A Powerful Notation for Problem Solvers

Richard Park

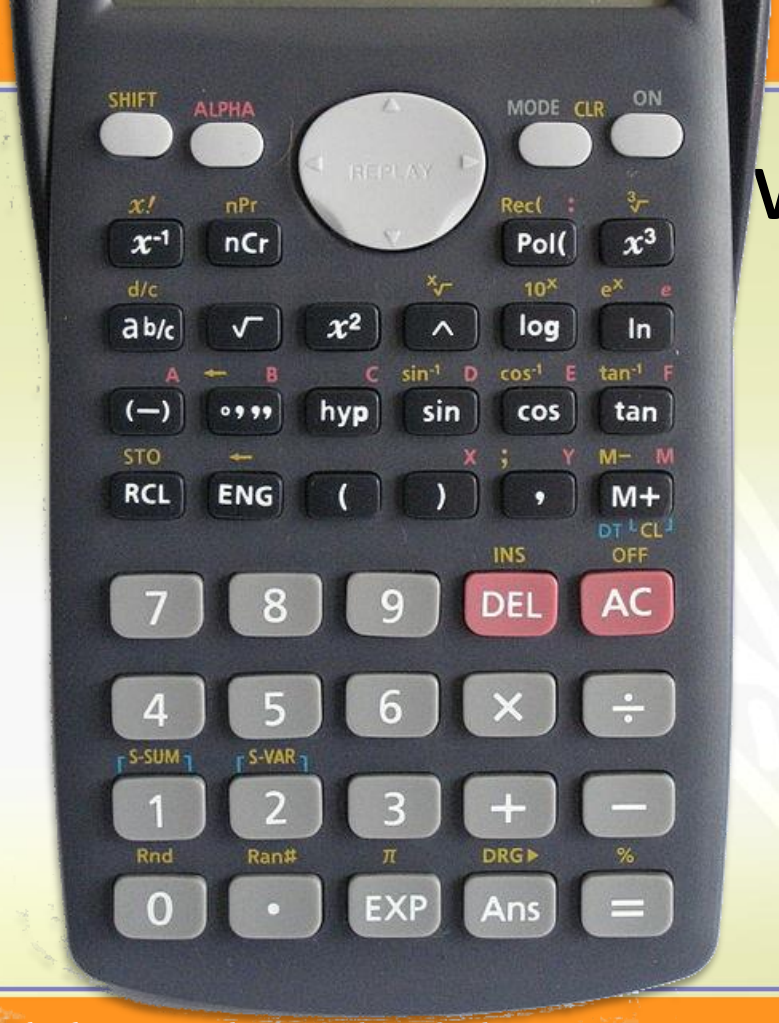
DYALOG

Applied mathematics is largely concerned with the design and analysis of explicit procedures for calculating the exact or approximate values of various functions.

Such explicit procedures are called algorithms or programs.

Because an effective notation for the description of programs exhibits considerable syntactic structure, it is called ***a programming language****.*

- Kenneth E. Iverson 1962


SHIFT
ALPHA
MODE CLR
ON
REPLAY
Pol(
log
ln
tan
RCL
ENG
DEL
AC
S-SUM
S-VAR
Rnd
Ran#
DRG
EXP
Ans

w APL

+ − × ÷ !

< ≤ = ≠ > ≥

∩ ∪ ∈ ∧ ∨

Some algorithms

- Quadratic equation
- Factorial
- Fibonacci

Quadratic Equation

$$f(x) = 2x^2 - 9x - 5$$

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Factorial

$$F(n) = 1 \times 2 \times 3 \times \cdots \times (n-2) \times (n-1) \times n$$

$$F(n) = \prod_{i=1}^{n} i$$

$$F(n) = \begin{cases} 1 & n = 1 \\ n \times F(n-1) & n > 1 \end{cases}$$

Fibonacci

$$F(n) = \begin{cases} n\ , & n \leq 1 \\ F(n-1) + F(n-2)\ , & \text{otherwise} \end{cases}$$

