THE APL HANDBOOK OF TECHNIQUES
THE APL HANDBOOK OF TECHNIQUES

"Compiled by DP Scientific Marketing"
PREFACE

The APL Handbook of Techniques is intended to augment the “bag of tricks” of the active APL user. As in the case of the primitive functions, the defined functions illustrated in this handbook may be used without full understanding of their methodology; however, any time spent analyzing the statements will be richly rewarded with new insights into the power of APL and the amazing foresight of Ken Iverson and Adin Falkoff.

What you are holding is a compendium of hundreds of functions submitted by professional programmers within IBM. These many contributions have been generalized, extended and harmonized into families (such as Text-Editing, Logical Operations, Report Formatting, Multi-Precision Arithmetic and Workspace Management). That APL is an art form is quickly evident by examining the various styles represented.

Various criteria were used in selecting and refining these functions: elegance, space and execution time. However, as you become familiar with each of the functions, you should experiment with your own variations, thereby imparting a personal style into your work. Once understood, the functions can be modified with confidence in the integrity of APL and its predictability.

Preparing this collection has been a very rewarding experience for me. I have often said that I am the greatest benefactor of this publication, as many of the functions were used to prepare the book itself. But the task was aided by the many contributors and the assistant editors. I would like to thank Len Lewis of DPD Scientific Marketing who believed from the beginning that such a publication was indeed possible. For the idea and the model, thanks to Curt Bury and Dr. Kent Haralson, respectively. For their contributions and long hours of testing, thanks to Larry Breed, Norm Brenner, Sylvia Eusebi, Ed Eusebi, Len Gilman, Tim Holls, Rainer Kogon, Dieter Lattermann, Beth Luc, Blair Martin, John McCleary, John McPherson, Joe Myers, Don Orth and Harry Saal.

Dave Macklin
December, 1977

De gustibus non est disputandum
INTRODUCTION

There are many publications from which one can learn how the operators work, and how to combine characters to form APL expressions, but few are written with the intention of developing the reader's style.

This handbook contains no explanation of the APL primitives; we assume you already understand them. Similarly, fundamental operations are not emphasized. The goal of this handbook is to furnish you with a collection of meaningful, useful APL functions, each demonstrating a particular technique. By carefully examining each function, you should begin to expand your APL awareness, thus becoming more proficient in the use of the language.

As with any programming language, there is no single way to solve a problem. However, preferred methods yield elegant functions which are either time or space efficient. Conversely, some approaches produce APL functions which can be inefficient or limited in scope. To improve your style, study this book and others like it. Examine functions written by experienced APL problem-solvers; modify those functions to suit your own needs. Fine tune your ability to recognize most efficient APL technique for solving that problem facing you.

Variable Usage

Within this publication you will notice both "global" and "local" APL variables. Without the global concept, variables which are used by sets of functions would have to be identified on each page they are used. Some of the global variables are:

- **AV**: 'ABCDEFGHIJKLMNOPQRSTUVWXYZ'
- **ALF**: AABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789'
- **DIGITS**: '0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ'
- **BK**: (BACKSPACE CHARACTER); SEE VTCC PAGE 44
- **CR**: (CARRIAGE RETURN)
- **ID**: (IDLE CHARACTER)
- **LN**: (LINEFEED CHARACTER)
- **TB**: (TAB CHARACTER)

Programming Note

A very interesting technique is employed in this publication. It can help you to understand how APL functions work. On many pages, you will find an "ANALYSIS" section. The first line of this section will be the expression being analyzed. As you read on, you will notice a line-by-line explanation of the interim results, as though the expression were being executed. By carefully examining this analysis, you will learn how and why the function accomplishes the stated technique.
The information contained in this document has not been submitted to any formal IBM test. Potential users should evaluate its usefulness in their environment.
Each page of this handbook contains exactly one primary and, optionally, one or more subordinate (secondary) functions. If they appear, subordinate functions are located to the side of the page. To locate any function (primary or subordinate), refer to the complete subject index or the KWIC index in the Appendix.

<table>
<thead>
<tr>
<th>function name</th>
<th>calling sequence</th>
<th>what it does</th>
<th>APL code</th>
</tr>
</thead>
<tbody>
<tr>
<td>DIFF</td>
<td>SINTAX: R=DIFF A</td>
<td>FUNCTIONS:</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>$\forall R=\text{DIFF }A$</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>[1] $R=((-\text{ppA})+1)\cdot A-1\cdot A$</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>$\forall R=\text{UNSCAN }V$</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>[1] $R=\text{V[0:10],DIFF V}$</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>[2] $\forall V++\forall R$</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**EXAMPLE:**

```
DIFF 11 22
  11      DIFF -3 4p12
  -1    -1
  -1    -1
```

**ANALYSIS:**

```
[1] R=((-\text{ppA})+1)\cdot A-1\cdot A

 25 18 24
 11 9 20
 51 53 14 3
```

```
[1] R=((-\text{ppA})+1)\cdot A-1\cdot A

 25 18 24 53
 11 9 20 40
 51 53 14 3
```

**SUBORDINATE FUNCTION**

```
APL code
```

**WHAT OCCURS DURING FUNCTION EXECUTION**

```
SIGNS CHANGED
-28 -7 6 29
-29 -2 11 20
-2 -39 11 48
```

```
PADDDED
0 -1
```

```
```

```
```

```
```

```
```
## TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>I. Matrix Manipulation Functions</td>
<td>1</td>
</tr>
<tr>
<td>II. Report Formatting Functions</td>
<td>24</td>
</tr>
<tr>
<td>III. Workspace Management Functions</td>
<td>40</td>
</tr>
<tr>
<td>IV. Multiprecision Arithmetic Functions</td>
<td>47</td>
</tr>
<tr>
<td>V. Mathematical and Numerical Functions</td>
<td>63</td>
</tr>
<tr>
<td>VI. Utility and Miscellaneous Functions</td>
<td>79</td>
</tr>
<tr>
<td><strong>APPENDIX</strong></td>
<td></td>
</tr>
<tr>
<td>APL Bibliography</td>
<td>92</td>
</tr>
<tr>
<td>KWIC Index</td>
<td>93</td>
</tr>
<tr>
<td><strong>SORT of Function Name vs. Abstract</strong></td>
<td></td>
</tr>
<tr>
<td>by Abstract</td>
<td>106</td>
</tr>
<tr>
<td>by Function Name</td>
<td>108</td>
</tr>
<tr>
<td>Subject Index</td>
<td>110</td>
</tr>
</tbody>
</table>
Section I

Matrix Manipulation Functions
ADDCOLS  ADD COLUMNS TO A MATRIX VECTOR OR SCALAR

**SYNTAX:**  
Z+A ADDCOLS B

- CONVERTS SCALAR AND VECTOR RIGHT ARGUMENTS INTO ONE-ROW MATRICES AND PADS THEM OUT ON THE LEFT (POSITIVE LEFT ARGUMENT) OR ON THE RIGHT (NEGATIVE LEFT ARGUMENT). RIGHT ARGUMENT MAY BE EITHER NUMERIC OR CHARACTER.
- USES: VMATRIX

**FUNCTION:**

```
\[ V Z+A ADDCOLS B \]
\[ [1] Z+((1+pB),-A+(xA)x1+pB)+B\text{MATRIX}\ B \]
\[ V \]
\[ V \]
```

**EXAMPLE:**

```
3 ADDCOLS 2 3p16
0 0 0 1 2 3
0 0 0 4 5 6
-2 ADDCOLS 3 4 5 7
3 4 5 7 0 0
0 ADDCOLS 2 3p'ABCDEF'
```

**ANALYSIS:**

- THE RIGHT ARGUMENT IS CONVERTED TO A MATRIX, THEN IS PADDED
- ON THE LEFT OR RIGHT, OR MADE EMPTY.
- AS SPECIFIED BY THE SIGNUM OF THE LEFT ARGUMENT (xA).
ADDROWS  ADD ROWS TO A MATRIX VECTOR OR SCALAR

SYNTAX:  Z+A ADDROWS B

- CONVERTS SCALAR AND VECTOR RIGHT ARGUMENTS INTO ONE-ROW MATRICES AND PADS THEM OUT ON TOP (POSITIVE LEFT ARGUMENT) OR ON THE BOTTOM (NEGATIVE LEFT ARGUMENT). RIGHT ARGUMENT MAY BE EITHER NUMERIC OR CHARACTER.
- USES: VMATRIX

FUNCTION:

\[ \nabla \ Z+A \text{ ADDROWS } B \\
[1] \ Z+(\! A+(\times A)\times 1+pB),1+pB)+B\times \text{ MATRIX } B \\
\n\n\]

EXAMPLE:

\[
\begin{align*}
2 & \ \text{ADDROWS} \ 2 \ 2p\ 14 \\
0 & 0 \\
0 & 0 \\
1 & 2 \\
3 & 4 \\
| & | \\
| & | \\
' | ' & \text{,} \ 2 \ \text{ADDROWS} \ 'ABCDEF' \\
|ABCDEF \\
| \\
1 & \ \text{ADDROWS} \ 3 \ 4 \ 5 \ 6 \\
0 & 0 \ 0 \ 0 \\
3 & 4 \ 5 \ 6 \\
\rho \ 1 & \ \text{ADDROWS} \ 3 \\
2 & 1
\end{align*}
\]

ANALYSIS:

- THE RIGHT ARGUMENT, CONVERTED TO A MATRIX, IS PADDED AT THE TOP OR BOTTOM OR MADE EMPTY, AS SPECIFIED BY THE SIGNUM OF THE LEFT ARGUMENT (\(\times A\)).
CCAT 
CATENATE BY COLUMNS [ VERTAB CMATRIX ROWFORM ]

SYNTAX:
R+A CCAT B

- SIDE-BY-SIDE CATENATION OF GENERAL STRUCTURES AND TYPES.
- SCALARS WILL BE REPLICATED IF TYPES AGREE.
- VECTORS BECOME ONE-COLUMN MATRICES BEFORE CATENATION.
- NUMERIC TYPES WILL BE PADDED WITH ZEROS.
- CHARACTER TYPES WILL BE PADDED WITH BLANKS.
- NUMERIC TYPES WILL BE FORMATTED IF TO BE CATENATED WITH CHARACTER TYPES. SEE \( \nabla \) BESIDE
- A MATRIX IS RETURNED.

FUNCTIONS:
\( \nabla \) R+A CCAT B
[1] A COLUMN CATENATION, SCALAR REPLICATION
[2] VERTAB
[3] CFORMAT \( \nabla \) VERTAB
[6] R+A,B \( \nabla \)

EXAMPLES:
\( \nabla \) ROWFORM; R
S='*'
U='['
V='ABC'
M=2 3p'X'
T=QM

V CCAT S 7 CCAT \( \nabla \) 7
A 7 1
B 7 2
C 7 3 \( \nabla \) R+CMATRIX

V CCAT U 7 4 [1] A ASSUMES LOCALIZED A AND B
A* 7 5 [2] A+MATRIX A' IF 0=pA
C 7 7 \( \nabla \)

V CCAT M
AXXX
BXXX
C
M CCAT T
XXXXX
XXXXX
XX

S CCAT M CCAT S
*XXX*
*XXX*
CHAR       BUILD CHARACTER ARRAY TO NUMERIC PATTERN

SYNTAX:      R+K CHAR N

* DISPLAYS OR OTHER STRUCTURES CAN BE FASHIONED FROM LOGICAL OR NUMERIC STRUCTURES, LATER TO BE OVERLAID BY VFILLS.
* USES: VONESIN

FUNCTION:

\[ \text{\texttt{\textbackslash v R+K CHAR N; \textbackslash n \textbackslash o}} \]
\[ \text{[1] A K IS SINGLE CHARACTER TO BE PLACED} \]
\[ \text{[2] A ACCORDING TO POSITIONS OF ONES IN N} \]
\[ \text{[3] R+(\times/pN)p \texttt{\textbackslash n \textbackslash a \texttt{\textbackslash n \textbackslash i \texttt{\textbackslash n \textbackslash o}})} \]
\[ \text{[4] R[ONESIN,N]+K} \]
\[ \text{[5] R+(pN)pR} \]
\[ \text{\texttt{\textbackslash v}} \]

EXAMPLE:

PATTERN
1 0 0 0 1
0 1 0 1 0
0 0 1 0 0
0 1 0 1 0
1 0 0 0 1

'X' CHAR PATTERN
X X
X X
X
X X
X X
X X
DIFF

DIFFERENCES BETWEEN ADJACENT ELEMENTS [UNSCAN]

SYNTAX: R+DIFF A

• A IS ANY NUMERIC STRUCTURE.
• R HAS ONE FEWER COLUMNS THAN A, AND CONSISTS OF THE SUCCESSIVE DIFFERENCES, A[...;...;I+1]-A[...;...;I]

FUNCTIONS:

\[ \nabla R+\text{DIFF} A \]
\[ R+((-pA)^{+1})(1\Phi A)-A \]

\[ \nabla R+\text{UNSCAN} V \]
\[ R+V[\square IO],\text{DIFF} V \]
\[ a V\leftrightarrow\rightarrow\rightarrow\rightarrow\rightarrow\rightarrow R \]
\nEXAMPLE:

DIFF 11 22
  11
  DIFF -3 4 \text{p} 12
  -1 1 -1
  -1 1 -1
  -1 1 -1

ANALYSIS: DIFF 3 4p1299

[1] \[ R+((-pA)^{+1})(1\Phi A)-A \]
  51 54 4 87
  80 68 37 96
  35 38 20 75

[1] \[ R+((-pA)^{+1})(1\Phi A)-A \]
  54 4 87 51
  68 37 96 80
  38 20 75 35

[1] \[ R+((-pA)^{+1})(1\Phi A)-A \]
  3 50 83 56
  -12 -31 59 -16
  -3 -18 55 -40

[1] \[ R+((-pA)^{+1})(1\Phi A)-A \]
  -2

[1] \[ R+((-pA)^{+1})(1\Phi A)-A \]
  FOR ANY STRUCTURE
  0 -1

[1] \[ R+((-pA)^{+1})(1\Phi A)-A \]
  3 50 83
  -12 -31 59
  -3 -18 55
EDIT

EDIT LATENT EXPRESSION OR CHARACTER STRUCTURE [ SEDIT ]

SYNTAX:

EDITED+EDIT UNEDITED

- LATENT EXPRESSIONS AND OTHER CHARACTER STRINGS ARE OFTEN DIFFICULT TO MODIFY, (CHANGE, ADD, DELETE), ESPECIALLY WHEN QUOTATION MARKS ARE INVOLVED. EDIT PERMITS YOU TO DEAL WITH THE FINAL APPEARANCE OF THE VECTOR.
- VECTORS MAY EXPAND OR CONTRACT.
- USE VMEDIT TO MODIFY MATRICES.

FUNCTIONS:

\[ \text{EDIT EDITED+EDIT UNEDITED;SLASH} \]

\[ \text{[1] } \text{A SAME AS EDIT, BUT INSERT IS AN EXECUTABLE EXPRESSION} \]

\[ \text{[2] } \text{A SAME AS EDIT, BUT INSERT IS AN EXECUTABLE EXPRESSION} \]

EXAMPLES:

A CHARACTER VECTOR WITH QUOTES WOULD HAVE TO BE KEYED, THUS:

\[ \text{A+EDIT'} \]

A+EDIT'' (TO CREATE ITEM)

(OPPORTUNITY TO ADD, NOW)

HE SAID, 'HELP ME, I'M DROWNING!''

HE SAID, 'HELP ME, I'M DROWNING!' (FOR PROOFREADING)

A+EDIT A

HE SAID, 'HELP ME, I'M DROWNING!'

(SPACE TO INSERTION POSITION)

PLEASE

HE SAID, 'HELP ME PLEASE, I'M DROWNING!''

A+EDIT A

HE SAID, 'HELP ME PLEASE, I'M DROWNING!''

(SPACE TO INSERTION POSITION)

PLEASE

HE SAID, 'PLEASE HELP ME PLEASE, I'M DROWNING!''

A+EDIT A

HE SAID, 'PLEASE HELP ME PLEASE, I'M DROWNING!''

///// (TO DELETE)

HE SAID, 'PLEASE HELP ME, I'M DROWNING!''

SEDIT 'TYPEWRITER IS A X-LETTER WORD.'

TYPEWRITER IS A X-LETTER WORD.

\[ \text{\^p 'TYPEWRITER'} \]

TYPEWRITER IS A 10-LETTER WORD.
ERECT

ERECT WORD MATRIX FROM CHARACTER STRUCTURE [ DLTMB ]

SYNTAX:

R+ERECT A

- BUILDS TABLE THAT CAN BE ADDRESSED RANDOMLY OR SEQUENTIALLY
- INPUT WORD SEPARATORS ARE SINGLE OR MULTIPLE BLANKS
- INPUT NEED NOT BE A VECTOR.
- HIGH-SPEED DESIGN COMPUTES ADDRESSES. (SEE VSHAPE)
- USES: VDIFF VDLTMB VΔ

FUNCTIONS:

\[ R^{+\text{ERECT}} A; \square IO; L; S; D; COLS; ROWS; Z \]

[1] \[ a \text{ AVOIDS OUTER PRODUCT FOR SPEED; ASSUMES BLANK DELIMITERS} \]
[2] \[ COLS+\lceil /D−1+DIFF 0, S:\\ ROWS+1+1+S++\lceil L\rceil ' = A+DLTMB A \Delta \square IO+1 \]
[3] \[ Z+(ROWS×COLS)d' ' \]
[4] \[ Z[\left((\sim L)/+\lceil L\right)\text{COLS}−1+D)+1+/~L]+(−L)/A \]
[5] \[ R^{+(ROWS,COLS)}\text{d} Z \]

\[ \square R^{+\text{DLTMB}} A; Z \]

[1] \[ a \text{ DELETE LEADING, TRAILING, MULTIPLE BLANKS} \]
[2] \[ R^{+}(Z\text{\&1}Z\neq' ') /A+, ' ', A \]

EXAMPLE:

\[ \text{VFORM} \]
\[ \text{VARIABLE FORMAT BY ROW OF A MATRIX [ ESCAPE ESCAPEX ]} \]
\[ \text{XVEC} \]
\[ \text{EXPAND LOGICAL VECTOR} \]
\[ \text{ZDIV} \]
\[ \text{ZERO TOLERANT DIVISION [ CDIV ]} \]

TIME"J"+ERECT M'

10 MSEC 0 BYTES

TIME"J+" 'SHAPE,M'

22 MSEC 0 BYTES

ERECT M

VFORM

VARIABLE FORMAT BY ROW OF A MATRIX

(\text{ESCAPE ESCAPEX})

XVEC

EXPAND LOGICAL VECTOR

ZDIV

ZERO TOLERANT DIVISION

[ CDIV

\[ \text{(NOTE USE OF Z AS LOCAL VARIABLE TO SAVE TIME AND SPACE) \]}\]
**FIRSTM**  SELECT FIRST OR ONLY APPEARANCE IN MATRIX [ FIRSTV ]

**SYNTAX:**

L-FIRSTM M

RETURNS LOGIC VECTOR THAT CAN SELECT:
- ROWS OF A MATRIX, IGNORING DUPLICATES
- THEIR INDICES
- CORRESPONDING ROWS OF INVERTED FILES

SEE \textit{\textit{VRIOTA}}

**FUNCTIONS:**

\[
\begin{align*}
&\text{v } L-FIRSTM M \\
&L-(V \& M) = 1 p M
\end{align*}
\]

FOR VECTOR ARGUMENTS.  SEE \textit{VDREP}.

**EXAMPLES:**

\[
\begin{align*}
\text{M+M=ELECT 'TOM DICK TOM HARRY DICK HARRY'}
\end{align*}
\]

TOM
DICK
TOM
HARRY
DICK
HARRY

(FIRSTM M)\#M

TOM
DICK
HARRY

(FIRSTM M)/l+t+p

1 2 4

FIRSTM M
1 1 0 1 0 0

**ANALYSIS:**

(FIRSTM M)\#M

\[
\begin{align*}
&\text{L+V+} \text{\textbackslash M} \cdot = q M \\
&\text{TDTHDH} \quad \text{TRANSPOSE ASSURES CONFORMITY}
\end{align*}
\]

GIOAIA
MCMRKR
K RKR
Y Y

\[
\begin{align*}
&\text{L+V+} \text{\textbackslash M} \cdot = q M \\
\end{align*}
\]

1 0 1 0 0 0
0 1 0 1 0 0
1 0 1 0 0 0
0 0 0 1 0 1
0 1 0 0 1 0
0 0 0 1 0 1

\[
\begin{align*}
&\text{L+V+} \text{\textbackslash M} \cdot = q M \\
&\text{\textbackslash CAPTURES POSITION OF FIRST 1 ENCOUNTERED.} \\
1 1 0 1 0 0 \\
&\text{LOGIC VECTOR TO SELECT ROWS}
\end{align*}
\]

TOM
DICK
HARRY
FRAME AN ARRAY [ MATRIX CHARACTER ]

**Syntax:**

\[
\text{Z+FRAME A}
\]

- Employs the characters ' - | \^ ' to build a frame around any array after reshaping it as a matrix.
- No data is truncated.
- Uses: \text{VCHARACTER} \text{VTABULATE} \text{VADJUSTUP} \text{VADJUSTDOWN} \text{VFRAMETEST} \text{VMATRIX} \text{VIF}

**Functions:**

\[
\begin{align*}
&\text{V Z+FRAME A};[1] \\
&\text{+L1 IF CHARACTER Z+A} \quad \text{[1]} \quad T+0\neq 0\backslash 0pA \\
&\text{Z+TABULATE Z} \quad \text{} \quad \text{} \\
&\text{LO:Z+1 ADJUSTUP' - ',[IIO+1]Z} \\
&\text{Z+' | ',[Z+1]' } \quad \text{} \quad \text{} \\
&\text{Z-ADJUSTDOWN Z} \\
&\text{L1:LO IF O=FRAMETEST Z+MATRIX Z} \quad \text{[2]} \quad \text{RESULT HAS TWO DIMENSIONS} \\
\end{align*}
\]

**Example:**

\[
\begin{align*}
\text{LV='THIS IS A LITERAL VECTOR'} \\
\text{FRAME LV} \\
\text{|------------------------|} \\
\text{|THIS IS A LITERAL VECTOR|} \\
\text{|------------------------|}
\end{align*}
\]

**Analysis:**

- Line 6 checks whether the argument, if character, is already framed.
- Line 3 frames the top, placing the character '|'
- In the first and last columns.
- Lines 4 and 5 frame the sides and bottom.
FRAMETEST CHECKS A MATRIX FOR FRAMING

SYNTAX: Z+FRAMETEST A

- EXAMINES A MATRIX FOR THE PRESENCE OF FRAMING ELEMENTS '____' AROUND ITS PERIPHERY AND RETURNS A 1 IF PRESENT, 0 OTHERWISE.
- USES: VIF

FUNCTION:

\[
\begin{align*}
\forall Z+\text{FRAMETEST } A \\
[1] & Z\leftarrow 0+1 \\
[2] & \rightarrow 0 \text{ IF } 0=x/pA \\
[3] & \rightarrow 0 \text{ IF } A[1,1]x'1' \\
[4] & \rightarrow 0 \text{ IF } 1=x^/,(A[1,1+pA],&A[1,1-1+pA])\varepsilon'1'____' \\
[5] & Z+1 \\
\end{align*}
\]

EXAMPLES:

\[
\begin{align*}
\text{FRAMETEST } 3 \ 4 'ABCD' \\
0 \\
\text{|______|} \\
\text{|SALES|} \\
\text{|______|} \\
\text{FRAMETEST X} \\
1 \\
\text{SALES} \\
\text{|-----|} \\
\text{1 \ 2 \ 3} \\
\text{4 \ 5 \ 6} \\
\text{FRAMETEST Y} \\
0 \\
\end{align*}
\]

ANALYSIS:

- LINES 2,3,4 SET THE RESULT TO 0 IF THE ARGUMENT IS EMPTY, OR THE A [1;1] ELEMENT IS NOT '1' OR THE ELEMENTS IN THE FIRST AND LAST ROWS, A FIRST AND LAST COLUMNS ARE NOT ALL MEMBERS OF '____'. OTHERWISE THE A RESULT IS 1.
GRADEUP

GENERATE ASCENDING ROW INDICES [ AV ALF NFORM LINFORM ]

SYNTAX:

I+C GRADEUP K

- TO SORT A LEFT-JUSTIFIED MATRIX ALPHABETICALLY
- C IS A COLLATING SEQUENCE; K IS A CHARACTER MATRIX.
- IF UNIQUE DISTINCTIONS OCCUR ONLY AT RIGHT SIDE, AND IF THE COLLATING SEQUENCE IS LONG, IT MAY BE NECESSARY TO SORT IN MORE THAN ONE PASS, FIRST ACCORDING TO THE RIGHTMOST COLUMNS.
- USES: NFORM

FUNCTIONS:

\[ \forall I+C GRADEUP K \]
[1] \[ I+\Phi C NFORM K \]
\[ \forall N+C NFORM K; \squareIO \]
[1] \[ N+(pC)1C1QK \land \squareIO+O \]
\[ \forall R+C LINFORM K \]
[1] \[ R+C NFORM((1+pR),11)+R+ERECT,K,\' \']
\[ \forall \]

EXAMPLES:

AV+' ABCDEFGHIJKLMNOPQRSTUVWXYZ'
A USEFUL GLOBAL, WITH 11-COLUMN RESOLUTION
ALF+AV,'ΔABCD EFGHIJKLMNOPQRSTUVWXYZ\alpha123456789'
A USED BY VVARS...LESS RESOLUTION BUT MORE CHARACTERS

AV GRADEUP 3 1p'ZYX'
3 2 1
DICK
HARRY
TOM

A[ΦAV GRADEUP A;]

TOM
HARRY
DICK

(ALPHABETICALLY SORTED)
(REVERSE ORDER)
INDEX
COLUMN INDEX IN MATRIX B WHOSE MEMBERS ALL BELONG TO A

SYNTAX:
Z+\text{A INDEX B}

- RETURNS THE INDEX POSITION OF EACH COLUMN OF B ALL OF WHOSE ELEMENTS ARE IN A. B MUST BE A MATRIX. THE SHAPE OF A IS NOT RESTRICTED. THE ARGUMENTS MAY BE EITHER CHARACTER OR NUMERIC.

