Dublin Haskell Book Club

October 2021

Building up our axioms (Sorry super text Heavy)

- A lot of this will be from my personal experience.
- A lot of this especially early will be stuff you already know and might be boring
- The group will have different experiences so we have to find the right balance of making it interesting to old hands and possible to new members
- Might make sense to have two session a month, one general session and an additional tutorial for people struggling.
- I’ll start making supporting material like summary videos for people that start late or miss weeks.

Reasons for starting Book club

- Excuse to use and talk about haskell.
- Fun way to build community.
- “Peer pressure” effect to help keep motivation and work through large text books.
- It’s easier and more fun to study topics in a group
- Opportunity to share experiences and learn from others.

Some Expectations(from my Perspective)

- Improve my Haskell knowledge.
- Introduction to new ideas.
- Meet new people who share a love of programming.
- Maybe become better at explaining concepts from haskell to others
- Build up the confidence to use haskell more

Hopes for the group

- Build a community of dedicated haskellers in Dublin.
- Get some networking effects where we all improve our haskell skills.
- Id absolutely love to start some open source projects in haskell from this group.(Open to any and all ideas)
- A support group where we can share and get excited about ideas and concepts in programming.

Logistics

- One meeting a month, generally late in the month and late in the week. These will be virtual for now, but later we can also do a meetup somewhere in Dublin with the option of signing in virtually.
- Github repo with the solutions to exercises, these slide decks and any other supporting media, feel free to make pull requests, also I’ll probably move it to a dedicated Dublin haskell org account.
- Generally around three chapters a month(depending on chapters) trying to average between 50 to 100 pages of content. Which means it will take at least 6 months to get through Real World Haskell.
- I’ll try to stay a few chapters ahead and post material ahead of time, also will post summary videos after each session.
- Would be great if people could volunteer to handle the meetups if i cant for any reason(this should be super rare).
- Also would be really cool if others thought of ideas and hosted separate meetups, eg code alongs, open source projects, speaker series etc.
- Will be using Discord.

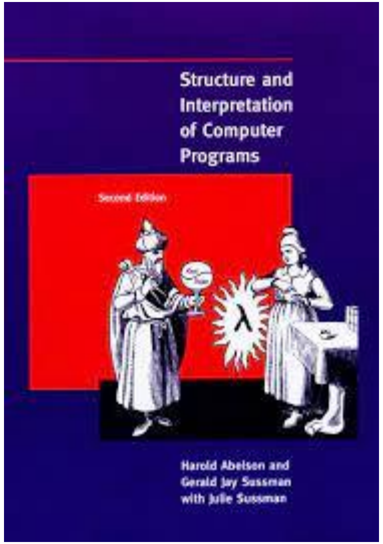
Why Real World Haskell

- It available free.
- It’s a classic.
- Personally I’ve never finished it.
- Comprehensive coverage, and nice balance between being beginner friendly and not shying away from complexity.
- Some really cool example projects (e.g JSON parser)
- Been around for a while so errata have been addressed.
- One of the Authors is from Dublin.