You already know APL

Arithmetic	+	−	×	÷	!	
Comparisons	<	≤	=	≠	>	≥
Sets & Logic	∩	∪	∈	∧	∨	

Half of it you can just guess

⍬

⌽

⌈

⌊

≠

Half of it you can just guess

`reverse` ⌽

⊖

⌈

⌊

≠

Half of it you can just guess

⌽

⍉

⌈

⌊

≠

Half of it you can just guess

TRANSPOSE

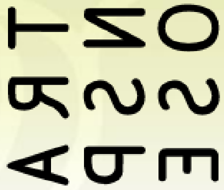
Half of it you can just guess

Half of it you can just guess

reverse

⌽

⍉



⌈7.3

⌊

≢

Half of it you can just guess

Half of it you can just guess

reverse

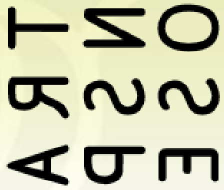
⌽

⍉

⌈ 8

⌊ 7.3

≠



Half of it you can just guess

Half of it you can just guess

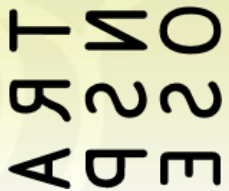
reverse (mirrored)

TRA
NSP
OSE
(transposed)

```
⌽
⍉
⌈ 8
⌊ 7
≢    'tally'
```

Half of it you can just guess

```
reverse      ⌽
             ⊖
             ⌈ 8
             ⌊ 7
5 = ≠        'tally'
```



A tool kit that grows

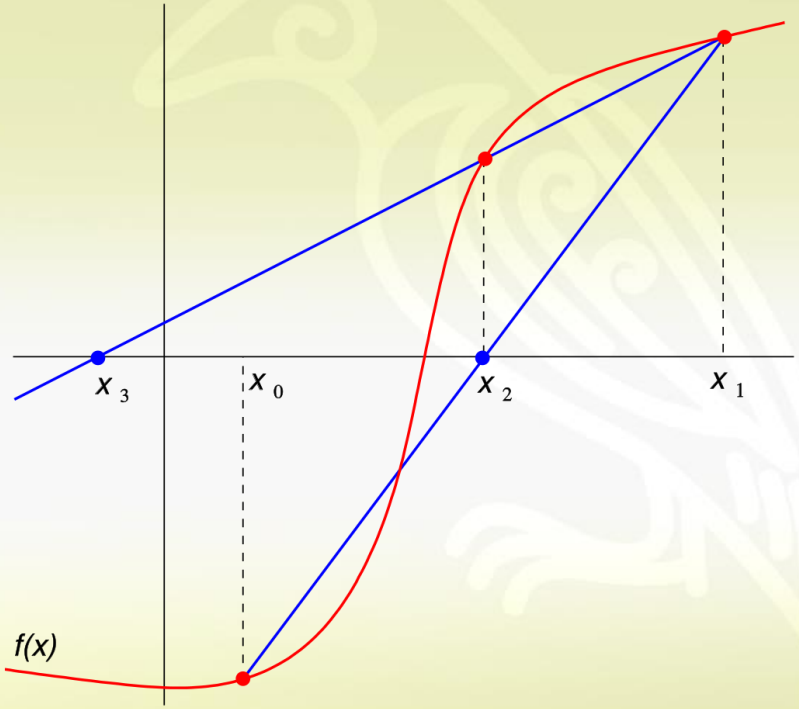
```
PlusMinus ← + , -

Average   ← +/ ÷ ≢
```

