Informatics 1
Functional Programming Lecture 1

Introduction

Don Sannella
University of Edinburgh
Welcome to Informatics 1, Functional Programming!

Informatics 1 course organiser: Paul Anderson

Functional programming (Inf1-FP)
Lecturer: Don Sannella
Teaching assistant: Karoliina Lehtinen

Computation and logic (Inf1-CL)
Lecturer: Michael Fourman
Teaching assistant: ???

Informatics Teaching Organization (ITO)
Gregor Hall
Where to find us

IF – Informatics Forum
FH – Forrest Hill

Inf1 course organiser: Paul Anderson dcspaul@inf.ed.ac.uk IF 1.24

Functional programming (Inf1-FP)
Lecturer: Don Sannella Don.Sannella@ed.ac.uk IF 5.12
Teaching assistant: Karoliina Lehtinen M.K.Lehtinen@sms.ed.ac.uk IF 5.34

Informatics Teaching Organization (ITO):
Gregor Hall FH 1.B15
Lectures

- Monday 14:10–15:00, David Hume Tower, Lecture Hall C
- Tuesday 11:10–12:00, Appleton Tower, Lecture Theatre 5

Sometimes, Inf1-FP swaps lecture slots with Inf1-CL.
Lectures

- Monday 14:10–15:00, David Hume Tower, Lecture Hall C
- Tuesday 11:10–12:00, Appleton Tower, Lecture Theatre 5

Sometimes, Inf1-FP swaps lecture slots with Inf1-CL.  
*Like this week* — extra Inf1-FP lecture on Friday!

- Friday 23 Sep, 14:10-15:00, Appleton Tower, Lecture Theatre 4
Required text and reading

*Haskell: The Craft of Functional Programming (Third Edition),*
  Simon Thompson, Addison-Wesley, 2011.

or

*Learn You a Haskell for Great Good!*

Reading assignment

This week       Thompson: parts of Chap. 1-3 and 5
                Lipovača: parts of intro, Chap. 1-2
Later weeks     See the course web page

The assigned reading covers the material very well with plenty of examples.

There will be no lecture notes, just the books. *Get one of them and read it!*
Linux / DICE Tutorial

Tuesday    20 September 2016    2–4pm    Forrest Hill Drill Hall
Wednesday  21 September 2016    2–4pm    Forrest Hill Drill Hall
Thursday   22 September 2016    2–4pm    Forrest Hill Drill Hall

Forrest Hill Drill Hall, FH 1.B30
Get You Installed a Haskell

<table>
<thead>
<tr>
<th>Day</th>
<th>Date</th>
<th>Time</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tuesday</td>
<td>20 September 2016</td>
<td>2–4pm</td>
<td>Forrest Hill Drill Hall</td>
</tr>
<tr>
<td>Wednesday</td>
<td>21 September 2016</td>
<td>2–4pm</td>
<td>Forrest Hill Drill Hall</td>
</tr>
<tr>
<td>Thursday</td>
<td>22 September 2016</td>
<td>2–4pm</td>
<td>Forrest Hill Drill Hall</td>
</tr>
<tr>
<td><strong>Friday</strong></td>
<td><strong>23 September 2016</strong></td>
<td><strong>3–5pm</strong></td>
<td><strong>Forrest Hill Drill Hall</strong></td>
</tr>
</tbody>
</table>

Forrest Hill Drill Hall, FH 1.B30
Forrest Hill Drill Hall
Forrest Hill Drill Hall
Lab Week Exercise and Drop-In Labs

Monday  3–5pm (demonstrator 3:00-4:00pm)  Forrest Hill Drill Hall
Tuesday  2–5pm (demonstrator 3:00-4:00pm)  Forrest Hill Drill Hall
Wednesday  2–5pm (demonstrator 3:00-4:00pm)  Forrest Hill Drill Hall
Thursday  2–5pm (demonstrator 3:00-4:00pm)  Forrest Hill Drill Hall
Friday  3–5pm (demonstrator 3:00-4:00pm)  Forrest Hill Drill Hall

Forrest Hill Drill Hall, FH 1.B30

Lab Week Exercise

submit by 5pm Friday 30 September 2016

*Do all the parts*
Tutorials

ITO will assign you to tutorials, which start in Week 3.

Attendance is compulsory.