FUNCTION:
\begin{verbatim}
\( \forall \ Z+\text{A INDEX B} \\
[1] \ Z*(\&\text{B} A) \& 1+pB \\
\end{verbatim}

EXAMPLE:
\begin{verbatim}
A+10
B+?3 8p15
B
1 1 1 3 5 1 4 4
9 8 7 11 9 12 3 1
2 1 1 2 1 3 1 7
A INDEX B
1 2 3 5 7 8
'A' INDEX C+3 6p'\text{ABCDEFAHIJKLABCXYZ}'
1
C
\text{ABCDE}
\text{AHIJKL}
\text{ABCXYZ}
\end{verbatim}

ANALYSIS:
- 'ANDS' OVER THE COLUMNS OF THE LOGICAL MATRIX CREATED BY \( \text{B} \& \text{A} \),
- THEN USES COMPRESSION TO SELECT THE CORRESPONDING COLUMN INDICES.
MEDIT EDIT MATRIX

SYNTAX: 

R+KD MEDIT M

- RETURNS A MODIFIED FORM OF STRUCTURE M AS A MATRIX.
- KD IS A PAIR OF INTEGERS, THE FIRST, K, SIGNIFIES THE NUMBER OF ROWS OF M PREcedING THE ONE TO BE CHANGED, INSERTed, OR DELETED. THE SECOND, D, IS THE LINE NUMBER TO BE DELETED.
- THE KEYBOARD WILL UNLOCK FOR THE EXPRESSION THAT WILL GENERATE THE LINE(S) TO BE INSERTED.
- THE INSERT MAY BE AN ALTERED FORM OF THE LINE(S) DELETED.
- WHEN K=D, THE INSERT WILL APPEAR AFTER LINE K.
- WHEN K<D, D-K LINES WILL BE DELETED BEFORE ACCEPTING THE INSERT. (IF K>D, K-D ORIGINAL ROWS WILL APPEAR BEFORE AND AFTER THE INSERT.)
- USES: VMATRIX VON vΔ

FUNCTION:

V R+KD MEDIT M:ΔIO

[1] a MATRIX EDIT (DELETE, INSERT, CHANGE)
[2] R+=(((1+KD),1+M)+M)ON(ΔM)ON((1+KD),0)+M=MATRIX M Δ ΔIO+1

EXAMPLES:

1 2 MEDIT 3 4p\12 (DELETING ROW 2)
10
1 2 3 4
9 10 11 12
1 1 MEDIT 3 4p\12
1
1 2 3 4
1 0 1 0
5 6 7 8
9 10 11 12
1 2 MEDIT 3 4p\12
\12 MEDIT 3 4p\12 (INSERTING FUNCTION OF EXISTING ROW)
1 2 3 4
8 7 6 5
9 10 11 12
3 0 MEDIT 3 4p\12 (SUPERIMPOSING FIRST THREE ROWS)
0
1 2 3 4
5 6 7 8
9 10 11 12
1 2 3 4
5 6 7 8
9 10 11 12
1 1 MEDIT 2 3p'ABCDEF'
' THIS IS THE LETTER B'
ABC
'THIS IS THE LETTER B'
DEF
M2V

COMPRESS CHARACTER MATRIX  EXPAND RESULT [ V2M ]

SYNTAX:

V+M2V M  AND  M+V2M V

These complementary functions allow two-way conversion between character matrices and character vectors.

M2V: Converts a character matrix M to a character vector V. Each row of M, with trailing blanks omitted, becomes a 'LINE' in V, ended by a carriage return.

V2M: Converts a character vector V to a character matrix M. Each 'LINE' (a character string ending in a carriage return) becomes a row of M, with padding as required. Both V and M will appear the same when displayed, but the vector representation is usually more economical in storage. The global CR must exist in workspace. (See VTCC page).

FUNCTIONS:

\[
\begin{align*}
\text{V} & \leftarrow \text{V}+\text{M2V M} \\
\text{V} & \leftarrow (\phi_1, V_1\ '\phi M)/, M, CR \\
\text{V} & \leftarrow \text{M}+\text{V2M V}; \text{IO} \\
\text{V} & \leftarrow (\phi M)(, M+M_0, \geq \{1/0, M+M-1+0, -1+M+M/ \phi M)(\sim M+V=CR)/V+V, CR \\
\text{V} & \leftarrow \text{V}+\text{'LINE 1'. CR}, \text{'LINE NUMBER 2'}
\end{align*}
\]

EXAMPLES:

\[
\begin{align*}
\text{V} & \leftarrow \text{'LINE 1'. CR}, \text{'LINE NUMBER 2'} \\
\text{LINE 1.} \\
\text{LINE NUMBER 2} \\
\text{M2V} & \leftarrow \text{V2+M2V M} \quad \text{(CONVERT VECTORIZATION TO MATRIX)} \\
\text{LINE 1.} \\
\text{LINE NUMBER 2} \\
\text{M} & \leftarrow \text{M2V} \\
\text{V2+M2V M} & \leftarrow \text{V2+M2V M} \quad \text{(CONVERT MATRIX BACK TO VECTORIZATION)} \\
\text{A/V}=\text{V2} & \leftarrow \text{A/V}=\text{V2} \quad \text{(COMPARE TWO VECTORS)}
\end{align*}
\]

ANALYSIS:

M2V[1]: A COLUMN OF CARRIAGE RETURNS IS CATENATED ONTO M AND THE RESULT RAVELED AND COMPRESSED BY A BOOLEAN VECTOR TO REMOVE TRAILING BLANKS IN EACH ROW. THE FINAL CARRIAGE RETURN IS THEN REMOVED.

V2M[2]: AFTER A CARRIAGE RETURN IS CATENATED ONTO V, IT IS SEARCHED FOR CARRIAGE RETURNS AND THEY ARE COMPRESSED OUT. THIS RESULT IS THEN EXPANDED BY A BOOLEAN VECTOR WHICH HAS THE EFFECT OF PADDING LINES TO THE SAME LENGTH. THE RESULT IS RESHAPED INTO A MATRIX.
PREEDIT  PREPARE MATRIX FOR FUNCTION-LIKE EDITING  [ POSTEDIT ]

SYNTAX:  R+TNAME PREEDIT M

- TNAME IS A TEMPORARY NAME TO BE ASSIGNED TO A COPY OF THE MATRIX, M, SO THAT IT MAY BE EDITED AS IF IT WERE A DEFINED FUNCTION. TNAME IS A CHARACTER STRING.
- WHEN EDITING IS COMPLETE, KEY: R+POSTEDIT TNAME, WHERE R CAN BE THE OLD OR NEW NAME OF THE EDITED MATRIX.
- USES: \( \text{VESCAPE VON} \)

FUNCTIONS:

\[ 1 \quad R+TNAME \text{ PREEDIT } M \]
\[ 2 \quad \text{(TNAME,' IN USE')ESCAPE 0=\text{NC TNAME}} \]
\[ 3 \quad \text{R+\text{FX TNAME ON'}}M,\downarrow M \]

\[ \downarrow \quad \text{R+POSTEDIT TNAME:} \]
\[ 1 \quad \text{\'NOT A NAME'ESCAPE~CHARACTER TNAME} \]
\[ 2 \quad \text{(TNAME,' NOT A FUNCTION')ESCAPE 3=\text{NC TNAME}} \]
\[ 3 \quad \text{R+1 1\text{CR TNAME}} \]
\[ 4 \quad j\text{+EX TNAME} \]

EXAMPLES:

> A=\text{ERECT} 'TOM DICK HARRY'
> A
> TOM
> DICK
> HARRY
> B=\text{J}OE\text{PREEDIT} A
> B
> JOE

\( \text{\downarrow JOE}\)  \( \text{(FOR FUNCTION EDITING)} \)

\[ 1 \quad \text{\#TOM} \]
\[ 2 \quad \text{\#DICK} \quad \text{(LAMP SYMBOLS PROTECT INTEGRITY OF DATA)} \]
\[ 3 \quad \text{\#HARRY} \]
\[ 4 \quad [\Delta 2] \quad \text{(TO DELETE ROW 2)} \]
\[ 2 \quad \text{\downarrow} \]
\[ 3 \quad \text{C+POSTEDIT B} \]

> TOM
> HARRY

\( \quad \text{(LAMP SYMBOLS HAVE BEEN REMOVED)} \)
RCAT       Catenate structures by rows [ colform vert ]

Syntax:     R+ A RCAT B

- Over-under catenation of general structures and types.
- Scalars will be replicated.
- Padding will be blank for characters, zero for numbers.
- Numeric types will be formatted if to be catenated to
  character types. See: VON VCCAT
- A matrix is returned.
- Uses: VCFORMAT VCMATRIX VCOLFORM VVERT

Functions:

∀ R+A RCAT B
[1]   a Row catenation, scalar replication
[2]   CFORMAT
[3]   CMATRIX
[4]   COLFORM
[5]   Z' R+VERT A', ((0*(ppA)+ppB)/'[1 IO]'), 'B'
        ∀
        ∀ COLFORM;R
        [1]   a Assumes A and B have been localized
        [2]   ⇒0 IF 0=(pρA)xρB
        [3]   A+((1+pA),R)+A  B+((1+pB),R+((1+pA)[1+pB])+B
        ∀
        ∀ Z+VERT X
        [1]   Z+((ρX),(1=ρX)ρ1)ρX
        ∀

Examples:

S+ 'o'
U+ '□'
V+ 'ABC'
S RCAT U

*□
U RCAT V
*□
ABC
1 RCAT 1 5
1 1 1 1 5 1
1 2 3 4 5
S RCAT 1 5

*□
1 2 3 4 5
U RCAT 1 5
*□
1 2 3 4 5
REPL  REPLACE ALL OCCURRENCES OF ELEMENT IN ARRAY

SYNTAX:  \( R+\text{VV} \) REPL \( A \)

\text{VV} IS A TWO-POSITION CHARACTER OR NUMERIC VECTOR.
\text{A} IS AN ARRAY OF THE SAME TYPE.
ALL APPEARANCES OF \( 1^{\text{VV}} \) WILL CHANGE TO \( 1+\text{VV} \).

FUNCTION:

\[
\begin{align*}
\text{\texttt{\#}} & \quad R+\text{VV} \text{ REPL } A \\
[1] & \quad R[(R=1+\text{VV})/\text{pR}+,A]+1+\text{VV} \\
[2] & \quad R+(\text{pA})\text{pR} \\
\end{align*}
\]

EXAMPLES:

'\text{''REPL''-1234.56''} MINUS SIGN BECOMES APL NEGATIVE
\text{1234.56} 0 1E75 REPL \( \text{\texttt{(13)^.=.13}} \) CHANGE ZEROS TO 1E75

ANALYSIS:

\begin{align*}
1 & \quad -1 \text{ REPL 2 5p1 0} \\
[1] & \quad R[(R=1+\text{VV})/\text{pR}+,A]+1+\text{VV} \quad \text{SEND IN REPLACEMENTS} \\
\text{\texttt{-1}} & \quad R[(R=1+\text{VV})/\text{pR}+,A]+1+\text{VV} \\
1 \quad 0 \quad 1 \quad 0 \quad 1 \\
0 \quad 1 \quad 0 \quad 0 \\
[1] & \quad R[(R=1+\text{VV})/\text{pR}+,A]+1+\text{VV} \quad \text{NOW A VECTOR} \\
1 \quad 0 \quad 1 \quad 0 \quad 1 \quad 0 \quad 1 \\
1 \quad 2 \quad 3 \quad 4 \quad 5 \quad 6 \quad 7 \quad 8 \quad 9 \quad 10 \\
[1] & \quad R[(R=1+\text{VV})/\text{pR}+,A]+1+\text{VV} \\
1 \quad 0 \quad 1 \quad 0 \quad 1 \quad 0 \quad 1 \\
[1] & \quad R[(R=1+\text{VV})/\text{pR}+,A]+1+\text{VV} \\
1 \quad 3 \quad 5 \quad 7 \quad 9 \\
[1] & \quad R[(R=1+\text{VV})/\text{pR}+,A]+1+\text{VV} \\
\text{\texttt{-1}} & \quad R+(\text{pA})\text{pR} \\
\text{\texttt{-1 0 -1 0 -1 0 -1 0}} & \quad R+(\text{pA})\text{pR} \\
2 \quad 5 \\
[2] & \quad R+(\text{pA})\text{pR} \\
\text{\texttt{-1 0 -1 0 -1}} & \quad R+(\text{pA})\text{pR} \\
0 \quad -1 \quad 0 \quad -1 \quad 0 \\
\end{align*}

17
RIOTA MATRIX ROW IOTA

SYNTAX: R+X RIOTA Y

- RIOTA EXTENDS TO MATRIX ARGUMENTS THE FUNCTION OF DYADIC \( \{ A \cup B \} \) THE LEAST INDEX IN VECTOR A OF THE ELEMENTS(S) IN SCALAR OR VECTOR B.
- THE RESULT R IS A VECTOR OF THE RESPECTIVE ROW INDICES OF THE FIRST OCCURRENCE OF THE ROWS OF Y IN X, IGNORING TRAILING BLANKS. IF A ROW OF Y DOES NOT OCCUR IN X, THE CORRESPONDING ELEMENT OF R IS SET TO 1+1+PX. NON-MATRIX ARGUMENTS ARE RESHAPED. SCALAR AND VECTOR ARGUMENTS ARE TREATED AS 1-ROW MATRICES.
- USES: \( \nabla \) MATRIX \( \nabla \)

FUNCTION:

\[
\nabla \, R+X \text{ RIOTA } Y
\]

[1] \( Y \triangleleft \text{MATRIX } Y \) \( \star \) \( X \triangleleft \text{MATRIX } X \)
[2] \( R+\nabla \text{IO}++/\sim \nabla (((0 \, 1 \times pX) \mid pY) \mid Y) \equiv q((0 \, 1 \times pY) \mid pX) \mid X \)

EXAMPLE: \( \text{IO-1} \)

\(X+3 \, 4p'\text{AAAAABBBBCCCC'}\)
\(Y+2 \, 5p'\text{CCCC XXXXX'}\)
\(X \text{ RIOTA } Y\)

3 4

ANALYSIS:

THE LEFT ARGUMENT, X, IS TRANSPOSED AND BECOMES THE RIGHT ARGUMENT IN THE MATRIX INNER PRODUCT

CCCC \( \wedge . \ mim ABC\)
XXXXX \( \wedge . \ mim ABC\)
ABC
ABC
ABC

THE RIGHT ARGUMENT IN THAT EXPRESSION HAS A FIFTH (BLANK) ROW TO SATISFY THE INNER PRODUCT REQUIREMENT THAT THE LAST DIMENSION (5) OF THE LEFT ARGUMENT MUST BE THE SAME AS THE FIRST DIMENSION OF THE RIGHT ARGUMENT. EACH ROW OF THE LEFT ARGUMENT IS COMPARED AGAINST EACH COLUMN OF THE RIGHT ARGUMENT GIVING THE MATRIX

\( 0 \, 0 \, 1 \)
\( 0 \, 0 \, 0 \) WHICH IS TRANSLATED INTO THE VECTOR 2 3 BY \( +/\sim \nabla \). ADDING \( \text{IO} \) GIVES THE RESULT 3 4 (FOR ORIGIN 1) OR 2 3 (FOR ORIGIN 0).
SHAPE MATRIX FROM CHARACTER STRING

SYNTAX: 

\[ R+C \text{ SHAPE } X;L \]

- \( X \) IS A CHARACTER VECTOR COMPOSED OF PHRASES OF VARIABLE LENGTH, SEPARATED BY ANY OF THE CHARACTERS IN VECTOR C.
- A MEMBER OF C MAY EVEN BE PART OF A PHRASE, IF IT IS SURROUNDED BY QUOTES IN X.
- SEE VERECT

FUNCTION:

\[ \nabla R+C \text{ SHAPE } X;L \]

1. \( R^+((=X*'')\wedge X\in C)\backslash pX+X,1+C \)
2. \( L^+R=R-[X]+1+1+R \)
3. \( R+(0*R)\not R^-1+((pR),1+L)p,(R\geq (\neg [X]+[L]),1)\backslash X \)

\[ \nabla \]

EXAMPLES:

' 'SHAPE'TOM DICK HARRY'
TOM
DICK
HARRY
';','SHAPE'SEMICOLON'','';COMMA','','PERIOD'.''

SEMICOLON';'
COMMA','
PERIOD'.

ANALYSIS:

[1]A 'LOGICAL VECTOR SELECTS 12 21 31 AS END POINTS
[3] \( R+(0*R)\not R^-1+((pR),1+L)p,(R\geq (\neg [X]+[L]),1)\backslash X \)

1 2 3 4 5 6 7 8 9 10 11 12
[3] \( R+(0*R)\not R^-1+((pR),1+L)p,(R\geq (\neg [X]+[L]),1)\backslash X \)

1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 0 0
1 1 1 1 1 1 1 0 0 0
1 1 1 1 1 1 0 0 0 1
[3] \( R+(0*R)\not R^-1+((pR),1+L)p,(R\geq (\neg [X]+[L]),1)\backslash X \)

1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 0 0 0 1
1 1 1 1 1 1 1 1 0 0 0 1
[3] \( R+(0*R)\not R^-1+((pR),1+L)p,(R\geq (\neg [X]+[L]),1)\backslash X \)

SEMICOLON';';COMMA','','PERIOD'.'
[3] \( R+(0*R)\not R^-1+((pR),1+L)p,(R\geq (\neg [X]+[L]),1)\backslash X \)

3 13
[3] \( R+(0*R)\not R^-1+((pR),1+L)p,(R\geq (\neg [X]+[L]),1)\backslash X \)

SEMICOLON';'
COMMA','
PERIOD'.

(NOW STRIP SUPERFLUOUS PUNCTUATION)
ULINE  UNDERLINE SPECIFIED ROWS OF CHARACTER MATRIX [ USCORE ]

SYNTAX:  R+N ULINE K

- RETURNS AN EXPANDED MATRIX WITH U D E R S C O R E S INSERTED, AS
  INDICATED BY N, A VECTOR OF ROW NUMBERS \[IO+1].
- UNDERLINES WILL NOT APPEAR IN COLUMNS THAT ARE ALWAYS BLANK.
- GIVEN A CHARACTER VECTOR, USCORE IS MUCH FASTER THAN ULINE.
- USES: \$MATRIX \$ESCAPEX \$XVEC \$CVEC \$FILLS

FUNCTIONS:

\[ R+N \text{ ULINE } K;L;[IO;J;I ] \]

\[ K+\text{MATRIX } \& K \]

\[ ' ' \text{MATRIX HAS ROWS NUMBERED: } ' ', \text{J'ESCAPEX}\text{\&J}/N\text{\&J}+(\lceil IO+1 \rceil)+pK \]

\[ R+(-J,0)+R \text{FILLS}(-L)(\lceil (J+I>/N)+p,N),1+pK)\text{p}(\text{\&K})'\text{\&K}\text{\&K}'\text{\&K}'\text{\&K}'\text{\&K}'\text{\&K} \]

\[ R+\text{USCORE } KV \]

\[ [1] \ R+K,[[IO-0.5](KV'= '))' ' ' \]

EXAMPLES:

NAMES BESIDE SCORES

<table>
<thead>
<tr>
<th></th>
<th>4</th>
<th>60</th>
<th>56</th>
</tr>
</thead>
<tbody>
<tr>
<td>TOM</td>
<td>107</td>
<td>84</td>
<td>62</td>
</tr>
<tr>
<td>HARRY</td>
<td>18</td>
<td>64</td>
<td>90</td>
</tr>
</tbody>
</table>

\((-1+/SCORES)\)ULINE NAMES BESIDE SCORES

<table>
<thead>
<tr>
<th></th>
<th>4</th>
<th>60</th>
<th>56</th>
</tr>
</thead>
<tbody>
<tr>
<td>TOM</td>
<td>107</td>
<td>84</td>
<td>62</td>
</tr>
<tr>
<td>HARRY</td>
<td>18</td>
<td>64</td>
<td>90</td>
</tr>
</tbody>
</table>

THE WINNER AND HIS SCORES

\(\text{pSCORES} \]

\[ 3 3 \]

\[ 4 \text{ULINE SCORES} \]

MATRIX HAS ROWS NUMBERED: 1 2 3

\[ 3 \text{ULINE SCORES} \]

\[ 4 60 56 \]

\[ 107 84 62 \]

\[ 18 64 90 \]

ANALYSIS:

\[ [1] \text{FORCES 2=K} \]

\[ [2] \text{CHECKS FOR ILLEGAL ROW NUMBERS, PRINTS MESSAGE, THEN ESCAPES.} \]

\[ [3] \text{GENERATES EXPANSION VECTOR, THEN EXPANDS K.} \]

\[ [4] \text{REPLICATES UNDERLINES, IF NECESSARY, EXPANDS THEM, THEN MERGES THEM WITH THE RESULT OF [3], AND FINISHES WITH SOME HOUSEKEEPING, IF NECESSARY.} \]
VFORM

VARIABLE FORMAT BY ROW OF A MATRIX [ ESCAPE ESCAPEX ]

SYNTAX:

\[ R+F \text{ VFORM } M \]

- The APL format function (\( \downarrow \)) acts uniformly on all rows of a matrix while allowing variable widths and decimal places from column to column. VFORM permits a uniform width for all columns while allowing individual row decimal places. \( M \) is a numeric matrix. \( F \) is a numeric vector of the form \( W, D_1, N_1, D_2, N_2, \ldots \) where \( W \) is the format width for all columns. \( D_1 \) is the number of decimal places in the first block of rows; \( N_1 \) is the number of rows in the first block, etc. \( pF \) must be odd, and the sum of \( N \)'s must equal the number of rows in the matrix.
- Uses \( \text{VHANG} \), although \( \text{VESCAPE} \) may be substituted.

FUNCTIONS:

\[ \text{V R+F VFORM M; I; J; IO} \]

\[ \text{IO} = 0 \]

\[ '\text{NOT A MATRIX' HANG} \text{ 2}=pM \]

\[ '\text{WRONG LENGTH LEFT ARGUMENT' HANG} \to pF \]

\[ F+F[0], \left(\left(1+pF\right)/2\right) p1+F \]

\[ '\text{WRONG ROW COUNT' HANG} \left(1+pM\right) \neq +/J+F[;2] \]

\[ F+F[;0 I] \]

\[ R+(0,(1+pM) \times \left(1+pF\right)) p1+0 \]

\[ L1: R+R, \left[0\right] F[I;] \downarrow \left[M\left(+/I+J\right)+1J[I;] \right] \]

\[ \text{V} \]

\[ \text{V MSG ESCAPE CONDITION} \]

\[ \text{\( \to \) WILL LEAVE NO TRACE...BETTER TO} \]

\[ \text{\( \to \) USE VHANG TO CHECK DOMAIN ERRORS} \]

\[ \text{\( \to \) IF\text{-}CONDITION} \]

\[ \text{\( \to \) MSG} \]

\[ \text{\( \to \) V QEXP ESCAPEX CONDITION} \]

\[ \text{\( \to \) IF\text{-}CONDITION} \]

\[ \text{\( \to \) QEXP} \]

\[ \text{\( \to \) V} \]

\[ \text{\( \to \) WILL EXECUTE THE QUOTED} \]

\[ \text{\( \to \) EXPRESSION, THEN ESCAPE} \]

\[ \text{\( \downarrow \)} \]

EXAMPLES:

\[
\begin{array}{cccc}
C+5 & 4 & 120 \\
1 & 2 & 3 & 4 \\
5 & 6 & 7 & 8 \\
9 & 10 & 11 & 12 \\
13 & 14 & 15 & 16 \\
17 & 18 & 19 & 20 \\
8 & 1 & 1 & 0 & 2 & 2 & 2 \\
1.0 & 2.0 & 3.0 & 4.0 \\
5 & 6 & 7 & 8 \\
9 & 10 & 11 & 12 \\
13.00 & 14.00 & 15.00 & 16.00 \\
17.00 & 18.00 & 19.00 & 20.00 \\
\end{array}
\]

ANALYSIS:

In line 8, \( J[I] \) is the number of rows in the block being formatted. \( F[I;] \) is the uniform width and the number of decimal places for the block. Each block is formatted and concatenated to \( R \) in line 8 until the number of blocks equals the count \( I \) in line 9.
WIDTH MEASURE FORMATTED MATRIX

SYNTAX: W+K WIDTH MATRIX

- RETURNS THE ACTUAL WIDTH OF ALL FIELDS AS A VECTOR.
- BLANK AREAS ARE NOT CONSIDERED SIGNIFICANT.
- A GLOBAL LOGIC VECTOR CAPABLE OF COMPRESSING (THEN EXPANDING) THE FORMATTED MATRIX WILL BE NAMED ACCORDING TO THE CHARACTER(S) OFFERED AS K.
- USES: VDMZ

FUNCTION:

\[
\begin{align*}
\text{\texttt{V W+K WIDTH MATRIX;V;IO}} \\
[1] \left[ \begin{array}{c}
W^-1++V.^=1+V++(\text{IO}+1),~\text{DMZ+K},'+'\text{MATRIXz'} \end{array} \right]' \\
\text{\texttt{V}}
\end{align*}
\]

EXAMPLES:

'B'WIDTH 9 2\text{\texttt{V}}\text{MM}
7554

\[
B
\begin{array}{cccccc}
0 & 0 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 \\
\end{array}
\]

92\text{\texttt{V}}\text{MM}

\[
\begin{array}{cccc}
1000.00 & 87.92 & 79.58 & 8.33 \\
920.42 & 87.92 & 80.25 & 7.67 \\
840.17 & 87.92 & 80.91 & 7.00 \\
\end{array}
\]

B/92\text{\texttt{V}}\text{MM}

\[
\begin{array}{cccc}
1000.0087.9279.588.33 \\
920.4287.9280.257.67 \\
840.1787.9280.917.00 \\
\end{array}
\]
SELECT NTH WORD IN CHARACTER STRUCTURE

SYNTAX:
W+NTH WORD K

* IF WORDS IN A VECTOR ARE DELIMITED BY BLANKS, OR APPEAR IN SEPARATE ROWS OF AN ARRAY AND ARE SIMILARLY DELIMITED, THEY CAN BE CHOSEN BY THE NATURAL NUMBERS INDICATING THEIR POSITION, FROM LEFT TO RIGHT AND TOP TO BOTTOM.