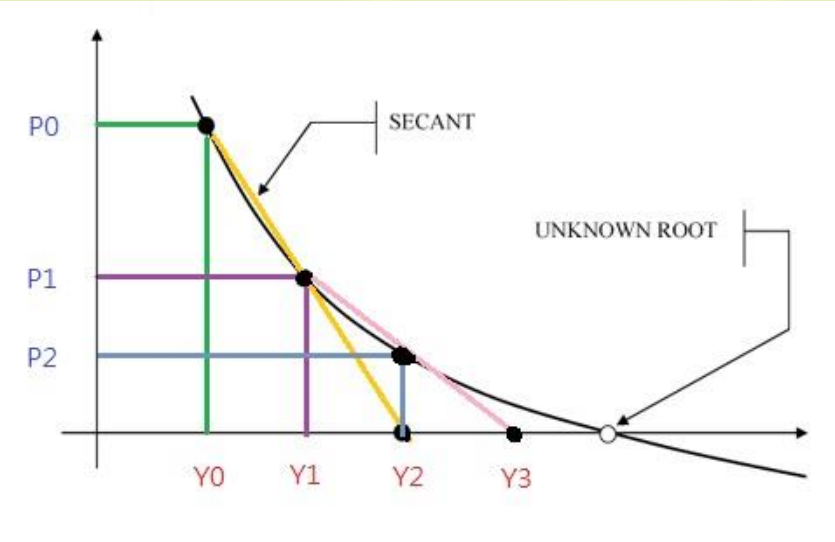
Some disparate notations are unified

Secant Method

$$x_n = \frac{x_{n-2}f(x_{n-1}) - x_{n-1}f(x_{n-2})}{f(x_{n-1}) - f(x_{n-2})}$$

Secant Method





Examples: Today's Problems

Cypher

Nails

Zoo

Tap Encoding

	1	2	3	4	5
1	A	B	C	D	E
2	F	G	H	I/J	K
3	L	M	N	O	P
4	Q	R	S	T	U
5	V	W	X	Y	Z

Table 1 – Polybius Square

```
      TapEnc'APL'
* * *** ***** *** *
```

Tap Encoding

	1	2	3	4	5
1	A	B	C	D	E
2	F	G	H	I/J	K
3	L	M	N	O	P
4	Q	R	S	T	U
5	V	W	X	Y	Z

Table 1 – Polybius Square

```
      TapEnc'APL'
* *  ***  *****  ***  *
```

Tap Encoding

	1	2	3	4	5
1	A	B	C	D	E
2	F	G	H	I/J	K
3	L	M	N	O	P
4	Q	R	S	T	U
5	V	W	X	Y	Z

Table 1 – Polybius Square

```
      TapEnc'APL'
* * *** ***** *** *
```

Tap Encoding

	1	2	3	4	5
1	A	B	C	D	E
2	F	G	H	I/J	K
3	L	M	N	O	P
4	Q	R	S	T	U
5	V	W	X	Y	Z

Table 1 – Polybius Square

```
      TapEnc'APL'
* * *** ***** *** *
```

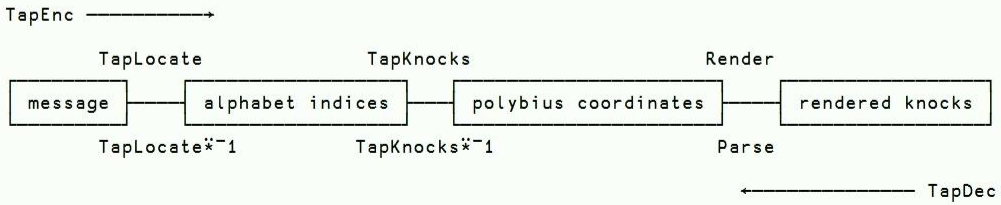
Define **TapEnc** and **TapDec** which converts between a message and a tap encoding as defined by the following polybius square:

