Course PM

TDA555/DIT440, Introduction to Functional Programming, 7.5hp, HT 2018

Examiner and Course responsible:
David Sands

Course Assistants/supervisors:

Marco Vassena, Elisabet Lobo-Vesga, Benjamin Eriksson
Adi Hrustic, Karin Wibergh
Sarosh Nasir, Henrik Rostedt

Course content

This is an introductory course in programming, and uses a functional language. The primary goal is to enable students to write small programs, while introducing some of the fundamental concepts of computer science. Secondary goals are to provide orientation regarding the courses to come (particularly data structures, algorithms, formal methods and programming language courses).

Concrete topics encountered in the course include:

- values, types and functions
- compound data types (lists, tuples, and user-defined types)
- higher-order functions
- using abstraction to avoid repetitive programming
- recursion and recursive data types
- efficient and inefficient programs
- input-output
- verification with the help of testing

Learning outcomes

After completion of this course, the student should be able to

- describe the basic concepts of modern functional programming languages, such as: data types, first- and higher-order functions,lazy evaluation, infinite data structures
- describe a basic repertoire of functional programming techniques, such as: recursion, testing, the role of data types in modelling and problem solving
- write small functional programs for various applications
- structure programs in a way that makes them easy to understand and modify, by appropriate application of data types, abstraction, and code reuse
- implement effective tests for functional programs with help of suitable tools

- show the ability, in various contexts, to judge which programming techniques are most appropriate for solving the problem at hand

Course structure/course implementation

The two main sources of information for the course are (i) the course home page, and (ii) the course slack workspace. The slack (ifp18.slack.com) will be used for last-minute announcements, discussion, general info, and e.g. finding lab partners. You are welcome to create your own private channels in the slack for your lab groups (course admin cannot see such channels).

The course is organized as follows:

- 8 weeks in total
- Two lectures per week, on Tuesdays and either Thursdays or Fridays, with a couple of reserved slots on Wednesdays for possible guest lectures.
- One exercise session per week, where the main idea is to get help with the exercises
- Four lab assignments, spread out over 7 submission deadlines. All lab assignments need to be completed in order to complete the course. See details below.
- One written exam at the end of the course; this is done individually
- Your final grade will be determined by your grade on the written exam only

The "Labs" (Programming Assignments)

There are 4 programming (lab) assignments in total. You have to pass all these to get a pass on the course. Labs are submitted in an electronic system. Lab 1 is graded offline, but labs 2 and 3 are graded by explaining your lab to the course assistants.

Working on your labs

You are required to work in groups of 3 (this is enforced strictly from lab 2 onwards). To keep things simple we suggest that you to use the same groups assigned in the intro course in weeks -1 and 0. Those of you who did not attend the intro course will need to find each other! For this we created a special channel in the course Slack.

Submitting your lab solutions

All lab assignments must be submitted using an electronic submission system called "Fire". You must submit by the deadlines otherwise Fire will not allow you to submit.

Remember to register both yourself and your lab partners in a lab group before you submit! By default, the submission system does not accept submissions made by single persons, and lab 2 and onwards requires you to submit as a group of 3 unless you have been granted an exception.

Lab Grading

For Lab 1 and 4, once you have submitted you will hear back from us via the fire system.

For labs 2 and 3 we will have a fixed time for presentation of solutions: Mondays 16-18 after the exercises (from week 3 onwards) and Tuesdays 10-12. Each grader will have a booking sheet in which you must book your slot.

When grading labs all members of the group must be present, and are required to understand all of the code submitted, and it is the responsibility of all of the group members to make sure that this is the case. If one of the group members clearly does not understand the code then the group will not be passed, and will be expected to try again when all members of the group are able to explain the solution.

Lab Supervision

Supervision is scheduled at various times and places. There are no lab supervision times on Mondays or Tuesdays (just times for presentation of your solutions to the graders). See the timeEdit schedule.

Note: These times are not obligatory! The idea is that all of you will spread out over the week. You cannot come all at once anyway to get lab supervision, but once or twice a week should be possible. You can also work at home on your own computer on the labs (but then you will not get any lab supervision).

If we run out of space then you can always work nearby with a laptop and come to one of the rooms when you get stuck. We will usually use some kind of electronic queue system

You are encouraged to bring pen and paper to the lab sessions, as that makes it convenient for the supervisors to explain certain things.

Booking a Space at the Lab Supervisions

Things are usually a bit busy at the beginning of the term so we usually set up a booking list to book a place in the labs. We ask you to book as a lab group and not individually. Fairness rules:

- Usually there are around 40 slots corresponding to the 20 workstations in each room.
- Each lab group should book one time slot at a time! A time slot is two hours, usually from 10-12, 13-15, or 15-17.
- Only when you are done can you book a new time.

Proper conduct

Cheating on labs is unacceptable. The following are forms of cheating:

- Using someone else's code
- Showing your code to someone else (copying, emailing, printing, pen-and-paper writing)
- Copying code from the web

On the other hand, it is fully allowed to orally discuss assignments and solutions. The web forum can be used for general and specific questions, but of course not for posting parts of solutions. If you have problems, you should talk to us instead of copying from others. If needed, you may get more time and more help. If this is not enough, it is advisable to redo the course next year. This option makes much more sense than cheating.

