from knowing nothing to understanding Nothing in 6 weeks

Haskell MOOC and Blended Course

Jeremy Singer & Alex Pancheva
Functional Programming (H) COMPSCI4021

- **Academic Session:** 2016-17
- **School:** School of Computing Science
- **Credits:** 10
- **Level:** Level 4 (SCQF level 10)
- **Typically Offered:** Semester 1
- **Available to Visiting Students:** Yes
- **Available to Erasmus Students:** Yes
ONLINE COURSE

Functional Programming in Haskell: Supercharge Your Coding

Get an introduction to Haskell, the increasingly popular functional programming language, with this University of Glasgow course.

Register
What is Haskell?

• a functional programming language

\[
\text{rms } xs = \sqrt{\text{foldr}(\lambda x a \rightarrow a + x \times x) 0 \ xs / \text{fromIntegral}(\text{length} \ xs)}
\]

• growing industry interest
Glasgow was the epicentre of Haskell development
Haskell at Glasgow
What we did

- 10 week, 10 credit Level 4 course
- 6 week MOOC
- 4 week f2f course
- drop-in clinic
Course Design

Learning type: Acquisition
Learning through acquisition is what learners are doing when they are listening to a lecture or podcast, reading from books or websites, and watching demos or videos.

Learning type: Collaboration
Learning through collaboration embraces mainly discussion, practice, and production. Building on investigations and acquisition it is about taking part in the process of knowledge building itself.

Learning type: Discussion
Learning through discussion requires the learner to articulate their ideas and questions, and to challenge and respond to the ideas and questions from the teacher, and/or from their peers.

Learning type: Investigation
Learning through investigation guides the learner to explore, compare and critique the texts, documents and resources that reflect the concepts and ideas being taught.

Learning type: Practice
Learning through practice enables the learner to adapt their actions to the task goal, and use the feedback to improve their next action. Feedback may come from self-reflection, from peers, from the teacher, or from the activity itself, if it shows them how to improve the result of their action in relation to the goal.

Learning type: Production
Learning through production is the way the teacher motivates the learner to consolidate what they have learned by articulating their current conceptual understanding and how they used it in practice.
Our Unique Selling Points

1. interactive coding
2. rockstar interviews
3. social learning
1. interactive coding

Type Haskell expressions in here.

\[ \text{41} \]

\[ \text{41} \]

\[ \text{:: Num a => a} \]

\[ \text{\lambda} \]

OK, no surprises so far, you got back the number 41 as expected.

Now type a simple integer arithmetic operation, e.g. \[ \text{6*7} \], and observe that it evaluates to the expected result.
the Haskell movement, I suppose, was born.
2. rockstar interviews
3. social learning

With the lambda function introduction in the first week the function used in zipWith should have been defined as (\x y -> (x, y)).

IMO this was the best teaching video in the course so far, by a large margin. At the risk of gushing, I thought it was very good in content and excellent in its form of presentation, even surviving the lack of HD because the fonts were all large and readable. If this is an indication of the direction in which the video content is evolving, we are in for a treat! :-)

Agreed and presented at a very good pace; neither too fast nor too slow.
some stats

• MOOC signups: 6.3K
• active learners: 3.1K
• completions: 900
• Glasgow students: 120
Functional Baby Talk: Analysis of Code Fragments from Novice Haskell Programmers

Jeremy Singer and Blair Archibald
School of Computing Science
University of Glasgow
UK
jeremy.singer@glasgow.ac.uk  b.archibald.1@research.gla.ac.uk

What kinds of mistakes are made by novice Haskell programmers, as they learn about functional programming? Is it possible to analyse these errors in order to improve the pedagogy of Haskell? In 2016, we delivered a massive open online course which had an interactive code evaluation environment. We captured and analysed over 160K interactions from learners. We report typical novice developer behaviour; for instance, the mean time spent on an interactive tutorial is around eight minutes. Although our environment was restricted, we gain some understanding of novice learner errors. Parenthesis mismatches, lexical scoping errors and do block misunderstandings are common. Finally, we make recommendations about how such beginner code evaluation environments might be enhanced.

1 Introduction

The Haskell programming language [11] has acquired a reputation for being difficult to learn. In his presentation on the origins of Haskell [17] Peyton Jones notes that, according to various programming
Advantages for Glasgow students

1) larger and more diverse learner community

2) flexible learning - where/when you like

3) backup face-to-face clinics if required

4) fresh, interactive learning materials
Feedback from Glasgow students

Beginning of the course was online with activities to help confirm learning.

Video lectures were great and gave students a chance to learn at own pace.

Particularly enjoyed the online platform and community.
Suggested improvements from Glasgow students

- more practical lab sessions

- course became exponentially harder with the last couple of topics [...] shift it so that these come earlier

- Probably fix some bugs with the online interpreter
Conclusion

• Investment

• Outcomes