

ATRAV

6/14/74

>SP ARUN

>RM

~3 mins

>SI

>CP

1/GLOBAL/

/REDUCE/

/ECUDER/

/OUTER/

5/INNER/

/DYSCENS/

/IDELEM/

/e/

/v/

10/ROTATE/

/ROTATE/

/↑/

/↓/

/UPGRADE/

15/DOWNGRADE/

/αCαOαMαAα/

/INDEX1/

/INDEX2/

/INDEX3/

20/TRANPOSE/

/TRANPOSE/

/./

/|/

/\|/

25/OUTONE/

/OUTNUMSC/

/FRACTPART/

/-/

/INONE/

30/+/

/INNUM/

/αCOMPAT/

/ASINDEX1/

/ASINDEX2/

35/ASINDEX3/

/SQRT/

/QSC/

/*/

/*/

40/LN/

/LN/

/T/

/1/

/CIRCLESC/

45/0/

/FACSC/

/EXCLAMATION/

/COMSC/

/EXCLAMATION/

50/!/

/!/

/OUT/

/αSTART/

/□START/

55/□KILL/

/αINIT/

/□CR_PROC/

/□BOXES_FILE/

/□BOXES_TERM/

60/□RECEIVE/

/□SETCONV/

/□TS/

/□S_AL/

64/□R_AL/

>EX

□

10/21/74

JHG:

Don't throw out

JHG

RAND

106

V WIDTH (//) DIGITS : V

V IO_BOXES + (e) GLOBAL: V

V (→) A : 204 V

V A + NOP B : 210 V

V ATTN A : 212 V

V αA αATN αB : 212 V

V (†) A : V

V A + (†) : V

V OUTCHARSC A : 212 V

V A + INCHARSC : 211 V

V RETURN : 208 V

V X + A (+) B : 112 V

V αA αASNO αB : 113 V

V X + (+) A : 128 V

V X + (-) A : 129 V

V X + (L) A : 130 V

V X + (I) A : 131 V

V X + (J) A : 132 V

V X + (~) A : 133 V

V X + (u) : 192 V

V (//) A : 193 V

V X + A INTCH B : 194 V

V X + A TYPECONV B : 143 V

V X + (L) A : 1 INTCH TYPECONV V

V X + (T) A : 0 INTCH TYPECONV V

V X + A COEFFBEXP B : 134 V

V X + COEF B : 1 INTCH COEFFBEXP V

V X + BEXP A : 0 INTCH COEFFBEXP V

V X + (p) A : 197 V

V X + A (p) B : 198 V

V X + (,) A : 199 V

V X + A COMAαINST B : 200 V

V X + (i) A : 201 V

V X + A (=) B , 1 : 142 V

V X + A (x) B , 2 : ~: V

V X + A (+) B , 3 : 135 V

V X + A (-) B , 4 : 136 V

V X + A (x) B , 5 : 137 V

V X + A (÷) B , 6 : 138 V

V X + αA (I) αB , 9 : αASNO αA αASNO αB 0 αA = αA + αB ÷ L: αA × αB - U: ← αA αASNO αB V

V X + A (<) B , 14 : 141 V

V X + A (≤) B , 15 : INTCH < ~: V

V X + A (≥) B , 16 : < ~: V

V X + A (>) B , 17 : INTCH < V

V X + A (v) B , 18 : 140 V

V X + A (Λ) B , 19 : 139 V

V X + A NOR B , 20 : v ~: V

V X + A NAND B , 21 : Λ ~: V

V X + (÷) A : 1.0 ÷ V

V X + (x) αA : + αA 0 INTCH < αA 0 < - U: αASNO αA V

V X + PI : 3.14159265389793 V

∇ X ← (○) A : PI : x ∇

∇ X ← MINNUMBER : > 202 > 255 > 255 > 255 > 255 ∇

∇ X ← MAXNUMBER : > 202 > 127 > 255 > 255 > 255 ∇

∇ X ← αA (Γ) αB , 7 : αASNO αA ← αB αA - αA αB < x αB + u : ← αA αASNO αB ∇

∇ X ← αA (L) αB , 8 : αASNO αA ← αB αA - αB αA < x αB + u : ← αA αASNO αB ∇

∇ X ← A IF B : INTCH ρ ∇

∇ X ← A UL B : INTCH ~: ρ ∇

∇ X ← A UNLESS B : UL ∇

∇ A ← ANDREDUCE B : 1 INTCH ρ ,: ρ: ∇

∇ A ← CRLF: 106 109 COMAαINST T: ∇

∇ □DE_PROC αA : ← αA 2 αATN u: αASNO αA ∇

∇ αB ← □CR_BOX αA: ← αA 3 αATN u: αASNO αA ∇

∇ □DE_BOX αA : ← αA 4 αATN u: αASNO αA ∇

∇ □CR_FILE αA: ← αA 5 αATN u: αASNO αA ∇

∇ □DE_FILE αA: ← αA 6 αATN u: αASNO αA ∇

∇ αB □SEND αA: αASNO αB ← αA αB 10 αATN u: ← αA αASNO αB ∇

∇ αB □DELIVER αA: αASNO αB ← αA αB 11 αATN u: ← αA αASNO αB ∇

∇ □DETACH αB : ← αB 13 αATN u: αASNO αB ∇

∇ □RW_FILE αA: ← αA 14 αATN u: αASNO αA ∇

```

V A ← REDUCE[SWITCH;DIM] SRC;RHOS;RANK;INCR;I;I2;TARGET;LEN;N
→ VECTORSICAL IF (RANK+' ' ρ ρ RHOS ρ SRC) < 2
→ DEFINEDIM←DIM ρ SKIP NEXT LINE IFF DIM ALREADY SPECIFIED.
DEFINEDIM: DIM←RANK ρ NEXT LINE YIELDS ERROR IF ρ,DIM ISN'T 1.
LEN←RHOS[I←DIM+(1 DIM<DIM) ρ DIM]
INCR←1
→ LASTDIM IF I=RANK
GENERATEINCR: INCR←INCR × RHOS[I]←RHOS[I2+I+1]
→ GENERATEINCR IF (I+I2) < RANK
LASTDIM: I+' ' ρ ρ SRC←,SRC
TARGET←1+ ' ' ρ ρ A←,(RHOS←(RANK-1) ρ RHOS) ρ 0
→ DONE IF INCR=LEN // → DZERO IF 0=LEN+INCR × LEN
ρ MAIN LOOP (NONSTANDARD)
SKIP: N←0
VECTOR: ↑SRC[I2+I-N]
RANK←I2+INCR-LEN
ELEMENT: ↑SRC[I2+I2-INCR] DYSCENS[SWITCH] ↑ ρ USE STACK, SAVE MEMREFS
→ ELEMENT IF RANK<I2
A[TARGET←TARGET-1]←↑
→ VECTOR IF (N←N+1)<INCR
→ SKIP IF 0<I+I-LEN
ρ END OF MAIN LOOP (NONSTANDARD)
RETURN // A←RHOS ρ A // → 0 IF 1= ρ RHOS ρ SHORTCUT OP/MATRIX
DONE: RETURN // A←RHOS ρ SRC
VECTORSICAL: A←SRC
→ 0 UL RANK ρ EXIT NOW IF OP/SCALAR.
→ DZERO IF 0=I←(RHOS←' ') ρ RHOS ρ PORNOGRAPHIC
→ DONE IF 1=I
↑SRC[I]
→V2+SWITCH=3 ρ CONDITIONALLY GOTO SPEEDY +/VECTOR SECTION
V2: RETURN // A←↑ // → V2 IF 1<I // ↑SRC[I←I-1] DYSCENS[SWITCH] ↑
VPL: RETURN // A←↑ // →VPL IF 1<I // ↑SRC[I←I-1] + ↑
DZERO: A←RHOS IDELEM SWITCH V