FUNCTIONS:

\[ W+NTH \text{ WORD } K \]
\[ W+1+(NTH=+\ ')=W)/W+DTMB,' ',K \]
\[ \text{delete trailing and multiple blanks} \]

EXAMPLE:

2 WORD \[Kristen\]

TOM DICK HARRY

ANALYSIS:

[1] \[W+1+(NTH=+/')=W)/W+DTMB,' ',K\] INITIAL BLANK

TOM DICK HARRY

[1] \[W+1+(NTH=+/')=W)/W+DTMB,' ',K\] TRAILING AND MULTIPLE BLANKS OUT

TOM DICK HARRY

[1] \[W+1+(NTH=+/')=W)/W+DTMB,' ',K\] LOCATION OF ONLY BLANKS

1 0 0 1 0 0 0 1 0 0 0 0

[1] \[W+1+(NTH=+/')=W)/W+DTMB,' ',K\] FIELDS NUMBERED

1 1 1 1 2 2 2 2 2 3 3 3 3

[1] \[W+1+(NTH=+/')=W)/W+DTMB,' ',K\] COMPARED

0 0 0 1 1 1 1 0 0 0 0 0

[1] \[W+1+(NTH=+/')=W)/W+DTMB,' ',K\] SELECTION

DICK

[1] \[W+1+(NTH=+/')=W)/W+DTMB,' ',K\] BLANK FIELD-MARKER DROPPED
ADJUSTDOWN  EXTEND THE ']' IN REPORT FORMATTING [ ROWINDICES ]

SYNTAX:  Z=A ADJUSTDOWN B

- EXTENDS TO ROW A, THE CHARACTER ']' USED AS A SEPARATOR
  IN MATRIX B
- USES: VIF VINDEX VROWINDICES VΔ

FUNCTIONS:

  V Z=A ADJUSTDOWN B;C;D; [IO

  [1] (G2|C+1+pD+B(-(|A|-1)ROWINDICES Z+B;))/0 Δ [IO=1
  [2] -L1 IF C≤3
  [3] D+(D)[I3;]

  V R=N ROWINDICES M;[IO

  [1] R FIRST OR LAST N ROWNUMBERS OF MATRIX
  [2] R+(R=N)/R=N+1+pM Δ [IO=1

  V

  E

  SALES

  -----  
  1 2 3 4|ABCDE
  5 6 7 8|FGHIJ
  9 10 11 12|KLMNO


  -----  

  7 ADJUSTDOWN E

  SALES

  -----  
  1 2 3 4|ABCDE
  5 6 7 8|FGHIJ
  9 10 11 12|KLMNO


  -----  

ANALYSIS:

  a LINE 1 PICKS OUT AND STORES IN D THAT PART OF MATRIX B WHOSE
  a ROW NUMBERS ARE LESS THAN |A|
  a LINES 2 AND 3 SELECT THE LAST 3 LINES OF D, WHERE LINE 4 LOCATES
  a THE COMMON OCCURRENCE OF THE SEPARATOR ']' AND EXTENDS
  a THEM 1 LINE DOWNWARD.
Section II

Report Formatting Functions
ADJUSTUP     EXTENDS '|' IN REPORT FORMATTING

SYNTAX:        Z+A ADJUSTUP B

- EXTENDS SEPARATOR '|' UP ONE LINE FROM ROW A
  IN MATRIX B.
- USES: VIF VINDEX VROWINDICES V+4

FUNCTION:

\[
\text{v Z+A ADJUSTUP B;C;D;\text{\texttt{IO}}}
\]

- \[1\] \(0 \geq C+1+pD+B[(\mid A\mid-1+pB)\text{\texttt{ROWINDICES}} Z+B;]A \text{\texttt{IO}}+1\)
- \[2\] \(L1 \text{\texttt{IF} } C \leq 3\)
- \[3\] \(D+D[\text{\texttt{3;}}]\)
- \[4\] \(L1:Z[A;\mid \text{\texttt{INDEX}} D]+\mid \text{\texttt{'}|'}\)

EXAMPLES:

\[
D
\]

\[
\begin{array}{cccc}
1 & 2 & 3 & 4 |ABCDE \\
5 & 6 & 7 & 8 |FGHI \\
9 & 10 & 11 & 12 |KLMNO \\
\end{array}
\]

\[
3 \text{ ADJUSTUP D}
\]

\[
\begin{array}{cccc}
1 & 2 & 3 & 4 |ABCDE \\
5 & 6 & 7 & 8 |FGHI \\
9 & 10 & 11 & 12 |KLMNO \\
\end{array}
\]

\[
1 \text{ ADJUSTUP 2 ADJUSTUP 3 ADJUSTUP D}
\]

\[
\begin{array}{cccc}
1 & 2 & 3 & 4 |ABCDE \\
5 & 6 & 7 & 8 |FGHI \\
9 & 10 & 11 & 12 |KLMNO \\
\end{array}
\]

ANALYSIS:
- LINE 1 PICKS OUT AND STORES IN D THAT PART OF THE ARRAY B WHOSE
  ROW NUMBERS ARE GREATER THAN |A.
- LINES 2 AND 3 SELECT THE NEXT 3 (OR FEWER IF THERE AREN'T 3) LINES,
  WHILE LINE 4 LOCATES ON THOSE LINES OCCURRENCES OF THE SEPARATOR '|' AND EXTENDS THEM UPWARD 1 LINE.
BARGRAPH

PLOT HORIZONTAL INTEGER BARGRAPHS

SYNTAX:

\( R+Q \) BARGRAPH \( V \)

- PRODUCE HORIZONTAL HISTOGRAMS OR GANTT CHARTS
- TWO CLASSES OF INPUT (CODED PLUS AND MINUS) TREATED, ONE INVISIBLE WHILE THE OTHER IS PLOTTED.
- THE CHARACTERS USED FOR THE BARS ARE USER-SPECIFIED.
- THE OUTPUT CAN BE CATENATED TO NAMES, FOR EXAMPLE.
- USES: \( V \Delta \)

FUNCTION:

\[ V \] \( R+Q \) BARGRAPH \( V; \Delta IO \)

[1] \( R \) IS CHARACTER TO BE USED FOR BARS
[2] \( V \) IS A VECTOR OF POSITIVE AND NEGATIVE INTEGERS
[3] \( V \) NEGATIVE INTEGERS STORED IN \( V \) WILL BE IGNORED
[4] \( R \) INVISIBLE WHILE \( V \) PLOTTED
[5] \( R+(\{2tQ\})[V_0+1]+1r/V\Delta \Delta IO+0 \)

ANALYSIS:

'\( \) BARGRAPH 1 -2 3 -4 5

\[ 1 \] \( R+(\{2tQ\})[V_0+1]+1r/V\Delta \Delta IO+0 \)

\[ 1 \] -2 3 -4 5

\[ 1 \] \( R+(\{2tQ\})[V_0+1]+1r/V\Delta \Delta IO+0 \)

5

\[ 1 \] 2 3 4 5

\[ 5 \] \( R+(\{2tQ\})[V_0+1]+1r/V\Delta \Delta IO+0 \)

WIDTH DETERMINED

NEGATIVES IGNORED

[5] \( R+(\{2tQ\})[V_0+1]+1r/V\Delta \Delta IO+0 \)

BLANK, \( Q \)

SHAPE OF INDEXING

MATRIX DETERMINES

SHAPE OF OUTPUT.

EXAMPLE:

FRAME('SHAPE'TOM JANE DICK MARY')BESIDE('O'BARGRAPH-V)FILLS'O'BARGRAPH V

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>TOM</td>
<td>( Q )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>JANE</td>
<td>( \circ )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DICK</td>
<td>( \bullet \bullet )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MARY</td>
<td>( \circ \circ \circ \circ \circ )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----</td>
<td>-----</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
BESIDE PRESENTS TWO STRUCTURES SIDE BY SIDE IN REPORT FORMAT

SYNTAX: Z+A BESIDE B

- CAN FORMAT A REPORT WITH ROW HEADINGS AND A SEPARATOR, OR SIMPLY JOIN TWO DISPARATE MATRICES. SEE VCCAT
- USES: VPREPARE VCOMPARE VIF VADDDROWS

FUNCTION:

\[ \n\begin{align*}
\n\text{\texttt{\textbackslash{}v \ Z+A BESIDE B; I; J}} \\
\text{\texttt{\textbackslash{}1 I+1\texttt{p}A\texttt{+PREPARE A}}} \\
\text{\texttt{\textbackslash{}2 J+1\texttt{p}B\texttt{+PREPARE B}}} \\
\text{\texttt{\textbackslash{}3 \texttt{\textbackslash{}if} (L1, L2, L0) IF I \texttt{COMPARE J}}} \\
\text{\texttt{\textbackslash{}4 L0\texttt{+L2 \& (I-J)ADDDROWS B}}} \\
\text{\texttt{\textbackslash{}5 L1\texttt{+A+(J-I)ADDDROWS A}}} \\
\text{\texttt{\textbackslash{}6 L2\texttt{+Z+A,'|',B}}} \\
\n\end{align*} \n\]

EXAMPLES:

\[ \text{\texttt{\textbackslash{}v+LM+\textquoteright{}SHAPE 'SIMPSON GERONIMO JONES LEGRAND'}} \]

SIMPSON
GERONIMO
JONES
LEGRAND

\[ \text{\begin{array}{cccc}
\text{AB} & 3 & 4 & 5 & 6 \\
\text{LM} & 589 & 931 & 847 & 527 & 92 \\
\text{NM} & 654 & 416 & 702 & 911 & 763 \\
\text{AB} & 263 & 48 & 737 & 329 & 633 \\
\text{LEGRAND} & 757 & 992 & 366 & 248 & 983 \\
\text{LM} & 589 & 931 & 847 & 527 & 92 \\
\text{NM} & 654 & 416 & 702 & 911 & 763 \\
\text{AB} & 263 & 48 & 737 & 329 & 633 \\
\text{LEGRAND} & 757 & 992 & 366 & 248 & 983 \\
\end{array}} \]

ANALYSIS:

- LINES 1 AND 2 PREPARE THE ARGUMENTS FOR SIDE BY SIDE PLACEMENT.
- NUMERIC ARGUMENTS ARE FORMATTED WITH AUTOMATIC WIDTH AND
- NO DECIMAL POSITIONS IF INTEGER, 2 DECIMAL POSITIONS OTHERWISE.
- ANY FRAMING ALREADY PART OF A CHARACTER ARGUMENT IS REMOVED.
- VECTOR OR SCALAR ARGUMENTS ARE CONVERTED INTO ONE-ROW MATRICES.
- LINES 3, 4, 5 CHECK THE NUMBER OF ROWS IN BOTH ARGUMENTS AND ADD THE
- APPROPRIATE BLANK ROWS TO PAD OUT THE SMALLER.


**BLANK**

DELETE SPECIFIC STRING FROM STRUCTURE [ LIM ]

**SINTAX:**

\[ R \rightarrow \text{STR BLANK A} \]

- **IF A REPORT, OR ANY STRUCTURE, CONTAINS UNWANTED ITEMS, NUMERIC OR CHARACTER STRINGS, THEN EVERY APPEARANCE OF THE SPECIFIED STRING WILL BE REPLACED BY BLANKS, OR BY ZERO, IF A IS A NUMERIC STRUCTURE.**
- **USES VLOC V\text{LIM} \text{\&} A**

**FUNCTIONS:**

\[ \n R \rightarrow \text{STR BLANK A} \]

1. IF STR APPEARS IN A IT WILL BECOME BLANK(OR 0)
2. \[ R[(x/\rho A)\text{LIM(STR LOC } R)_{0.}+\text{\&}, \text{STR } \Delta [IO+0]+0'\rho A \Delta R,A} \]
3. \[ R+(\rho A)_{\rho R} \]

\n
**EXAMPLES:**

\[ B+\n+2\n+2 \mid 6 \]

0.00 1.00 0.00 1.00 0.00 1.00
'0.00'BLANK B
1.00 1.00 1.00
\[ B+\n+3 \mid 4 \rho 17 \]

0 1 2 3
4 5 6 0
1 2 3 4
0 1 2 BLANK B
0 0 0 3
4 5 6 0
0 0 3 4

**ANALYSIS:**

'AAA'BLANK'AAABBBAAA'

1. \[ R[(x/\rho A)\text{LIM(STR LOC } R)_{0.}+\text{\&}, \text{STR } \Delta [IO+0]+0'\rho A \Delta R,A} \]

AAA
2. \[ R[(x/\rho A)\text{LIM(STR LOC } R)_{0.}+\text{\&}, \text{STR } \Delta [IO+0]+0'\rho A \Delta R,A} \]

0 1 2
2. \[ R[(x/\rho A)\text{LIM(STR LOC } R)_{0.}+\text{\&}, \text{STR } \Delta [IO+0]+0'\rho A \Delta R,A} \]

AAABBBAAA
2. \[ R[(x/\rho A)\text{LIM(STR LOC } R)_{0.}+\text{\&}, \text{STR } \Delta [IO+0]+0'\rho A \Delta R,A} \]

0 6 7 8
2. \[ R[(x/\rho A)\text{LIM(STR LOC } R)_{0.}+\text{\&}, \text{STR } \Delta [IO+0]+0'\rho A \Delta R,A} \]

0 1 2
6 7 8
7 8 9
8 9 10
2. \[ R[(x/\rho A)\text{LIM(STR LOC } R)_{0.}+\text{\&}, \text{STR } \Delta [IO+0]+0'\rho A \Delta R,A} \]

0 1 2 6 7 8 7 8 8
(FALSE INDICES DELETED)
CENTERON CENTERS AND CATENATES TWO STRUCTURES [ CENTER ]

SYNTAX: Z+A CENTERON B

- TAKES TWO CHARACTER ARRAYS OF ANY SIZE, CENTERS THE ONE WITH FEWER COLUMNS, THEN CATENATES A ABOVE B.
- USES: VIF VCOMPARE VADDCOLS VCENTER VON VPREPARE

FUNCTIONS:

V Z+A CENTERON B;I
[1] A=\( A \) A B=\( B \)
[2] (L1,L3,L2)IF 0 COMPARE I+(1+pA)\(-1+pB
[3] L1:=L3 A B CENTER I ADDCOLS PREPARE B
[4] L2:A+CENTER(-I)ADDCOLS PREPARE A
[5] L3:Z+A ON B

V Z+CENTER B;C
[1] C+NUMBLANKCOLS Z+B
[2] Z+\((0.5/-C)\)Z

EXAMPLES:

LV='THIS IS A LITERAL VECTOR'
LV
THIS IS A LITERAL VECTOR

LV CENTERON 'HELLO'
THIS IS A LITERAL VECTOR
HELLO

'SALES' CENTERON 'REPORT FOR OCTOBER'
SALES
REPORT FOR OCTOBER

ANALYSIS:

A LINE 1 FORCES CHARACTER REPRESENTATION.
A LINE 2 CHECKS THE NUMBER OF COLUMNS IN THE ARGUMENTS.
A IF EQUAL, LINE 5 USES VON TO CATENATE THEM.
A IF UNEQUAL, THE SMALLER IS PADDED OUT ON THE LEFT TO THE WIDTH OF THE LARGER WITH VADDCOLS, THEN USES VCENTER TO SPLIT THE NUMBER OF BLANK COLUMNS, PUTTING HALF ON THE RIGHT, BEFORE CATENATING.
A VPREPARE MAKES ONE-ROW MATRICES OF VECTOR AND SCALAR ARGUMENTS.
CITED EXTRACT CITED STRINGS FROM CHARACTER ARRAYS

SYNTAX: R+KV CITED A

- STRINGS OF NON-BLANK INFORMATION, DEMARKED BY THE CHARACTERS PROFFERED AS KV, WILL BE EXTRACTED, SHAPED AS A MATRIX, AND SORTED.
- USES: VSHAPE VGRADEUP (AND GLOBAL VARIABLE "AV")

FUNCTION:

\[
\begin{align*}
\text{\texttt{R+KV CITED A}} \\
\text{\texttt{[1] R+ 'SHAPE(\sim R\in KV) / R+(\sim \backslash R\in KV) / R+, A}} \\
\text{\texttt{[2] R+R[AV GRADEUP R;]}} \\
\end{align*}
\]

EXAMPLES:

'[]' CITED 'TOM [DICK] HARRY'

DICK

'\leftrightarrow' CITED 'TOM \leftrightarrow HARRY DICK \leftrightarrow JANE'

HARRY

ANALYSIS:

'[]' CITED 'TOM [DICK JANE] HARRY'

\[
\begin{align*}
\text{\texttt{[1] R+ 'SHAPE(\sim R\in KV) / R+(\sim \backslash R\in KV) / R+, A}} \\
\end{align*}
\]

TOM [DICK JANE] HARRY

\[
\begin{align*}
\text{\texttt{[1] R+ 'SHAPE(\sim R\in KV) / R+(\sim \backslash R\in KV) / R+, A}} \\
\end{align*}
\]

OR ANY CHARACTER NOT IN TEXT

\[
\begin{align*}
\text{\texttt{[0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0} \\
\text{\texttt{[1] R+ 'SHAPE(\sim R\in KV) / R+(\sim \backslash R\in KV) / R+, A}} \\
\end{align*}
\]

EXTRACTED

\[
\begin{align*}
\text{\texttt{[DICK JANE}} \\
\text{\texttt{[1] R+ 'SHAPE(\sim R\in KV) / R+(\sim \backslash R\in KV) / R+, A}} \\
\end{align*}
\]

CLEANED AND SHAPED,

DICK

JANE

30
COLLECT COLLECT AND SUMMARIZE COEFFICIENTS OF COMMON CODES

SYNTAX:

R+VV COLLECT CV

- VV AND CV ARE A PAIR OF NUMERIC VECTORS TO WHICH WE HAD BEEN CATENATING PAIRS OF CORRESPONDING DATA.
- WE DESIRE TO REDUCE THE DATA BY COLLECTING AND SUMMING THOSE TERMS IN VV THAT ARE IN THE SAME CATEGORY, AS REPRESENTED BY REPETITION IN CV, THE CODE VECTOR.
- THE CODE VECTOR, ALTHOUGH NUMERIC, MAY REPRESENT ALPHABETIC CHARACTERS. SEE VJNFORM IN KWIC INDEX.
- USES VDREP.

FUNCTION:

\[ V R+VV \text{ COLLECT } CV; T \]

\[ R+T, [RIO+0.5]((T+DREP CV)\cdot=CV)+.xVV \]

EXAMPLE:

\[ 0+I+=?1p6 \]
\[ 3 1 4 5 2 3 1 4 5 2 3 \]  
(A RANDOM PATTERN OF OCCURRENCE)

\[ 0+VV+=11?99 \]
\[ 98 99 23 27 68 58 32 93 79 17 29 \]  
(TALLIES, CORRESPONDING TO CAR-CODES)

(AV KFORM, 0 -1+R), \( V0 1+R+VV \) COLLECT AV LJNFORM CARS[I;]

PONTIAC 185
BUICK 131
CHEVROLET 116  
(RE-TRANSLATED, FOR REPORTING PURPOSE)
CADILLAC 106
OLDSMOBILE 85

ANALYSIS:

\( VV \) COLLECT \( CV \)-?1p10

\[ R+T, [RIO+0.5]((T+DREP CV)\cdot=CV)+.xVV \]

VALUES

7 16 47 90 89 58 30 9 24 74 35

CODES

10 11 6 8 1 5 1 5 8 7

REPLICATES DELETED

10 6 8 5 7

HITS LOCATED

1 0 0 0 0 0 0 0 0 0 0
0 1 1 0 0 1 0 1 0 0 0
0 0 0 1 0 0 0 0 0 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 0 0 0 0 0 0 0 1

REDUCED AND LAMINATED
DREP

SELECT UNIQUE ELEMENTS FROM ANY STRUCTURE

SYNTAX:

R+DREP V

USEFUL IN PREPARING SPECIAL COLLATING SEQUENCES. RESULT CONTAINS SINGLE APPEARANCE OF EACH ELEMENT, NOT REARRANGED.

FUNCTION:

\[ \forall R+DREP V \]

\[ R+(V\setminus V)=\top V/\land \top V \]

\[ \forall \]

EXAMPLE:

DREP 1 2 3 4 5 4 3 2 1
1 2 3 4 5

ANALYSIS:

DREP 'MISSISSIPPI'

\[ R+(V\setminus V)=\top V/\land \top V \]

MISSISSIPPI

\[ R+(V\setminus V)=\top V/\land \top V \]

MISSISSIPPI

\[ R+(V\setminus V)=\top V/\land \top V \]

11

\[ R+(V\setminus V)=\top V/\land \top V \]

1 2 3 4 5 6 7 8 9 10 11

\[ R+(V\setminus V)=\top V/\land \top V \]

MISSISSIPPI

\[ R+(V\setminus V)=\top V/\land \top V \]

1 2 3 3 2 3 3 2 9 9 2

\[ R+(V\setminus V)=\top V/\land \top V \]

1 1 1 0 0 0 0 0 1 0 0

\[ R+(V\setminus V)=\top V/\land \top V \]

MISP
HEADERON PUTS A HEADING ON A REPORT [ COMPARE ]

SYNTAX: Z+A HEADERON B

* CENTERS A CHARACTER HEADING (LEFT ARG.) ON A REPORT
   (RIGHT ARG.), WITH AN EXTRA ROW OF BLANKS AND A
   SEPARATOR. HEADING MAY BE SCALAR, VECTOR OR MATRIX.
* USES: VPREPARE VIF VCENTER VADDCOLS VADDRROWS VADJUSTUP
   VADJUSTDOWN VCOMPARE V\Delta

FUNCTIONS:

\[ Z+A \text{ HEADERON B}; I; J; IO \]
\[ i-1+pA+\text{PREPARE A} \Delta \text{IO}+1 \]
\[ j-1+pB+\text{PREPARE B} \]
\[ \left( L_1, L_2, L_0 \right) \text{IF I COMPARE J} \]
\[ L_0:=L_2, pB+\text{CENTER}(J-I) \text{ADDCOLS B} \]
\[ L_1:=A+\text{CENTER}(I-J) \text{ADDCOLS A} \]
\[ L_2:=Z+A, \left[ 1 \right]'_|', \left[ 1 \right]1 \text{ADDRROWS B} \]
\[ Z+(2+1+pA)\text{ADJUSTUP}(1+1+pA)\text{ADJUSTDOWN} Z \]

EXAMPLE:

'SALES' HEADERON ?4 5p100

SALES
-----------------
  52 32 99 50 27
  10 95 8 51 39
  28 92 53 47 95
  6 77 78 83 13
(2 8p'DECEMBER REPORT ' ) HEADERON ?4 5p1000
  DECEMBER
  REPORT
-----------------
  16 689 869 630 737
  726 1000 889 234 307
  352 514 592 846 413
  842 270 416 538 468
* LINES 1 AND 2 PREPARE THE HEADING BY CONVERTING SCALARS AND VECTORS
  TO ONE-ROW MATRICES AND REMOVING ANY PREEXISTING FRAMING ELEMENTS.
  THEY ALSO FORMAT THE RIGHT ARGUMENT.
* LINES 3, 4, 5 CHECK THE WIDTHS OF THE ARGUMENTS AND CENTER THE HEADING
  LINE 6 ADDS THE SEPARATOR '|_' AND AN EXTRA BLANK ROW.
* LINE 7 ADDS THE FRAMING ELEMENT '|' WHERE NEEDED IN THE EXTRA BLANK
  ROW TO PRETTY UP THE REPORT. MOST OF THE TIME IT WON'T BE NEEDED.
LJUST

LEFT JUSTIFY ANY ARRAY
[ DLB RJUST DL ]

SYNTAX:

R+V LJUST A

- TO SHIFT SIGNIFICANT CHARACTERS OR NUMERIC VALUES TO
  LEFTMOST POSITIONS.
- INSIGNIFICANT VALUES OR CHARACTERS ARE DEFINED IN V.
  THEY CAN BE TRUNCATED BY DT. (SEE DLB, BELOW)
- RIGHT JUSTIFICATION IS THE REVERSAL OF LEFT JUSTIFICATION.