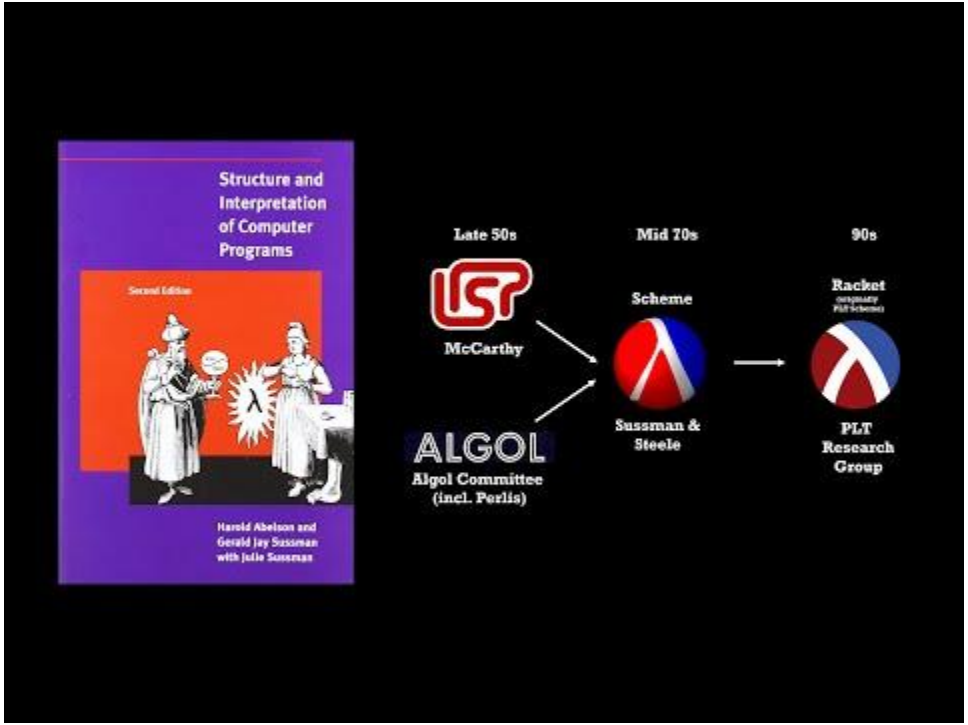
One of Irealnd's great Haskellers

Functional Background(From a C developer)

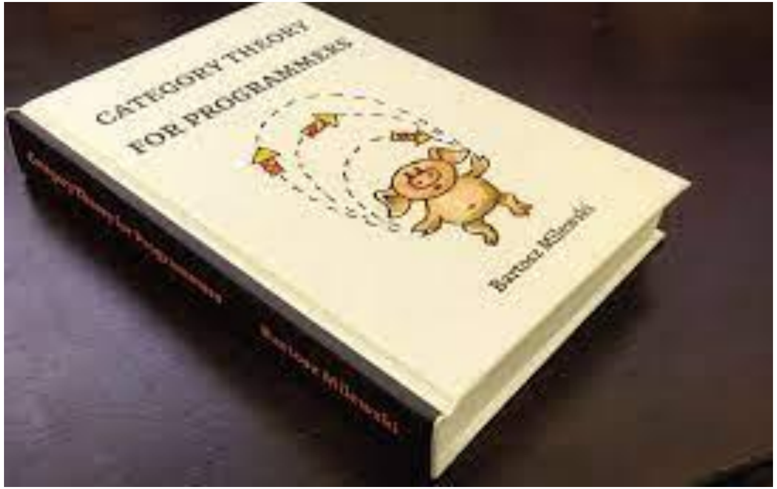
- Church and Turning
- The lambda Calculus
- Lisp’s and C
- Stalmans lisp machines and Bell labs systems
- Scheme’s and SICP
- Category Theory
- OOP obsession
- Functionals return to the mainstream
- Java, C++, Pythons “borrowing”
- The hybrids Scala, Closure and Rust.


Structure and Interpretation of Computer Programs
Second Edition
λ
Harold Abelson and Gerald Jay Sussman with Julie Sussman

My introduction came from a C++ Conference


Structure and Interpretation of Computer Programs
Second Edition
Harold Abelson and Gerald Jay Sussman with Julie Sussman
Late 50s
Mid 70s
90s
LISP
McCarthy
ALGOL
Algol Committee (incl. Perlis)
Scheme
Sussman & Steele
Racket
(originally PLT Scheme)
PLT Research Group

Pure mathematics vs Applied || Functional vs Object



SOME REASONS FOR MY PERSONAL INTEREST IN Haskell

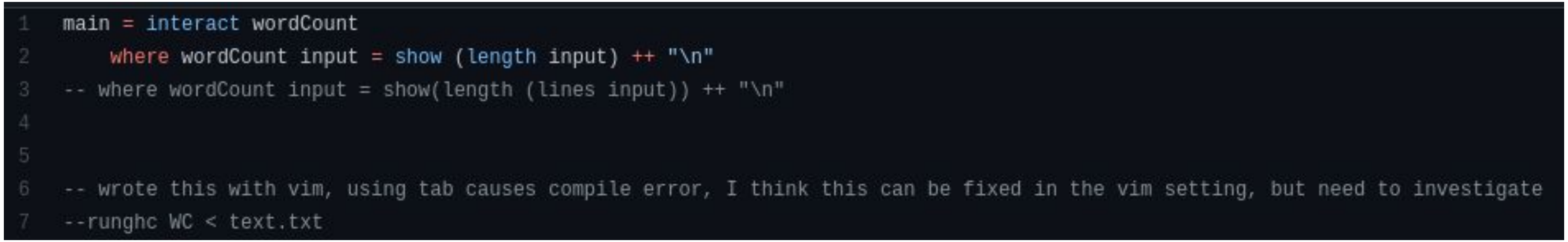
- Reduced lines of code, means less bugs and more readability and easier to maintain.
- Different way of thinking make my generally coding ability better
- Power of abstraction
- Use for writing parsers and compilers.
- Ambition to create my own language and rewrite Minix.