```

```

V A ← ECUDER[SWITCH;DIM] SRC;RHOS;RANK;INCR;I;I2;LEN
→ VECTORSCAL IF (RANK≠'' ∅ ∅ RHOS≠ ∅ SRC)<2
→ DEFINEDIM←DIM
DEFINEDIM: DIM←RANK
LEN←RHOS[I+DIM+(1DIM<DIM) ∅ DIM]
INCR←1
→ GENERINCR←I=RANK
GENERINCR: → GENERINCR←I=RANK // INCR←INCR × RHOS[I+I+1]
→ DONE IF (0=I≠'' ∅ ∅ SRC≠,SRC) ∨ INCR=LEN+INCR × LEN
SKIP: A←0
VECTOR: RANK←(I2+I-A)+INCR-LEN
ELEMENT: SRC[I2]+SRC[I2+I2-INCR] DYSCFNS[SWITCH] SRC[I2]
→ ELEMENT IF RANK<I2
→ VECTOR IF (A+A+1)<INCR
→ SKIP IF 0<I+I-LEN
DONE: RETURN // A←RHOS ∅ SRC
VECTORSCAL: → DONE UL RANK
→ 0 IF (I≠'' ∅ ∅ A+SRC)<2
V2: → V2+1=I // A[I]←A[I+I-1] DYSCFNS[SWITCH] A[I] ∇

```

```

V A ← B OUTER[SWITCH] C; RHOA; RB; RC; I; AI; CI; L
RHOA ← ( p B), p C
→ END IF (A=0) = (RB ← ' ' p p B ←, B) × RC ← ' ' p p C ←, C
n SHOULD THE VECTOR CONCATENATION APPROACH BE USED?
→ ELEM IF 15 < RB n MAXIMUM OF 15 CONCATENATIONS MAY BE PERFORMED
n YES. IT IS MOST EFFICIENT WHEN (p, B) <<< p, C
VFNS: → VFNS ← SWITCH // A ← I ← 0
VEQ: → END // → VEQ UL I = RB // A ← A, B[I+I+1] = C
VNE: → END // → VNE UL I = RB // A ← A, ~ B[I+I+1] = C
VPL: → END // → VPL UL I = RB // A ← A, B[I+I+1] + C
VMS: → END // → VMS UL I = RB // A ← A, B[I+I+1] - C
VTI: → END // → VTI UL I = RB // A ← A, B[I+I+1] × C
VDI: → END // → VDI UL I = RB // A ← A, B[I+I+1] ÷ C
VMA: → END // → VMA UL I = RB // A ← A, L + (L < C) × C - L + B[I+I+1]
VMI: → END // → VMI UL I = RB // A ← A, L + (C < L) × C - L + B[I+I+1]
VMO: → END // → VMO UL I = RB // A ← A, C - L + [C ÷ L + 0 = L + B[I+I+1]
VEX: → END // → VEX UL I = RB // A ← A, B[I+I+1] * C
VLO: → END // → VLO UL I = RB // A ← A, B[I+I+1] LN C
VCI: → END // → VCI UL I = RB // A ← A, B[I+I+1] CIRCLESC C
VGA: → END // → VGA UL I = RB // A ← A, B[I+I+1] EXCLAMATION C
VLT: → END // → VLT UL I = RB // A ← A, B[I+I+1] < C
VLE: → END // → VLE UL I = RB // A ← A, ~ C < B[I+I+1]
VGE: → END // → VGE UL I = RB // A ← A, ~ B[I+I+1] < C
VGT: → END // → VGT UL I = RB // A ← A, C < B[I+I+1]
VOR: → END // → VOR UL I = RB // A ← A, C ∨ B[I+I+1]
VAN: → END // → VAN UL I = RB // A ← A, C ∧ B[I+I+1]
VNO: → END // → VNO UL I = RB // A ← A, ~ C ∨ B[I+I+1]
VNA: → END // → VNA UL I = RB // A ← A, ~ C ∧ B[I+I+1]
ELEM: AI ← 1 + ' ' p p A ←, RHOA p 0
NEWROW: I ← RC // L ← B[RB]
→ NEWROW ← 10 < I ← I - 1 // A[AI ← AI - 1] ← L DYSCFNS[SWITCH] C[I]
→ NEWROW IF 0 < RB ← RB - 1
END: RETURN // A ← RHOA p A V

```

```

V A ← B INNER[SW1;SW2] C;B1;B2;C1;C2;RHOA;RHOB;RHOC;I;J;K
RHOA ← B1+C1-(0<B1← ρ RHOB ← ρ B)+0<C1← ρ RHOC← ρ C
B1←B2+C1+C2+1
→ 1+DIMB-(K+I+0)<J+1+'' ρ ρ RHOB
DIMB: → DIMB IF I<J // B1+B1 × RHOA[K+K+1]←RHOB[I+I+1]
→ DIMB+2+J=1
B2←RHOB[J+1]
→ 1+DIMC-(I+1)<J+'' ρ ρ RHOC
DIMC: → DIMC IF I<J // C2+C2 × RHOA[K+K+1]←RHOC[I+I+1]
→ DIMC+2+J=0
C1←'' ρ RHOC
→ PROPOGATE UL (1=B2) < B2=C1
→ END IF (B1=0) ∨ C2=0 // A←(B1,C2) ρ 0
→ FORMED IF (2=ρ RHOB) ∧ 2= ρ RHOC
B←(B1,B2) ρ B // C←(C1,C2) ρ C
FORMED: → FILLIN IF 0=B2
THIRD: J←C2
SECOND: K←B[B1;I] DYSCFNS[SW2] C[I+B2;J]
FIRST: K←(B[B1;I] DYSCFNS[SW2] C[I+I-1;J]) DYSCFNS[SW1] K
→ FIRST IF 1<I
A[B1;J]←K
→ SECOND IF 0<J+J-1
→ THIRD IF 0<B1+B1-1
END: → 0 IF 2= ρ RHOA
RETURN // A←RHOA ρ A
FILLIN: RETURN // A←RHOA IDELEM SW1
PROPOGATE: → LENERR UL (C1=1) ∨ I+B2=1
RETURN//A←RHOA ρ REDUCE[SW1;2+I]((B1,B2) ρ B) OUTER[SW2](C1,C2) ρ C
LENERR: 1 2 = 1 2 3 nLENGTH ERROR
V