FUNCTIONS:

† R+V LJUST A
[1] R+(+/\AeV)FA

† R+V RJUST A
[1] R+FA LJUSTFA

EXAMPLE:

' LJUST -3 5p' TOM DICK HARRY'

TOM
DICK
HARRY

ANALYSIS:  ' *?LJUST A

[1] R+(+/\AeV)FA

***TOM
**DICK
*HARRY

?BETTY
??MARY
?JANE

[1] R+(+/\AeV)FA

??*
[1] R+(+/\AeV)FA

1 1 1 0 0 0
1 1 0 0 0 0
1 0 0 0 0 0

1 0 0 0 0 0
1 1 0 0 0 0
1 1 0 0 0 0
[1] R+(+/\AeV)FA

3 2 1
1 2 2
[1] R+(+/\AeV)FA

TOM***
DICK**
HARRY*

BETTY?
MARY??
JANE
ON

CONFORM AND CATENATE ANY STRUCTURES

SYNTAX: \( Z+A \) ON \( B \)

- NAMES OR NUMBERS MAY BE ADDED TO LISTS OF ANY SHAPE OR CHARACTER, AT EITHER END. THE OUTPUT IS A MATRIX.
- NUMERIC MATRICES WILL BE PADDED WITH ZEROS IF THEY REMAIN NUMERIC.
- CHARACTER MATRICES WILL BE PADDED WITH BLANKS.
- ANY OPERAND MAY HAVE ANY STRUCTURE.
- USE VCENTERON IF LEFT-JUSTIFICATION IS NOT DESIRED.
- USES: VCFORMAT VMATRIX

FUNCTION:

\[
\begin{align*}
\& V \ Z+A \ ON \ B;DIO \\
[1] & \text{CFORMAT} \\
[2] & Z+(1+pA+\text{MATRIX } A)[1+pB+\text{MATRIX } B \\
[3] & Z+((1+pA),Z)+A,[DIO+0](1+pB),Z)+B \\
[4] & Z+Z'(A+Z)\{(2\times0+p,B)+0+p,A\} \\
[5] & a \text{ IN CASE OPERAND WAS EMPTY} \\
\end{align*}
\]

EXAMPLES:

'4' ON '4
1 2 3 4

FRAME 'HEADING'CENTERON 3 4p12
<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>HEADING</td>
</tr>
<tr>
<td>0 1 2 3 4 5 6 7</td>
</tr>
<tr>
<td>----------------</td>
</tr>
</tbody>
</table>

' HEADING' ON 3 3p 9
HEADING
0 1 2
3 4 5
6 7 8

(BLANKS COUNT AS CHARACTERS)
OUTPUT CENTER HEADINGS OVER FORMATTED COLUMNS [ CNTR DMZ NEXTA ]

SYNTAX: MAT+HEADINGS OUTPUT MATRIX

- HEADINGS, A CHARACTER VECTOR WITH BLANK DELIMITERS, WILL BE CENTERED OVER THE APPROPRIATE COLUMNS, AND UNDERSCORED TO THE FULL WIDTH OF THOSE COLUMNS IN THE PREFORMATTED MATRIX.
- ANY HEADING TOO WIDE FOR ITS COLUMNS WILL BE NOTED.
- AN INSUFFICIENT NUMBER OF HEADINGS WILL CAUSE LENGTH ERROR.
- ONLY THE LEFTMOST HEADINGS WILL BE USED.
- USES: VDMZ VNEXTA VCNTR VWIDTH VDLB
- GENERATES A GLOBAL VARIABLE, V, WHICH CAN BE USED TO REDUCE (THEN EXPAND) THE RESULT BY ELIMINATING BLANK COLUMNS.

FUNCTIONS:

\[ \begin{align*}
\text{MAT+HEADINGS OUTPUT MATRIX} &= S; I; A; V; W; H; D \text{IO} \\
[1] &\quad \text{MAT+} (1+p\text{MATRIX}) p' ' S + (I+D \text{IO}+1) p A + \text{HEADINGS} \\
[2] &\quad W + ' B' \text{WIDTH MATRIX} \\
[3] &\quad \text{BACK} : \rightarrow (I+pW) V 0 + pH + \text{NEXTA}) / \text{FINAL} \\
[4] &\quad \rightarrow (0 \neq pA \ A S + S, A + W[I+I+1] \text{CNTR} H) / \text{BACK} \\
[5] &\quad \rightarrow 0, p[1]' \text{ 'COLUMN WIDTH (', (W), ') TOO NARROW FOR '}, H \\
[7] &\quad \text{MAT} + (\text{MAT}, [0.1] E \text{ '} ) [1] \text{MATRIX} \\
\end{align*} \]

EXAMPLES:

\[ \begin{align*}
\text{A} &\quad \text{(SEE THE EXAMPLE FOR VAMORTIZE)} \\
\text{'TOM DICK HARRY } &\quad \text{º 'OUTPUT 9 2\text{vMM}} \\
\text{---} &\quad \text{---} &\quad \text{---} &\quad \text{---} &\quad \text{---} \\
1.00 &\quad 1000.00 &\quad 87.92 &\quad 79.58 &\quad 8.33 \\
2.00 &\quad 920.42 &\quad 87.92 &\quad 80.25 &\quad 7.67 \\
3.00 &\quad 840.17 &\quad 87.92 &\quad 80.91 &\quad 7.00 \\
\text{'ABERCROMBIE DICK HARRY } &\quad \text{º 'OUTPUT 9 2\text{vMM}} \\
\text{COLUMN WIDTH (4 7 5 5 4) TOO NARROW FOR ABERCROMBIE} \\
\text{A} &\quad \text{↑ __ OPPENDER} \\
\end{align*} \]
PAD

PADS ARRAYS WITH BLANKS OR ZEROS

SYNTAX:

Z+P PAD X

- PADS ARRAY X WITH BLANKS (IF LITERAL) OR ZEROS (IF NUMERIC). X CAN BE AN ARRAY OF ANY SHAPE AND TYPE.
- P IS A NUMERIC VECTOR SPECIFYING THE AMOUNT OFPadding IN EACH DIMENSION OF X (OR THE LAST OP).

- THE SENSE OF Padding (RIGHT OR LEFT), (BOTTOM OR TOP), ETC., IS DETERMINED BY THE SIGNS OF THE ELEMENTS OF P JUST AS WITH THE TAKE FUNCTION (+). IF P IS NOT LONG ENOUGH TO MATCH THE RANK OF X, Padding IS DONE ONLY FOR THE LAST OP DIMENSIONS.
- THIS FUNCTION IS UNLIKE THE TAKE FUNCTION IN THAT THE CHANGE IN SIZE IS SPECIFIED AND IT DOES NOT REQUIRE DETAILED KNOWLEDGE OF THE DIMENSIONS OF X.

FUNCTION:

\[ \text{\texttt{\(v \text{ } Z+P \text{ } \text{PAD} \text{ } X\)}} \]

\[ \text{\texttt{\(Z+((Z=0)+\times Z+P)\times(Z+1,pX)+(Z=-1[pP]+P)+X\)}} \]

\[ \text{\texttt{\(v\)}} \]

EXAMPLES:

| 3 PAD 4 | \(\text{-}1\) \(-\text{20} \text{ } \text{PAD} \text{ } 3 \text{ } 3p'\text{ABC}'\) |
| 4 0 0 0 | \(\text{ABC} \) |
| 0 0 0 0 | \(\text{ABC} \) |
| 0 0 0 0 | \(\text{ABC} \) |
| -8 PAD 'ABCD' | \(\text{ABC} \) |
| \(\text{ABCD} \) | \(\text{ABC} \) |
| \(\text{-3} \text{ } \text{PAD} \text{ } 2 \text{ } 4p18 \) | \(\text{1} \text{ } \text{2} \text{ } \text{PAD} \text{ } 2 \text{ } 2 \text{ } 4p16 \) |
| 0 0 0 0 1 2 3 4 | 1 2 3 4 0 0 |
| 0 0 0 0 5 6 7 8 | 5 6 7 8 0 0 |
| 0 0 0 0 0 0 0 0 | 0 0 0 0 0 0 |
| \(\text{1} \text{ } \text{-3} \text{ } \text{PAD} \text{ } 2 \text{ } 4p18 \) | \(\text{1} \text{ } \text{2} \text{ } \text{PAD} \text{ } 2 \text{ } 2 \text{ } 4p16 \) |
| 0 0 0 0 1 2 3 4 | 9 10 11 12 0 0 |
| 0 0 0 0 5 6 7 8 | 13 14 15 16 0 0 |
| 0 0 0 0 0 0 0 0 | 0 0 0 0 0 0 |

ANALYSIS:

\[ Z+((Z=0)+\times Z+P)\times(Z+1,pX)+(Z=-1[pP]+P)+X \]

| \(\text{ABSOLUTE VALUE OF Padding}\) |
| \(\text{DIMENSION VECTOR OF } X\) |

\(\text{SIGN OF Padding}\)
PREPARE

STANDARDIZE STRUCTURE FOR REPORT FORMATTING [ IF ]

SYNTAX:

Z-PREPARE A

- A SUBROUTINE THAT NORMALIZES DATA FOR THE FORMATTING
FUNCTIONS: ▽BESIDE ▽CENTERON ▽HEADERON
- USES: ▽CHARACTER ▽FRAMETEST ▽MATRIX ▽TABULATE ▽IF

FUNCTIONS:

▽ Z-PREPARE A
[ 1 ] →L1 IF ~CHARACTER A
[ 2 ] →0 IF 0=FRAMETEST Z+MATRIX A
[ 3 ] →0, pZ+1 1+~1+1+Z
[ 4 ] L1: Z-TABULATE A
▽

EXAMPLES:

X
ABCD
ABCD
ABCD  
PREPARE X
ABCD
ABCD
ABCD  
pPREPARE X
3 4
<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>SALES</td>
</tr>
<tr>
<td>-----</td>
</tr>
</tbody>
</table>
PREPARE Y
SALES
Z
1 2 3 4
5 6 7 8
9 10 11 12
PREPARE Z
1 2 3 4
5 6 7 8
9 10 11 12  
pPREPARE Z
3 12

ANALYSIS:

- LINE 1 BRANCHES TO LINE 4 IF THE ARGUMENT IS NUMERIC. 'TABULATE'
- THEN FORMATS IT.
- LINES 2 AND 3 TAKE THE ARGUMENT, WHICH IS MADE INTO A MATRIX, AND
- EXAMINE IT FOR FRAMING ELEMENTS ON THE PERIPHERY OF THE ARRAY.
- THESE ARE REMOVED IF PRESENT.
TABULATE NUMERIC STRUCTURES IN CONTROLLED FORMAT [ HANG ]

SYNTAX:  Z+TABULATE A

- A MUST BE A NUMERIC STRUCTURE. TABULATE WILL DIAGNOSE THE DOMAIN ERROR IF IT IS NOT. THE STATE INDICATOR WILL BE PRESERVED FOR ANALYSIS. THIS IS BETTER THAN TO ESCAPE FROM THE CALLING FUNCTION WITH NO TRACE.
- SEE VESCAPE.
- RETURNS FORMATTED NUMERIC MATRICES.
- ANY FRACTIONS PRESENT WILL FORCE TWO DECIMAL PLACES.
- VECTORS WILL BECOME ONE-COLUMN MATRICES.
- USES: VINTEGER VMATRIX VCHARACTER VHANG

FUNCTIONS:

V Z+TABULATE A
[1] 'A IS NOT NUMERIC'HANG CHARACTER A
[2] Z+(2× INTEGER A)VMATRIXQA

V

EXAMPLES:

TABULATE 2 3\#16
1 2 3
4 5 6

TABULATE 2 3\#16
3.14 6.28 9.42
12.57 15.71 18.85

TABULATE 13
1
2
3

TABULATE 'ABC'
A IS NOT NUMERIC
HANG[5]

THE PROGRAM HAS STOPPED FOR INSPECTION OF THE INPUT DATA AND ITS POSSIBLE SOURCE.
Section III

Workspace Management Functions
LISTFN LISTS A FUNCTION IN STANDARD FORM

SYNTAX: R+LISTFN FN

- Returns a character matrix which appears to be a listing of the function whose name, in quotes, is the argument. It contains V's, statement numbers, and indented labels and comments.
- An empty matrix is returned for non-existent and locked functions.

FUNCTION:

```
V ZQQ+LISTFN XQQ;FQQ;OIO
[1] OIO+1
[2] ->0=1+PZQQ+;CR XQQ)+0
[3] ZQQ+X(PFX 5 0+;CR 'LISTFN'), ZQQ'
[4] ->0
[5] Z+FQQ A;B;N
[6] N+1+pB+(A[; 1]=')V B(+/V BfZ)++/' '+(B+V/Z+A=')fA
[8] Z++;[1][1]';ZP(3 0v(N,1)vN),''),Bv ' ',A),[1]' '1
[9] Z[1,N+2;5]+'V'
V
```

EXAMPLE:

```
VTEST[[]]V
V TEST
[1] A+13
[2] aCOMMENT
[3] LI:'END'
V
```

```
O+Q+LISTFN 'TEST'
V TEST
[1] A+13
[2] aCOMMENT
[3] LI:'END'
V
```

```
V
```

NOTES: The technique used here Illustrates a Method of dealing with the problem of name-shadowing in APL: access to global or semi-global objects is inhibited if identical local names occur in an active function. To avoid this in situations where access to global objects is necessary, some programmers use highly improbable names such as the ZQQ, XQQ and FQQ used here. This in turn makes the code harder to understand and maintain. The method used here is a compromise. Improbable names are used to obtain data to pass to a local function with "good" names. Line 3 creates and executes this local function, which is "defined" in lines 5 through 9 of LISTFN.
**SYNTAX:**

\[ R+\text{DEF} \text{ Prompt MSG} \]

- **Prompt** can examine keyboard entry, and either accept it uncritically, return a specified default character vector, or, if numeric values are requested, will check the character set used, and can check for the specified length.
- **Msg** is a character string, i.e., the prompting message.
- **Def**, if character, is the default character string, or '' if a scalar numeric value, \( n \):
  - Will accept any numeric vector if \( n = 0 \)
  - Will reject vector unless \( n = \text{p-vector} \)
- If an integer vector, \( v \), will accept a numeric vector if its length is one member of \( v \).
- **Uses:** \( \text{V} \text{character Vempty Vif V} \)

**Function:**

\[ \text{R}+\text{DEF Prompt MSG;J;K} \]

[1] \( +0 \text{ IF O=}\text{p,DEF} \Delta \text{J}+\text{R+R+}[\text{R}+\text{p,R+MSG} \]
[2] \( +0 \text{ IF CHARACTER DEF} \Delta \text{R}+\text{R,}(0=}\text{p,R)/}\text{DEF} \]
[3] \( \text{NSCREEN:ERR IF~(~EMPTY K)\&\&/K+J} \leq \text{.0123456789E'} \)
[4] \( +0 \text{ IFv/DEF=(DEF>0)\times R+1J} \]
[5] \( \text{ERR:+1 Δ [^* NOT EXACTLY ' ,((V/DEF>0)/}\text{DEF},'} \text{' NUMBER',/(V/DEF>1)'/S'} \)
[6] \( \text{a characters may be added or deleted from NSCREEN} \)

**Examples:**

- **Prompt** the ages of the members of your family, in descending order...
  
  THE AGES OF THE MEMBERS OF YOUR FAMILY, IN DESCENDING ORDER..54 43 22 16
  54 43 22 16

- **Prompt** the month day year of your birth, as numbers...
  
  THE MONTH DAY YEAR OF YOUR BIRTH, AS NUMBERS..3 13
  NOT EXACTLY 3 NUMBERS
  THE MONTH DAY YEAR OF YOUR BIRTH, AS NUMBERS..3 13 1954
  3 13 1954

- **Prompt** give me two numbers...
  
  GIVE ME TWO NUMBERS..14 1.234E56
  -14 1.234E56

- **Prompt** an example of a scalar numeric value:
  
  AN EXAMPLE OF A SCALAR NUMERIC VALUE: ASDF
  NOT EXACTLY 1 NUMBER
  AN EXAMPLE OF A SCALAR NUMERIC VALUE: -1
  -1

- **Prompt** another...
  
  ANOTHER..0
  0

- **No response**
  
  WHAT IS YOUR NAME?  WHAT IS YOUR NAME?

- **No response**
  
  GIVE ME TWO OR THREE NUMBERS...
  GIVE ME TWO OR THREE NUMBERS...77
  NOT EXACTLY 2 3 NUMBERS
  GIVE ME TWO OR THREE NUMBERS...77 6
  77 6
STATUS CURRENT SESSION AND WORKSPACE STATUS [ NOW ]

SYNTAX: STATUS

• DISPLAYS DATE, TIME, CPURTIME, AVAILABLE SPACE, SUSPENSIONS

FUNCTIONS:

\[ \text{STATUS} \]
\[ \text{NOW} \]
\[ \text{CPUMTIME THIS SESSION: ')(0.001*1+1+0AI),' SECONDS' \]
\[ \text{SPACE LEFT: ')(vWA),' BYTES' \]
\[ \text{FUNCTIONS SUSPENDED: ',v-1+p\text{pLC} \}
\[ \text{R+NOW} \]
\[ \text{CPUMTIME THIS SESSION: 8.117 SECONDS} \]
\[ \text{SPACE LEFT: 404708 BYTES} \]
\[ \text{FUNCTIONS SUSPENDED: 0} \]
TABS

COMPARE REQUIRED TAB SETTINGS TO EXISTING ONES [ COLNO ]

SYNTAX:

\[ \text{TABS} \]

- AN UPCOMING REPORT MAY REQUIRE SPECIAL TAB SETTINGS.
- THE TERMINAL USER CAN BE PROMPTED TO VERIFY THE CORRECTNESS
  OF THE EXISTING TABS BEFORE CONTINUING WITH THE REPORT.
- SEE VTCC, PAGE 44

FUNCTIONS:

\[ \text{TABS;X;Y} \]

1. VISUAL CHECK OF REQUIRED AND ACTUAL TABS
2. 'NONE REQUIRED.' ESCAPE EMPTY \( \text{\&HT, \&IO} \)
3. 'IMPOSSIBLE. CORRECT \( \text{\&HT, \&PW} \). ESCAPE-\( \text{\&HT} / \( \text{\&PW} / \text{\&HT} / 129 \)
4. 'TABS REQUIRED: ) (LEFT MARGIN AT ZERO)'
5. ' ' \( X=/(y+1+[/\text{\&HT}]/\text{\&HT}) \)
6. 0 COLNO Y
7. (1+X) '\text{\&}'
8. 5\text{\&IDLC}
9. 'EXISTING TABS, INDICATED BY \'. CLEAR AND SET AS REQUIRED.'

\[ \text{\&HT+}' \]

\[ \text{TABS} \]

NONE REQUIRED.
\[ \text{\&HT+1 33 55} \]
\[ \text{TABS} \]

TABS REQUIRED: ) (LEFT MARGIN AT ZERO)

\[ \text{\&HT+1333333334444444455555555} \]
\[ \text{012345678901234567890123456789012345678901234567890123456789012345} \]

\[ \text{\&HT+1333333334444444455555555} \]

EXISTING TABS, INDICATED BY ' \'. CLEAR AND SET AS REQUIRED.

43
TCC SYSTEM INDEPENDENT TERMINAL CONTROL CHARACTERS

SYNTAX:

Z+BKSP  Z+CRTN  Z+IDLC  Z+LNFD  Z+TABC

- Z IN EACH CASE IS ONE BACKSPACE, CARRIAGE RETURN, IDLE, LINE FEED, OR TAB CHARACTER, IF THAT CHARACTER EXISTS IN THE ATOMIC VECTOR (AV) IN THE APL SYSTEM YOU ARE USING. THE CHARACTERS ARE STORED AS THE GLOBAL VARIABLES BK, CR, ID, LN, AND TB FOR RAPID ACCESS. THESE FUNCTIONS DEPEND ON THE SPECIFIC CONFIGURATION OF AV IN THE VARIOUS APL SYSTEMS TO DATE. EACH FUNCTION ASSUMES THAT AT LEAST ONE ELEMENT OF AV IS UNIQUE TO EACH SYSTEM.

- KNOWN EXCEPTIONS: THE TAB AND IDLE CHARACTERS DO NOT EXIST IN VS APL AV. IDLC WILL DELIVER THE \( \text{O} \) CHARACTER; TABC THE \( + \).

FUNCTIONS:

\[ \text{V} \]
\[ Z+BKSP \]
\[ \rightarrow (0 \cdot NC 'BK') / A1 \]
\[ 1 \]
\[ B K + AV (IO + 158 200 41 [ 'O' \cdot AV [ IO + 73 ] ] ) \]
\[ 2 \]
\[ A 1 : Z + BK \]
\[ \text{V} \]

\[ \text{V} \]
\[ Z+CRTN \]
\[ \rightarrow (0 \cdot NC 'CR') / A1 \]
\[ 1 \]
\[ C R + AV (IO + 156 202 73 [ 'O' \cdot AV [ IO + 73 ] ] ) \]
\[ 2 \]
\[ A 1 : Z + CR \]
\[ \text{V} \]

\[ \text{V} \]
\[ Z+IDLC \]
\[ \rightarrow (0 \cdot NC 'ID') / A1 \]
\[ 1 \]
\[ I D + AV (IO + 163 191 64 [ 'O' \cdot AV [ IO + 73 ] ] ) \]
\[ 2 \]
\[ A 1 : Z + ID \]
\[ \text{V} \]

\[ \text{V} \]
\[ Z+LNFD \]
\[ \rightarrow (0 \cdot NC 'LN') / A1 \]
\[ 1 \]
\[ L N + AV (IO + 159 201 169 [ 'O' \cdot AV [ IO + 73 ] ] ) \]
\[ 2 \]
\[ A 1 : Z + LN \]
\[ \text{V} \]

\[ \text{V} \]
\[ Z+TABC \]
\[ \rightarrow (0 \cdot NC 'TB') / A1 \]
\[ 1 \]
\[ T B + AV (IO + 162 185 192 [ 'O' \cdot AV [ IO + 73 ] ] ) \]
\[ 2 \]
\[ A 1 : Z + TB \]
\[ \text{V} \]

EXAMPLES:

\[ 'O', BKSP, 'O' \]

\[ 'A', CRTN, 'B' \]

\[ 'A', IDLC, 'B' \]

\[ 'A', LNFD, 'B' \]

\[ 'A', TABC, 'B' \]
TIME

RUNNING TIME AND NEW SPACE FOR AN APL STATEMENT [ ALT ]

SYNTAX:

TIME STMT

- Executes the APL statement in the quoted character string, STMT. Displays its running time and new space.
- Different ways of performing the same function or equivalent operation can be compared.
- Language elegance, clarity, and maintainability should be weighed in judgment against time and space usage.
- Zero space may be reported if the variables had been named earlier in this workspace and were sufficiently large at that time.
- Traffic on the host machine may cause timing to vary slightly. Time your statements more than once, then average the results.
- Temporary storage is not measured, although it may be significantly large, e.g. outer products.

FUNCTIONS:

\[ \text{TIME STMT; n; Z} \]

[1] How much time/space does an APL statement use?
[2] ' n' ALT STMT
[3] \[ Z \leftarrow (1+1+\square AI), \square WA \]
[4] \[ n \]
[5] \[ (\mathbf{v}_{-7} + (1+1+\square AI) - 1 + Z), \text{MSEC ',}(\mathbf{v}_{-7} - 48 + \square WA), \text{' BYTES'} \]
[6] \[ n \]

\[ \text{NAME ALT EXP; n} \]

[1] \text{J+\square FX NAME ON EXP}
[2] \text{INTRANSITIVE SYNONYM}

EXAMPLES:

TIME 'Z+1000'
1 MSEC 24 BYTES
TIME 'Z+Z-7'
0 MSEC 0 BYTES

- Notice that some APL systems keep an arithmetic progression vector stored in compact form as long as possible. When we square it:
  TIME 'A+Z*2'
8 MSEC 392 BYTES
- ...we suddenly see storage being allocated
- For it since it can no longer be stored merely as starting point, step, and length. For timing, compare:
  TIME 'B+Z*2'
10 MSEC 392 BYTES
- Surprisingly, \( Z*Z \) seems to take longer than \( Z*2 \).
- Timing and storage comparisons may well be different on different APL systems. The 5110 does not support \( \square AI \).
  TIME ''
0 MSEC 0 BYTES
- If the results of testing the previous null statement were
- Not 0 0, adjust the constants in line 5 of the APL function.
VARS DISPLAY CHARACTERISTICS OR CONTENTS OF VARS SELECTIVELY

SYNTAX:  
R+R VARS K

- IN THE AUTOMATIC PRODUCTION OF APPLICATION DOCUMENTATION, IT IS OFTEN DESIRED TO DISPLAY ALL VARIABLES, OR ONLY THOSE OF PARTICULAR INITIAL CHARACTERS.
- WHEN THESE STRUCTURES ARE TOO LARGE TO BE DISPLAYED, THEIR CHARACTERISTICS ONLY, MAY BE REQUESTED.

1 VARS'' WIL DISPLAY EVERYTHING
1 VARS''Z''' WIL DISPLAY ALL VARIABLES WITH INITIAL Z
0 VARS'' WIL DISPLAY CHARACTERISTICS ONLY OF ALL VARS
0 VARS''XYZ''' FOR CHARACTERISTICS OF VARS INITIALLY "XYZ"
- USES: VDLTMB VIF VLOGICAL VINTEGER VFLOATING VCHARACTER VEMPTY VGRADEUP VIS ALF

FUNCTION:
\[ R+R VARS K;J;DIO \]

- LOCAL OR GLOBAL VARIABLES AND THEIR CHARACTERISTICS

\[ K+8 10+CR 'VARS' \]

\[ BACK;J+(14)IF(LOGICAL J),(INTEGER J),(FLOATING J),CHARACTER J+1R[1;] \]

\[ ' ' \]

\[ EMP IF EMPTY J \]

\[ R[1;], 'IS 'DLTMB(.K[I;]),(,K[I+4+ppJ;]),(0*ppJ)'/OF SHAPE ',wpx \]

\[ J[R[1;] \]

\[ FWD:+(0*1+pR+1 04R)0,BACK \]

\[ EMP:+FWD,pD+R[1;],'IS EMPTY' \]

- THE FOLLOWING IS NEVER EXECUTED.