	1	**2**	**3**	**4**	**5**
1	A	B	C	D	E
2	F	G	H	I/J	K
3	L	M	N	O	P
4	Q	R	S	T	U
5	V	W	X	Y	Z

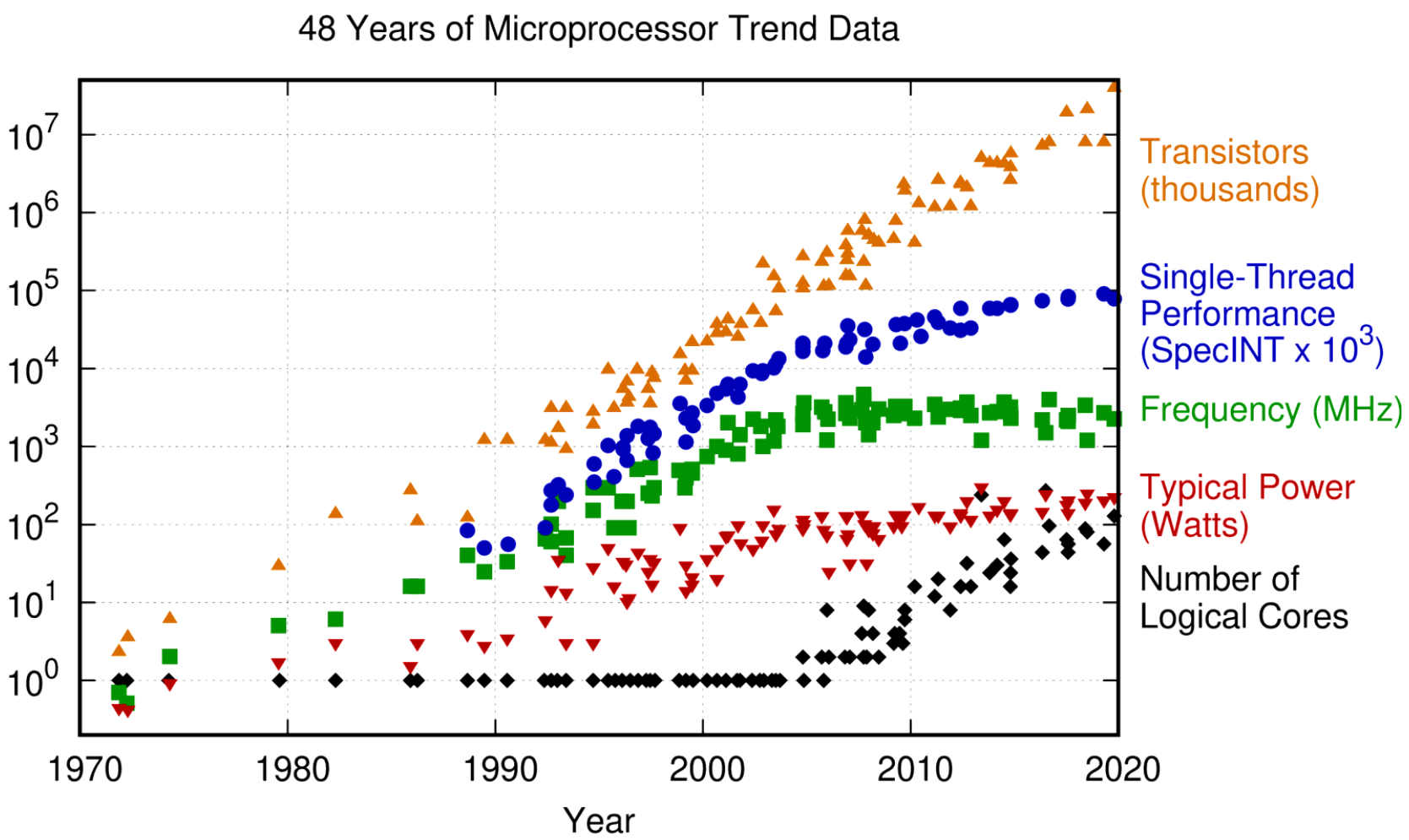
(I/J both map to the same encoding. When decoding, always produce I instead of J.) e.g.

```
        TapEnc 'APL'
* * *** ***** *** *

        TapDec '* * *** ***** *** *'
APL
```

```
TapAlpha←'ABCDEFGHIKLMNOPQRSTUVWXYZJ'
TapKnocks←↓∘⍉1+5 5⊤-∘1
TapLocate←(⊢-(17×∘⌊26÷⍨⊢))(TapAlpha∘⍳)

Render←' '(1↓∘∈,¨)'*'/⍨¨∈
TapEnc←Render TapKnocks∘TapLocate

Parse←(⍴∘1 0∘≠⊂⊢)(≠⌸' '∘(≠⊆⊢))
TapDec←(TapKnocks∘TapLocate⍣¯1)Parse
```



Thinking in rectangles



Original data up to the year 2010 collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten
New plot and data collected for 2010-2019 by K. Rupp