Chapter one - Getting started

- Getting Haskell setup
- GHCI
- Calculator
- Lists
- Strings and Chars
- Types
- A Simple Program

Chapter one Problems + Solution

- Question 4 and 5 page 16
- WC.hs

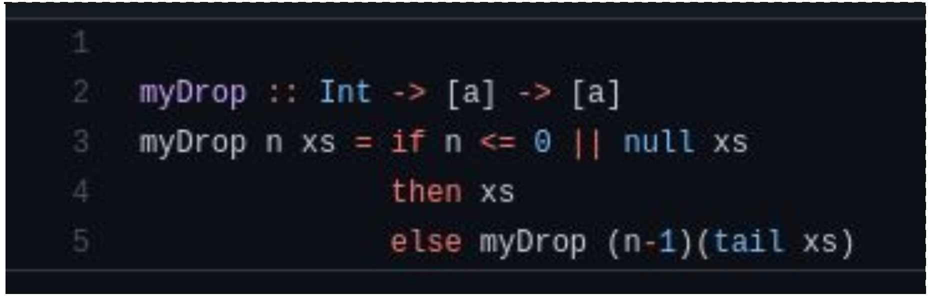


```
main = interact wordCount
    where wordCount input = show (length input) ++ "\n"
-- where wordCount input = show(length (lines input)) ++ "\n"


-- wrote this with vim, using tab causes compile error, I think this can be fixed in the vim setting, but need to investigate
--runghc WC < text.txt
```

Chapter two Types and Functions

- Why types
- Haskell Type System
- Basic Types
- Functional Application
- Composite Data Types
- Functions Lists and Tuples
- Types and Purity
- Simple Functions
- Evaluation by example
- Polymorphism
- Purity

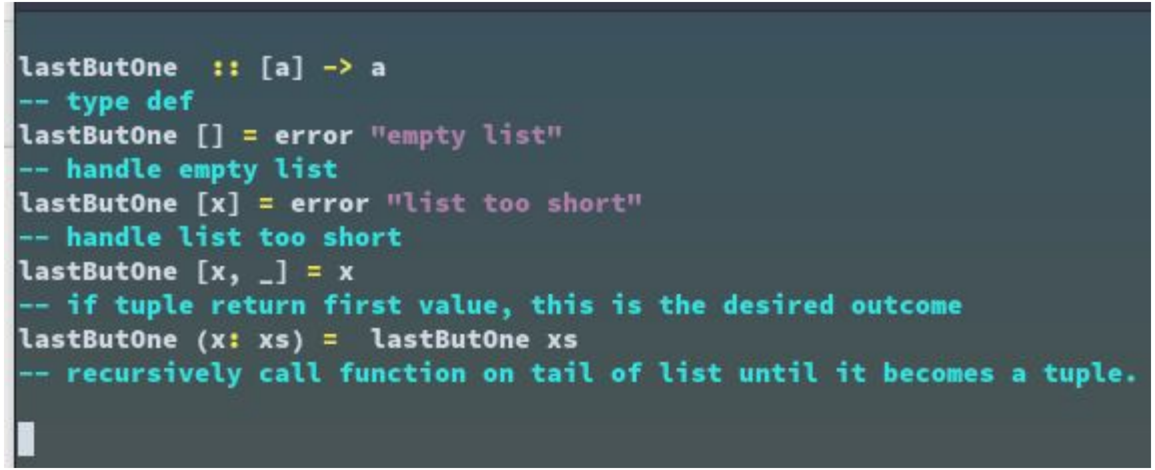


```haskell

myDrop :: Int -> [a] -> [a]
myDrop n xs = if n <= 0 || null xs
              then xs
              else myDrop (n-1)(tail xs)
```

Lastbutone.hs

- My solution, probably not the most efficient but I wanted to use recursion and pattern matching.



```haskell
lastButOne  :: [a] -> a
-- type def
lastButOne [] = error "empty list"
-- handle empty list
lastButOne [x] = error "list too short"
-- handle list too short
lastButOne [x, _] = x
-- if tuple return first value, this is the desired outcome
lastButOne (x: xs) =  lastButOne xs
-- recursively call function on tail of list until it becomes a tuple.
```

Conclusion and final discussion

- Thoughts on the chapters?
- Parts you found interesting?
- Difficult parts?
- Pacing so far?

Next Month

- Chapter 3 Defining Type, Streamlining Functions pages 41 to 68
- Chapter 4 Functional Programming pages 71 to 111
- Chapter 5 Writing a Library: Working with JSON Data pages 111 to 134