```

V A ← B DYSCFNS[SWITCH] C

→1+SWITCH

RETURN // A ← B = C

RETURN // A ← ~ B = C

RETURN // A ← B + C

RETURN // A ← B - C

RETURN // A ← B × C

RETURN // A ← B ÷ C

RETURN // A ← (B × A) + C × ~ A + C < B

RETURN // A ← (C × A) + B × ~ A + C < B

RETURN // A ← C - B × [C ÷ B + B = 0 nAPLSV DYADIC |

RETURN // A ← B * C

RETURN // A ← B LN C

RETURN // A ← B CIRCLES C

RETURN // A ← B COMSC C

RETURN // A ← B < C

RETURN // A ← ~ C < B

RETURN // A ← ~ B < C

RETURN // A ← C < B

RETURN // A ← B v C

RETURN // A ← B ^ C

RETURN // A ← ~ B v C

RETURN // A ← ~ B ^ C

n ORDER: = ≠ + - × ÷ [L |

n * LN O GAMMA < ≤ ≥ > v

n ^ NOR NAND

v

V A ← RHOA IDELEM I

I ← 1 v 1+2=A+(1 0 0 0 1 1,MINNUMBER,MAXNUMBER,0 1 2 2 1 0 1 1 0 0 1 2 2)[I]

n THIS FUNCTION COMPUTES THE IDENTITY ELEMENT FOR THE ITH PRIMITIVE

n FUNCTION. P-FNS WITH NO SUCH ELEMENT, SUCH AS NAND, INVOKE A

n DOMAIN ERROR.

A ← RHOA p A V

$\forall A \leftarrow B(\epsilon) C$

$\rightarrow OR_IT+0=A+'' \circ \circ C+, C \quad // \quad \downarrow(\circ B) \circ 0$

$OR_IT: \rightarrow OR_IT+0=A+A-1 \quad // \quad \downarrow(B=C[A]) \quad \forall \quad \uparrow$

$A \leftarrow \uparrow$

$\forall$

Y A ← B (1) C;D;E;I;J

→ 6 IF (ρ A+,C)=J+0 // E+' ' ρ ρ B←B,0 // D← ρ C

C ← B = B[E]←A[J+J+ ~ I+0] nC[E] IS FORCED TO BE ONE

→ 3+C[I+I+1] nC[I] IS ALWAYS ONE v ZERO

A[J]←I nC[I] WAS ONE

→ 2 IF J < ρ A

A←D ρ A

nERROR UNLESS 1 GE ρ ρ B

n360 ENHANCEMENT: B MAY BE A SCALAR, IF SO, TREATED LIKE A VECTOR

v

```

V A ← ROTATE[ODIM] SRC; I; I2; INCR; LEN; N; RANK; RHOS; START
→ VECTORSCAL IF (RANK ← ' ' ρ ρ RHOS ← ρ SRC) < 2
→ DEFINEDIM ← DIM
DEFINEDIM: DIM ← RANK
LEN ← RHOS[I ← DIM ← (1 DIM < DIM) ρ DIM]
INCR ← 1
→ GENINCR ← I ← RANK
GENINCR: → GENINCR ← I ← RANK // INCR ← INCR × RHOS[I ← I + 1]
→ DONE IF 0 = I ← ' ' ρ ρ A ← SRC ← , SRC // LEN ← LEN × INCR
SKIP: N ← 0
VECTOR: START ← -INCR - RANK ← (I2 ← I + INCR - N) - LEN
ELEM: A[I2 ← I2 - INCR] ← SRC[START ← START + INCR]
→ ELEM IF RANK < I2
→ VECTOR IF (N ← N + 1) < INCR
→ SKIP IF 0 < I ← I - LEN
DONE: RETURN // A ← RHOS ρ A
VECTORSCAL: → 0 UL RANK // A ← SRC
→ 0 IF (I ← 1 ← ' ' ρ ρ A) < 3 // START ← 0
V2: → V2 + 1 = I // A[I ← I - 1] ← SRC[START ← START + 1]
V

```

```

V A + B ROTATE[ODIM] SRC;AI;BI;SI;BASE;I;INCR;LEN;N;RHOS
  → VECTORSCAL IF (BASE+' ' p p RHOS+ p SRC)<2
  → DEFINEDIM+eDIM
DEFINEDIM: DIM+BASE
  LEN+RHOS[AI+(1DIM<DIM) p DIM]
  → DONE+0<I+' ' p p A+SRC+,SRC
DONE: RETURN // A+RHOS p SRC+A n EXTRANEIOUS ASSIGNMENT SAVES WORKSPACE
  B+B-LEN×[B÷LEN n B+LEN | B
  → MULTI IF 2<BI+1+' ' p p B+,B // N+ p B
  → MULTI+1 // BI+1+' ' p p B+,( (~ AI=1BASE)/RHOS) p B
MULTI: N αCOMPAT (~AI=1BASE)/RHOS
  INCR+1
  → GENINCR+AI=BASE
GENINCR: → GENINCR+AI=BASE // INCR+INCR × RHOS[AI+AI+1]
  LEN+LEN × INCR
SKIP: → DONE IF (I+I-LEN)<N+0
  → ELEM2 IF (AI+0)=SI+B[BI+BI-1] × INCR // BASE+I-N
ELEM1: A[BASE+AI+AI+INCR]+SRC[BASE+SI+SI+INCR]
  → ELEM1 IF SI<LEN
  SI+0
ELEM2: A[BASE+AI+AI+INCR]+SRC[BASE+SI+SI+INCR]
  → ELEM2 IF AI<LEN
  → SKIP+(N+N+1)<INCR
VECTORSCAL: → 0 UL BASE // A+SRC
  B+(1B<B) p B n → LERR IF 1< p ,B
  → 0 IF 0=RHOS+' ' p RHOS
  → E2 IF (AI+0)=SI+B-RHOS×[B÷RHOS
E1: A[AI+AI+1]+SRC[SI+SI+1]
  → E1 IF SI<RHOS
  SI+0
E2: A[AI+AI+1]+SRC[SI+SI+1]
  → E2 IF AI<RHOS
V

