CISC 260: Programming Paradigms

Joshua Dunfield
Queen's University
Fall 2018

Today

- Who am I?
- Hello, what is this course about?
  - What is a paradigm?
  - Functional programming
  - Logic programming
- Logistics (attendance, marks, etc., etc.)
- Next time…

Who am I?

"Prof. Dunfield"
(general function: `(\x -> "Prof. " ++ x)`)
"Joshua"

Not recommended:

- "Sir" (from McGill students)
- "Herr Dr. Dunfield"
  (mail addressed to me in Germany)

Carnegie Mellon —2007→ McGill —2010→
MPI-SWS —2014→ UBC —2017→ Queen's

Who am I?

- I study “programming languages”:
  - Logical and mathematical foundations of programming languages
  - Reasoning about programs and programming languages
  - Formal methods
  - Types (so many types)

Who am I?

"Prof. Dunfield"

(general function: `(\x -> "Prof. " ++ x)`)

"Joshua"

http://dunfieldlab.ca/

joshuad@cs.queensu.ca

Goodwin 534 **Office Hours:**

TBA + by appointment + **when door is wide open**

(if the door is closed,
don't knock unless, basically, the building is on fire)

Hello, what is this course about?

- World domination?

World domination: this time for sure!

- 196x: Algol was going to dominate
- 1970: PL/I was going to dominate
- 1980: C was going to dominate
- 1990: C++ was definitely going to dominate
  - James Gosling had to justify why he would dare to design a new language
- 2000: Java was going to dominate

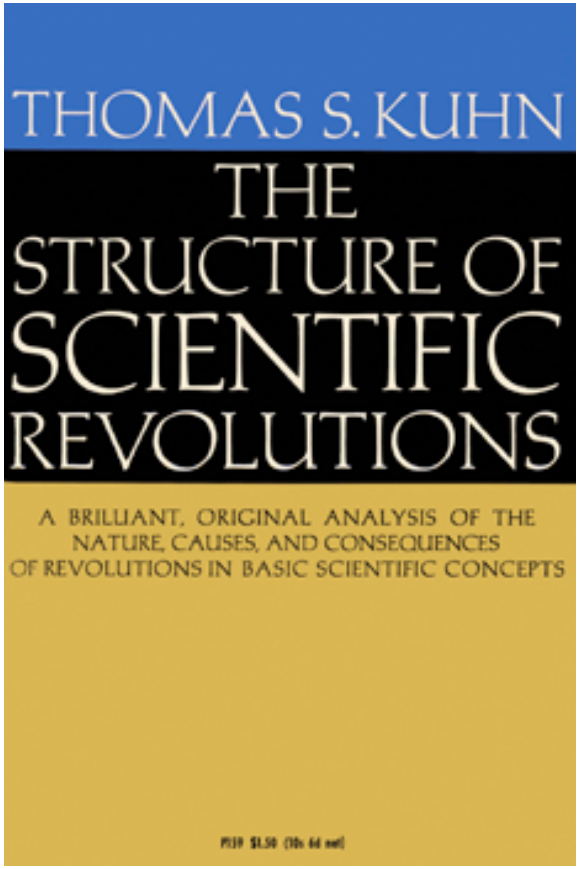
Not a monoculture

- **Machine-level programming**: assembly
- **Low-level programming**: C, C++, Rust
- **Web development**: JavaScript, PureScript, …
- **"Scripting"**: bash, zsh, Python, Ruby, …
- **Numerical computing**: Matlab, Fortran, …
- **Symbolic computing**:
  Lisp, Scheme, OCaml, Haskell, …
- **Deductive reasoning:** Prolog, Twelf, Agda…

- **Destroying the world's economy:** Haskell

“Programming Paradigms”

- What is a paradigm?



"Programming Paradigms"

- What is a paradigm?
  - "a typical example or pattern of something; a **model**… a **worldview** underlying the theories and methodology of a particular scientific subject" (Oxford American Dictionary)

- too. much. ideology.

“Programming Paradigms”

- What is a *programming* paradigm?
  - object-oriented programming
    (“object-oriented paradigm”—163K results)
  - imperative programming
    (“imperative paradigm”—11K results)
  - functional programming
    (“functional paradigm”—58K results)
    “the *functional paradigm* completely changes the way we think about programming”

      —`theburningmonk.com`

“Programming Paradigms”

- What do these mean?
  - object-oriented programming languages
  - imperative programming languages
  - functional programming languages

[switch from slides]

“Programming Paradigms”

no, I said “switch from slides”

"Programming Paradigms"

- Object-oriented

features	**languages**
objects, classes, inheritance, (methods), (instance variables)	Simula-67, Smalltalk, C++, Java, Python (?)

- Imperative

features	**languages**
commands ("statements"), variable assignment, (procedures), (structs)	Fortran, Algol-60, PL/I, Pascal, C, C++, Java (?), Python (?)

- Functional

features	**languages**
functions, (recursion), *no* variable assignment	Lisp, Scheme, Racket, OCaml, Haskell, Python (?), Java (??), C++ (???)

