APL
PRESS
Box 378, Pleasantville
10570 (914)769-3323

BULK RATE
U.S. Postage Paid
PERMIT NO. 28
Pleasantville
NY 10570

6

1978

APL NEWS

* STAPL is planning to sponsor a three-day APL Conference in Rochester, New York, beginning May 30, 1979. Further details will appear in STAPL Quote-Quad.

* I.P. Sharp Associates are sponsoring a three-day APL Conference in Toronto on September 18-20 of this year. Sessions will include:

  Direction of APL Developments
  Direction of Communication Systems Developments
  Applications in the Research Industries, in Banking and Finance, and in the Actuarial and Insurance Industry
  Applications for Graphics
  APL Standards for Documentation and Programming
  Financial Planning
  Logistic Systems in Multi-national Manufacturing Industries
  Quantitative Analysis and Statistical Techniques

  There will also be a session on APL in Exposition organized by Professor Robert Spence of Imperial College, as well as a two-part Advanced APL course held in the evenings and consisting of three hours each on the topics APL System Design, and Data Base Design in APL. Further information on the conference is available from Ms. R. Wild, I.P. Sharp Associates, 145 King Street West, Toronto, Canada.

* Our most recent publication (STARMAP, by P. C. Berry and J. R. Thorstensen) contains functions which produce coordinates for a plot of the heavens at specified times and places. We have learned from one of the authors that these functions, together with appropriate plotting functions, are now available on the I.P. Sharp APL service. We would be pleased to announce their availability on other APL systems.

MORE ON GRADE

John C. McPherson

The grade function produces a permutation vector, a vector composed of one use of each of the elements ⍳*N*. Each permutation vector is a representation of a permutation matrix, a square binary matrix with just one mark in each row and column. The function:

```
          V←5 3 1 6 2 4
          V
    5 3 1 6 2 4
          ⎕←A←(⍳⍴V)∘.=V
     0 0 1 0 0 0
     0 0 0 0 1 0
     0 1 0 0 0 0
     0 0 0 0 0 1
     1 0 0 0 0 0
     0 0 0 1 0 0
```

transforms a permutation vector into a matrix.

We can test that a vector is in fact a permutation vector by examining the truth value of the expression ∧/(⍳⍴*V*)=*V*[⍋*V*].

A permutation matrix can be reduced to a permutation vector by the expression:

```
    ∇ Z←MTPV A                              MTPV A
[1]   Z←+⌿+⍀⊖A ∇                        5 3 1 6 2 4
```

A permutation vector can be displayed as the same pattern rotated 0, 90, 180, 270 degrees by successive applications of ⍋ looked at from the front and back by reversing the permutation vector and rotating again.

```
    ∇ Z←M A
[1]   Z←(⍳⍴V)∘.=A ∇

    (6 16↑⍕M V),(6 16↑⍕M ⍋ V),(6 16↑⍕M ⍋⍋ V),6 16↑⍕M ⍋⍋⍋ V
 0 0 1 0 0 0      0 1 0 0 0 0      0 0 1 0 0 0      0 0 0 1 0 0
 0 0 0 0 1 0      0 0 0 1 0 0      0 0 0 0 0 1      0 1 0 0 0 0
 0 1 0 0 0 0      0 0 0 0 0 1      1 0 0 0 0 0      0 0 0 0 0 1
 0 0 0 0 0 1      1 0 0 0 0 0      0 0 0 0 1 0      1 0 0 0 0 0
 1 0 0 0 0 0      0 0 0 0 1 0      0 1 0 0 0 0      0 0 1 0 0 0
 0 0 0 1 0 0      0 0 1 0 0 0      0 0 0 1 0 0      0 0 0 0 1 0
```

Symmetry may reduce the number of different displays to 4 or 2. The eight possible vectors generating the same pattern can be generated by *GEN*:

```
    ∇ Z←GEN P                                  GEN V
[1]   Z←P,⍋P                               5 3 1 6 2 4     4 2 6 1 3 5
[2]   Z←(4,⍴P)⍴Z,9-Z                       3 5 2 6 1 4     4 1 6 2 5 3
[3]   Z←(⍕Z),(4,-4+2×⍴P)↑⍕⌽Z     ∇         4 6 8 3 7 5     5 7 3 8 6 [illegible]
    ∇                                      6 4 7 3 8 5     5 8 3 7 4 6
```

The use of grade in sorting is only a small part of the usefulness of permutation vectors. Any recoding of a set of characters and symbols from one representation to another is the application of a permutation vector to the original representation. Note that this means any and all characters can be given any new representation desired.