```

∇ A ← B (†) C; AS1; AS2; CS1; CS2; I; J

→ RERR IF 1 < p p B

→ SCAC + 0 < I + ' ' p p J + p C

SCAC: I + ' ' p p J // C + (J + B = , B) p C

→ LERR UL I = p B + , B

→ NULL UL ANDREDUCE 0 < J

→ 0 UL ANDREDUCE 0 < p A + (| B) p ' ' 0 [1 + 1 ' ' p C]

→ 0 IF I = 0 nC IS SCALAR

CS2 + 0 [(B < 0) × B + J

B + J [| B // AS2 + 0 [-B + J

AS2 + AS2[I] // AS1 + AS2[1]

CS2 + CS2[I] // CS1 + CS2[1]

J + B[1]

→ VECT IF I = 1 nC IS VECTOR

→ RERR UL I = 2 nC ~ MATRIX

I + B[2]

MAT: A[AS1 + J; AS2 + I] + C[CS1 + J; CS2 + I]

→ MAT IF 0 < I + I - 1

→ MAT - 1 0 < J + J - 1

RETURN

VECT: → VECT - 0 = J + J - 1 // A[AS1 + J] + C[CS1 + J]

RERR: I = I p I + 1 2

LERR: 1 2 = 1 2 3

NULL: A + (| B) p 0

∇

```

-----
V A ← B (↑) C; I; J; II; JJ; STARTCOL; STARTROW
→ SCALAR IF 0 = p J ← p C
n NEXT STATEMENT INDUCES A LENGTH ERROR WHEN APPROPRIATE
→ RERR UL ((I ← ' ' p p J + J - | B) < 3) ^ (1 = p , B) v 1 = p p B
→ MAT IF I = 2
→ NULLVEC UL 0 < J
A ← J p C
→ 0 UL (I + 0) < B + ' ' p B
VEC: → VEC + J = I // A[I] ← C[B + I + I + 1]
RETURN
NULLMAT: RETURN // A ← (J × 0 < J) p ' '
NULLVEC: RETURN // A ← ' '
SCALAR: RETURN // A ← (B = 0) p C
RERR: I = I p I ← 12 n GENERATES A RANK ERROR
MAT: → NULLMAT UL (0 < JJ + J[2]) ^ 0 < II ← ' ' p J
A ← J p C
STARTCOL ← J[2] // STARTROW ← ' ' p J + 2 p B × (I + 0) < B
MATROW: I ← I + 1 // STARTROW ← STARTROW + 1 // J + 0
MATCOL: A[I; J] ← C[STARTROW; STARTCOL + J + J + 1]
→ MATCOL IF J < JJ
→ MATROW IF I < II
n DROP IS IMPLEMENTED ONLY FOR SCALAR, VECTOR, v MATRICE
n RIGHT ARGUMENTS.
n SCALAR DROP MATRICE EXECUTES LIKE (SCALAR, SCALAR) DROP MATRICE
n RANK ERROR OCCURS
n IF RIGHT ARGUMENT RANK > 2, OR
n IF LEFT ARGUMENT HAS AN INVALID RANK.
V

```

```

V A ← UPGRADE VECT; I; INDEX; REMEMBER; RHOV; V; VAL
→ 0 IF 0=RHOV+(, INDEX = 0 0 p I+0) p INDEX ← p A+VECT
SEARCH: INDEX+0
n REMEMBER ← SMALLEST INDEX OF THE SMALLEST NUMBER ENCOUNTERED.
→ REMEM - 'M' = VECT[INDEX+INDEX+1]
REMEM: VAL+VECT[REMEMBER+INDEX]
CHECK: → (INDEX = RHOV) p THRU
→ ('M'=V+VECT[INDEX+INDEX+1]) p CHECK
→ CHECK - V < VAL
THRU: VECT[A[I+I+1]+REMEMBER]+'M'
→ SEARCH IF I<RHOV
n E R R O R T R A P S
n RANK ERROR (5,0): 1 ≠ p p VECT
n TYPE ERROR (7,0): OR/VECT ELEMENT ANY CHAR EXCEPT 'M'
V

```

▽ A ← DOWNGRADE VECT
A ← UPGRADE -VECT ▽

V A ← B αCaOαMaAα C

n ***** MICROPROGRAM ASSUMES 20B ↔x[2]COMA *****

A+B,[] C n INVOKE THE CRMS-APL , INSTRUCTION

V

V A ← INDEX1[VAR;I1];RHOA;RHO1;J1

→ IERR UL 1=ρρVAR

→ DEFI1+εI1

DEFI1: I1←1(ρVAR)[1]

RHO1←'' ρ ρ A←I1←,I1 // RHOA← ρ I1 // J1←0

→6+RHO1=J1

→5 // A[J1] ← VAR[I1[J1+J1+1]]

→ 0 ^ 11=ρ RHOA

RETURN // A←RHOAρA

IERR: 1 2[1;1]

V

```

V A ← INDEX2[VAR;I1;I2];I1R1;R1;R2;RA;RHOA
→IERR UL 2=opVAR
→ DEFI1+eI1
DEFI1: I1←i(opVAR)[1]
→ DEFI2+eI2
DEFI2: I2←i(opVAR)[2]
RHOA←(oI1),oI2
A←(RA←(R1←'opI1←,I1) × 'opI2←,I2)opVAR
→END IF 1=RA←RA+1 n INITIALIZE PRE-DECREMENTING POINTER
I1R1←I1[R1] // R2←'opI2
AI1: →AI1+0=R2←R2-1 // A[RA←RA-1]+VAR[I1R1;I2[R2]]
→AI1-10<R1←R1-1
END: RETURN // A←RHOApA
IERR: 1 2[1;1]
V

```

```

V A ← INDEX3[VAR;I1;I2;I3];I1R1;I2R2;R1;R2;RA;RHOA
→IERR UL 3=ppVAR
→DEFI1+eI1
DEFI1: I1+1(ppVAR)[1]
→DEFI2+eI2
DEFI2: I2+1(ppVAR)[2]
→DEFI3+eI3
DEFI3: I3+1(ppVAR)[3]
RHOA+(pI1),(pI2),pI3
A←(RA←(R1+'ppI1',I1) × ('ppI2',I2) × 'ppI3',I3)ppVAR
→END IF 1=RA+RA+1
      I1R1←I1[R1] // R2+'ppI2
AI1: I2R2←I2[R2] // R3+'ppI3
AI2: →AI2+0=R3+R3-1 // A[RA+RA-1]←VAR[I1R1;I2R2;I3[R3]]
→AI2-10<R2+R2-1
→AI1-10<R1+R1-1
END: RETURN // A←RHOApA
IERR: 1 2[1;1]
V

```

V A ← TRANSPOSE B; RHOA; RANK; IX; X; Y

→ TWOD IF 2=RANK+1' p p RHOA ← p A+B

→ 0 IF RANK<2

RETURN // A←((1+RANK) - 1RANK) TRANSPOSE B n APLSV TREATMENT OF MONADIC TRANSPOSE

n

n ~ WASTEFUL SINCE MOST MONADIC TRANSPOSES INVOLVE

n TWO-D MATRICES.