Some cheating can be detected by the lab graders, when they discover similar solutions (e.g. same code, but different comments, layout, variable names, etc.). At the end of the course, we will also use automatic software that checks for similarity between all submitted solutions.

If we suspect cheating we are required to report this to the disciplinary board (Chalmers or GU). If you are determined to have cheated then you might be suspended ("avstängd") 1-3 months (no studiemedel).

Examination forms

Examination consists of compulsory laboratory exercises and a written exam at the end of the course; see above.

Sample exams are provided at the exams page.

The grades awarded on the exam are 3, 4, 5 for Chalmers students, and G, VG for GU students. (G on the GU side corresponds to 3 or 4 on the Chalmers side; VG corresponds to a 5.)

The exam has a different structure compared to most other exams. Please read this carefully.

The exam is divided into two parts, Part I and Part II.

Part I: If you want to get a 3 or a G, you only have to do Part I.

- Part I consists of 7 separate assignments. You have to complete 5 out of 7 assignments in order to pass the exam.
- Typically, each assignment in Part I will have one of the following themes:
  - write a recursive function over a list
  - write a recursive function over a number
  - write a function that needs to use guards
  - use a standard higher-order function such as map, filter, or takeWhile to solve a given problem
  - write a recursive function over a given recursive datatype
  - design your own (recursive) datatype
  - simplify the definition of a function that does too much pattern matching, unnecessary case distinctions, has complicated Boolean expressions, etc.
  - write a property for a given function
  - write a function that does some simple IO (reading/writing files, printing on the screen, user input) or defines a quickCheck generator with a specific property.

Part II: If you want to get a 4, 5, or a VG, you also have to do Part II.

- Part II consists of 2 separate, bigger and/or more advanced assignments. If you want to get a 4, you only have to complete one of these (your choice). If you want to get a 5 or a VG, you have to complete both of these.
- Typically, you will encounter the following problems in these assignments:
  - dealing with lists-of-lists, lists-of-tuples
  - advanced uses of list comprehensions
  - doing recursion over a list that does not follow that standard recursion pattern over lists
  - you need to define and use a helper function that was not specified in the assignment
  - writing your own higher-order function
  - dealing with a more complicated recursive datatype
  - dealing with a recursive data structure

"Completing" an assignment means that your answer is judged to be "good enough". In other words, you are allowed to make some small mistakes in your answer. "Good enough" means that your answer has to show clearly that you would have gotten the answer completely correct if you had had a computer as help, for example.

Points on Part II can be counted towards Part I if needed, but this is very unlikely to happen in practice.

The contents of the exam is: Everything we have talked about in the course, including

- The contents of each lecture
- QuickCheck properties and simple generators

The exam times and places are advertised in the student portals.

Course literature

The course does not follow a specific book closely, but there is a wealth of online material, including the book *Learn You a Haskell for Great Good!*:

http://learnyouahaskell.com/

Most lectures are available as video recordings. The web pages give an overview of the lectures and the recommended reading.

If you are interested, there are a number of Haskell resources available at the links page.

Changes from Last Year

The main changes from 2017 will be in Labs 1 and 4. Lectures relating to Lab 4 will be changed.

Schedule

There are two lectures a week with a schedule updated on the course home page. A preliminary version of lectures and Lab presentation times are listed here. Up to date version including Lab supervision times and locations are available in timeEdit.

	Tid	Kurskod	Kursnamn	Lokal	Typ	Egen text	Bokningskommentar
v 36	Tisdag 2018-09-04						
	13:15 - 15:00	DIT440 GU, TDA555	Introduktion till funktionell programmering, Introduktion till funktionell programmering	HB1	Föreläsning		
	Fredag 2018-09-07						
	08:00 - 09:45	DIT440 GU, TDA555	Introduktion till funktionell programmering, Introduktion till funktionell programmering	HB1	Föreläsning		
v 37	Fredag 2018-09-14						
	08:00 - 09:45	DIT440 GU, TDA555	Introduktion till funktionell programmering, Introduktion till funktionell programmering	HB1	Föreläsning		
v 38	Måndag 2018-09-17						
	16:15 - 18:00	DIT440 GU, TDA555	Introduktion till funktionell programmering, Introduktion till funktionell programmering	EG-2516, EG-3503, EG-3504, EG-3505, EG-3506, EG-3507, EG-3508	Laboration, Redovisning		
	Tisdag 2018-09-18						
	10:00 - 11:45	DIT440 GU, TDA555	Introduktion till funktionell programmering, Introduktion till funktionell programmering		Laboration, Redovisning		
	13:15 - 15:00	DIT440 GU, TDA555	Introduktion till funktionell programmering, Introduktion till funktionell programmering	HB1	Föreläsning		
	Torsdag 2018-09-20						
	15:15 - 17:00	DIT440 GU, TDA555	Introduktion till funktionell programmering, Introduktion till funktionell programmering	HB1	Föreläsning		
v 39	Måndag 2018-09-24						
	16:15 - 18:00	DIT440 GU, TDA555	Introduktion till funktionell programmering, Introduktion till funktionell programmering	EG-2516, EG-3503, EG-3504, EG-3505, EG-3506, EG-3507, EG-3508	Laboration, Redovisning		
	Tisdag 2018-09-25						
	10:00 - 11:45	DIT440 GU, TDA555	Introduktion till funktionell programmering, Introduktion till funktionell programmering		Laboration, Redovisning		Idéläran grupprum 11, Idéläran grupprum 8, Edithuset - grupprum 6207,6209,6211
	13:15 - 15:00	DIT440 GU, TDA555	Introduktion till funktionell programmering, Introduktion till funktionell programmering	HB1	Föreläsning		
	Torsdag 2018-09-27						
	15:15 - 17:00	DIT440 GU, TDA555	Introduktion till funktionell programmering, Introduktion till funktionell programmering	HB1	Föreläsning		