Recoding is effected by

```
⎕AV[256|ASCII⍳V]
```

Why this is so can be readily visualized by noting that a permutation matrix, viewed as a nomogram or transformation matrix permits any input on one axis to designate any output on the other axis.

```
  1 2 3 4 5 6
1-+-+-o | | |
2-+-++--+-o |
3-+-o   |   |
4-+-----+---o
5-o     |
6-------o
```

Similarly, permutation vectors can be used to rearrange a set of objects into any new arrangement, and have been used in sorting bank checks in many fewer passes into sequence by branch bank, by customer account number, by check number, using ⍋⍋V or its equivalent V[⍋V]←⍳⍴V to renumber the checks from 1 to ⍴V. Those numbers are then converted to a number base matching the sorting machine for maximum efficiency in sorting the checks.

Note that the indexed form of "double up-grade" may execute much faster than the second grade function on long vectors.

Another use for ⍋ is a fast way of detecting duplicates in a set of results. It is known that some of the sums of two integers cubed are the same. To find such instances, the sums of pairs are sorted and adjacent elements which are equal are selected:

```
(R=1⌽R)/R←S[⍋S]
```

A frequent use of variable insertion is restoring of blanks between words in compressed text, where if N←+\WLV the word length vector, the expression:

```
(TX,(⍴N)⍴' ')[⍋(⍳⍴TX),N]
```

restores the blanks.

ULTIMATELY PERIODICALLY SELF-REPRODUCING EXPRESSIONS

James G. Mauldon
Amherst College

1. Self-reproducing expressions. A <u>self-reproducing expression</u> (SRE) is an expression *V* satisfying

```
t(V=⍎V)∧(~V∊'ABC...ZΔA̲B̲C̲...Z̲Δ̲')
```

Such an expression is necessarily a vector of characters and, as reported in [1,2], one SRE of minimal length (22) is 22⍴1⌽11⍴'''22⍴1⌽11⍴'''

Readers of [2] were challenged to find a SRE of odd length. One such expression, of length only 23, is

```
23⍴2⌽12⍴'''123⍴2⌽12⍴'''
```

In fact there exists a SRE of any length L ≥ 22. Examples are provided by the results (*SRE L*) and (*SRE*2 *L*), where the functions *SRE* and *SRE*2 defined in section 4 of this note:

```
      ⍎⎕←SRE 35
35⍴12⌽23⍴'''35⍴12⌽23⍴3⍴35⍴12⌽23⍴'''
35⍴12⌽23⍴'''35⍴12⌽23⍴3⍴35⍴12⌽23⍴'''
```

2. Periodically self-reproducing expressions. By definition, a <u>PSRE</u> of <u>period</u> P is an expression which reproduces itself after P applications of the execute function, but not earlier, and readers of [2] were challenged to find a PSRE of period 2.

It will be seen that the expressions *A* and *B* below are PSRE, of quite different types, of period 2 and 3 respectively.

```
      ∧/A=⎕ ← ⍎⎕ ← ⍎⎕ ← A
1⌽26⍴13⍴'''2↓1⌽26⍴13⍴'''
2↓1⌽26⍴13⍴'''2↓1⌽26⍴13⍴'''
1⌽26⍴13⍴'''2↓1⌽26⍴13⍴'''
1
      ∧/B=⎕ ← ⍎⎕ ← ⍎⎕ ← ⍎⎕ ← B ← PSRE2 3
4↓1⌽(48⍴47⍴21⍴'''),1⌽4↓1⌽(48⍴47⍴21⍴'''),1⌽'⍟   '
4↓1⌽(48⍴47⍴21⍴'''),1⌽4↓1⌽(48⍴47⍴21⍴'''),1⌽'   ⍟'
4↓1⌽(48⍴47⍴21⍴'''),1⌽4↓1⌽(48⍴47⍴21⍴'''),1⌽' ⍟ '
4↓1⌽(48⍴47⍴21⍴'''),1⌽4↓1⌽(48⍴47⍴21⍴'''),1⌽'⍟   '
1
```

