CS 421 Lecture 15

- Today's class: APL
  - Functional programming – "no side effects"

Functional Programming

- "The assignment statement splits programming into two worlds. The first world comprises the right sides of assignment statements. This is an orderly world of expressions, a world that has useful algebraic properties.... It is the world in which most useful computation takes place.
- "The second world... is the world of statements. … This world of statements is a disorderly one, with few useful mathematical properties."

John Backus (creator of Fortran), "Can Programming be liberated from the von Neumann Style? A Functional Style and its Algebra of Programs." Turing Award lecture, 1977.

In C++ are these assertions true:

(1) $f()+1 > f()$?

(2) $e_1 + e_2 == e_2 + e_1$?

(3) $x = y+1;$ if $(x>y)$...

APL

- Computations on matrices using operators that have matrix arguments.
- Ken Iverson – “A Programming Language” – 1960
- Defined a set of operators on matrices, plus a typeface for those operators, and built terminals

APL operations

- Binary operations on numbers extended naturally to matrices
  - Comparison and boolean ops treated as arithmetic
- Reduction operations: $+/$, $\times/$, $\wedge/$, ...
  - For vectors, put operator between every element
  - For matrices, reduce each row
- Compression: B / V
  - selects elements (or rows) of V where $B = 1$

APL font

Monadic - unary
Dyadic - binary

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

0 1 2 3 4 5 6 7 8 9

"shape" - monadic: shape of argument
dyadic: creates a value of given shape

and

≤ ≥ ≠ < > = ι ρ ⌽ ∧ / ÷ ← ⋆ × + − [ ]

subscripting

transpose

dyadic arithmetic op's

assignment

index generation (monadic)

dyadic comparisons

dyadic: compression
also used in reduction op's

APL examples

$M = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}$ (2×3 matrix)

$V = 2\ 4\ 6\ 8$

- $1 + M = \begin{pmatrix} 2 & 3 & 4 \\ 5 & 6 & 7 \end{pmatrix}$
- $(+/V) \div n$

  If $n=4$: $(2+4+6+8) \div 4 = 5$
- $(+/V) \div \rho V$ = average of V
- $(((V \div 2) \times 2) = V)/V$ 0 1 0 0 1

  $V = 1\ 2\ 5\ 9\ 10$ 2 10

  0 1 2 4 5

  0 2 4 8 10 = 1 2 5 9 10

$n = 8$

APL examples

- `prime n = ∧/(0≠n%(1+ι(n-2)))`

6

1 2 3 4 5 6

index gen

2 3 4 5 6 7

8%2 8%3 . . . 8%7

= 0 2 0 3 2 1

0 1 0 1 1 1

∧/0 1 0 1 1 1 = 0∧1∧0..∧1 = 0

APL examples

$\boxed{V[\iota\rho V]} = V$

- Subscripting: `V[V']` – elements of `V` in positions given by `V'`.

```
reverse V = V[1+(ρV)-ιρV]
```

n n-1 ... 1

$\underbrace{n - 1\ 2\ 3 \ldots n}_{n-1\ \ n-2\ \ldots\ 0}$

length of V

APL examples

- Dyadic ρ – "restructure"
- $V \rho A$ returns a value with shape V, values drawn from A

$10 \rho 7 = \underbrace{7\ 7\ 7 \cdots 7}_{10\ \text{times}}$

$10 \rho 1\ 2 = 1\ 2\ 1\ 2\ 1\ 2\ 1\ 2\ 1\ 2$

$2\ 3 \rho 7 = \begin{pmatrix} 7 & 7 & 7 \\ 7 & 7 & 7 \end{pmatrix}$

$2\ 3 \rho \iota 6 = 2\ 3 \rho 1\ 2\ 3\ 4\ 5\ 6 = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}$

$2\ 3 \rho \iota 5 = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 1 \end{pmatrix}$

$(2 \rho n) \rho \underbrace{1 , n \rho 0}_{\text{catenation}}$ (say $n = 5$)

$0\ 0 \cdots 0$

$\boxed{5\ 5 \rho 1\ 0\ 0\ 0\ 0\ 0}$

$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$

n = 4

APL examples

- ← assignment
- ⍉ transpose

```
(⍉M) = M←(2⍴n)⍴⍳n
```

$4\ 4\ \rho\ 1\ 2\ 3\ 4$

$$M = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 1 & 2 & 3 & 4 \\ 1 & 2 & 3 & 4 \\ 1 & 2 & 3 & 4 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 1 & 1 & 1 \\ 2 & 2 & 2 & 2 \\ 3 & 3 & 3 & 3 \\ 4 & 4 & 4 & 4 \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

APL examples

let one = newint 1;

let zero = newint 0;;

let four = newint 4;;

let a = rho(newveci [2;3]) (indx (newint 6));; 2 3 ρ ι 6

let v = newveci [2;4;6];;

let c = newveci [1;0];;

let d = newveci [1;0;1];;

$$\begin{matrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{matrix}$$

a *@ a — ×

v -@ one —

a >@ four >

$$\begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 1 \end{pmatrix}$$

!+v — +/ v

APL examples

maxR a , minR

d % v
c % a } compression

shape a } monadic ρ

ravel a } - turn a into a vector

rho (shape a) v - (ρa)ρv - value with shape of a, elements from v

rho (shape v) c

a ^@ c - catenate, i.e. dyadic ,

trans - transpose

@@ - subscript

show - convert to string

APL examples

indx (newint 5)   1 2 3 4 5

trans a

v @@ (indx two)   v [1 2] , eg v = newveci [5; 6; 7; 8]
v @@ (indx two) = [5; 6]

a @@ one

(trans a) @@ (indx two)

First element of a a vector,
first row if a a matrix

APL examples

```
let incr a = a +@ (newint 1);;
let fac n = !* (indx n);;
let avg v = (!+v) /@ (shape v);;
let reverse v =
  let sz = (shape v) @@ one
  in v @@ (incr (sz -@ (indx sz)));;
let prime n = !& (zero <>@ (n %@ (incr
  (indx (n -@ two)))));;
```

Lessons from APL

- APL makes you think about programming different
- Lesson 1 in language design:

  Don't !