TWOD: A←((Y+RHOA[2]),IX+1' p RHOA) p B

→ 0 IF 0=IX × Y n IS MATRIX 0 BY N OR N BY 0 ??

X←IX n NO SUCH LUCK.

MOVE2: A[Y;X] ← B[X;Y] n MOST REGRETABLY, SUBSCR × TAKES TIME.

→ MOVE2 IF 0<X+X-1

→ MOVE2 -10<Y+Y-1

n NO ERRORS ARE EXPLICITLY GENERATED.

V

```

V Z ← A TRANSPOSE B;RB;RZ;I;J;W1;W2;CZ
→RERR IF 1<poA
→LERR IF (J←''poRB+oZ+B) ≠ oA←,A
→0 IF J<2
W1←JpCZ←''p(2-ANDREDUCE(A=[A]^A>0) ^ 2-ANDREDUCE(1I+[/A)∈A nCHECK DOMAIN
INITAW1: W1[J←J-1]←W1[J]×RB[J]
→INITAW1 IF 1<J
RZ←W2←Ip0
LOOP: CZ←CZ×RZ[J]←L/(Z+J=A)/RB
W2[J]←+/Z/W1
→LOOP UL I<J+J+1
B←,B // Z←CZp0
→SHAPE IF CZ=0
VECT: A←CZ-J←1
I←''ppRZ
ELEM: J←J+W2[I]×W1-RZ[I]×A←L(W1+A)÷RZ[I]
→ELEM IF 0<I←I-1
Z[CZ]←B[J]
→VECT IF 0<CZ←CZ-1
SHAPE: →0 // Z←RZpZ
RERR: I=IpI+12
LERR: 1 2 = 1 2 3
V

```

→ CATENLOOP IF I<RANK
FINI: A+RA o A V

```

V A ← B (/[ODIM] C;AI;BI;CI;CMAX;N;NI;RHOB;RHOC
→ DEFINEDIM-10< ρ RHOC+ ρ C nBRANCH IF C IS AN ARRAY
→ 0^11=ρN // A+(N+,Bv0)ρC n256<ρ,B FOR B/SCALAR ASSUMED
RHOC+ ρ C+,C nVECTOR/SCALAR IS TREATED LIKE VECTOR/1 ρ SCALAR
→ DEFINEDIM+eDIM
DEFINEDIM: DIM+' ' ρ ρ RHOC nASSIGN DEFAULT DIMENSION INDEX.
→ QUICK1 IF 1= ρ , B
→ RERR UL 1= ρ RHOB+ ρ B v 0 nB MUST BE A BINARY VECTOR
→ NOT1TO1 UL RHOC[DIM]=RHOB+' ' ρ RHOB
→ NULL IF (0=CMAX+' ' ρ ρ C+,C) v RHOB=AI+CI+0
→ LASTDIM IF DIM= ρ RHOC // B+B,2 // A+CMAX ρ C
→ NDONE IF BI= ρ RHOC // N+RHOC[BI+DIM+1]
→ NDONE-1BI< ρ RHOC // N+N x RHOC[BI+BI+1]
NDONE: BI+0 // CI+CI-N
FINDNL: → FINDNL+B[BI+BI+1] // CI+CI+N // NI+0
→ FINDNL+NI<N // A[AI+AI+1]+C[CI+NI+NI+1] nASSIGN A[AI+1N]
→ LGO // → FINDNL-1CI<CMAX
LASTDIM: BI+0
FINDL: → FINDL+B[BI+BI+1]
→ FINDL // A[AI+AI+1]+C[CI+BI]
→ FINDL-1(CI+CI+RHOB)<CMAX
LGO: RHOC[DIM]+RHOC[DIM]×A]÷CI nCOMPUTE RHOC[DIM]++/B
RETURN // A+RHOC ρ A
QUICK1: → QUICK1F UL B nTRAP IF B ~ BINARY
RETURN // A+C
QUICK1F: RETURN // A+RHOC ρ RHOC[DIM]+0
NOT1TO1: → LERR UL (1= ρ ,C) ^ 1= ρ ρ C nLENGTH ONE VECTOR
RETURN // A+(+/B) ρ C
NULL: RETURN // A+RHOC ρ RHOC[DIM]++/B n'/ARRAY v B/NULLARRAY
LERR: 1 2 = 1 2 3 nLENGTH ERROR
RERR: N=N ρ N+12 nRANK ERROR
V

```

```

V A ← B (\)[ODIM] C;RA;RC;I;IA;IC;N;NN
→ SCLR+0<I+' ' p p RA← p C
SCLR: I+1 // RA← p C+,C
→ DEFINEDIM+eDIM
DEFINEDIM: DIM←I
→ RERR IF 1< p p B v IA+IC+0 nB BETTER BE BIN SCA v VECT
→ REDUCEEXPAND+0=N+' ' p p B+,B
REDUCEEXPAND: → REDUCEEXPAND+IC=N // IA+IA+B[IC+IC+1]
→ NOMATCH UL IA=RA[DIM]
REMATCH: RA[DIM]+N
→ NULL IF 0=RC+' ' p p C+,C
A←,RA p ' ' 0 [1+1 ' ' p C] // B+B,2 // AI+CI+0
→ LAST-1I=DIM
→ INCR+I=NN // N+RA[NN+1+DIM]
INCR: → INCR+I=NN // N+N x RA[NN+NN+1]
I+0 // AI+AI-N
NLAST: → NLAST+B[I+I+1] // AI+AI+N // NN+0
→ NLAST+NN<N // A[AI+NN+NN+1]+C[CI+CI+1]
→ NLAST-1CI<RC
RETURN // A+RA p A
I+0
LAST: → LAST+B[I+I+1]
→ LAST // A[AI+I]+C[CI+CI+1]
→ LAST-1CI<RC // AI+AI+I-1
RETURN // A+RA p A
NULL: RETURN // A+RA p 0
FILL: RETURN // A+N p ' ' 0 [1+1 ' ' p C]
NOMATCH: → LERR UL (I=1) v 1=RA[1]
→ FILL IF IA=0
→ REMATCH // C+IA p C nDIM=1
LERR: 1 2 = 1 2 3
RERR: I=I p I+12 v