- LOGICAL
- INTEGER
- FLOATING
- CHARACTER
- SCALAR
- VECTOR
- MATRIX
- ARRAY

EXAMPLE:

0 VARS''RST'''

R IS CHARACTER MATRIX OF SHAPE 47 76
T IS CHARACTER VECTOR OF SHAPE 12
TT IS EMPTY

46
Section IV

Multiprecision Arithmetic Functions
ADD

MULTIPRECISION INTEGER ADDITION

SYNTAX:

C+A ADD B

- ADD A TO B, GIVING C, USING MULTIPRECISION INTEGER ARITHMETIC (I.E. TRUNCATING ALL RESULTS TO MP INTEGERS)
- A AND B ARE NUMERIC VALUES. THEY MAY BE: INTEGER OR FLOATING POINT, OF EITHER ORDINARY OR EXTENDED PRECISION.
- C WILL BE A MULTIPRECISION INTEGER (ZERO EXPONENT)
- AN EXTENDED PRECISION NUMBER, INTEGER OR FLOATING POINT, IS REPRESENTED BY AN INTEGER VECTOR OF CONSECUTIVE 1-7 DECIMAL DIGIT SEGMENTS, THE FIRST OF WHICH IS THE EXPONENT, ZERO FOR INTEGERS. ALL SEGMENTS HAVE THE SAME SIGN, WITH THE EXCEPTION OF THE EXPONENT. (SEE VPADD). LEADING ZEROS ARE ELIMINATED.
- USE VFIX TO CONVERT A NUMERIC VARIABLE TO MP INTEGER FORM.
- THE MULTIPRECISION INTEGER ARITHMETIC PACKAGE COMPRISSES: VADD VSUB VMUL VDIV VSQRT VFIX VCAN VROUNDS
- FUNCTIONS OF THE MULTIPRECISION INTEGER ARITHMETIC PACKAGE ARE USED BY THE MULTIPRECISION FLOATING POINT ARITHMETIC PACKAGE
- USES: VFIX VCAN

FUNCTION:

V C+A ADD B
[1] MULTIPRECISION INTEGER ADD
[2] C+CAN[(C+FA)+(C+(pA+FIX A)/(pB)+pB+FIX B]

EXAMPLES:

A
0 1 2345678 9012345 6789012
B
0 222 3333333 4444444
A ADD B
0 1 2345901 2345679 1233456
A ADD -3
0 1 2345678 9012345 67890.09
0 0 0 0 ADD -1
0 9999999 9999999 9999999
ALPREC ALTER PRECISION OF A SCALAR OR MULTIPRECISION NUMBER

SYNTAX: Z+N ALPREC X

- FOR X A SCALAR OR MULTIPRECISION INTEGER OR FLOATING POINT NUMBER OF PRECISION M (I.E. (pX)=M+1), Z WILL BE SET TO THE FLOATING POINT NUMBER OF PRECISION M+N WHOSE VALUE IS THE SAME AS OR TRUNCATED FROM THAT OF X. (SCALARS HAVE IMPLICIT PRECISION 3.)
- IN BRIEF, X IS EXTENDED ON THE RIGHT WITH ZEROES (IF N>0) OR TRUNCATED ON THE RIGHT (IF N<0). THE EXPONENT (1+X) IS INCREASED BY N TO COMPENSATE.
- SINCE ALL THE FUNCTIONS IN THE MULTIPRECISION FLOATING POINT PACKAGE DETERMINE THE PRECISION OF THE RESULT FROM THE PRECISIONS OF THE INPUT, IT IS OFTEN NECESSARY TO INDICATE THE PRECISION OF SOME EXACT QUANTITY (SUCH AS AN INTEGER) SO AS NOT TO CAUSE INVOLUNTARY SHORTENING OF PRECISION. SEE THE EXAMPLES BELOW.
- DECREASING THE PRECISION OF A NUMBER (BY USING NEGATIVE N) MAY OCCasionally BE NECESSARY IF THE RESULT OF AN INTERMEDIATE CALCULATION IS ASSIGNED A SPURIOUSLY HIGH PRECISION BY ONE OF THE ARITHMETIC FUNCTIONS.
- USES: VFLOAT

FUNCTION:

\[ Z+N \text{ ALPREC } X \]

[1] \text{INCREASE THE PRECISION OF } X \text{ BY } N \text{ (OR DECREASE IF } N \text{ IS NEGATIVE).}

[2] \[ Z+(-N-1+X),1+((p,X)+N+NTZ-p,X)+X+\text{FLOAT } X \]

EXAMPLES:

\[ D \]

\[ ^{-5} \text{76543 2109876 5432109 8765432 1098765} \]
\[ \text{FORMAT } D \]
\[ 0.0076543 2109876 5432109 8765432 1098765 \]
\[ \text{FORMAT 1 FDIV D} \]
\[ 130.6451593 9386058 9254313 \]
\[ \text{NOTE THE LOSS OF PRECISION.} \]
\[ \text{A INSTEAD, ONE, WHICH WE KNOW IS EXACT, MUST BE ASSIGNED} \]
\[ \text{A PRECISION AT LEAST AT GOOD AS THAT OF THE NUMBER } D. \]
\[ D+\text{ONE}+5 \text{ ALPREC 1} \]
\[ ^{-7} 1.00000000000 \]
\[ D+Z+\text{ONE FDIV D} \]
\[ ^{-5} 130.6451593 9386058 9254312 9891047 5895018 \]
\[ \text{FORMAT } Z \]
\[ 130.6451593 9386058 9254312 9891047 5895018 \]
\[ \text{SIMILARLY, WE CAN AVOID A LOSS OF PRECISION IN ADDITION, HERE:} \]
\[ D+Z+1 \text{ FADD D} \]
\[ ^{-2} 1 \text{76543 2109876} \]
\[ \text{AND THE RIGHT WAY, USING THE PREPARED ONE:} \]
\[ \text{FORMAT } Z+\text{ONE FADD D} \]
\[ 1.0076543 2109876 5432109 8765432 1098765 \]
EDIT MULTIPRECISION INTEGERS INTO CANONICAL FORMAT

**SYNTAX:**

\[ Z + \text{CAN} \ A \]

- INPUT A IS EDITED INTO THE CANONICAL MULTIPRECISION INTEGER FORMAT AS DESCRIBED UNDER FUNCTION ADD
- THIS FUNCTION IS USED BY EVERY OTHER FUNCTION IN THE MULTIPRECISION INTEGER AND FLOATING POINT ARITHMETIC PACKAGES, EXCEPT VROUNDS.
- USES: VROUNDS

**FUNCTION:**

\[
\begin{align*}
\text{\texttt{\textbackslash v}} & \text{ Z + CAN A ; S ; L} \\
[1] & \text{\texttt{\textbackslash nEDIT A MULTIPRECISION INTEGER INTO CANONICAL FORM}} \\
[2] & Z + 1 \text{ ROUNDS A} \\
[3] & \text{\texttt{\textbackslash nPROPAGATE CARRIES LEFTWARD}} \\
[4] & L1 : Z + (S, 0) + 0, Z - 10000000 \times S + (\times Z) \times L | Z + 10000000 \\
[5] & + (Z \times S) / L1 \\
[6] & \text{\texttt{\textbackslash nDROP LEADING ZEROES (BUT PREVENT 0 \rightarrow 1 0)}} \\
[7] & L2 : + (1 = p Z + (\left(0 \times 1 + Z, 1\right) - \square IO) + Z) / L3 \\
[8] & \text{\texttt{\textbackslash nMAKE ALL TERMS (EXCEPT THE EXPONENT) THE SAME SIGN}} \\
[9] & + (\neg V / S + (\neg L + 1 + S) = S + \times Z) / L3 \\
[10] & + L2, p Z + Z + (L \times 10000000 \times S) + (1 + L \times S, 0) \\
\end{align*}
\]

**EXAMPLES:**

\[
\begin{align*}
A & 0 - 1 - 2345678 - 9012345 - 6789012 \\
B & 0 222 333333 4444444 5555555 \\
A + B & 0 221 987655 4567901 1233457 \\
\text{CAN A + B} & 0 221 987654 5432098 8766543 \\
& 123456789012345 \\
& 1 2345678 9012345 \\
& \text{CAN 7.8} \\
& 0 8
\end{align*}
\]
DIV

MULTIPRECISION INTEGER DIVISION

SYNTAX:

C+A DIV B

- DIVIDE A BY B, GIVING C, USING MULTIPRECISION INTEGER ARITHMETIC. A AND B MAY BE IN ANY NUMERIC FORMAT.
- SEE THE DESCRIPTION UNDER VADD.
- THE REMAINDER OF THE DIVISION IS LEFT IN GLOBAL VARIABLE REM
- USES: VADD VSUB VMUL VFIX VCAN

FUNCTION:

\[ \text{V C+A DIV B;Q;T}\]

\[\begin{align*}
1 & : \text{MULTIPRECISION INTEGER DIVIDE WITH REMAINDER} \\
2 & : (2^pB+\text{FIX } B)/L2 \\
3 & : \text{SPECIAL CODE FOR SPEED IF B IS SCALAR} \\
4 & : C- (2^pB+1+B)=1+REM+0,1+A+\text{FIX A) p 0} \\
5 & : L1: C+C, Q= (T+100000001REM, 1+A) \div B \\
6 & : \text{REM+0,T-QxB} \\
7 & : (0<pA+1+A) / L1 \\
8 & : 0, pC+C=\text{CAN C} \\
9 & : L2: \text{REM+A+FIX A} \\
10 & : (B^{A} = "pC+0 0) i L4, 0 \\
11 & : L3: Q=\text{CAN}\{100000013+REM} \div 100000012+B \\
12 & : C+C ADD Q= (-1+(pQ)+(pREM)-pB) \div Q \\
13 & : \text{REM+REM SUB B MUL Q} \\
14 & : L4: \text{ADD Q}= (x(pREM)-pB)+L3, 0 \\
15 & : L3\times (|1+2+REM> |1+2+B)
\end{align*}\]

EXEMLPES:

\[\begin{align*}
A & : 1-2345678-9012345-6789012 \\
B & : 0222333333444444 \\
A & : \text{DIV } B \\
0 & : 55528 \\
RE & : 04643271574297420 \\
B & : \text{DIV } A \\
0 & : 0 \\
RE & : 0222333333444444 \\
A & : \text{DIV } 0 \\
0 & : (\text{DOMAIN ERRORS PREVENTED, AS IN VCDIV}) \\
0 & : 01000000 DIV 7 \\
0 & : 1428571428571428571428571428 \\
RE & : 04
\end{align*}\]
FADD

MULTIPRECISION FLOATING POINT ADDITION

SYNTAX:

C+A FADD B

- A AND B ARE NUMERIC VARIABLES. THEY MAY BE: INTEGER OR FLOATING POINT OF EITHER ORDINARY OR EXTENDED PRECISION.
- THE RESULT WILL BE MULTIPRECISION FLOATING POINT.
- NUMERIC VARIABLES OF ANY FORMAT MAY BE CONVERTED INTO MP FLOATING POINT BY THE FUNCTION FLOAT.
- THE PRECISION OF A MULTIPRECISION FLOATING POINT NUMBER IS INDICATED BY ITS LENGTH, AND ALL MULTIPRECISION FLOATING POINT OPERATIONS SET THE LENGTH OF THE RESULT ACCORDING TO THE PRECISION OF THE OPERANDS. IN PARTICULAR, THE RESULT OF AN ADD OR SUBTRACT HAS A PRECISION SUCH THAT ITS LEAST SIGNIFICANT ELEMENT IS GOVERNED BY THE SIGNIFICANCE OF THE OPERAND OF GREATER MAGNITUDE.
- THE MULTIPRECISION FLOATING POINT ARITHMETIC PACKAGE COMPRISSES: VADD VFSUB VMUL VDIV VFLOAT VFSQRT VEXP VPI VALPREC VFORMAT VSCALE
- FADD USES: VFLOAT VADD

FUNCTION:

\[
\begin{align*}
\downarrow C+A FADD B;M\\
\text{[1]} & \text{MULTIPRECISION FLOATING POINT ADD} \\
\text{[2]} & M+\lfloor C+(1+A\cdot\text{FLOAT } A),1+B\cdot\text{FLOAT } B \\
\text{[3]} & C+(\lfloor /C),1+(0,1+(M-1+C)\cdot A)\text{ADD } 0,1+(M-1+C)\cdot B\\
\downarrow
\end{align*}
\]

EXAMPLES:

A
0 47152 2357206
B
0 222 3333333 4444444 5555555
A FADD B
0 222 3333333 4491596 7912761
A FADD C
0 47152 2369551

NOTE THE TRUNCATION

TO SEE THE NUMBERS IN USUAL FORM, USE VFORMAT

FORMAT A
47152 2357206.
FORMAT C
12345.6789012 3456789
FORMAT A FADD C
47152 2369551.
FDIV MULTIPRECISION FLOATING POINT DIVISION

SYNTAX: C+A FDIV B

- A, B, AND C ARE MULTIPRECISION FLOATING POINT VALUES.
  (SEE DESCRIPTION FOR VADD)
- THE PRECISION OF THE RESULT IS THE SAME OR SLIGHTLY GREATER
  THAN THE SMALLER OF THE PRECISIONS OF THE TWO OPERANDS.
- USES: VFLOAT VDIV
- DIVISION BY ZERO WILL RETURN ZERO, AS IN VCDIV.

FUNCTION:

\[ \text{C} + A \text{ FDIV B} \]

\[ \text{REM} \]

\[ 1 \quad \text{MULTIPRECISION FLOATING POINT DIVIDE} \]
\[ 2 \quad \text{C} + ((1000 \times 1 + A) \times 1 + 1 + \text{B}) + 1 + (\text{pB}) + 0 \times (\text{pB} + \text{FLOAT B}) - \text{pA} + \text{FLOAT A} \]
\[ 3 \quad \text{C} + ((1 + A) - (1 + B) + C), 1 + (0, 1 + (C + pA) + A) \text{DIV 0, 1 + B} \]

\[ \text{V} \]

EXAMPLES:

B
0 222 3333333 4444444 5555555
FORMAT B
222 3333333 4444444 5555555.

C
-2 12345 6789012 3456789
FORMAT C
12345.6789012 3456789
FORMAT B FDIV C
180090 0016298 1001574.
FORMAT (B FMUL C) FDIV B
12345.6789012 3456788 9994910
HALF
-6 5000000 0 0 0 0
FORMAT HALF
0.5000000 0000000 0000000 0000000 0000000 0000000
FORMAT B FDIV HALF
444 6666666 8888888 1111110.0000000
FEXP
MULTIPRECISION FLOATING POINT EXPONENTIAL FUNCTION

SYNTAX: Z+FEXP X

- RETURNS *X AS A MULTIPRECISION FLOATING POINT NUMBER.
- X MAY BE SCALAR OR MULTIPRECISION FLOATING POINT,
  MP INTEGER BEING A CASE OF FLOATING POINT.
- SEE: VADD VFADD
- THE PRECISION OF Z IS CHOSEN TO BE THE SAME AS THAT
  OF X. USE VALPREC TO INCREASE PRECISION
- USES: VFLOAT VADD VMUL VDIV

FUNCTION:

\[ V Z+FEXP X;I;N;T \]
\[ ^{[1]} \text{MULTIPRECISION FLOATING POINT } *X \text{ BY THE STANDARD POWER SERIES} \]
\[ ^{[2]} Z+T+0,(1-N+1+X+\text{FLOAT } X)+1+i0 \]
\[ ^{[3]} X+0,1+iX \]
\[ ^{[4]} \text{LOOP: } Z+Z \text{ ADD } T+N+(T \text{ MUL } X) \text{DIV } I+i1 \]
\[ ^{[5]} \to(\sqrt{T} \div 0) \text{ LOOP} \]
\[ ^{[6]} Z+N,1+iZ \]
\[ V \]

EXAMPLES:

CREATE THE CONSTANT 1 BY USE OF FUNCTION ALPREC

\[ ^{[1]} \text{ON6 }^4 \text{ ALPREC } 1 \]
\[ ^{[2]} 1.0000000 000000 000000 000000 000000 000000 \]
\[ ^{[3]} \text{FEXP ONE} \]
\[ ^{[4]} -6 2 712818 2845904 5235360 2874713 5266249 7757231 \]
\[ ^{[5]} \text{FEXP ONE} \]
\[ ^{[6]} 2.7182818 2845904 5235360 2874713 5266249 7757231 \]
\[ ^{[7]} \text{MINUSONE }^4 \text{ APPREC } -1 \]
\[ ^{[8]} -6 -1 0 0 0 0 0 \]
\[ ^{[9]} \text{FEXP MINUSONE} \]
\[ ^{[10]} -1.000000 000000 000000 000000 000000 000000 \]
\[ ^{[11]} \text{FEXP MINUSONE} \]
\[ ^{[12]} 0.3678794 417144 2321595 5237701 6146086 7445811 \]
FIX

CONVERT TO MULTIPRECISION INTEGER

SYNTAX:

Z+FIX X

- IF X IS SCALAR, IT IS ROUNDED AND CATENATED BEHIND A ZERO.
  IF X IS MP INTEGER, IT IS LEFT UNCHANGED.
  IF X IS MULTIPRECISION FLOATING POINT, IT IS TRUNCATED TO
  THE MULTIPRECISION INTEGER WITH THE SAME LEADING SEGMENTS.

FUNCTION:

V Z+FIX X

[1] ACONVERT A NUMBER TO A MULTIPRECISION INTEGER
[2] +(1<z+X)/MP
[3] 0, Z+C AN Z
[4] MP: +(0=1+Z)/0
[5] Z+C AN(0{1+(1+X)+pX}+1+X

EXAMPLES:

B
0 222 3333333 4444444 5555555
FORMAT B
222 3333333 4444444 5555555. FIX B
0 222 3333333 4444444 5555555
C
-2 -12345 -6789012 -3456789
FORMAT C
-12345.6789012 3456789 FIX C
0 -12345
D
-3 76543 2109876 5432109
FORMAT D
0.0076543 2109876 5432109 FIX D
0 0

FIX 7
0 7

FIX 7.8
0 8
FLOAT CONVERT TO MULTIPRECISION FLOATING POINT [ SCALE ]

SYNTAX:
Z-FLOAT X

* IF X IS SCALAR, IT WILL BE CONVERTED TO AN MP FLOATING POINT NUMBER OF PRECISION 3.
* SEE THE DESCRIPTION OF MP FLOATING POINT FORMAT FOR FADD.
* USES VCAN.

FUNCTIONS:

\[ \text{V Z-FLOAT X} \]
[1] A CONVERT A NUMBER TO MULTIPRECISION FLOATING POINT
[2] \((0<pZ\times X)/0\)
[3] \(X+\lfloor 10000000\times (X=0)+X \)
[4] \(Z+(X-2),1\text{VCAN } Z\times 10000000\times 2-X \)

\[ \text{V R+SCALE MP} \]
[1] A SCALAR APPROXIMATION OF MP
[2] \((10000000\times 1+MP)\times 1000000011+MP \)

EXAMPLES:

FLOAT 876
-2 876 0 0
FLOAT -1 2345
-2 -1 2345000 0
FLOAT 1E20
0 1000000 0 0
FLOAT -2 3 1415926 5358979
-2 3 1415926 5358979

SCALE 6 ALPREC 1234.1234567890
1234.123457
FMUL
MULTIPRECISION FLOATING POINT MULTIPLICATION

**SYNTAX:**
C+A FMUL B

- A AND B ARE SCALAR OR MULTIPRECISION NUMBERS.
- RETURNS A MULTIPRECISION PRODUCT.
- SEE DESCRIPTION UNDER VFADD.
- THE PRECISION OF C IS SUCH THAT THE RELATIVE ACCURACY OF ITS LEAST SIGNIFICANT DIGIT IS JUST BETTER THAN THE LEAST SIGNIFICANT OF THE TWO OPERANDS.
- USES: VFLOAT VMUL

**FUNCTION:**

\[ C = A \times B \]

\[ 1 \] MULTIPRECISION FLOATING POINT MULTIPLY
\[ 2 \] \( C = (0,1+A+\text{FLOAT } A) \times (0,1+B+\text{FLOAT } B) \)
\[ 3 \] \( C = ((1+A)+(1+B)-C),(C+2-(pC)\times 2-(pA)\times pB)+1+C \)

**EXAMPLES:**

\[ A \]
0 47152 2357206
HALF
-6 5000000 0 0 0 0 0
FORMAT HALF
0.5000000 0000000 0000000 0000000 0000000 0000000
A FMUL HALF
-1 23576 1178603 0
FORMAT A FMUL HALF
23576 1178603.000000
C
-2 12345 6789012 3456789
C FMUL HALF
-3 6172 8394506 1728394 5000000
FORMAT C FMUL HALF
6172.8394506 1728394 5000000
TWO
-5 2 0 0 0 0 0
\cdot FORMAT TWO
2.0000000 0000000 0000000 0000000 0000000 0000000
TWO FMUL HALF
-6 1 0 0 0 0 0
FORMAT TWO FMUL HALF
1.0000000 0000000 0000000 0000000 0000000 0000000
FORMAT

CONVERT MULTIPRECISION NUMBER TO CHARACTER STRING

SYNTAX:

Z+FORMAT X

Returns a character string with seven-digit groups of decimal digits punctuated by single blanks or decimal point.

FUNCTION:

\[ Z+\text{FORMAT } A; B; E \]

1. CONVERT A MULTIPRECISION NUMBER TO A CHARACTER STRING
2. GET THE EXPONENT, AND INSERT LEADING AND TRAILING ZEROS
3. \[ A+((0\ 2-E+pA)p0,1+A,(0\ E+1+A)p0 \]
4. \[ Z+(8\ 0\ \downarrow([A]+10000000-(pA)*10000000),',', 7\times[IO]+') ' \]
5. \[ Z[B+(8\times1+pA)-7\times[IO]]+', ' \]
6. INSERT A DECIMAL POINT ON TOP OF A BLANK, E GROUPS FROM THE RIGHT
7. \[ Z[1+(1[E-1]+B)+', ' \]
8. DROP LEADING BLANKS, INSERT ' ' JUST IN FRONT
9. \[ Z+(0>1+A)/''),(−[IO]−(Z=' ')\ 0)+Z \]

EXAMPLES:

B
0 222 333333 4444444 5555555
FORMAT B
222 333333 4444444 5555555.

C
-2 12345.6789012 3456789
FORMAT C
-12345.6789012 3456789

D
-3 76543 2109876 5432109
FORMAT D
0.0076543 2109876 5432109
FORMAT -5 74
0.0000000 0000000 0000000 0000000 00000074
**FSQRT**

**MULTIPRECISION FLOATING POINT SQUARE ROOT**

**SYNTAX:**

\[ C + \text{FSQRT} \; A \]

- **Returns the Multiprecision Floating Point Form of the Square Root of** \( A \).
- **A may be scalar, MP integer, or MP floating point.**
- **The precision of the result is that of the operand.** (See the description for \( \text{VFADD} \).)
- **Uses:** \( \text{VFLOAT} \; \text{VSQRT} \)

**FUNCTION:**

\[ \text{VF} \; C + \text{FSQRT} \; A; \; E; \; M \]

[1] **MULTIPRECISION FLOATING POINT SQUARE ROOT**
[2] \( M \leftarrow 0.5 \times (C + 2 |E + A|) - pA + \text{FLOAT} \; A \)
[3] \( C + (M + 10.5 \times E)^{-1} \times \sqrt{0.1 \times (pA) + C - 2 \times M} + A \)

**EXAMPLES:**

\[ B \]

0 222 333333 4444444 555555
FORM B
222 333333 4444444 5555555.
\[ \text{Z+FSQRT} \; B \]
-2 47152 2357205 3020324 9027073
FORM Z
47152 2357205.3020324 9027073
FORM Z FMUL Z
222 333333 4444444 5555555.992900
\[ \text{Z+TWO+3 ALPREC 2} \]
-5 2 0 0 0 0
FORM TWO
2.000000 000000 000000 000000 000000
\[ \text{Z+FSQRT TWO} \]
-5 1 4142135 6237309 5048801 6887242 969807
FORM Z FMUL Z
1.9999999 9999999 9999999 9999999 9999997
FSUB MULTIPRECISION FLOATING POINT SUBTRACTION

SYNTAX: C+A FSUB B

- SUBTRACT B FROM A, USING MULTIPRECISION FLOATING POINT ARITHMETIC. A AND B MAY BE IN ANY NUMERIC FORMAT. (SEE DESCRIPTION UNDER FADD)
- USES: FADD.

FUNCTION:
\[ C + A \text{ FSUB } B \]

EXAMPLES:

A
0 47152 2357206
B
0 -222 333333 444444 555555

A FSUB B
0 -222 -333333 -4397292 -3198349

C
-2 12345 6789012 3456789

A FSUB C
0 47152 2344861

D+3 76543 2109876 5432109

C FSUB D
-2 12345 6712469 1346913

FORMAT C
12345.6789012 3456789

FORMAT D
0.0076543 2109876 5432109

FORMAT C FSUB D
12345.6712469 1346913
MULTIPRECISION INTEGER MULTIPLICATION

SYNTAX:

\( C \leftarrow A \text{ MUL } B \)

1. A IS MULTIPLIED BY B, GIVING C, USING MULTIPRECISION INTEGER ARITHMETIC; A AND B MAY BE IN ANY NUMERIC FORMAT
2. SEE THE DESCRIPTION UNDER VADD
3. USES: VFIX VCAN

FUNCTION:

\( \n C \leftarrow A \text{ MUL } B \)

1. MULTIPRECISION INTEGER MULTIPLY
2. MAKE A THE SHORTER OF THE ARGUMENTS TO SAVE SPACE
3. \( \{(pA)\leftarrow(pA\text{+FIX } A)L\l pB\text{+FIX } B)\}/L1 \)
4. C=A
5. A+B
6. B+C
7. CHECK FOR POSSIBLE OVERFLOW (720 = (2*56)*1E7*2).
8. \( L1\leftarrow(720\times1+pC\times A\times B)/L2 \)
9. \( C\leftarrow((LC+100000000),0)+0,100000000\cdot c \)
10. \( L2\leftarrow C\text{+CAN}\leftarrow[0,\phi C\leftarrow(\phi A),1+pA]p0 \)

EXAMPLES:

\[
\begin{array}{lllll}
A & 1 & 2345678 & -9012345 & -6789012 \\
B & 222 & 3333333 & 4444444 \\
A \text{ MUL } B & -274 & -4855942 & -5116597 & -4938071 & -3415514 & -3649328 \\
& 123456 \text{ MUL } 876543 \\
B & 10821 & 4492608 \\
5 \text{ MUL } B & 1111 & 6666667 & 2222220
\end{array}
\]
**SQRT MULTIPRECISION INTEGER SQUARE ROOT**

**SYNTAX:**

\[ Z \leftarrow \text{SQRT} A \]

- A MAY BE SCALAR OR MULTIPRECISION BUT WILL BE TRUNCATED TO AN INTEGER.
- RUNNING TIME IS PROPORTIONAL TO \( (pA)^x \) AND IS MUCH FASTER FOR NUMBERS WITH MANY TRAILING ZEROES.
- USES: \texttt{\textbackslash fix \textbackslash sub \textbackslash mul \textbackslash div \textbackslash can}.