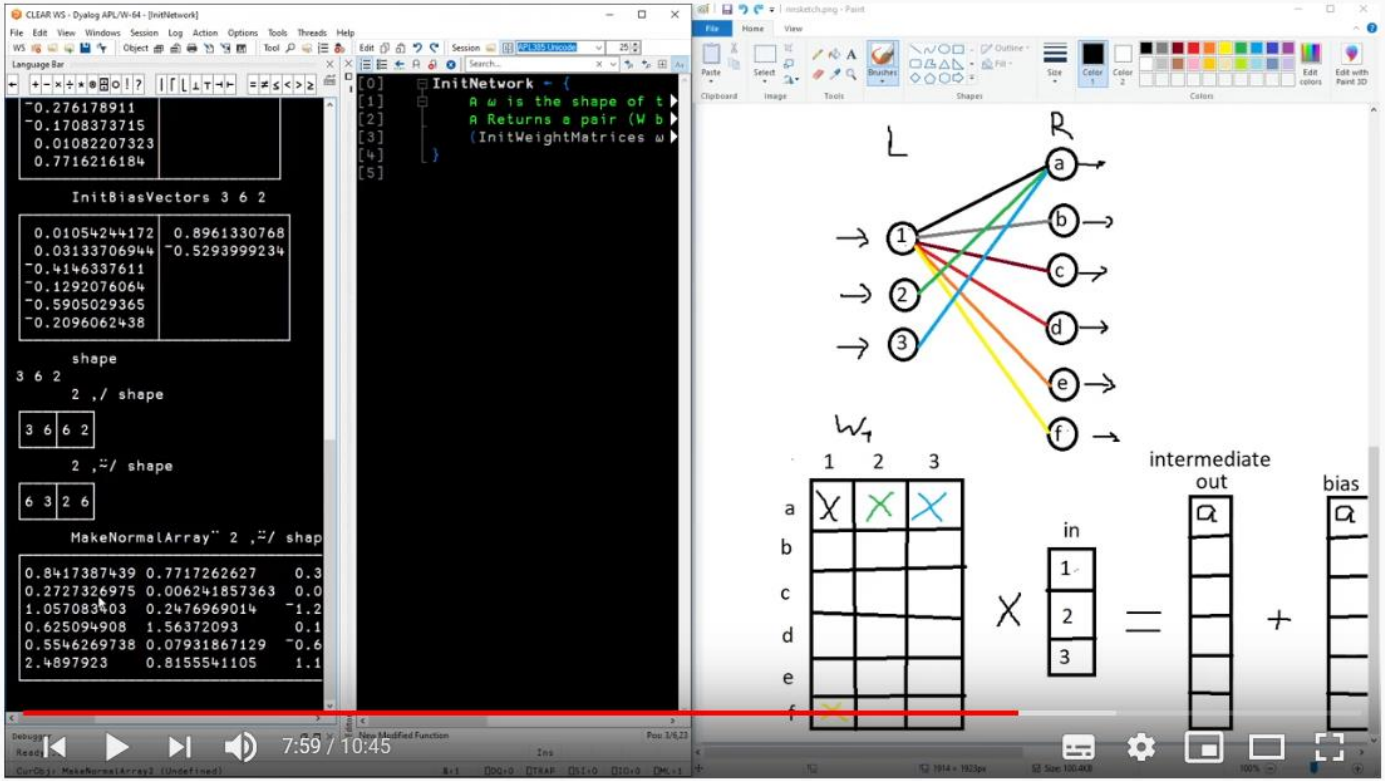
Rectangles are fashionable again

$$F(x) = \frac{1}{1 + e^{-x}}$$

```
⍝ Sigmoid function

F ← {
      ÷1+*-⍵
}
```

```
BackProp ← {
      ⍝ Compute the derivatives of the network parameters w.r.t. some network output.
      ⍝ ω: consecutive layer outputs.
      ⍝ α: expected network output.
      dWs ← ⍬
      dbs ← ⍬
      xs ← 1↓⊃ω
      dxs ← ⊂α LossFn.DLoss⊃⊃ω
      BP ← {
          ⍝ Auxiliary function.
          ⍝ The layers and xs are in the reverse natural order of the network,
          ⍝ while the derivative arrays are in the natural order.
          (layers xs dWs dbs dxs) ← ⍵
          l ← ⊃layers
          x ← l.bias+l.weights+.×⊃xs
          db ← (⊃⊃dxs)×l.ActivationFn.DF x
          dx ← (⍉l.weights)+.×db
          dW ← db+.×⍉⊃xs
          (1↓layers) (1↓xs) (dWs,⊂dW) (dbs,⊂db) (dxs,⊂dx)
      }
      r ← BP⍣(≠layers) ⊢ (⊃layers) xs dWs dbs dxs
      (_ _ dWs dbs _) ← r
      (⊃dWs) (⊃dbs)
  }
```

```
BP ← {
      ⍝ Auxiliary function.
      ⍝ The layers and xs are in the reverse natural order of the network,
      ⍝ while the derivative arrays are in the natural order.
      (layers xs dWs dbs dxs) ← ⍵
      l ← ⊃layers
      x ← l.bias+l.weights+.×⊃xs
      db ← (⊃⍞dxs)×l.ActivationFn.DF x
      dx ← (⍉l.weights)+.×db
      dW ← db+.×⍉⊃xs
      (1↓layers) (1↓xs) (dWs,⊂dW) (dbs,⊂db) (dxs,⊂dx)
  }
  r ← BP⍣(≠layers) ⊢ (⍞layers) xs dWs dbs dxs
  (_ _ dWs dbs _) ← r
  (⍞dWs) (⍞dbs)
}
```



11 - Initialising a Neural Network | "Learn APL with Neural Networks"

8 views • 4 Nov 2020

1 0 SHARE SAVE ...