```

V Z←OUTONE[0QQ] A;B;I;NOBLANK
→IPIO IF (ε□IO_BOXES) // Z←A n SAVE RETURN VALUE
A←,A // NOBLANK←1+I←0
DECIDE: →LAST IF (pA)←I←I+1
→ (CH,NUMERIC)[1+1 B←A[I]]
CH: OUTCHARSC B // NOBLANK←1
→ DECIDE
NUMERIC: → PBL IF NOBLANK
OUTCHARSC ' '
PBL: OUTNUMSC B
→ DECIDE // NOBLANK←0
IPIO: □IO_BOXES[2] □DELIVER A
LAST: →0Λ1QQ // →0Λ1~εQQ n EXIT NOW IF QQQUAD OUTPUT
←←CRLF
V

```

-----
V OUTNUMSC A;B;C;TEXP
// OUTONE (A<0) p '-'
→ INT IF (B<1000000) ^ (99999<B) v B=[ B+ | A // C+' '
→ FLLT1 IF B<1 // TEXP+0
→ FLLT100000 IF B<100000
DOWN: → DOWN+(B÷10)<10 // TEXP+TEXP+1
OUTNUMSC B
OUTCHARSC 'E'
OUTNUMSC TEXP
RETURN
FLLT100000: OUTNUMSC [ B
OUTCHARSC ' '
FRACTPART B
RETURN
FLLT1: → FLLT1 IF (B÷B × 10)<1 // TEXP+TEXP-1
→ DOWN+1
INT: → INT+0=B÷A//C+'0123456789'[1+[B-10×A÷[B÷10],C
// OUTONE C
RETURN V

```

V FRACTPART A;B;SCALE

SCALE+1 // B+6 n MAX SIGNIFICANT DIGITS

GETSC: → GETSC+A<SCALE+SCALE × 10 // B+B-1

DIGIT: OUTCHARSC '0123456789'[1+L A+10 × A-L A]

→ DIGIT IF (A>L A) ^ 0<B+B-1

V

▽ A+(-)

// OUTONE '*'

□IO_BOXES[1] □SETCONV 1

A←□RECEIVE αA←□IO_BOXES[1]

A←.A 0 NO FUNNY STUFF ALLOWED.

▽

```

V A←INONE B;C:D
// OUTONE B // A+' '
→IPIO IF εIO_BOXES
READ: → BSP IF '+'=C←INCHARSC
→ NEWL IF '\'=C
→ DONE IF 109=TC nCARRIER RETURN ENCOUNTERED.
→ READ // A+A,C
BSP: → READ // A+((0<D)×D←1+ρA) ρA
NEWL: → 1 // OUTONE '+<',CRLF
DONE: // OUTONE CRLF
RETURN
IPIO: IO_BOXES[1] SETCONV 0
A←RECEIVE αA←IO_BOXES[1]
V

```

∇ A←(←)
A←INONE ' ' ∇

```

V A←INNUM;B;D;I;SIGN
→IPIO IF εIO_BOXES
→BADIS IF (A≠0) = p B←INONE ' ': ,T106
I←SIGN+'-' = ' 'pB
MORE: →NOMORE IF ( o B)<I+I+1
→BADIS IF (D<0) v 9<D+(TB[I])-T'0'
→MORE // A←D+A × 10
BADIS: 'MALFORMED INTEGRAL SCALAR INPUT.'
→ 1
NOMORE: A←A × 1-SIGN+SIGN
RETURN
NOGOOD: 'TRY AGAIN'
IPIO: // OUTONE '--> ',T106
IO_BOXES[1] SETCONV 1
A←RECEIVE αA←IO_BOXES[1]
→NOGOOD IF 0=pA+(1A)/A
V

```

∇ B αCOMPAT C;A;I

→ SQUEEZE1B-(ρ B)= ρ C ∩ B ∧ C ARE VECTORS

→ 0 IF ANDREDUCE B=C

SQUEEZE1B: → SQUEEZE1C IF 0=I+'' ρ ρ B // A+''

→ SQUEEZE1B+10<I+I-1 // A+A,B[I]+11 ≠ B[I]

SQUEEZE1C: → TESTEQ IF 0=I+'' ρ ρ C // B+''

→ SQUEEZE1C+10<I+I-1 // B+B,C[I]+11 ≠ C[I]

TESTEQ: → LERR UL (ρ A)= ρ B

→ 0 IF ANDREDUCE A=B

LERR: 1 2 = 1 2 3 ∇

V A+ASINDEX1[0B;I1] C;RC

→ IERR UL 1= 0 0 B

→ DEFI1+eI1

DEFI1: I1← 1 0 B

→ SINGLE IF 1=RC+' ' 0 0 C←,A+C

I1←,I1 // (0A) αCOMPAT 0I1

→ 0 IF 0=RC

AI1: → AI1+0=RC+RC-1 // B[I1[RC]]+C[RC]

RETURN

SINGLE: → AI1-1 // C←(RC+' ' 0 0 I1←,I1) 0 C

IERR: 1 2[1;1] V

```

V A+ASINDEX2[OB;I1;I2] C;I1R1;RC;R1;R2
  → IERR UL 2=OB n CHECK RANK
  → DEFI1+eI1
DEFI1: I1+1(OB)[1]
  → DEFI2+eI2
DEFI2: I2+1(OB)[2]
  → SINGLE IF 2=RC+1+' ' p p C+,A+C
  I1+,I1 // I2+,I2 // (pA) αCOMPAT (pI1),pI2
  → 0 IF 1=RC // R1+' ' p p I1
  R2+' ' p p I2 // I1R1+I1[R1]
AI1: → AI1+0=R2+R2-1 // B[I1R1;I2[R2]]+C[RC+RC-1]
  → AI1-10<R1+R1-1
  RETURN
SINGLE: RC+1+' ' p( p I1+,I1) × p I2+,I2
  → AI1-2 // C+(RC-1). p C
IERR: 1 2[1;1] V

```

```

V A←ASINDEX3(pB;I1;I2;I3) C;I1R1;I2R2;RC;R1;R2;R3
→IERR UL 3=opB  n CHECK RANK
→DEFI1←eI1
DEFI1: I1←1(pB)[1]
→DEFI2←eI2
DEFI2: I2←1(pB)[2]
→DEFI3←eI3
DEFI3: I3←1(pB)[3]
→SINGLE IF 2=RC+1+'pC+,A+C
(oA) αCOMPAT (pI1),(pI2),pI3
I1←,I1 // I2←,I2 // I3←,I3
LASTCHK: R1←'pI1 // →0Λ1=RC
R2←'pI2 // I1R1+I1[R1]
AI1: R3←'pI3 // I2R2+I2[R2]
AI2: →AI2+0=R3+R3-1 // B[I1R1;I2R2;I3[R3]] ← C[RC+RC-1]
→AI2-10<R2+R2-1
→AI1-10<R1+R1-1
RETURN
SINGLE: RC+1+'p (pI1←,I1) × (pI2←,I2) × pI3←,I3
→LASTCHK // C←(RC-1)pC
IERR: 1 2[1;1]
V