**FUNCTION:**

\[
\texttt{\textbackslash v \ 2+\text{SQRT} \ A;F;N;\text{REM}}
\]

\[
[1] \ 	exttt{\textbackslash a\text{MULTIPRECISION INTEGER SQUARE ROOT BY ELEMENTARY RECURSION}}
\]

\[
[2] \ 	exttt{\textbackslash a\text{GET AN ACCURATE STARTING VALUE}}
\]

\[
[3] \ Z+(N+1+L0.5\times pA)+\texttt{CAN}(10000000\downarrow (6+2|pA)+A+\texttt{FIX} A)\times 0.5
\]

\[
[4] \ 	exttt{\textbackslash a\text{DROP TRAILING ZEROES FROM A AND DOUBLE IT}}
\]

\[
[5] \ A\times 0\times 1=1+1+A+\texttt{CAN} 2\times((pA)+[10-\varphi0\times A];1)+A
\]

\[
[6] \ 	exttt{\textbackslash LOOP:Z+N+z\text{MUL} F+((1+pF)\times 0\ 1\ 5000000 )\text{SUB} F+((N+pA)+Z\text{MUL} Z)\text{DIV} A}
\]

\[
[7] \ 	exttt{\textbackslash (1\times/N+F)/\text{LOOP}}
\]

**EXAMPLES:**

\[
B
\]

\[
0 \ 222 \ 3333333 \ 4444444 \ 5555555
\]

\[
\texttt{\textbackslash a+A+SQRT} \ B
\]

\[
0 \ 47152 \ 2357205
\]

\[
A \ \texttt{\textbackslash mul} \ A
\]

\[
0 \ 222 \ 3333333 \ 4415961 \ 5412025
\]

NOTICE THAT THIS ESTIMATE OF \( B^{0.5} \) IS SLIGHTLY LOW

\[
\texttt{\textbackslash a+A+\text{ADD} 1}
\]

\[
0 \ 47152 \ 2357206
\]

\[
A \ \texttt{\textbackslash mul} \ A
\]

\[
0 \ 222 \ 3333333 \ 4510266 \ 126436
\]

BUT THE NEXT HIGHER INTEGER IS SLIGHTLY TOO HIGH, SO THAT THE TRUE SQUARE ROOT IS BRACKETED BETWEEN THESE TWO VALUES.
SUB

MULTIPRECISION INTEGER SUBTRACTION

SYNTAX:

C+A SUB B

- SUBTRACT B FROM A, GIVING C, USING MULTIPRECISION INTEGER ARITHMETIC. A AND B MAY BE IN ANY NUMERIC FORMAT, C WILL BE MP INTEGER. SEE \texttt{VADD}.
- USES: \texttt{VADD VFIX}

FUNCTION:

\begin{verbatim}
\texttt{\textbackslash v C+A SUB B}
\end{verbatim}

\[1\]
\texttt{\textbackslash aMPRECISION INTEGER SUBTRACT}

\[2\]
\texttt{C+(A+\texttt{FIX A})ADD-B+\texttt{FIX B}}

\texttt{\textbackslash v}

EXAMPLES:

\begin{verbatim}
A
0 1 2345678 9012345 6789012
B
0 222 3333333 4444444
A SUB B
0 1 2345456 5679012 2344568
A SUB 4
0 1 2345678 9012345 6789008
B SUB B
0 0
\end{verbatim}
Section V

Mathematical / Numerical Functions
AMORTIZE  MORTGAGE CALCULATION BY MONTHS

SYNTAX:  M+AMORTIZE DEBT, RATE, MONTHS

- DISPLAYS MORTGAGE TABLE INDICATING THE PROGRESSIVE DEBT REDUCTION, AND THE LEVEL PAYMENT SCHEDULE, AS IT BREAKS DOWN BETWEEN AMORTIZED DEBT AND INTEREST.
- DEBT IS TOTAL UNITS (E.G., DOLLARS) BORROWED.
- RATE IS ANNUAL INTEREST, AS PERCENTAGE (E.G., .095).
- MONTHS (E.G., 120 IF TEN YEAR MORTGAGE).
- RETURNS A MATRIX THAT RETAINS FULL PRECISION FOR SUMMARY CALCULATIONS. (SEE EXAMPLE)
- USES: \\n\nFUNCTION:

\[ M+AMORTIZE NV3;□IO \]
[4]  M=(MONTHS,5)×I+0
[5]  PAYMENT+DEBT×RATE×(1+(1+RATE)×MONTHS)
[7]  M[I]+(I+I+1),DEBT,PAYMENT,AMORTIZED,INTEREST
[8]  +(0<DEBT+NEWDEBT)/BACK
[9]  'M DEBT PAYMT AMORT INT OUTPUT(9 0\w10 -4\wM),', ', ', 9 2\wO 1\wM

EXAMPLES:

M+AMORTIZE 1000 .06 12

<table>
<thead>
<tr>
<th>M</th>
<th>DEBT</th>
<th>PAYMT</th>
<th>AMORT</th>
<th>INT</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1000.00</td>
<td>86.07</td>
<td>81.07</td>
<td>5.00</td>
</tr>
<tr>
<td>2</td>
<td>918.93</td>
<td>86.07</td>
<td>81.47</td>
<td>4.59</td>
</tr>
<tr>
<td>3</td>
<td>837.46</td>
<td>86.07</td>
<td>81.88</td>
<td>4.19</td>
</tr>
<tr>
<td>4</td>
<td>755.58</td>
<td>86.07</td>
<td>82.29</td>
<td>3.78</td>
</tr>
<tr>
<td>5</td>
<td>673.29</td>
<td>86.07</td>
<td>82.70</td>
<td>3.37</td>
</tr>
<tr>
<td>6</td>
<td>590.59</td>
<td>86.07</td>
<td>83.11</td>
<td>2.95</td>
</tr>
<tr>
<td>7</td>
<td>507.48</td>
<td>86.07</td>
<td>83.53</td>
<td>2.54</td>
</tr>
<tr>
<td>8</td>
<td>423.95</td>
<td>86.07</td>
<td>83.95</td>
<td>2.12</td>
</tr>
<tr>
<td>9</td>
<td>340.01</td>
<td>86.07</td>
<td>84.37</td>
<td>1.70</td>
</tr>
<tr>
<td>10</td>
<td>255.64</td>
<td>86.07</td>
<td>84.79</td>
<td>1.28</td>
</tr>
<tr>
<td>11</td>
<td>170.85</td>
<td>86.07</td>
<td>85.21</td>
<td>0.85</td>
</tr>
<tr>
<td>12</td>
<td>85.64</td>
<td>86.07</td>
<td>85.64</td>
<td>0.43</td>
</tr>
</tbody>
</table>

\[ +f 0 2\wM \]
1032.797156 1000 32.79715648
\[ a +______TOTAL INTEREST PAID \]
\[ a +______TOTAL DEBT REPAYED \]
\[ a +______TOTAL REPAYED \]
CONV

CONVERT DECIMAL VALUES TO ANY BASE [ DIGITS CONFRAC ]

SYNTAX:
R+BASE CONV DEC

- VALUES ARE CONVERTED TO CHARACTER STRINGS THAT RETAIN THEIR
  ARITHMETIC CAPABILITY.
- THE CHARACTER STRINGS CAN BE RECONVERTED BY VDEC.
- THE GLOBAL VARIABLE, DIGITS WILL SUPPORT UP TO BASE 36.
  THE CATENATION OF UNDERSCORED LETTERS AND OTHER CHARACTERS
  TO DIGITS WILL PERMIT LARGER BASES.
- NEGATIVE NUMBERS WILL BE TREATED CORRECTLY.
- FRACTIONS WILL BE CLOSELY APPROXIMATED.
- INTEGER CONVERSIONS, E.G., HEXADECIMAL, WILL BE EXACT.
- USES: VENC VDL VCONFRAC

FUNCTIONS:

\[ R+\text{BASE CONV DEC};\text{IO} \]

[1] \( \text{IO}<0 \)

[2] \( \text{DIGITS}='0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ' \)

[3] \( R\to 'DL' \to '[\text{DEC}<0], '0' DL \text{DIGITS}[\text{BASE ENC}][\text{DEC}] \)

[4] \( R+R, \text{BASE CONFRAC DEC} \)

\[ R+\text{BASE CONV DEC};\text{IO} \]

\[ R+B \text{ CONFRAC N;}\text{IO};\text{NN};\text{BB};\text{CT} \]

\[ \begin{align*}
[1] & \quad \text{CT}+1E^{-15} \\
[2] & \quad \to 0 \text{ IF } 0=\text{NN}+1||N,R+;\text{IO}+J+0 \\
[3] & \quad \text{BACK}:R+R,|\text{NN}+\text{BB}|B*J+J+1 \\
[4] & \quad \text{BACK IF } 1+\text{NN}+\text{BB}||\text{NN} \\
[5] & \quad R\to 'DL',\text{DIGITS}[R] \\
[6] & \quad \text{CONVERTS DECIMAL FRACTIONS} \]

EXAMPLES:

10DEC 10CONV 10DEC'-'1234.5678'
1234.5678
16DEC'20000'
131072
36CONV 123456789
21I3V9
36DEC 36CONV 1234567890123456
1234567890123456

ANALYSIS:

16 \text{ CONV 131072} \\
[3] \( R\to 'DL' \to '[\text{DEC}<0], '0' DL \text{DIGITS}[\text{BASE ENC}][\text{DEC}] \) \quad \text{ABSOLUTE VALUE} \\
131072 \\
[3] \( R\to 'DL' \to '[\text{DEC}<0], '0' DL \text{DIGITS}[\text{BASE ENC}][\text{DEC}] \) \quad \text{(SEE VENC)} \\
0 2 0 0 0 0 \\
[3] \( R\to 'DL' \to '[\text{DEC}<0], '0' DL \text{DIGITS}[\text{BASE ENC}][\text{DEC}] \) \quad \text{SELECTED} \\
020000 \\
[3] \( R\to 'DL' \to '[\text{DEC}<0], '0' DL \text{DIGITS}[\text{BASE ENC}][\text{DEC}] \) \quad \text{LEADING ZERO DELETE} \\
20000 \\
[3] \( R\to 'DL' \to '[\text{DEC}<0], '0' DL \text{DIGITS}[\text{BASE ENC}][\text{DEC}] \) \quad \text{IF NEGATIVE} \\
20000
DATE

COMPUTE NORMAL DATE FROM ASTRONOMERS' DAY NUMBER

**SYNTAX:**

\[ Z + \text{DATE} \ JS \]

- Returns month, day, year, style. (See \textit{VDayno})
- \textit{JS} is the Julian day number as would be found by \textit{VDayno}.
- \textit{JS} may be a single value or a vector. Optionally, it may be an array of shape \((N,2)\) where the second column is 0 or 1, stating for each Julian day whether the old (0) or new (1) calendar was in use. Normally, this style is computed automatically.

**FUNCTION:**

\[ \begin{align*}
V & \ Z + \text{DATE} \ JS; C; D; J; M; S; Y \\
[1] & \ a\text{CONVERT JULLIAN DAY NUMBER (AND OPTIONAL STYLE) TO DAY, MO, YEAR, STYLE.} \\
[2] & \ a\text{JS MAY ALSO BE A VECTOR OF JD'S OR AN ARRAY OF JD'S AND STYLES.} \\
[3] & \ JS+(2+(pJS),1 1)pJS \\
[4] & \ S+(J+2423434)v(J+2299171)\wedge(JS,2361221<J+JS[;\text{IO}])[;\text{IO}+1] \\
[5] & \ C+L(J+J-1684595)+36524.25 \\
[6] & \ J+J+(-S)\times(2-C)+[C+4]-[36524.25\times C] \\
[7] & \ Y+L(J+1)+365.25025 \\
[8] & \ J+J+31-1.365.25X Y \\
[9] & \ D+J-30.5875XM+1J+30.5875 \\
[10] & \ M+M+2-12XJ+M+11 \\
[11] & \ Z+M,D,(J+Y+100\times C-1),[IIO+0.5]S \\
\end{align*} \]

**EXAMPLES:**

- \[ \text{Z+DATE} 5 17 1977 \]
  \[ 2443281 \]
  \[ \text{DATE} Z \]
  \[ 5 17 1977 1 \]
  \[ \text{DATE} Z, Z+30 \]
  \[ 5 17 1977 1 \]
  \[ 6 16 1977 1 \]
  \[ \text{IF THE OLD STYLE CALENDAR WAS IN USE AFTER 1752, OR} \]
  \[ \text{THE NEW STYLE IN USE BEFORE THEN, THE USER MUST} \]
  \[ \text{GIVE THE STYLE. FOR EXAMPLE, IN THE USA:} \]
  \[ \text{Z+DATE} 1 1 1800 \]
  \[ 2378497 \]
  \[ \text{DATE} Z \]
  \[ 1 1 1800 1 \]
  \[ \text{BUT IN RUSSIA} \]
  \[ \text{DATE} 1 2pZ,0 \]
  \[ 12 21 1799 0 \]
SYNTAX:

Z=DAYNO DATE

- Days since 1/1/4713 B.C. (see VDATE)
- Date is month, day, year. It may be a single such triplet or a matrix, each row of which is a triplet.
- Optionally, the input may have a fourth component or column of 0 or 1 for each date, stating whether the old (0) or new (1) style calendar was in use. Normally, this is computed automatically.
- Day of the week may be computed by 1+7|1+DAYNO DATE. Sunday = 1, ..., Saturday = 7. (See VDAYS)

FUNCTIONS:

1. Z=DAYNO DATE;C;D;JP;M;S;Y
   - Compute Julian day number. Date is a vector or array whose rows are
   - Monthly, day, year, style. Style is an optional logical value = 1 if the
   - New style (Gregorian) calendar should be used. The Julian day is
   - A continuous count that began at 0 at noon, 1/1/4712 (i.e. 4713 BC).
   - DATE+(2+1,pDATE)pDATE
   - Z+100*(Y+DATE[;2]+DATE[;2]),[DATE][(M+DATE[;2])],[DATE]-0.5]+DATE[;1]+DATE[;1]
   - S+(Z>19230114)\(V\)(Z>15821025)\(V\)(DATE,Z>17520902)[;3+DATE]
   - C+(2*X-\(S\))+0.75\(X\)\(S\)\(0.01\)\(X\)\(Y-JF\)+2+M
   - Z+31+D+(\(367\times JF\)+\(M-2\)+12)-\(\frac{\(C-365.25\times 4712\times JF\)}{Y-JF}\)

EXAMPLES:

1. \(0.00\) is new moon; \(0.75\) is last quarter
2. \(R+2\times 1+29.53059\times 29.53059-9+DAYNO MDYS\)

```
DAYNO 5 17 1977
2443281
      Y+Z=4 3p2 28 1900 3 1 1900 2 28 2000 3 1 2000
2 28 1900
3 1 1900
2 28 2000
3 1 2000
DAYNO Z
2415079 2415080 2451603 2451605
As can be seen, 1900 was not a leap year, but 2000 will be one.
1+7|1+DAYNO 5 17 1977
3
A I.E. Tuesday
MOONPHASE 5 17 1977
0.99
A I.E. just before new moon
(DAYNO 5 17 1977)-DAYNO 1 1 1901
27805
A The age in days of the twentieth century.
A If the old style calendar was in use after 1752, or the
A new style in use before then, the style must be entered.
A For example, in Russia before the revolution (compare above)
DAYNO 2 28 1900 0
2415091
```
SYNTAX:

\[ N \text{+DAYS } D \quad D \text{+DATES } N \quad R \text{+NDATES } KM \]

- **GIVEN A NUMERIC VECTOR OR MATRIX OF THE FORM:**
  - MONTH, DAY, YEAR (1 30 1977), DAYS WILL RETURN
  - FOR EACH, THE NUMBER OF DAYS SINCE 1/1/1, INCLUSIVE,
    AS IF THE GREGORIAN CALENDAR HAD BEEN IN USE CONTINUALLY,
    WITH NO LOSS AT THE CHANGE (IN ENGLAND ON SEPTEMBER 14,
    1752.) FOR A BETTER FUNCTION, SEE \text{VDAYNO}. THE .7| OF DAYS, CAN SELECT THE DAYS OF THE WEEK,
    WITH 0+=SUNDAY; 1+=MONDAY, ETC.
- **DATES CONVERTS THE DAYS BACK INTO CALENDAR DATES.**
- **NDATES CONVERTS DATES AVAILABLE AS CHARACTER MATRICES
  OF THE FORM: '013077' TO THAT REQUIRED BY DAYS.

FUNCTIONS:

\[ \text{V } N \text{+DAYS } D ; P ; I ; \square IO \]

\[ [1] \quad D+=\left(1 I+1 0 D, 3) p D \quad \triangle IO+1 \right. \]

\[ [2] \quad P+=\left(1 9(0(0.1)4 1 00 4 000) D[; 3])[2 ; ;] \quad \text{[3]} \quad N+=\left(3 6 5 X D[; 3]-1 \right)+\quad \text{[4]} \quad N+=I P N+D[; 2]+(1 30.56 x D[; 1])-3 0+(D[; 1] x 3) x 2-P \quad \text{[5]} \quad \text{N+ NOT ACCURATE PRIOR TO 1753. USE \text{VDAYNO}.} \]

\[ \text{V } D \text{+DATES } N ; Y ; M ; P ; R ; \square IO \]

\[ [1] \quad M+=\left(0(0.1)4 1 00 4 000) Y+=\left(1 3 6 4+D, N)+3 6 5 .2 4 2 5) \quad +0, \quad \square IO+1 \quad \text{[2]} \quad D+=D-1 0 \quad +(R+D>M[; 2]) P+=\left(3 6 5 X Y-1)+\quad \text{[3]} \quad D+=D-1 3 0.56 \times M+\left(D+3 0+D+(D>5 9+P)x 2-P+(R+P)[; 1])+3 0.56 \times F Y \quad \text{[4]} \quad D+=\left(1 P N), 3) p M , D, 0 \quad \text{[5]} \quad \text{R+ NOT ACCURATE PRIOR TO 1753. SEE \text{VDAYNO}.} \]

\[ \text{V } R \text{+NDATES } KM \]

\[ [1] \quad R+=\left(1+P K M), 3) p Z, 1 1 0 1 1 0 1 1 0 \quad \text{[2]} \quad R[; \square IO+2]+R[; \square IO+2]+1 9 0 0 \quad \text{[3]} \quad \text{R+ASSUMES 20TH CENTURY} \]

EXAMPLES:

\[ \text{V } R \text{+PAYDAY MDY; \square IO} \]

\[ [1] \quad \text{R+FRIDAY, ON OR BEFORE MDY} \quad \text{[2]} \quad R+(D A Y S M D Y)-1 7+\square IO+0 \quad \text{[3]} \quad R+DATES R[; (7 R) \setminus 5] \quad \text{V} \quad \text{PAYDAY 6 3 0 1 9 7 7} \]

6 24 1977

AA

081118
021926
031354
062758

DATES DAYS NDATES AA

<table>
<thead>
<tr>
<th>8</th>
<th>11</th>
<th>1918</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>19</td>
<td>1926</td>
</tr>
<tr>
<td>3</td>
<td>13</td>
<td>1954</td>
</tr>
<tr>
<td>6</td>
<td>27</td>
<td>1958</td>
</tr>
</tbody>
</table>
DEC
CONVERT TO DECIMAL

SYNTAX:
R+BASE DEC Q

- Character vectors belonging to the global digits, '\', representing scalar numbers in any base, will seem to be converted to their decimal values, with which ordinary calculations can be made.
- . will be understood as separating the integer portion from any possible fraction. Fractions will be closely approximated. Integers will be exact, unless they are forced to float.
- Arithmetic results can be converted to other bases through the use of \( \text{CONV} \).
- \( \text{DIGITS} \) will support bases \( 2 \leq \text{BASE} \leq 36 \).

FUNCTION:

\[
V \cdot R + \text{BASE DEC Q} ; \square_{IO} ; P ; S \\
[1] \quad Q + (S + Q \cdot '1' - '1') / (Q + (P + Q \cdot '1') / Q) \\
[2] \quad \text{CHARACTER ERROR ESCAPE} - A / Q \cdot \text{BASE} + R \cdot \text{DIGITS} \\
[3] \quad R + (1 - 2 \times 0 \cdot S) \times (\text{BASE} \cdot 0 \cdot 1 + v \cdot \sqrt{P} \times 1 \cdot R \cdot \text{Q} \cdot \square_{IO} + 0
\]

EXAMPLES:

\[
(10 \text{ DEC}'1234') = 1\times1234 \\
1 \\
16 \text{ DEC}'20000' \\
131072 \\
(16 \text{ DEC}'20000') + 16 \text{ DEC}'FFFF' \\
196607 \\
16 \text{ CONV } (16 \text{ DEC}'FFFF') - 16 \text{ DEC}'1234' \\
EDCB \\
10 \text{ CONV } -12345.6789 \\
-12345.6789999999988 \\
10 \text{ CONV } 2 \times -16 \\
.00001525878906 \\
16 \text{ DEC } 16 \text{ CONV } 2 \times -16 \\
1.525878906E^5 \\
.0001 \\
16 \text{ CONV } 2 \times -64 \\
.0 \\
16 \text{ CONV } 2 \times -32 \\
.00000001 \\
\]

ANALYSIS:

\[
16 \text{ DEC}'-EDCB.125' \\
[3] \quad R + (1 - 2 \times 0 \cdot S) \times (\text{BASE} \cdot 0 \cdot 1 + v \cdot \sqrt{P} \times 1 \cdot R \cdot \text{Q} \cdot \square_{IO} + 0 \\
\quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \}
DROUNDS DISTRIBUTIVE ROUNDING OF A VECTOR TO ARBITRARY SCALAR UNIT

SYNTAX: R+U DROUNDS V

- Rounding the elements of a vector before summation may cause an error in the sum. Round-off errors do not necessarily compensate. It would be good practice to carry maximum precision until the final summation, then rounding the sum.
- When this is not possible, we would still want the rounded sum to equal the sum of the rounded elements. See vrounds

FUNCTION:

\[ V \text { R+U DROUNDS V} \]

\[ E+1 \lfloor V+U+\square CT+\square IO+0 \]

\[ N+(\lfloor 0.5++/V \rfloor)-+/LV \]

\[ R+U(\lfloor V \rfloor)+N>\Delta \gamma E \]

\[ V \]

EXAMPLE:

A

0.86 0.21 0.95 0.9 0.7 0.69 0.44 0.59 0.16 0.57 0.06 0.47 0.6 0.46 0.93 0.61
+/A

9.2
+/ .1 ROUNDS A

9.4
+/ .1 ROUNDS A

9.2

ANALYSIS: .1 DROUNDS A

\[ E+1 \lfloor V+U+\square CT+\square IO+0 \]

\[ 0.6 0.1 0.5 1 1 0.9 0.4 0.9 0.6 0.7 0.6 0.7 1 0.6 0.3 0.1 \]

\[ N+(\lfloor 0.5++/V \rfloor)-+/LV \]

\[ 8 2 9 8 6 6 4 5 1 5 0 4 5 4 9 6 \]

\[ R+U(\lfloor 0.5++/V \rfloor)-+/LV \]

92

\[ N+(\lfloor 0.5++/V \rfloor)-+/LV \]

10 ADJUSTMENTS NEEDED, BUT WHERE?

\[ R+U(\lfloor V \rfloor)+N>\Delta \gamma E \]

LOCATION, BY SEVERITY

10 14 11 2 0 3 12 4 8 5 7 6 1 9 13 15

\[ R+U(\lfloor V \rfloor)+N>\Delta \gamma E \]

HERE! (10 WORST REPRESENTED BY FLOOR)

0 0 0 1 1 1 0 1 1 1 1 1 0 0

\[ R+U(\lfloor V \rfloor)+N>\Delta \gamma E \]

8.6 2.1 9.5 9 7 6 9 4.4 5.9 1.6 5.7 0.6 4.7 6 4.6 9.3 6.1

\[ R+U(\lfloor V \rfloor)+N>\Delta \gamma E \]

8 2 9 8 6 6 4 5 1 5 0 4 5 4 9 6

\[ R+U(\lfloor V \rfloor)+N>\Delta \gamma E \]

8 2 9 9 7 7 4 6 2 6 1 5 6 5 9 6

69
ENC

GENERATE SUFFICIENT ENCODING POSITIONS

SYNTAX:

R+S ENC A

ENCODE(τ) AND DECODE(τ) WOULD BE FULLY COMPLEMENTARY IF SUFFICIENT RADIX POSITIONS COULD BE SUPPLIED. VENC WILL PERFORM THE FULL REPRESENTATION OF ITS ARGUMENT, ACCORDING TO THE RADIX, S. (S>1)∧(1≤l/A)

FUNCTION:

\[ \triangledown R+S ENC A \]
[1] a 1<s 1≤l/A
[2] R+((1+S⊗1[[/,A])pS)TA

\[ \triangledown \]

EXAMPLES:

12 ENC 143 144 145
0 0 0
0 1 1
11 0 0
11 0 1

1212 ENC 143 144 145
143 144 145
143 0 1 (TWO TWELVES NOT ENOUGH)

ANALYSIS:

12 ENC 143 144 145
[2] R+((1+S⊗1[[/,A])pS)TA

143 144 145
[2] R+((1+S⊗1[[/,A])pS)TA

12
[2] R+((1+S⊗1[[/,A])pS)TA

GUARDING AGAINST ZERO

145
[2] R+((1+S⊗1[[/,A])pS)TA

2.002784991
[2] R+((1+S⊗1[[/,A])pS)TA

3
[2] R+((1+S⊗1[[/,A])pS)TA

GUARD HIGH-ORDER POSITION

4
[2] R+((1+S⊗1[[/,A])pS)TA

12 12 12 12
[2] R+((1+S⊗1[[/,A])pS)TA

0 0 0
0 1 1
11 0 0
11 0 1
FREQ

FREQUENCY DISTRIBUTION OF ELEMENTS [ SORTDA KFORM ]