	Tid	Kurskod	Kursnamn	Lokal	Typ	Egen text	Bokningskommentar
v 40	Måndag 2018-10-01						
v 40	Måndag 2018-10-01						
	16:15 - 18:00	DIT440 GU, TDA555	Introduktion till funktionell programmering, Introduktion till funktionell programmering	EG-2516, EG-3503, EG-3504, EG-3505, EG-3506, EG-3507, EG-3508	Laboration, Redovisning		
v 40	Tisdag 2018-10-02						
	10:00 - 11:45	DIT440 GU, TDA555	Introduktion till funktionell programmering, Introduktion till funktionell programmering		Laboration, Redovisning		Idéläran grupprum 11, Idéläran grupprum 8, Edithuset - grupprum 6207,6209,6211
	13:15 - 15:00	DIT440 GU, TDA555	Introduktion till funktionell programmering, Introduktion till funktionell programmering	HB1	Föreläsning		
	Torsdag 2018-10-04						
	15:15 - 17:00	DIT440 GU, TDA555	Introduktion till funktionell programmering, Introduktion till funktionell programmering	HB1	Föreläsning		
v 41	Måndag 2018-10-08						
	16:15 - 18:00	DIT440 GU, TDA555	Introduktion till funktionell programmering, Introduktion till funktionell programmering	EG-2516, EG-3503, EG-3504, EG-3505, EG-3506, EG-3507, EG-3508	Laboration, Redovisning		
	Tisdag 2018-10-09						
	10:00 - 11:45	DIT440 GU, TDA555	Introduktion till funktionell programmering, Introduktion till funktionell programmering		Laboration, Redovisning		Idéläran grupprum 11, Idéläran grupprum 8, Edithuset - grupprum 6207,6209,6211
	13:15 - 15:00	DIT440 GU, TDA555	Introduktion till funktionell programmering, Introduktion till funktionell programmering	HB1	Föreläsning		
	Onsdag 2018-10-10						
	10:00 - 11:45	DIT440 GU, TDA555	Introduktion till funktionell programmering, Introduktion till funktionell programmering	KE	Föreläsning		
v 42	Måndag 2018-10-15						
	16:15 - 18:00	DIT440 GU, TDA555	Introduktion till funktionell programmering, Introduktion till funktionell programmering	EG-2516, EG-3503, EG-3504, EG-3505, EG-3506, EG-3507, EG-3508	Laboration, Redovisning		
	Tisdag 2018-10-16						
	10:00 - 11:45	DIT440 GU, TDA555	Introduktion till funktionell programmering, Introduktion till funktionell programmering		Laboration, Redovisning	Klicka för lokalinfo	Idéläran grupprum 11, Idéläran grupprum 8, Edithuset - grupprum 6207,6209,6211
	13:15 - 15:00	DIT440 GU, TDA555	Introduktion till funktionell programmering, Introduktion till funktionell programmering	HB1	Föreläsning		

	Tid	Kurskod	Kursnamn	Lokal	Typ	Egen text	Bokningskommentar
	Onsdag 2018-10-17						
v 42	Onsdag 2018-10-17						
	10:00 - 11:45	DIT440 GU, TDA555	Introduktion till funktionell programmering, Introduktion till funktionell programmering	KE	Föreläsning		
v 43	Måndag 2018-10-22						
	16:15 - 18:00	DIT440 GU, TDA555	Introduktion till funktionell programmering, Introduktion till funktionell programmering	EG-2516, EG-3503, EG-3504, EG-3505, EG-3506, EG-3507, EG-3508	Laboration, Redovisning		
	Tisdag 2018-10-23						
	10:00 - 11:45	DIT440 GU, TDA555	Introduktion till funktionell programmering, Introduktion till funktionell programmering		Laboration, Redovisning	Klicka för lokalinfo	Idéläran grupprum 11, Idéläran grupprum 8, Edithuset - grupprum 6207,6209,6211
	13:15 - 15:00	DIT440 GU, TDA555	Introduktion till funktionell programmering, Introduktion till funktionell programmering	HB1	Föreläsning		