```

V Z←SQRT X;D;E;Y

E←X÷0.5 COEFFBEXP L 129+D+D←[0.5×(BEXP X+X=0)-128

Y←2.185183-3.028991÷1.5451577+E+0^1+E<0

Z←0.5×Z+E÷Z+0.5×Y+E÷Y

Z←Z × 0.5+0.5 × E←Z<1

Z←(0<X) × Z COEFFBEXP L 129+D-E V

V P+QSC N;Q

→ 2+εRAND

RAND←1[L|'p INNUM // OUT[CRLE, ' ENTER PSEUDO-RANDOM INTEGER BETWEEN 1 AND 1 MILLION']

RAND←(1000×RAND-1000×L(RAND←(389×Q)+17×P)÷1000.)+17×Q+RAND-1000×P+L(RAND÷1000, nRAND←1E6|RAND×73*3

P←1+LN×.999999[RAND÷1E6

V

V AN←(*) U;F;G;H;I;J;L

J←|U÷0.69314718

G←L J+0.5

H←J-G

I←H × 0.69314718

J←I × I

F←2+J×(2520+28×J)÷15120+(420×J)+J×J

L←(F+I)÷F-I

H←L<1

L←L × H+0.5 × ~ H

F←L COEFBEXP L 129+G-H

AN←((U<0)÷F)+F×U>0

V

V ABB←A (*) B,10

nA TO THE B = EXP OF B TIMES NATURAL LOG OF A

ABB←*B×LN AV

VAS←LN X;A;B;G0;A1;A2

A←1 ^ 1+X ≤ 0

A←A+A+COEF X

B←0.5 × 1+A

A1←0.5 × B+G0+SQRT A

A2←(0.5 × A1+SQRT A1 × G0)-0.25 × A1

G0←0.69314718 × (BEXP X)-129

AS←G0+0.703125×(A-1)÷A2-0.0625×A1-0.25×B

V

```

VC←A LN B,11
C←(LN B) ÷ LN A V

```

```

V A ← B (T) C;RHOA
→ DONE IF 0=(I←(A←'') ρ ρ B←,B) × ρ C←,C//RHOA+( ρ B), ρ C
LOOP: A←((-B[I]×C←[C÷B[I]]+C),A nPORNO
→ LOOP UL 0=I-I-1
DONE: A←RHOA ρ A
n THIS DECODE DIFFERS FROM APL/360 IN THE TREATMENT OF:
n   NEGATIVE LEFT ARGUMENTS
n   ^ NONCONFORMING DIMENSIONS
V

```

```

V A ← B (1) C;COUNT;I;LASTI;RHOB
→ SCALAR IF 0=A←'' ρ ρ RHOB← ρ B
→ MAKEONE IF 1=COUNT←RHOB[A]
→ MATMULT IF 1=I+1+'' ρ ρ B←,B
LASTI←I-COUNT // A←1
ITER: → ITER+LASTI=I // A←(B[I]←A) × B[I+I-1]
→ ITER -11<LASTI
MATMULT: RETURN // A←(RHOB ρ B)+.× C
SCALAR: A←1 // RHOB← ρ B ←,B
MAKEONE: A←(RHOB ρ B◦.*A-1A←RHOB[A]←'' ρ( ρ C),1)+.× C
V

```

VANNS+A CIRCLESC B;Z;Z2;K;L;M;N;P;Q;R
 n BRANCH TO THE APPROPRIATE PORTION OF CIRCLE
 →(ATH,ACH,ASH,M4,AT,AC,AS,ZER,SS,CS,TA,P4,SH,CSH,TH)[8+A]

n
 n ARCTANGENT HYBERBOLIC
 n

ATH:ANNS+0.5×LN (1+B)÷1-B
 RETURN

n
 n ARCCOSINE HYBERBOLIC
 n

ACH:ANNS+LN B+SQRT $^{-1+B \times B}$
 RETURN

n
 n ARCSINE HYBERBOLIC
 n

ASH:ANNS+LN B+SQRT 1+B × B
 RETURN

M4:ANNS+ SQRT (B+1) × B-1
 RETURN

ZER:ANNS+SQRT (1+B) × 1-B
 RETURN

P4:ANNS+SQRT 1+B × B
 RETURN

n
 n ARC TANGENT
 n

AT: K← | B

L←(0 ≤ K) ∧ (K ≤ 1)

M←1<K

N←(L×K)+M÷K+K=0 n ELSE ÷0 ERROR

K← M × 1.5707963

P←1-M+M

L← (0 ≤ N) ∧ N ≤ 0.414213562373095

M←(N ≤ 1) ∧ 0.414213562373095<N

Q←(L × 0.198912367)+M × 0.668178637

R←(L × 0.1963495408493621)+M × 0.5890486225480862

L←(N-Q)÷1+Q×N

Z←L × L

Z2← Z × Z

M←135135+(171962.4603936874 × Z)+52490.48316373623 × Z2

M←M+2218.098888446071 × Z2 × Z

N←135135+(217007.463936857 × Z)+97799.30329541391 × Z2

N←N+10721.37452059297 × Z2 × Z

ANNS←(×B)×K+P×R+L×M÷N

RETURN

n
 n ARCCOSINE
 n

AC:ANNS+1.5707963- $^{-1}$ CIRCLESC B

RETURN

n
 n ARC SINE
 n

```

AS:K← | B
L←(0 ≤ K) ∧ (K ≤ 0.5)
M←(0.5 < K) ∧ (K ≤ 0.8660254037844387)
N←(0.8660254037844387 < K) ∧ (K ≤ 1)
P←(M × 0.7853981633974485)+N × 1.5707963
Q←L+(0.5 × M)-N+N
R←(L×K)+(M×-1+2×K×K)+N×SQRT 0.5×1-K
Z←R × R
K←0.01731276426+Z×0.0223750091+Z×0.009342806551+Z×0.01433124507+Z×0.0183566709+Z×-10.0118622397+Z×0.03162712225
K←0.999999999+Z×0.166666666+Z×0.074999999+Z×0.0446428605+Z×0.0303818164+Z×K
ANNS←(× B) × P+Q × K × R
RETURN
SH:K← * B
ANNS←0.5 × K÷K
RETURN
SSH:K← * B
ANNS←0.5×K÷K
RETURN
TH:K← * B
L←÷K
ANNS←(K-L)÷K+L
RETURN
TA:K←LB÷PI
L←PI × K+1.5707963<B-K × PI
M← | B-L
N←LM÷0.3926990816987241
P←M-N × 0.3926990816987241
R←(N = 2)+(0.414213562373095 × N = 1)+(2.414213562373095 × N = 3)
Z←P × P
K←2027025+Z×-2270270+Z×6930+Z×-336
M←2027025+Z×-945945+Z×51975+Z×-630
Q←P×K÷M
ANNS←(×B-L)×(R+Q)÷1-R×Q
RETURN
CS: P←Q-LQ←(|B)×0.15915494
Q←((P-0.75)×(0.5≤P)∧P<1)+(0.25-P)×(0≤P)∧P<0.5
R←Q×Q
ANNS←Q×6.283185+R×-41.34168+R×81.60248+R×-76.58128+R×39.76072
RETURN
SS: P←Q-LQ←(|B)×0.15915494
Q←(P×(0≤P)∧P<0.25)+((P-1)×(0.75≤P)∧P<1)+(0.5-P)×(P≥0.25)∧P<0.75
R←Q×Q
ANNS←Q×6.283185+R×-41.34168+R×81.60248+R×-76.58128+R×39.76072
V