Rodrigo Girão Serrão
18 subscribers

SUBSCRIBED

So who's using this anyway?

Business & Finance

Medical Records

Karolinska Universitetssjukhuset.

Visby lasarett/Gotlands kommun.

Danderyds Sjukhus.

Södertälje Sjukhus.

ASIH inom Stockholms Läns Sjukvårdsområde.

Geriatrik inom Stockholms Läns Sjukvårdsområde.

Psykiatri inom Stockholms Läns Sjukvårdsområde.

Primärvård inom Stockholms Läns Sjukvårdsområde.

S:t Eriks Ögonsjukhus.

Nynäs vård AB.

Rehab Station Stockholm.

Stiftelsen Stora Sköndal.

Liljeholmsmottagningen.

Sophiahemmet.

Södersjukhuset, in process.

Installations in the private sector:

Stockholms Sjukhem.

Brommageriatriken AB.

So who's using this anyway?

Business & Finance

Medical Records

Science

DNA Oil refinery NMR

Computer Science

Co-dfns

```
⍝ A  B  E  F  G  L  M  N  O  P  V  Z
⍝ 0  1  2  3  4  5  6  7  8  9 10 11
tt←{⎕←'C' ⋄ ((d t k n)exp sym)←⍵ ⋄ I←{(⊂⍵)[]⍺}
 r←I@{t[⍵]≠3}⍣≡⍨p⍳2{p[⍵]←⍺[⍺⍳⍵]}⌿⊢∘⊂⌸d⍳p←⍳≠d                                  ⍝ PV
 p,←n[i]←(≠p)+⍳≠i←⍳(t=3)∧p≠⍳≠p ⋄ t k n r,←3 1 0(r[i])⍴⍨⍨≠i                  ⍝ LF
 p r I⍨←⊂n[i]@i⊢⍳≠p ⋄ t k(⍳@i⍨)←10 1
 i←(⍳(~t∈3 4)∧t[p]=3),{⍵⌿⍨2|⍳≠⍵}⍳t[p]=4 ⋄ p t k n r⌿⍨←⊂m←2@i⊢1⍴⍨≠p             ⍝ WX
 p r i I⍨←⊂j←(+⍀m)-1 ⋄ n←j I@(0≤⊢)n ⋄ p[i]←j←i-1
 k[j]←-(k[r[j]]=0)∨0@({⊃⌽⍵}⌸p[j])⊢(t[j]=1)∨(t[j]=2)∧k[j]=4 ⋄ t[j]←2
 p[i]←p[x←¯1+i←{⍵⌿⍨~2|⍳≠⍵}⍳t[p]=4] ⋄ t[i,x]←t[x,i] ⋄ k[i,x]←k[x,i]            ⍝ LG
 n[x]←n[i] ⋄ p←((x,i)@(i,x)⊢⍳≠p)[p]
 n[p⌿⍨(t[p]=2)∧k[p]=3]+←1                                                    ⍝ CI
 p[i]←p[x←p I@{~t[p[⍵]]∈3 4}⍣≡i←⍳t∈4,(⍳3),8+⍳3] ⋄ j←(⌽i)[⍋⌽x]                ⍝ LX
 p t k n r{α[⍵]@i⊢α}←⊂j ⋄ p←(i@j⊢⍳≠p)[p]
 s←¯1,⍨∈⍳⍨n[∪x]←⊢∘≠⌸x←0[]⍉e←∪I∘⍋⍨rn←r[b],⍪n[b←⍳t=1]                          ⍝ SL
 d←(≠p)↑d ⋄ d[i←⍳t=3]←0 ⋄ _←{z⍳d[i]+←⍵≠z←r[⍵]}⍣≡i ⋄ f←d[0[]⍉e],¯1              ⍝ FR
 xn←n⌿⍨(t=1)∧k[r]=0                                                          ⍝ XN
 v←⍳(n<¯4)∧(t=10)∨(t=2)∧k=4 ⋄ x←n[y←v,b] ⋄ n[b]←s[e⍳rn] ⋄ i←(≠x)⍴c←≠e         ⍝ AV
 _←{z/⍨c=i[1[]z]←e⍳⍉x I@1⊢z←r I@0⊢⍵}⍣≡(v,r[b])⍪⍉⍪⍳≠x
 f s←(f s I⍨⊂i)⍳@y⍨⊂¯1⍴⍨≠r
 p t k n f s r d xn sym}
```