The vector *B* is the result (*PSRE*2 *P*) (see section 4) where ⍴*P*=3, but an even more striking instance of the phenomenon of periodicity is provided by the result (*PSRE P*), where the period P can be any positive integer not exceeding 99999:

```
      ⍎⎕ ← ⍎⎕ ← PSRE 134
(12⌽48⍴24⍴'''),⍕134   |1+(12⌽48⍴24⍴'''),⍕134   |1+0
(12⌽48⍴24⍴'''),⍕134   |1+(12⌽48⍴24⍴'''),⍕134   |1+1
(12⌽48⍴24⍴'''),⍕134   |1+(12⌽48⍴24⍴'''),⍕134   |1+2

      ∧/(PSRE 134)=⎕ ← ⍎⎕ ← ⍎⎕ ← ⍎(132⍴'⍎'),'PSRE 134'
(12⌽48⍴24⍴'''),⍕134   |1+(12⌽48⍴24⍴'''),⍕134   |1+132
(12⌽48⍴24⍴'''),⍕134   |1+(12⌽48⍴24⍴'''),⍕134   |1+133
(12⌽48⍴24⍴'''),⍕134   |1+(12⌽48⍴24⍴'''),⍕134   |1+0
1
```

3. Ultimately periodically self-reproducing expressions. The result (*T UPSRE P*) of the function *UPSRE* is an expression which, after yielding a 'tail', of length T, of non-repeated expressions, continues with a PSRE of period P. For example, we have:

```
      ⍎⎕←⍎⎕←⍎⎕←⍎⎕←⍎⎕ ← 2 UPSRE 3
(12⌽48⍴24⍴'''),⍕⌊/0 3|1+(12⌽48⍴24⍴'''),⍕⌊/0 3|1+¯2
(12⌽48⍴24⍴'''),⍕⌊/0 3|1+(12⌽48⍴24⍴'''),⍕⌊/0 3|1+¯1
(12⌽48⍴24⍴'''),⍕⌊/0 3|1+(12⌽48⍴24⍴'''),⍕⌊/0 3|1+0
(12⌽48⍴24⍴'''),⍕⌊/0 3|1+(12⌽48⍴24⍴'''),⍕⌊/0 3|1+1
(12⌽48⍴24⍴'''),⍕⌊/0 3|1+(12⌽48⍴24⍴'''),⍕⌊/0 3|1+2
(12⌽48⍴24⍴'''),⍕⌊/0 3|1+(12⌽48⍴24⍴'''),⍕⌊/0 3|1+0
```

but there are shorter expressions possessing the same property.

4. The functions *SRE*, *SRE*2, *PSRE*, *PSRE*2 and *UPSRE*.

```
SRE:   α⍴¯9⌽'''''''',10⌽(α-15)⍴'⍴⍴⌽',⍕α-3 1⍴0 23 12÷(α≥24)∧(α≤99)
SRE2:  α⍴(α-12)↑3⌽'''''''',,1⌽'⍴⌽⍴',⍕α-3 1⍴0 23 12÷(α≥24)∧(α≤99)
PSRE:  1⌽'0',3⌽48⍴24↑'|1+(12⌽48⍴24⍴''''''),⍕',⍕ω
PSRE2:(42⍴'4↓1⌽(48⍴47⍴21⍴''''''),1⌽'),1⌽'''''';ω↑'⍟'
UPSRE:(3⌽48⍴'|1+(12⌽48⍴24⍴''''''),⍕⌊/0 ',⍕ω),⍕-α÷(ω≤9)
```

The characters after ÷ provide protection against misuse of the functions *SRE*, *SRE*2 and *UPSRE*. The result (*SRE L*) contains no spaces if L ≥ 33, whereas the result (*SRE*2 *L*) reveals its operation more clearly:

```
      2 34⍴(SRE 34),(SRE2 34)
34⍴11⌽22⍴'''34⍴11⌽22⍴⍴34⍴11⌽22⍴'''
34⍴11⌽22⍴'''             34⍴11⌽22⍴'''
```

5. Non-repeating expressions. Plainly (*PSRE* 0), subject only to the limitation imposed by the size of the system, allows any number of applications of the execute function without ever producing a repetition. Another non-repeating expression is

```
(6⌽34⍴17⍴'''),⍕1+(6⌽34⍴17⍴'''),⍕1+0'
```

but very much shorter examples may be found.

6. Three problems.

(A) Find the shortest ultimately periodically self-reproducing expression.

(B) Find the shortest non-repeating expression (NRE).

(C) Find the shortest NRE which does not use the format function.

REFERENCES

[1] APL Quote-Quad, Vol.4, No.2 (January, 1973)

[2] This Newsletter, Vol.2, Nos.1,2,3 (1977)