```

```

V A ← B (0) C,12;RA;I
→ ONEB IF ANDREDUCE 1=0B n VERRY VERRY LIKELY
RA← 0 A+B=C
→ DONE IF 0=I+1' 0 0 G←( 0 B←( 0 A←,A) 0 B) 0 G
LOOP: A[I]←B[I] CIRCLES C[I]
→ LOOP IF 0<I+I-1
DONE: RETURN // A←RA 0 A
ONEB: → 0 UL ANDREDUCE 1=0A+(''0B) CIRCLES C
A←(0B=C)0A n FORMAT ONE ELEMENT ANSWER
V

```

```

VG←FACSC X;K
→ L14 IF X<0
K←X-1 X
G←K×-5.100461+K×-1.850729
G←-195.6928+K×-66.46644+K×-36.83094+G
G←G÷-195.6928+K×16.2694+K×36.8849+K×K-11.43218
→ 0 IF (1 ≤ X) ^ X<2
→ L11 IF X<1
K←K+2
L8: G←G × K
→ L8 IF X ≥ K+K+1
RETURN
L11: G←G÷1+K
RETURN
K←1+X-1 X
L14: G←K×-5.100461-K×1.850729
G←-195.6928+K×-66.46644+K×-36.83094+G
G←G÷-195.6928+K×16.2694+K×36.8849+K×K-11.43218
G←G÷K+1
L16: G←G÷K
→ L16 IF X<K+K-1
V

```

```

V A ← EXCLAMATION B,13;I
→ LOOP+1=I+1+' '00A←,B
LOOP: → LOOP+1=I // A[I]←FACSC A[I+I-1]
A←(0B)0A V
-----

```

```

V COM ← B COMSC A
→0 UL COM←B<A
COM←(FACSC A) ÷ (FACSC B) × FACSC A-B V

```

```

VCUT←B EXCLAMATION A
CUT←( EXCLAMATION A) ÷ ( EXCLAMATION B) × EXCLAMATION A-B
V

```

```

V A ← (?) B:I
→LOOP+1=I+1+' 'pA←,B
LOOP: → LOOP+1=I // A[I]←QSC A[I+I-1]
A←(pB)pA V

```

```

V A←B (?) C:I;I2
→DERR UL ((B<I+0)=C<B) ^ B=[B
→SAVEΔSP IF 300<C
A←iC n ASSUME ENOUGH SPACE TO USE QUICKIE DYADIC ? ALGORITHM.
→DONE IF C<2
AGAIN: + A[I2+I+?C-I+I+1]
A[I2]←A[I] n EXCHANGE A[I] WITH A[I2]
A[I]←+
→AGAIN IF I<C-1
DONE: RETURN // A←BpA
DERR: A←2^2
SAVEΔSP: A←?C n PERFORM THE MORE EXPENSIVE DYADIC ? ALGORITHM.
→DONE IF B<2
NEW: →NEW+B<pA+A,I // →NEW IF (I+?C)∈A
→DONE
V

```

```

-----
V OUT[OA;OB;OC;OD;OE;OF;OG;OH;OI;OJ;OK;OL]
  +A // +2 ULεA
  +B // +3 ULεB
  +C // +4 ULεC
  +D // +5 ULεD
  +E // +6 ULεE
  +F // +7 ULεF
  +G // +8 ULεG
  +H // +9 ULεH
  +I // +10ULεI
  +J // +11ULεJ
  +K // +12ULεK
  +L // +13ULεL
V

```

```

-----
V αSTART DUMMY
  START 20 2 0 20 ↔ 24B ↔ LOGGING TERMINAL
  'ALL UP AND RUNNING (BELIEVE IT OR NOT)'
  +1
V

```

```

-----
V START TM_FN;A;B;C
  IO_BOXES ← BOXES_TERM TM_FN[1] // +2 IF εIO_BOXES
  SLAVES←0 6p0 // +3 IF εSLAVES
  CRLF, 'TYPE SUBJECT TERMINALS'
  +0Λ10εA // + 'YOU ENTERED: ',A+INNUM
AG: B←(BOXES_TERM A[1]), (CR_BOX 1), CR_BOX 1
  OUT[C+'FROM START' CR_PROC [130 6,B,CR_BOX 1;3] TM_FN[2] ]
  SLAVES←SLAVES,[1] C,A[1],B
  +AG IF 0<pA+1+A
V

```

```

-----
V KILL ID;A;B
  → 0 UL 6=PB←,(A←SLAVES[;1]=ID)/SLAVES // → 0 UL ←SLAVES
  SLAVES←(~A)/SLAVES
  DE_PROC B[1]
  DETACH B[3 4] n IO_BOXES
  DE_BOX B[3] // DE_BOX B[4]
  DE_BOX B[5] // DE_BOX B[6] n IPC_BOXES
V

```

```

-----
V ID αINIT STATE
  ID←ID
  → 0^1~←STATE n HEADER ACTS LIKE ID αINIT[STATE]
  WIDTH←STATE[1]
  DIGITS←STATE[2]
  IO_BOXES←STATE[3],STATE[4]
  IPC_BOXES←STATE[5],STATE[6]
  ALARM←STATE[7]
  CRLF
V

```

```

-----
V R ← P CR_PROC[STATE;ON] FNUM
  → INIT_N + ←N n SHOULD DEFAULT BE USED?
  INIT_N: N←4 n YES.
  ↓ P // ↓ N // ↓ STATE // ↓ FNUM
  R←↑ // ATTN 1 V

```

```

-----
V BOXES←BOXES_FILE FILE
  ↓BOXES←(CR_BOX 1), CR_BOX 1 // ↓FILE
  7 αATTN ↑ n ATTACHFILE ATTENTION TRAP
V

```

V BOXES+BOXES_TERM TERM
-+BOXES+(BOX_BOX 1), BOX_BOX 1 // +TERM
ATTN 8 n ATTACHTERMINAL TRAP
V

V M+RECEIVE pB
ATTN 9 // + B
B++ // M++ V

V B SETCONV L
ATTN 12 // + B // + L V

V STAMP+TS
STAMP++
V

V S_AL[OALARM]
→DEFΔALARM+εALARM
DEFΔALARM: ALARM←□ALARM
V

V T←□R_AL[OALARM]
→DEFΔALARM+εALARM
DEFΔALARM: ALARM←□ALARM
V

□