BONFIRE

⍝ Co-dfns: Thesis Edition
⍝ A B E F G L M N O P V Z
⍝ 0 1 2 3 4 5 6 7 8 9 10 11
tt←{⎕←'C' ⋄ ((d t k n)exp sym)←⍵ ⋄ I←{(⊂⍵)⌷⍺}
r←I@{t[⍵]≠3}⍣≡⍨p⊣2{p[⍵]←⍺[⍺⍸⍵]}⌿⊢⌸d⊣p←⍳≢d ⍝ PV
p,←n[i]←(≢p)+⍳≢i←⍸(t=3)∧p≠⍳≢p ⋄ t k n r,←3 1 0(r[i])⍴⍨¨≢i ⍝ LF
p r I⍨←⊂n[i]@i⊢⍳≢p ⋄ t k(⊣@i⍨)←10 1
i←(⍸(~t∊3 4)∧t[p]=3),{⍵⌿⍨2|⍳≢⍵}⍸t[p]=4 ⋄ p t k n r⌿⍨←⊂m←2@i⊢1⍴⍨≢p ⍝ WX
p r i I⍨←⊂j←(+⍀m)-1 ⋄ n←j I@(0≤⊢)n ⋄ p[i]←j←i-1
k[j]←-(k[r[j]]=0)∨0@({⊃⌽⍵}⌸p[j])⊢t[j]=1 ⋄ t[j]←2
p[i]←p[x←¯1+i←{⍵⌿⍨~2|⍳≢⍵}⍸t[p]=4] ⋄ t[i,x]←t[x,i] ⋄ k[i,x]←k[x,i] ⍝ LG
n[x]←n[i] ⋄ p←((x,i)@(i,x)⊢⍳≢p)[p]
n[p⌿⍨(t[p]=2)∧k[p]=3]+←1 ⍝ CI
p[i]←p[x←p I@{~t[p[⍵]]∊3 4}⍣≡i←⍸t∊4,(⍳3),8+⍳3] ⋄ j←(⌽i)[⍋⌽x] ⍝ LX
p t k n r{⍺[⍵]@i⊢⍺}←⊂j ⋄ p←(i@j⊢⍳≢p)[p]
s←¯1,⍨∊⍳¨n[∪x]←⊢∘≢⌸x←0⌷⍉e←∪I∘⍉⍨rn←r[b],⍪n[b←⍸t=1] ⍝ SL
d←(≢p)↑d ⋄ d[i←⍸t=3]←0 ⋄ _←{z⊣d[i]+←⍵≠z←r[⍵]}⍣≡i ⋄ f←d[0⌷⍉e],¯1 ⍝ FR
xn←n⌿⍨(t=1)∧k[r]=0 ⍝ XN
v←⍸(t=10)∧n<¯4 ⋄ x←n[y←v,b] ⋄ n[b]←s[e⍳rn] ⋄ i←(≢x)⍴c←≢e ⍝ AV
_←{z/⍨c=i[1⌷z]←e⍳⍉x I@1⊢z←r I@0⊢⍵}⍣≡(v,r[b]);⍉;⍳≢x
f s←(f s I¨⊂i)⊣@y¨⊂¯1⍴⍨≢r ⋄ p t k n f s r d xn sym}

So who's using this anyway?

Business & Finance

Medical Records

Science

DNA Oil refinery NMR

Computer Science

Co-dfns

Simulation: https://stormwind.fi/

Where to find out more?

tryapl.org

apl.wiki

apl.chat

dyalog.tv

Why learn APL?

A tool of thought

Think in rectangles

Kattis

Why should I start learning today?

Annual APL Problem Solving Competition

Launched this week

Compete for $$$

Previous winners learned APL *while* participating

Referral awards

dyalogaplcompetition.com