"Programming Paradigms"

- Lots of (?) and overlap (like Python)
- Possible to write in a functional **style** in many languages
  (use lots of anonymous functions)
- Possible to write in an object-oriented **style** in many languages
  (C with function pointers inside structs)
- "Programming paradigm" is grandiose
- But "programming style" seems too trivial

Functional programming (in Haskell)

- "Pure" or "side effect-free" programming
  - no assignment to variables (no overwriting)
  - no loops
  - lots of recursive functions
    (anything you can do with a loop,
    you can do with recursion)
  - no printing; more accurately, weird printing
  - No side effects often makes it **easier to reason about programs**
- Functions can take functions as arguments
- Functions can build new functions

Functional programming (in Haskell)

```
(\x -> x + 1) 3
4

add1 = map (\x -> x + 1)

:type add1
add1 :: Num b => [b] -> [b]

add1 [1, 10, 100]
[2,11,101]
```

Logic programming (in Prolog)

- Unlike Haskell, Prolog is not usually considered **general-purpose**
- Instead, for **deductive reasoning**
  - A Prolog program is a set of **facts** combined with **rules** about deriving (deducing) "new" facts

  Fact: smoky(British_Columbia)

  Rule: if smoky(X) then on_fire(X)

  Is British Columbia on fire?

Logic programming (in Prolog)

- Unlike Haskell, Prolog is not usually considered **general-purpose**
- Instead, for **deductive reasoning**
  - A Prolog program is a set of **facts** combined with **rules** about deriving (deducing) "new" facts

Fact: smoky(British_Columbia)

Rule: if smoky(X) then on_fire(X)

Is British Columbia on fire?

1. smoky(British_Columbia) *[Fact]*
2. if smoky(X) then on_fire(X) *[Rule]*
3. on_fire(British_Columbia) *[X = British_Columbia]*

Logic programming (in Prolog)

Fact: smoky(British_Columbia)

Rule: if smoky(X) then on_fire(X)

Query: Find something that is on fire.

1. if smoky(X) then on_fire(X) *[Rule]*

   Query: Find something that is smoky

   a. smoky(British_Columbia) *[Fact]*

2. on_fire(British_Columbia) *[see previous slide]*

260 Fall 2018: more than a “trip to the zoo”

- Reasoning about programs by **stepping** (tracing)
- Proving properties of programs (by stepping)
- Defining what stepping is by **logical rules**
- “Running” the logical rules (in Prolog)
- Defining types by logical rules

Some learning outcomes

- Write short programs in a functional language such as Haskell or LISP, including the use of recursion, lists, higher-order functions.
- Use structural induction to prove simple assertions about functional programs.
- Write short programs in a logical language such as Prolog.
- Predict the behaviour of small programs written in either paradigm.

Logistics

Logistics

- Lectures:
  - Slides, code on laptop, board, camera projector, …
  - I don’t track attendance or participation in lecture
  - But you’ll probably do better if you attend and participate

Course grade

- Assignments (about 6): **25%**
- Quizzes (4): **45%**
- Final exam: **30%**

Course grade

- Assignments (about 6): **25%**
  - I drop your lowest assignment mark (unless you "depart from academic integrity"), so really about 5 assignments worth ~5% each
  - **"~": I can re-weight assignments**
- Quizzes (4): **45%**
  - I drop your lowest quiz mark (unless you "depart from academic integrity"), so really 3 quizzes, worth ~15% each
  - **"~": I can re-weight quizzes**
- Final exam: **30%**

Textbooks

- Two required textbooks:
  - Simon Thompson. *Haskell: The Craft of Functional Programming*, third edition.
  - Ivan Bratko. *Prolog Programming for Artificial Intelligence*, fourth edition.
- About $200 new at the Queen's bookstore
- Consider used copies
  (both books have been used for 260 before)

Academic integrity

- Don’t copy (from anyone or anywhere)
- Don’t allow anyone to copy from you
- Don’t work on individual assignments together
- Get help from me or the TAs
- On Piazza: you can post questions about assignments, but be very careful if you answer them—let me and the TAs decide how much to “give away”
  - (Feel free to answer general questions!)

Academic integrity

- Sometimes people are tempted to "depart from academic integrity" because they feel desperate
  - Try to ask for help
  - Helping individual students is the best part of teaching

Academic Consideration

- New(ish) policy (2018) for the Faculty of Arts and Science
  - Self-Declaration of Brief Absence (up to 48 hours)
  - Request for Academic Consideration for Extenuating Circumstances (>48 hours)
- https://webapp.queensu.ca/artsci/acrp/
- Single point of contact, instead of explaining your situation to every professor individually
  - The new system should be harder to seriously abuse

Software

- Software is bad

Software

- Software is bad

…oh, wrong course, never mind

Software

- Haskell: two major alternatives
  1. GHC (Glasgow Haskell Compiler) ("Haskell Platform")
     - WinGHCi
     - macOS / Linux: ghci
  2. hugs

  They give different, but often equally bad, error messages

Software

- Prolog: SWI-Prolog
  - more information in a few weeks, when we get into Prolog

Software

- A text editor
  - not a word processor (Word, TextEdit (!), …)
  - Atom (Windows, macOS, Linux)
  - Notepad++ (Windows)
  - Aquamacs (macOS)
  - Editors for old people (like me):
    Emacs, vim (Linux, …)

To do, as soon as possible

1. Sign up on the 260 Piazza:
   https://piazza.com/queensu.ca/fall2018/cisc260
   - Good for Q & A and discussions
   - onQ for other stuff, like announcements
2. Make sure you can access CASLab
3. Install GHCi
4. Install (if you don't already have one) a text editor

Next lecture (tomorrow):

Basics of Haskell