Tuesday/Wednesday    Computation and Logic
Thursday/Friday       Functional Programming

Contact the ITO if you need to change to a tutorial at a different time.

You must do each week’s tutorial exercise! Do it before the tutorial!

Bring a printout of your work to the tutorial!

You may collaborate, but you are responsible for knowing the material.

Mark of 0% on tutorial exercises means you have no incentive to plagiarize.

But you will fail the exam if you don’t do the tutorial exercises!
Beginner-Friendly Tutorials

Some tutorials are labelled as *beginner friendly*.

Ask for one of these if:

- you have no previous programming experience; and/or
- you just aren’t so confident

All tutorial exercises will cover the same tutorial exercises.

The beginner-friendly tutorials will proceed more carefully. Their priority is to make sure that all students are keeping up.

Contact the ITO if you need to change into or out of a beginner-friendly tutorial.
Automated Feedback

CamlBack (UCLA) will give automated feedback on tutorial exercises.

Use is optional, but it’s good to get immediate feedback.

```
1 fib :: Integer -> Integer
2 fib 0 = 0
3 fib 1 = 1
4 fib n = fib (n-1) * fib (n-2)
```
Automated Feedback

CamlBack (UCLA) will give automated feedback on tutorial exercises. Use is optional, but it’s good to get immediate feedback.
Formative vs. Summative

0% Lab week exercise
0% Tutorial 1
0% Tutorial 2
0% Tutorial 3
10% Class Test
0% Tutorial 4
0% Tutorial 5
0% Tutorial 6
0% Tutorial 7
0% Mock Exam
0% Tutorial 8
90% Final Exam
Course Webpage

See http://www.inf.ed.ac.uk/teaching/courses/inf1/fp/ for:

- Course content
- Organisational information: what, where, when
- Annotated lecture slides, reading assignment, tutorial exercises, solutions
- Past exam papers
- Programming competition
- Other resources
Any questions?

Questions make you *look good*!

Don’s *secret technique* for asking questions.

Don’s *secret goal* for this course
Ask

Use the Ask online Inf1-FP forum:

- For *asking questions* outside lectures
- For *reading answers* to questions asked by others
- For *writing answers* to questions asked by others

See the course webpage for the link and for sign-up instructions
Part I

Introduction
Why learn Haskell?

- Important to learn many languages over your career
- Functional languages increasingly important in industry
- Puts experienced and inexperienced programmers on an equal footing
- Operate on data structure *as a whole* rather than *piecemeal*
- Good for concurrency, which is increasingly important

Operating on data structure as a whole rather than piecemeal:
In an imperative language, you often use a loop to operate on the items in a data structure, one at a time. In a functional language, you tend to operate on the whole data structure: Whoosh! This leads to higher-level thinking about algorithms.
Linguistic Relativity

“Language shapes the way we think, and determines what we can think about.”
Benjamin Lee Whorf, 1897–1941

“The limits of my language mean the limits of my world.”
Ludwig Wittgenstein, 1889–1951

“A language that doesn’t affect the way you think about programming, is not worth knowing.”
Alan Perlis, 1922–1990
Look at these web pages:

Jane Street Capital: www.janestreet.com/technology/

Facebook:
www.wired.com/2015/09/
    facebooks-new-anti-spam-system-hints-future-coding/

Functional Programming is Black Magic
http://www.quora.com/... (see course web page)

ICFP 2016: conf.researchr.org/home/icfp-2016
Families of programming languages

- **Functional**
  - Erlang, F#, Haskell, Hope, Javascript, Miranda, OCaml, Racket, Scala, Scheme, SML
    - More powerful
    - More compact programs

- **Object-oriented**
  - C++, F#, Java, Javascript, OCaml, Perl, Python, Ruby, Scala
    - More widely used
    - More libraries
Functional programming in the real world

- Google MapReduce, Sawzall
- Ericsson AXE phone switch
- Perl 6
- DARCS
- XMonad
- Yahoo
- Twitter
- Facebook
- Garbage collection
Functional programming is the new new thing

Erlang, F#, Scala attracting a lot of interest from developers

Features from functional languages are appearing in other languages

- **Garbage collection** Java, C#, Python, Perl, Ruby, Javascript
- **Higher-order functions** Java, C#, Python, Perl, Ruby, Javascript
- **Generics** Java, C#
- **List comprehensions** C#, Python, Perl 6, Javascript
- **Type classes** C++ “concepts”