VI, Chap. 19, 20	Part V, Chap. 15, 16

Tentative Dates for Assignments, Tutorials

Assignm.	Issueance	Fst. Delivery	Score	Corresp. Tut. Mtg.
1	Mo, 03/16/20	Mo, 03/23/20	100	CW 14: 3/30-4/3/20
2	Mo, 03/23/20	Mo, 03/30/20	100	CW 17: 04/20-24/20
3	Mo, 03/30/20	Mo, 04/20/20	100	CW 18: 04/27-30/20
4	Mo, 04/27/20	Mo, 05/04/20	100	CW 20: 05/11-15/20
5	Mo, 05/11/20	Mo, 05/18/20	100	CW 22: 05/25-29/20
6	Mo, 05/18/20	Mo, 05/25/20	100	CW 23: 06/02-05/20
7	Mo, 06/01/20	Mo, 06/08/20	100	CW 25: 06/15-19/20

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`

Recommended Reading: Basic

- Jens Knoop. Slides and companion material of the course ‘LVA 185.A03 Funktionale Programmierung.’ Compilers and Languages Group, TU Vienna, WS 2019/2020.
- Simon Thompson. Haskell: The Craft of Functional Programming. Addison-Wesley/Pearson, 3rd edition, 2011.
- Richard Bird. Thinking Functionally with Haskell. Cambridge University Press, 2015.
- Greg Michaelson. An Introduction to Functional Programming through Lambda Calculus. Dover Publications, 2nd edition, 2011.
- Peter Pepper, Petra Hofstedt. Funktionale Programmierung: Sprachdesign und Programmiertechnik. Springer-V., 2006.
- Simon Peyton Jones (Ed.). Haskell 98: Language and Libraries. The Revised Report. Cambridge University Press, 2003.
- …

Recommended Reading: Advanced

- Jeremy Gibbons, Oege de Moor (Eds.). The Fun of Programming. Palgrave Macmillan, 2003.
- Bryan O'Sullivan, John Goerzen, Don Stewart. Real World Haskell. O'Reilly, 2008.
- Tomas Petricek, Jon Skeet. Real World Functional Programming: With Examples in F# and C#. Manning Publications Co., 2009.
- Chris Okasaki. Purely Functional Data Structures. Cambridge University Press, 1999.
- Aaron Stump. Verified Functional Programming in Agda. ACM Books Series, No. 9, 2016.
- Lectures on Advanced Functional Programming. International Summer Schools 1995, 1996, 1998, 2002; Springer-V., LNCS volumes 925, 1129, 1608, 2638.
- ...

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
- Usage of other computers: is encouraged (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_ss2020.html`

Organizers

- Co-organizer:
  - Ass.Prof. Dipl.-Ing. Dr. techn. Ulrich Neumerkel
- Teaching assistant (Tutor):
  - 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!