SYNTAX: R+FREQ A

- THE ARGUMENTS MUST BE NUMERIC CODES, OR NUMERIC REPRESENTATIONS OF CHARACTER GROUPS.
- FREQUENCY WILL APPEAR IN DESCENDING ORDER, WITHIN WHICH THE CATEGORIES WILL ASCEND.
- USES: VDREP

FUNCTIONS:

\[ \text{V } R+FREQ A \]

[1] A IS A NUMERIC STRUCTURE. USE LjNFORM OR NFORM TO CONVERT.
[3] IF THE ARGUMENT IS A CONVERTED CHARACTER MATRIX,
[4] THE SECOND COLUMN OF THE RESULT CAN BE RECONVERTED BY KFORM,
[5] \( R+(2)\text{SORTDA}[];[]IO+0.5]+'A'\text{.}=R+DREP A+,A \)

\[ \text{V } K+C KFORM N;[]IO \]

EXAMPLES:

\[ R+FREQ 55P5 \]

\[ FREQ ?5p5 \]

\[ 2 2 \]

\[ 2 4 \text{ (2 TWO'S, 2 FOURS, 1 THREE, NO ONES, NO FIVES) } \]

\[ 1 3 \]

\[ N+(AV,"\"\")NFORM VERT'THE QUICK BROWN FOX JUMPED OVER THE LAZY DOG'S BACK' \]

\[ (V0 \text{-1+R}),\text{MATRIX (AV,"\"\") KFORM 0 1+R+FREQ N} \]

\[ \text{(NINE BLANKS)} \]

\[ \text{V } R+DA \text{ SORTDA M;N} \]

\[ [1] DA IS A PAIR OF COLUMN NUMBERS IN USER'S ORIGIN. \]

\[ 2 R \text{ A CONTROLS THE INITIAL ASCENDING SORT. } \]

\[ 2 T \text{ D CONTROLS THE FINAL DESCENDING SORT. } \]

\[ 2 U \text{ M IS A NUMERIC MATRIX WHICH MAY RESULT FROM NFORM. } \]

\[ 1 P \text{ R+N[\text{V,(N+M[1,M[-1+DA]];)]1+DA]};] \]

\[ \text{V } \]

\[ 1 G \]

\[ 1 J \]

\[ 1 L \]

\[ 1 M \]

\[ 1 N \]

\[ 1 P \]

\[ 1 Q \]

\[ 1 S \]

\[ 1 V \]

\[ 1 W \]

\[ 1 X \]

\[ 1 Y \]

\[ 1 Z \]

1 
PI

COMPUTE PI TO ARBITRARY PRECISION

SYNTAX:

SUM+PI N

- COMPUTE PI (3.14159+) TO 7×N DECIMAL DIGITS OF PRECISION
- THE ARCSIN POWER SERIES (6·x-10.5) IS SUMMED--NOT THE FASTEST KNOWN METHOD, BUT FAR FROM THE SLOWEST
- RUNNING TIME IS PROPORTIONAL TO N×2
- USES: VADD VMUL VDIV

FUNCTION:

\[
\begin{align*}
\text{SUM+PI N;I;TERM;REM} \\
[1] &\text{?COMPUTE PI TO 7×N DECIMAL PLACES BY THE POWER SERIES FOR 6·x-10.5} \\
[2] &SUM+TERM+0,(N+I+1)+3 \\
[3] &\text{LOOP:TERM+TERM MUL I)DIV 4×I+1} \\
[4] &SUM+SUM ADD TERM DIV I+2 \\
[5] &+(v/TERM=0)/LOOP \\
[6] &SUM+(-N),1+SUM \\n\end{align*}
\]

EXAMPLES:

- 6 3 1415926 5358979 3238462 64433832 7950288 4197136
  \text{COMPUTE THE RAMANUJAN NUMBER *OK*.5 FOR K=163} \\
- 5 163 0 0 0 0 0
  \text{FORMAT Z+FSQRT K} \\
12.7671453 3480370 4661710 9520097 8089234
  \text{FORMAT Z+P FMUL Z} \\
40.1091699 9113251 9755350 0836229 0414003
  \text{FORMAT Z+FEXP Z} \\
2625 3741264 0768743.9999999 99999825 0066319 1466030 7724958
  \text{FOR NUMEROUS OTHER VALUES OF K, THESE NUMBERS ARE VERY}
  \text{CLOSE TO PERFECT INTEGERS. ALL THE MORE REMARKABLE THAT}
  \text{RAMANUJAN DISCOVERED THEM IN 1915 WITHOUT THE AID OF A}
  \text{COMPUTER.}
QPROBF  COMPUTE CHI SQUARE PROBABILITY FUNCTION

SYNTAX:  Z+CHISQ QPROBF NU

- COMPUTE THE PROBABILITY OF A GIVEN CHI SQUARE VALUE
  OCCURRING FOR A GIVEN NU (NUMBER OF DEGREES OF FREEDOM)
- NU IS ROUNDED DOWN TO THE NEXT LOWER EVEN INTEGER
- NOTE THE EXTREME ELEGANCE WITH WHICH IT IS POSSIBLE
  IN APL TO EXPRESS A POWER SERIES

FUNCTION:

∇ Z+CHISQ QPROBF NU
[1]  aCOMPUTE Q(CHISQ|NU), WHERE NU IS ROUNDED DOWN TO AN EVEN INTEGER
[2]  Z+(*-CHISQ+2)×(CHISQ+2)××\(1\-\OPTION-\1\NU+2

∇

EXAMPLES:

5.78 QPROBF 20
0.999164
27.3 QPROBF 20
0.127033
27.3 QPROBF 40
0.93691

ANALYSIS:  5.78 QPROBF 20
[2]  Z+(*-CHISQ+2)×(CHISQ+2)××\(1\-\OPTION-\1\NU+2

0 1 2 3 4 5 6 7 8 9
[2]  Z+(*-CHISQ+2)×(CHISQ+2)××\(1\-\OPTION-\1\NU+2
1 1 2 3 4 5 6 7 8 9
[2]  Z+(*-CHISQ+2)×(CHISQ+2)××\(1\-\OPTION-\1\NU+2
1 1 2 6 24 120 240 4032 36288
[2]  Z+(*-CHISQ+2)×(CHISQ+2)××\(1\-\OPTION-\1\NU+2
(NOT ALL RECIPROCALS SHOWN)
1 1 0.5 0.1666666667 0.0416666667 0.0083333333 0.001388888889 0.000183333333
[2]  Z+(*-CHISQ+2)×(CHISQ+2)××\(1\-\OPTION-\1\NU+2
2.89
[2]  Z+(*-CHISQ+2)×(CHISQ+2)××\(1\-\OPTION-\1\NU+2
17.9782608
[2]  Z+(*-CHISQ+2)×(CHISQ+2)××\(1\-\OPTION-\1\NU+2
2.89
[2]  Z+(*-CHISQ+2)×(CHISQ+2)××\(1\-\OPTION-\1\NU+2
-2.89
[2]  Z+(*-CHISQ+2)×(CHISQ+2)××\(1\-\OPTION-\1\NU+2
0.05557621261
[2]  Z+(*-CHISQ+2)×(CHISQ+2)××\(1\-\OPTION-\1\NU+2
0.9991636444
ROMAN CONVTRNT INTEGER TO ROMAN NUMERALS

SYNTAX:
R=ROMAN N

• ROMAN NUMERALS MAY BE REQUIRED FOR CERTAIN TYPES OF PAGE
OR PARAGRAPH NUMBERING. THEY ALSO ILLUSTRATE THAT THERE IS
A DISTINCTION BETWEEN THE VALUE OF A NUMBER AND ITS REPRES-
SENTATION. N IS AN INTEGER GREATER THAN ZERO. R IS A
CHARACTER VECTOR REPRESENTING N AS A ROMAN NUMERAL.

FUNCTION:

```
V R+ROMAN N;I;I0  I0=0
[1] I0+0 1000T'pN
[2] R+0 5T10 10 10 10±N+I[1]
[3] N+,Q(i4)♦,Qo[R-1 30.×4=R[1;)],[0]R[0;]4=R
[4] R+(I[0]p'M'),N/,Q4 16p'×N×DCMDLXCLVIXV'
[5] V
```

EXAMPLES:
ROMAN 7
VII
ROMAN 77
LXXVII
ROMAN 977
CMLXXVII
ROMAN 1977
MCMLXXVII
ROMAN 10000
MMMMMMMMM
ROUND SELECTIVE SYMMETRICAL ROUNDING

SYNTAX:

\[ R \triangleq U \text{ ROUNDS } A \]

- TO ROUNDOFF NUMBERS TO ANY GIVEN UNITS
- TO ROUND NEGATIVE NUMBERS AWAY FROM ZERO
- RESULT WILL BE THE NEAREST MULTIPLE OF THE CORRESPONDING UNIT.

FUNCTION:

\[ \nabla R \triangleq U \text{ ROUNDS } A \]

1. \( U \) IS A SCALAR OR CONFORMABLE STRUCTURE OF SPECIFIED UNITS
2. \( R \triangleq (xA) \times U \times 0.5 + |A + U| \)

EXAMPLE:

10 0.01 ROUNDS 5287 1234.006
5290 1234.01

ANALYSIS:

\[
\begin{align*}
A &= 3.6 145 -150 -151 1.027 \\
U &= 1.5 3 7 7 0.03 \\
\end{align*}
\]

1. \( R \triangleq (xA) \times U \times 0.5 + |A + U| \)

- THE CORRESPONDING UNITS

\[
\begin{align*}
1.5 3 7 7 0.03 \\
\end{align*}
\]

2. \( R \triangleq (xA) \times U \times 0.5 + |A + U| \)

- NORMALIZED

\[
\begin{align*}
2.4 48.33333333 -21.42857143 -21.57142857 34.23333333 \\
\end{align*}
\]

3. \( R \triangleq (xA) \times U \times 0.5 + |A + U| \)

- ABSOLUTE VALUES

\[
\begin{align*}
2.4 48.33333333 21.42857143 21.57142857 34.23333333 \\
\end{align*}
\]

4. \( R \triangleq (xA) \times U \times 0.5 + |A + U| \)

- HALF-ADJUSTMENT ADDED

\[
\begin{align*}
2.9 48.33333333 21.92857143 22.07142857 34.73333333 \\
\end{align*}
\]

5. \( R \triangleq (xA) \times U \times 0.5 + |A + U| \)

- FLOOR

\[
\begin{align*}
2 48 21 22 34 \\
\end{align*}
\]

6. \( R \triangleq (xA) \times U \times 0.5 + |A + U| \)

- NORMALIZATION REVERSED

\[
\begin{align*}
3 144 147 154 1.02 \\
\end{align*}
\]

7. \( R \triangleq (xA) \times U \times 0.5 + |A + U| \)

- THE ORIGINAL SIGNS

\[
\begin{align*}
1 1 -1 -1 1 \\
\end{align*}
\]

8. \( R \triangleq (xA) \times U \times 0.5 + |A + U| \)

- NEGATIVE NUMBERS RESTORED

\[
\begin{align*}
3 144 -147 -154 1.02 \\
\end{align*}
\]
TO NUMERIC VECTORS IN EQUAL INCREMENTS [ BY IN FROM ]

SYNTAX:  

\[ \begin{align*}
R+A & \text{ TO } B \\
R+A & \text{ TO } B \text{ BY } C \\
R+N & \text{ FROM } A \\
R+N & \text{ FROM } A \text{ BY } C \\
R+A & \text{ TO } B \text{ IN } M \\
R+N & \text{ FROM } A \text{ BY } C
\end{align*} \]

A: STARTING VALUE
B: LAST VALUE (OR BOUNDARY VALUE)
C: INCREMENT (POSITIVE OR NEGATIVE BUT NOT ZERO)
M: NUMBER OF INTERVALS DESIRED (M ≠ 0).
N: NUMBER OF VALUES DESIRED
R: RESULTING NUMERIC VECTOR WITH EQUAL INCREMENTS

WHEN THE FUNCTIONS 'TO' AND 'FROM' ARE USED ALONE, THE INCREMENT IS UNDERSTOOD TO BE ONE. SEQUENCES OF ANY OF THE ABOVE FORMS ARE ALSO POSSIBLE, PROVIDED THAT THEY ARE SEPARATED BY COMMAS AS SHOWN IN THE EXAMPLES.

EXAMPLES:

\[ \begin{align*}
5 & 6 7 8 9 10 \leftrightarrow 5 \text{ TO } 10 \\
4.1 & 5.1 6.1 \leftrightarrow 4.1 \text{ TO } 7 \\
5 & 4 3 2 \leftrightarrow 5 \text{ TO } 2 \\
0 & 2 4 6 \leftrightarrow 0 \text{ TO } 6 \text{ IN } 3 \\
6 & 4 2 0 \leftrightarrow 6 \text{ TO } 0 \text{ IN } 3 \\
5 & 7 9 \leftrightarrow 5 \text{ TO } 10 \text{ BY } 2 \\
5 & 5 \leftrightarrow 5 \text{ TO } 10 \text{ BY } 6 \\
3 & 4 5 6 7 \leftrightarrow 5 \text{ FROM } 3 \\
15 & 12 9 6 \leftrightarrow 4 \text{ FROM } 15 \text{ BY } -3
\end{align*} \]

FUNCTIONS:

\[ \begin{align*}
\n & V \quad Z+A \text{ TO } B; D; R; X; RIO \\
[1] & \quad \text{IO}+0 \\
[2] & \quad R+pZ+1,B \\
[3] & \quad Z+,Z \\
[5] & \quad D+Z[1]-A \\
[6] & \quad +(3>R)+L1 \\
[7] & \quad B+A+(D+X)×1+X \\
[8] & \quad ⇔L2 \\
[9] & \quad L1:B+A+(X×D)×1+|D+X \\
[10] & \quad L2:Z+B,(2+R>1)+Z \\
\n & V \quad Z+B \text{ BY } C \\
[1] & \quad 'ZERO IS INVALID ARGUMENT' \text{ HANG } 0=1+C \\
[2] & \quad Z+(1,pZ)pZ+B,C \\
\n & V \quad Z+B \text{ IN } M \\
[1] & \quad 'ZERO IS INVALID ARGUMENT' \text{ HANG } 0=1+M \\
[2] & \quad Z+(1,1,pZ)pZ+B,M \\
\n & V \quad Z+N \text{ FROM } A; R; RIO \\
[1] & \quad \text{IO}+0 \\
[2] & \quad R+pZ+1,A \\
\end{align*} \]

NOTE: THIS IS AN EXAMPLE OF LINKING APL FUNCTIONS TOGETHER. THE CORE FUNCTIONS, 'TO' AND 'FROM', DETERMINE WHETHER OR NOT THERE WAS A 'BY' OR 'IN' CLAUSE FROM THE RANK OF THEIR RIGHT ARGUMENTS.
TRUNC  TRUNCATE HIGHER AND LOWER ORDER DIGITS

SYNTAX:  R+U TRUNC A

- SELECT PARTICULAR DECIMAL DIGIT POSITIONS
- EXPLICIT (INPUT) DECIMAL FRACTIONS WILL BE RETURNED CORRECTLY. LOW-ORDER DIGITS OF COMPUTED FRACTIONS MAY NOT BE EXACT IN DECIMAL REPRESENTATION.

FUNCTION:

\[ \text{TRUNC}(R+U, A) \]

[1] IF \( U \) IS ANY POWER OF TEN, THEN THE CORRESPONDING DECIMAL POSITION OF \( A \) IS RETURNED.
[2] IF \( U \) IS A UNIT DIVISOR, \( A \) IS FIRST CONVERTED TO THE NEW UNIT,
[3] THEN THE NEW UNITS PLACE IS RETURNED.

EXAMPLE:

\( (10 \times 5) \text{TRUNC} 12345.0 \)

5 4 3 2 1

ANALYSIS:

\( \text{TRUNC}(100 \div 7) \)

[1] \( R + 10 | L | A \div U \)

0 1 1 1 0

\( \text{TRUNC}(142.8571428571428) \)

142 14 1

\( \text{TRUNC}(142.8571428571429) \)

142 14 1

\( \text{TRUNC}(142.8571428571428) \)

142 14 1

\( \text{TRUNC}(142.8571428571429) \)

142 14 1
ZDIV ZERO TOLERANT DIVISION [ CDIV ]

SYNTAX: \( R+N \) ZDIV \( D \)

- Domain errors are undesired in commercial matrix operations where zeros usually indicate unavailable information.
- \( N \) and \( D \) are conformable numeric structures or scalars in any combination.
- Zero will be returned instead of the domain error.

FUNCTIONS:

\( \lor R+N \) ZDIV \( D \)

- [1] \( R \) returns zero when divisor is zero
- [2] \( R \) returns unity when \( N \) and \( D \) are both zero
- [3] \( R+(N\times R)+(D\times R)+\neg R+(N=0)\lor D\neq 0 \)

\( \lor \) CDIV \( D \)

- [1] \( R \) commercial division: returns zero if \( D=0 \)
- [2] \( R+(N\times R)\lor D+\neg R+D\neq 0 \)

EXAMPLES:

\[ \begin{align*}
A &= 2 \quad 0 \quad 2 \quad 0 \\
B &= 3 \quad 3 \quad 0 \quad 0 \\
A \text{ ZDIV } B &= 0.6666666667 \quad 0 \quad 0 \quad 1
\end{align*} \]

+ Non-zero divided by zero

ANALYSIS:

\[ \begin{align*}
4 \quad 0 \quad 4 \quad 0 \text{ ZDIV } 2 \quad 2 \quad 0 \quad 0
\end{align*} \]

- [3] \( R+(N\times R)+(D\times R)+\neg R+(N=0)\lor D\neq 0 \)

\[ \begin{align*}
1 \quad 1 \quad 0 \quad 0
\end{align*} \]

- [3] \( R+(N\times R)+(D\times R)+\neg R+(N=0)\lor D\neq 0 \)

\[ \begin{align*}
0 \quad 1 \quad 0 \quad 1
\end{align*} \]

- [3] \( R+(N\times R)+(D\times R)+\neg R+(N=0)\lor D\neq 0 \)

\[ \begin{align*}
1 \quad 1 \quad 0 \quad 1
\end{align*} \]

--------DIVISION MAY PROCEED FOR THESE CASES

\[ \begin{align*}
1 \quad 1 \quad 0 \quad 1
\end{align*} \]

--------BUT NOT THIS CASE

\[ \begin{align*}
0 \quad 0 \quad 1 \quad 0
\end{align*} \]

- [3] \( R+(N\times R)+(D\times R)+\neg R+(N=0)\lor D\neq 0 \)

\[ \begin{align*}
2 \quad 2 \quad 0 \quad 0
\end{align*} \]

- [3] \( R+(N\times R)+(D\times R)+\neg R+(N=0)\lor D\neq 0 \)

\[ \begin{align*}
2 \quad 2 \quad 1 \quad 0
\end{align*} \]

- [3] \( R+(N\times R)+(D\times R)+\neg R+(N=0)\lor D\neq 0 \)

\[ \begin{align*}
4 \quad 0 \quad 0 \quad 0
\end{align*} \]

- [3] \( R+(N\times R)+(D\times R)+\neg R+(N=0)\lor D\neq 0 \)

\[ \begin{align*}
2 \quad 0 \quad 0 \quad 1
\end{align*} \]
Section VI

Utility & Miscellaneous Functions
COMB

ALL COMBINATIONS OF ELEMENTS [DEBLANK UNIQ]

SYNTAX:  

R+A COMB B

- Juxtaposes each unique element of A with each unique element of B, disregarding blanks.
- A and B can be character or numeric structures.
- Uses: VFORMAT VDEBLANK VUNIQ

FUNCTIONS:

\[ \vee R+A \text{ COMB B} \]

[1] \( A+\text{CFORMAT}, \text{DEBLANK}, \text{AND UNIQ CLEAN UP} \)
[2] \( A+\text{GLOBALS} A \text{ AND } B, \text{WHICH ARE LOCAL HERE.} \)
[3] \( A+\text{CFORMAT} \)
[4] \( A+\text{DEBLANK} \)
[5] \( A+\text{UNIQ} \)
[6] \( R++,p((pB),pA)pA),,[[IO+0.5]],((pA),pB)pB \)

\[ \vee \]

EXAMPLES:

'AABC'COMB\6

\[ \vee \text{DEBLANK} \]

\[ [1] \quad A+(A=' ') / A+,A \quad \Delta \quad B+(B=' ') / B+,B \]

\[ \vee \]

\[ \text{UNIQ} \]

\[ [1] \quad A+\text{DREP} A \quad \Delta \quad B+\text{DREP} B \]

\[ \vee \]

1 2 3 COMB 234 345 1.1

1 234
1 345
1 1.1
2 234
2 345
2 1.1
3 234
3 345
3 1.1
**CVEC**
BUILD COMPRESSION OR LOGICAL VECTOR

**SYNTAX:**
R+N CVEC LOC

- Binary vectors of arbitrary length with arbitrary zeros at numbered positions, in user's origin.
- Can generate input to VXVEC.

**FUNCTION:**

\[
\begin{align*}
\text{\#} & \quad \text{R+N CVEC LOC} \\
1 & \quad R+Np1 \\
2 & \quad R[LOC]=0 \\
3 & \quad \text{a returns a compression vector that can select all but LOC} \\
4 & \quad \text{a LOC is desired row of \( VLOC (\downarrow 0=0) \), or similar numeric vector} \\
5 & \quad \text{a N is original length of axis to be compressed} \\
\end{align*}
\]

**EXAMPLES:**

17 CVEC 1 2 3 5 7 11
0 0 0 1 0 1 0 1 1 1 0 1 1 1 1 1

32 CVEC (0=4\(\uparrow 21)\(\downarrow 21)
1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1

1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1
DELETE TRAILING INSIGNIFICANT CHARACTERS OR VALUES

**SYNTAX:**

```
R+V DT A
```

- INSIGNIFICANT CHARACTERS OR VALUES, AS DEFINED IN V, THAT APPEAR ON THE RIGHT SIDE OF AN ARRAY, WILL BE DROPPED.
- THE ORIGINAL RANK OF A WILL BE PRESERVED.
- AN EMPTY ARRAY IS RETURNED IF NOTHING SIGNIFICANT REMAINS.

**FUNCTION:**

```
V R+V DT A
[1] R+((-1+pA),I,+/v\~AeV)+A
```

**EXAMPLES:**

```
0 DT 2 4p4+1
1
1
'**DT 2 4p'MN**MNM**'
MN*
MNM
p' 'DT 3'+K'
1
TO DELETE TRAILING BLANKS
```

**ANALYSIS:**

```
'* ?'DT'GOOD '?'
```

```
[1] R+((-1+pA),I,+/v\~AeV)+A
```

```
GOOD ?
[1] R+((-1+pA),I,+/v\~AeV)+A
```

```
* ?
[1] R+((-1+pA),I,+/v\~AeV)+A
```

```
0 0 0 0 1 1 1
[1] R+((-1+pA),I,+/v\~AeV)+A
```

```
0 0 0 1 1 1
[1] R+((-1+pA),I,+/v\~AeV)+A
```

```
4
[1] R+((-1+pA),I,+/v\~AeV)+A
```

```
4
[1] R+((-1+pA),I,+/v\~AeV)+A
```

```
7
[1] R+((-1+pA),I,+/v\~AeV)+A
```

```
4
[1] R+((-1+pA),I,+/v\~AeV)+A
```

```
GOOD
```

81
EASTER

COMPUTE THE DATE OF EASTER

SYNTAX:

Z+EASTER YEAR

• COMPUTE THE DATE OF EASTER FOR ANY YEAR SINCE 33 AD
• YEAR MAY BE A SINGLE YEAR OR VECTOR OF YEARS.
IT MAY ALSO BE AN ARRAY OF SHAPE (N,2) WHERE THE
SECOND COLUMN IS 0 OR 1 FOR EACH YEAR STATING
WHETHER THE OLD (0) OR NEW (1) STYLE CALENDAR
WAS IN EFFECT THEN. NORMALLY, THIS IS COMPUTED
AUTOMATICALLY.

FUNCTION:

\[ Z+EASTER \text{ } YS; C; EPACT; G; N; X; Y \]

[1] \text{ } COMPUTE EASTER FOR YEAR } Y, \text{ OPTIONAL STYLE } S.
[2] \text{ } YS MAY ALSO BE A VECTOR OF YEARS OR AN ARRAY OF YEARS AND STYLES.
[3] \text{ } YS+(2+(YS), 1, 1)YS
[4] \text{ } S+(Y>1922) v(Y>1583) \land (YS, 1752<Y+YS[, IO][; IO+1])
[5] \text{ } +0 X1 p Z+(33 \bot Y)/'EASTER WASN'T CELEBRATED THAT EARLY.'
[6] \text{ } X+S2-0.75 C+1 ||0.01 Y
[7] \text{ } EPACT+30120+(S\times 10+1 \bot 0.32 \times C-15)+(11 \times G+1+19 \bot Y)+X
[8] \text{ } N=44-EPACT+S \times (EPACT=24) v(EPACT=25) \land G>11
[9] \text{ } N+N+30 \times N<21
[10] \text{ } N=N+7-7 \bot N+7 \times 1.25 Y
[11] \text{ } +0 X1 p Z+N
[12] \text{ } Z+'EASTER ON ', ((6 \times 30.5-\bot N)+'MARCH APRIL '), (\bot 1+31 \bot 1+N), ', ', \times 1+Y

EXAMPLES:

EASTER 1978
EASTER ON MARCH 26, 1978
EASTER 1865
EASTER ON APRIL 16, 1865
EASTER 1
EASTER WASN'T CELEBRATED THAT EARLY.
\text{ } A \text{ VECTOR INPUT PRODUCES A VECTOR OUTPUT OF THE}
\text{ } DAY NUMBERS IN MARCH.
EASTER 1978 1865
26 47
\text{ } WHEN OLD STYLE WAS KNOWN TO BE IN USE AFTER 1752,
\text{ } OR NEW STYLE BEFORE THEN, YOU MUST GIVE THE STYLE.
\text{ } FOR EXAMPLE, RUSSIA BEFORE THE REVOLUTION:
EASTER 1 2p1865 0
EASTER ON APRIL 11, 1865
**EXTEND**

**EXTEND VECTOR WITH LAST VALUE**

**SYNTAX:**

\[ R+N \text{ EXTEND } V \]

- THE APL {} would extend a vector by padding it with zeros or blanks.
- EXTEND will fill the space remaining on the right with the rightmost value.
- THIS will happen only if \( N > pV \).
- EXTEND returns a vector of length \( N \), or \( pV \), whichever is greater.

