
DECLARATIVE PROGRAMMING LANGUAGES

Jim Royer
EECS Department

A HISTORICAL DIGRESSION



- ▶ Algebra is largely an Arabic invention.
- Muhammed ibn-Musa al-Khwarizmi – 9th century AD
- “completing the square”
- “solving quadratic equations”
($ax^2 + bx + c$)

A HISTORICAL DIGRESSION, CONTINUED

- ▶ The Renaissance Italians built on the Arabic work.
 - Motivated by financial and commercial problems
 - General solutions to cubic and quartic equations
 - But what did their equations look like ...

Gerolamo Cardano
1501 – 1576



CARDANO'S RHETORICAL ALGEBRA (CIRCA 1550)

Solve:

Let a cube and six times its side equal 20.

$$x^3 + 6x = 20$$

A Solution:

Cube one third the number of sides (i.e., the coef. of x). Add to it the square of one-half the constant of the equation and take the square root of the whole. You will put this twice, and to the one of the two you add one-half the number you have squared and from the other you subtract one haft the same. You will then have a binomium (i.e., $\sqrt{108}+10$) and its apotome (i.e., $\sqrt{108}-10$). Then, subtracting the cube root of the apotome from the cube root of the binomium the remainder is the required side.

$$x = \sqrt[3]{\sqrt{108} + 10} - \sqrt[3]{\sqrt{108} - 10}$$

THE MODERN NOTATION

▶ François Viète
1540 – 1603

- French lawyer, royal counselor, mathematician, and **cryptographer**
- 1592: “In Artem Analyticem Isagoge”



THE OLD VERSUS THE NEW NOTATION

The old notation

concrete

spells out a computation

hard to manipulate

The new notation

abstract

spells out a relationship

easy to manipulate

PROGRAMMING: IMPERATIVE VS. DECLARATIVE

Ordinarily technology changes fast. But programming languages are different: programming languages are not just technology, but what programmers think in. They're half technology and half religion.

— Paul Graham, “Beating the Averages”

Imperative programming languages

- ▶ E.g, C, C++, ...
- ▶ spell out computations
- ▶ tricky to manipulate

Declarative programming languages

- ▶ E.g., Lisp, Scheme, ML, Haskell, Python, ...
- ▶ spell out relations (ideally)
- ▶ easy to manipulate
- ▶ more abstract

DECLARATIVE LANGUAGES: PROS AND CONS

Cons

- ▶ Weak programmers do not fare well.
- ▶ There is sometimes a price in performance, **but not always.**
- ▶ Some things can be tricky to express, **e.g., “:=” in Haskell.**

Pros

- ▶ Creative programmers fare well.
- ▶ Some things are easy to express that are unheard of in the imperative setting, **e.g., Lisp macros.**
- ▶ Programs can be very close to specifications.
- ▶ There is a long, successful track record in prototyping. **E.g., Perl 6 is being developed in Haskell.**

A QUICK ASIDE: IT ISN'T A BOYS-ONLY CLUB

- ▶ Haskell borrows **many** ideas from modern/abstract/structural algebra
- ▶ Emmy Noether
1882 – 1935
 - Key figure in the development of modern algebra
 - A hero in physicists for her work on symmetry and conservation laws.
 - Threw the best parties in Göttingen.



SOME PLACES TO FIND OUT MORE

- ▶ P. Graham's essays, especially **“Beating the Averages”**
<http://www.paulgraham.com/articles.html>
- ▶ Wikipedia's Haskell entry:
[http://en.wikipedia.org/wiki/Haskell_\(programming_language\)](http://en.wikipedia.org/wiki/Haskell_(programming_language))
Under external links see especially:
 - **“Wearing the hair shirt: A retrospective on Haskell”** by S.P. Jones
 - **“Haskell vs. Ada vs. C++ vs. Awk vs. ... An Experiment in Software Prototyping Productivity”** by P. Hudak and M. Jones
- ▶ H. Abelson and G. Sussman's **“Structure and Interpretation of Computer Programs, 2/e”**, MIT Press, 1996.
- ▶ L. Paulson's **“ML for the Working Programmer”**, Cambridge Univ. Press, 1996.

AN EXAMPLE

```
-- The list of all primes via the Sieve of Eratosthenes  
-- A bit slow, but cute!
```

```
primes = sieve [2 ..]  
  where sieve (x:xs) = x : sieve [ y | y <-xs, y `mod` x > 0]
```

```
[2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59,  
 61, 67, 71, 73, 79, 83, 89, 97, 101, 103, 107, 109, 113, 127,  
131, 137, 139, 149, 151, 157, 163, 167, 173, 179, 181, 191,  
193, 197, 199, 211, 223, 227, 229, 233, 239, 241, 251, 257,  
263, 269, 271, 277, 281, 283, 293, 307, 311, 313, 317, 331,  
337, 347, 349, 353, 359, 367, 373, 379, 383, 389, 397, 401,  
409, 419, 421, 431, 433, 439, 443, 449, 457, 461, 463, 467,  
479, 487, 491, 499, 503, 509, 521, 523, 541, 547, 557, 563,  
569, 571, 577, 587, 593, 599, 601, 607, 613, 617, 619, 631,  
...
```