**FUNCTION:**

\[ \text{\theta} R+N \text{ EXTEND } V \]

```
```

**EXAMPLES:**

```
10 EXTEND 0 0 0 1
0 0 0 1 1 1 1 1 1
(30 EXTEND 'INDEX ITEM-'),' 20'
INDEX ITEM----------------- 20
12 EXTEND 'THIS WILL NOT BE PADDED WITH-
THIS WILL NOT BE PADDED WITH-
```

**ANALYSIS:**

```
33 EXTEND ITEM 4.'
ITEM 4.
```

```
7
```

```
26
```

```
26
```

```
......................
ITEM 4......................
```
FILLS REPLACE VACANT ELEMENTS [ CFFORMAT CONFORM STRUCT A ]

SYNTAX: R+A FILLS B

- THE STRUCTURE A, WHICH MAY BE SCALAR, WILL APPEAR IN VACANT SPACE OF B. IN A NUMERIC STRUCTURE ZERO SIGNIFIES VACANCY. DISPARATE STRUCTURES WILL BE MADE TO CONFORM. UNLESS OFFSET, THE FIRST ELEMENT OF A WILL MAP INTO THE FIRST ELEMENT OF B.
- IF ONE, BUT NOT BOTH OF THE OPERANDS, IS NUMERIC, IT WILL BE CONVERTED TO CHARACTER FORM.
- USES: VCFORMAT VCONFORM

FUNCTIONS:

✓ R+A FILLS B

[1] CFFORMAT
[2] CONFORM
[3] R+(P B) P (B=1+0 P B) Φ B, [0.10-0.5] A

✓ CONFORM; J : K : R

[1] R+(J+p A) [X+p B]
[2] →0 IF 0=J×K

EXAMPLES:

EX

X X
X X
X X
QUADX

[1] D IS DESIRED RANK (DIMENSIONS)
[2] X+((-D)+(D1),p A) p A

✓ R+ A Δ B

[1] THE SEPARATOR. A AND B MUST RETURN VALUES.
[2] R+ A

✓ CFORMAT

[1] ASSUMES A AND B HAVE BEEN LOCALIZED
[2] A+¥A Δ B+¥B'IF (CHARACTER A)≠CHARACTER B

✓ NULLX

✓ ' , NULLX) FILLS( ' , QUADX) FILLS EX

X [ ] X [ ]
X [ ] X [ ]
X [ ]
X [ ]
X [ ]
X [ ]

ANALYSIS:

CFORMAT WILL FORCE BOTH A AND B INTO CHARACTER FORM IF ONLY ONE IS SO.
CONFORM WILL PAD THE SMALLER ARRAY TO THE SHAPE OF THE LARGER,
UNLESS EITHER ONE IS SCALAR.
STRUCT REDEFINES THE RANK OF ITS OPERAND.
LOC
LOCATE STRUCTURED DATA

SYNTAX:

\[ I+P \text{ LOC } A \]

- RESULT IS A MATRIX OF THE STARTING LOCATIONS \[ \square IO+0 \]
  IF THE ENTIRE STRUCTURE WAS FOUND AT LEAST ONCE.
  \( P \) IS THE SEARCH ARGUMENT. (SEE VONESIN)
- USES: \( \forall \Delta \)

FUNCTION:

\[ \forall I+P \text{ LOC } A; \square IO \]

EXAMPLE:

'TOP SECRET' LOC 'STOP SECRETARY'

ANALYSIS:

5 6 LOC i7
1 2 3 4 5 6 7

\[ I+(pA)\tau(\wedge \chi(p, P)(., P)\phi(., P)\cdot \cdot \cdot A)/\chi/pA \square IO+0 \]

IN CLEAR WS, \[ \square IO+1 \]

0 1 2 3 4 5 6

\[ I+(pA)\tau(\wedge \chi(p, P)(., P)\phi(., P)\cdot \cdot \cdot A)/\chi/pA \square IO+0 \]

LOCALLY, \[ \square IO+0 \]

0 0 0 0 1 0 0

\[ I+(pA)\tau(\wedge \chi(p, P)(., P)\phi(., P)\cdot \cdot \cdot A)/\chi/pA \square IO+0 \]

0 0 0 0 1 0

\[ I+(pA)\tau(\wedge \chi(p, P)(., P)\phi(., P)\cdot \cdot \cdot A)/\chi/pA \square IO+0 \]

0 1

\[ I+(pA)\tau(\wedge \chi(p, P)(., P)\phi(., P)\cdot \cdot \cdot A)/\chi/pA \square IO+0 \]

0 0 0 0 1 0 0

\[ I+(pA)\tau(\wedge \chi(p, P)(., P)\phi(., P)\cdot \cdot \cdot A)/\chi/pA \square IO+0 \]

0 0 0 0 1 0

\[ I+(pA)\tau(\wedge \chi(p, P)(., P)\phi(., P)\cdot \cdot \cdot A)/\chi/pA \square IO+0 \]

4

\[ I+(pA)\tau(\wedge \chi(p, P)(., P)\phi(., P)\cdot \cdot \cdot A)/\chi/pA \square IO+0 \]

7

\[ I+(pA)\tau(\wedge \chi(p, P)(., P)\phi(., P)\cdot \cdot \cdot A)/\chi/pA \square IO+0 \]

4

85
LOGICAL  MISCELLANEOUS [ INTEGER FLOATING EMPTY ]

SYNTAX:  \( T \rightarrow \text{LOGICAL} \ A \)

- RETURN 1 IF THE STRUCTURE SATISFIES CONDITION, OTHERWISE, 0.

FUNCTIONS:

\[
T \rightarrow \text{LOGICAL} \ A \\
[1] \quad T \rightarrow \neg/ \langle ;, A \rangle; 1 \ 0
\]

\[
T \rightarrow \text{INTEGER} \ A \\
[1] \quad \rightarrow (\text{CHARACTER} \ A) / T \rightarrow 0 \\
[2] \quad T \rightarrow 0 \lor \rightarrow 1 \lor \ A
\]

\[
T \rightarrow \text{FLOATING} \ A \\
\text{DEF'N: FLOATING} = 1, \text{AS USED HERE, MEANS AT LEAST ONE} \\
\text{MEMBER OF THE ARGUMENT IS NOT AN INTEGER.} \\
[1] \quad T \rightarrow \neg \text{INTEGER} \ A \lor \neg \text{LOGICAL} \ A \lor \neg \text{CHARACTER} \ A
\]

\[
T \rightarrow \text{EMPTY} \ A \\
[1] \quad T \rightarrow \neg 0 = \rho \ A
\]

EXAMPLES:

\[
\text{LOGICAL} 14 \\
0 \quad \text{LOGICAL} \langle \text{IO} = 1 \\
1 \quad \text{EMPTY} \ 0 \\
1 \quad \text{INTEGER} 1 \\
1 \quad \text{(ONES AND ZEROS ARE INTEGERS)} \\
\text{FLOATING} 1 \\
0 \quad \text{FLOATING} \ 0.1 \\
1
\]

\[
\text{CHARACTER} 1 \\
0 \quad \text{CHARACTER} ' 1'
\]

86
NUMBLANKCOLS COUNTS BLANK COLUMNS AT SIDES OF STRUCTURE

SYNTAX: Z=NUMBLANKCOLS A

FUNCTION: Z=NUMBLANKCOLS A
[1] A+Af' = MATRIX A
[2] Z+(pA)|(-DIO)+(A\10),(FA)\10

EXAMPLES:

    NUMBLANKCOLS 3 4ρ' "
    0 0    NUMBLANKCOLS ' ', 3 4ρ' K'
    1 0    NUMBLANKCOLS 3 4ρ' K'
    2 1

ANA LYSIS:

    NUMBLANKCOLS 2 5ρ' AB "

[1] A+Af' = MATRIX A

    AB
    AB
[1] A+Af' = MATRIX A

   1 0 0 1 1
   1 0 0 1 1
[2] Z+(pA)|(-DIO)+(A\10),(FA)\10
[2] Z+(pA)|(-DIO)+(A\10),(FA)\10 FROM THE RIGHT

   1 1 0 1
[2] Z+(pA)|(-DIO)+(A\10),(FA)\10

   2
[2] Z+(pA)|(-DIO)+(A\10),(FA)\10     ORIGIN INDEPENDENT

   2 2
[2] Z+(pA)|(-DIO)+(A\10),(FA)\10

   -1
[2] Z+(pA)|(-DIO)+(A\10),(FA)\10

   1 2
[2] Z+(pA)|(-DIO)+(A\10),(FA)\10

   5
[2] Z+(pA)|(-DIO)+(A\10),(FA)\10     ZEROS IF ALL BLANK

   1 2
ONESIN

LOCATE ONES IN NUMERIC STRUCTURE

SYNTAX:

R+ONESIN A

AN ARRAY OF ONES AND ZEROS MAY HAVE BEEN THE RESULT OF A TEST OF ANOTHER ARRAY. THIS FUNCTION WILL CONVERT THE ONES TO THEIR OWN LOCATIONS (IIO+O) BY COLUMNS, THAT CAN READILY BE USED TO GENERATE SUBSCRIPTS THAT RELATE TO THE SOURCE.

FUNCTION:

V R+ONESIN A;IIO
[1] R+(pA)TR/\iR+,1=A+IIO+O

EXAMPLE:

A+2 3 4\p5
A
0 1 2 3
4 0 1 2
3 4 0 1
2 3 4 0
1 2 3 4
0 1 2 3
ONESIN A
0 0 0 1 1
0 1 2 1 2
1 2 3 0 1
THRU

GENERATE INDICES OR OTHER EQUAL INCREMENTS BETWEEN LIMITS

SYNTAX:  

\[ R+F \text{ THRU } TB \]

* TO PRODUCE NUMERIC VECTORS WITH INTEGRAL OR FRACTIONAL INCREMENTS OR DECREMENTS

FUNCTION:

\[ \forall \, R+F \text{ THRU } TB; \square IO; B \]

[1]  \[ R+F+(\times R)\times B\times 1+\mid R+(TB[0]-F)\times B+|TB[1]+\square IO+0 \]

[2]  \( a \) GENERATES EQUAL INTERVALS BETWEEN LIMITS \( F \) (A SCALAR) AND \( 1+TB \)

[4]  \( a \) \( 1+TB \leftrightarrow \) THE DESIRED INTERVAL, E.G., \( 1, 0.1, 0.125, 360, \text{ETC.} \)

\[ \forall \]

EXAMPLE:

6 THRU 11 2

6 8 10

ANALYSIS:  

47 THRU 43 0.5

[1]  \[ R+F+(\times R)\times B\times 1+\mid R+(TB[0]-F)\times B+|TB[1]+\square IO+0 \]

[2]  \[ 0.5 \]

[1]  \[ R+F+(\times R)\times B\times 1+\mid R+(TB[0]-F)\times B+|TB[1]+\square IO+0 \]

[3]  \[ 47 \]

[1]  \[ R+F+(\times R)\times B\times 1+\mid R+(TB[0]-F)\times B+|TB[1]+\square IO+0 \]

[4]  \[ -4 \]

[1]  \[ R+F+(\times R)\times B\times 1+\mid R+(TB[0]-F)\times B+|TB[1]+\square IO+0 \]

[5]  \[ -8 \]

[1]  \[ R+F+(\times R)\times B\times 1+\mid R+(TB[0]-F)\times B+|TB[1]+\square IO+0 \]

[6]  \[ SIGN CAPTURED \]

[2]  \[ 8 \]

[1]  \[ R+F+(\times R)\times B\times 1+\mid R+(TB[0]-F)\times B+|TB[1]+\square IO+0 \]

[7]  \[ FOR FRACTIONS \]

[2]  \[ 8 \]

[1]  \[ R+F+(\times R)\times B\times 1+\mid R+(TB[0]-F)\times B+|TB[1]+\square IO+0 \]

[8]  \[ FOR END-POINT \]

[2]  \[ 9 \]

[1]  \[ R+F+(\times R)\times B\times 1+\mid R+(TB[0]-F)\times B+|TB[1]+\square IO+0 \]

[1]  \[ 0 1 2 3 4 5 6 7 8 \]

[1]  \[ R+F+(\times R)\times B\times 1+\mid R+(TB[0]-F)\times B+|TB[1]+\square IO+0 \]

[2]  \[ SCALE \]

[1]  \[ 0 0.5 1 1.5 2 2.5 3 3.5 4 \]

[1]  \[ R+F+(\times R)\times B\times 1+\mid R+(TB[0]-F)\times B+|TB[1]+\square IO+0 \]

[2]  \[ SIGN APPLIED \]

[1]  \[ -1 \]

[1]  \[ R+F+(\times R)\times B\times 1+\mid R+(TB[0]-F)\times B+|TB[1]+\square IO+0 \]

[1]  \[ 2.5 2 \]

[1]  \[ R+F+(\times R)\times B\times 1+\mid R+(TB[0]-F)\times B+|TB[1]+\square IO+0 \]

[2]  \[ 3 \]

[1]  \[ R+F+(\times R)\times B\times 1+\mid R+(TB[0]-F)\times B+|TB[1]+\square IO+0 \]

[3]  \[ 3.5 3 \]

[1]  \[ R+F+(\times R)\times B\times 1+\mid R+(TB[0]-F)\times B+|TB[1]+\square IO+0 \]

[4]  \[ 4 \]

[1]  \[ R+F+(\times R)\times B\times 1+\mid R+(TB[0]-F)\times B+|TB[1]+\square IO+0 \]

47 46.5 46 45.5 45 44.5 44 43.5 43
TABLE LOOK-UP OF STRUCTURED ARGUMENTS [ IS ]

SYNTAX:

- Z+TABLE TLU ARGS

- RETURNS A MATRIX OF SUBSTITUTIONS CORRESPONDING TO A MATRIX OF ARGUMENTS. THE SUBSTITUTIONS ARE FOUND IN A TABLE WHOSE INITIAL COLUMNS WILL BE MATCHED AGAINST ANY NUMBER OF ARGUMENTS, IN ANY ORDER.
- THE ARGUMENTS ARE USUALLY PRESENTED AS A MATRIX, BUT A SINGLE ARGUMENT MAY BE VECTOR OR SCALAR.
- UNDISCOVERED FUNCTIONS WILL BE RETURNED AS BLANKS (OR ZEROS).
- THE UNMATCHED ARGUMENTS WILL BE REPORTED AT THE TERMINAL.
- USES: VHANG, WHICH PRESERVES THE STACK FOR ANALYSIS. VFIRSTM TO REMOVE Duplicates FROM TABLE. VVIS TO CHECK WHETHER TABLE AND ARGUMENT ARE EITHER BOTH NUMERIC, OR BOTH CHARACTER. VMATRIX VIF VON

FUNCTIONS:

- Z+TABLE TLU ARGS;W;R;L

EXAMPLES:

ARGS                     TABLE                     SARGS
D03                      D01EDUCATION              D03
D01                      D02SYSTEMS SUPP           D01
D4A                      D03MKTG SERV              D4A
D02                      D04MARKETING             D02
D01                      D03MKTG SERV              D01
D03                      D04MARKETING             D03
D02                      D04MARKETING             D02

TABLE TLU ARGS             TABLE TLU SARGS
MKTG SERV                  NOT FOUND:
EDUCATION                  XXX
MARKETING                  MKTG SERV
SYSTEMS SUPP               EDUCATION
EDUCATION                  MARKETING
MKTG SERV                  SYSTEMS SUPP
MARKETING                  EDUCATION
SYSTEMS SUPP               MKTG SERV
**XVEC**  
EXPAND LOGICAL VECTOR

**SYNTAX:**  
\[ R+W \text{ XVEC } B \]

- A binary indication of a compressed data structure will be transformed into an expansion vector that can inject \( W \) spaces (or \( W \) zeros in a numeric structure) ahead of the field or group to be shifted.
- Since the expansion can be made along any axis, the length of the binary vector, \( B \), must equal the length of the axis.

**FUNCTION:**

\[
\begin{align*}
& \mathbf{v} \text{ } R+W \text{ XVEC } B \\
[1] & R+(\overline{1}+R+\overline{B}IO)\in \overline{R+(\overline{pB})++W\times B=B} \\
[2] & A \text{ } B \text{ } IS \text{ } A \text{ } LOGICAL \text{ } VECTOR, \text{ } WITH \text{ } ZEROS \text{ } INDICATING \text{ } THE \text{ } BEGINNING \text{ } OF \text{ } EACH \text{ } FIELD, \text{ } BEFORE \text{ } WHICH \overline{WpO} \text{ } WILL \text{ } BE \text{ } INSERTED. \\
[3] & A \text{ } THE \text{ } ORIGINAL \text{ } ZEROS \text{ } WILL \text{ } BE \text{ } CONVERTED \text{ } TO \text{ } ONES.
\end{align*}
\]

**EXAMPLE:**

```
'SHAPE' TOM DICK HARRY'
```

TOM
DICK
HARRY

\( B+\overline{0}+1 \text{ } \text{ XVEC } \text{ } 1 \text{ } 0 \text{ } 1 \)

1 0 1 1
\( B\backslash A \)

TOM

DICK
HARRY

**ANALYSIS:**  
3 XVEC 1 0 1 1

\[
\begin{align*}
[1] & R+(\overline{1}+R+\overline{B}IO)\in \overline{R+(\overline{pB})++W\times B=B} \\
0 & 1 0 0 \\
[1] & R+(\overline{1}+R+\overline{B}IO)\in \overline{R+(\overline{pB})++W\times B=B} \\
0 & 3 0 0 \\
[1] & R+(\overline{1}+R+\overline{B}IO)\in \overline{R+(\overline{pB})++W\times B=B} \\
0 & 3 3 3 \\
[1] & R+(\overline{1}+R+\overline{B}IO)\in \overline{R+(\overline{pB})++W\times B=B} \\
0 & 4 5 6 \\
[1] & R+(\overline{1}+R+\overline{B}IO)\in \overline{R+(\overline{pB})++W\times B=B} \\
1 & 5 6 7 \\
[1] & R+(\overline{1}+R+\overline{B}IO)\in \overline{R+(\overline{pB})++W\times B=B} \\
0 & 1 2 3 4 5 6 \\
[1] & R+(\overline{1}+R+\overline{B}IO)\in \overline{R+(\overline{pB})++W\times B=B} \\
1 & 0 0 0 1 1 1
\end{align*}
\]

91
APPENDIX
BIBLIOGRAPHY

IBM Publications

APL Language, GC26-3847  
VS APL General Information, GH20-9064  
VS APL for CMS: Writing Auxiliary Processors, SH20-9068  
VSPC Installation Reference Material, SH20-9072  
An Introduction to the IBM 3270 Data Analysis APL Feature, GA27-2788  
VS TSIO Guide and Reference, SH20-9107

Non-IBM Publications


Suppose you require a technique to solve a particular problem. You suspect that within the handbook there is a function which can help, but you do not know its name. How do you locate it?

Scan the keywords for a subject reference. When you find it, you will see (within the same abstract) the name of the APL function you need.

Conversely, you may determine the purpose of a function if you know only its name. Use the function name as a keyword to yield the appropriate abstract.
\textsc{vadd}  
\textsc{multiprecision integer addition}

\textsc{add}  
columns to a matrix vector or scalar  \textsc{vaddcols}

\textsc{add}  
rows to a matrix vector or scalar  \textsc{vaddrows}

\textsc{vaddcols}  
\textsc{add}  
columns to a matrix vector or scalar  \textsc{vadd.rows}

\textsc{vaddrows}  
\textsc{add}  
rows to a matrix vector or scalar

\textsc{adjacent elements [ unscan] \textsc{vdiff}}  
\textsc{differences between}

\textsc{vadjustdown}  
\textsc{extend the} ']' \textsc{in report formatting [ rowindices]}

\textsc{vjustup}  
\textsc{extends} ']' \textsc{in report formatting}

\textsc{valprec}  
\textsc{alter precision of a scalar or multiprecision number}

\textsc{alt}  
\textsc{alter precision of a scalar or multiprecision number \textsc{valprec}}

\textsc{vamortize}  
\textsc{mortgage calculation by months}

\textsc{apls statement [ \textsc{alt}] \textsc{vtime}}  
\textsc{running time and new space for an apl statement}

\textsc{arbitrary precision \textsc{vpi}}  
\textsc{compute pi to}

\textsc{arbitrary scalar unit\textsc{vrounds}}  
\textsc{distributive rounding of a vector to}

\textsc{arguments [ is] \textsc{vlu}}  
\textsc{table look-up of structured array}

\textsc{array \textsc{vrepl}}  
\textsc{replace all occurrences of element in}

\textsc{array [ dlb \textsc{rjust dl}]}  
\textsc{vljust}  \textsc{left justify any}

\textsc{array to numeric pattern \textsc{vchar}}  
\textsc{build character}

\textsc{array [ matrix character]}  
\textsc{vframe}  \textsc{frame an}

\textsc{arrays \textsc{vpad}}  
\textsc{pads arrays with}

\textsc{arrays with blanks or zeros \textsc{vpad}}  
\textsc{pads}

\textsc{ascending row indices [ av nform ljnform]}  
\textsc{vgradeup}  \textsc{generate}

\textsc{astronomers \textsc{vdayno}}  
\textsc{day number for}

\textsc{astronomers' day number \textsc{vdate}}  
\textsc{compute normal date from}

\textsc{at sides of structure \textsc{numblankcols}}  
\textsc{counts blank columns}

\textsc{av nform ljnform]}  
\textsc{vgradeup}  \textsc{generate ascending row indices}

\textsc{vbargraph}  
\textsc{plot horizontal integer bargraphs}

\textsc{vbargraphs}  
\textsc{plot horizontal integer}

\textsc{base [ digits conffrac]}  
\textsc{vconv}  \textsc{convert decimal values to any}

\textsc{belong to a \textsc{vindex}}  
\textsc{column index in matrix b whose members all}

\textsc{vbindex}  
\textsc{presents two structures side by side in report format}

\textsc{vblank}  
\textsc{delete specific string from structure [ lim]}

\textsc{blank columns at sides of structure \textsc{numblankcols}}  
\textsc{counts blank columns}

\textsc{blanks or zeros \textsc{vpad}}  
\textsc{pads arrays with}

\textsc{build character array to numeric pattern \textsc{vchar}}  
\textsc{build compression or logical vector \textsc{vcvec}}

\textsc{by columns [ vertab cformat cmatrix rowform]}  
\textsc{vccat}  \textsc{catenate}
BY IN FROM ] \to NUMERIC VECTORS IN EQUAL INCREMENTS [ 
BY MONTHS \AAMORTIZE MORTGAGE CALCULATION 
BY ROW OF A MATRIX [ ESCAPE ESCAPEX ] \VFORM VARIABLE FORMAT 
BY ROWS [ COLFORM CHARACTER VERT ] \VCAT CATEGENET STRUCTURES 
BY SIDE IN REPORT FORMAT \VBESIDE PRESENTS TWO STRUCTURES SIDE 
CALCULATION BY MONTHS \AAMORTIZE MORTGAGE 
CALCULATIONS [ DATES NDATES PAYDAY ] \V DAYS DATE 
\VCAN EDIT MULTIPRECISION INTEGERS INTO CANONICAL FORMAT 
CANONICAL FORMAT \VCAN EDIT MULTIPRECISION INTEGERS INTO 
CATEGENET ANY STRUCTURES \VCAN CONFORM AND 
CATEGENET BY COLUMNS [ VERTAB CFORTH CMATRIX ROWFORM ] \VCAT 
CATEGENET STRUCTURES BY ROWS [ COLFORM CHARACTER VERT ] \VCAT 
CATEGENET TWO STRUCTURES [ CENTER ] \VCENTERON CENTERS AND 
\VCAT CATEGENET BY COLUMNS [ VERTAB CFORTH CMATRIX ROWFORM ] 
CDIV ] \VZDIV ZERO TOLERANT DIVISION [ 
CENTER HEADINGS OVER FORMATTED COLUMNS [ CNTR DMZ NEXTA ] \V OUTPUT 
CENTER ] \VCENTERON CENTERS AND CATEGENET TWO STRUCTURES [ 
\VCENTERON CENTERS AND CATEGENET TWO STRUCTURES [ CENTER ] 
CATEGENET CFORTH CMATRIX ROWFORM ] \VCAT CATEGENET BY COLUMNS [ VERTAB 
CFORTH CONFORM STRUCT \ ] \VFILLS REPLACE VACANT ELEMENTS [ 
\VCCHAR BUILD CHARACTER ARRAY TO NUMERIC PATTERN 
CHARACTER ARRAY TO NUMERIC PATTERN \VCCHAR BUILD 
CHARACTER ARRAYS \VCITED EXTRACT CITED STRINGS FROM 
CHARACTER MATRIX EXPAND RESULT [ \V2M ] \VN2V COMPRESS 
CHARACTER MATRIX [ USCORE ] \VLINE UNDERLINE SPECIFIED ROWS OF 
CHARACTER STRING \VSYSTEM SHAPE MATRIX FROM 
CHARACTER STRING \VFORMAT CONVERT MULTIPRECISION NUMBER TO 
CHARACTER STRUCTURE [ DLTMB ] \VERECT ERECT WORD MATRIX FROM 
CHARACTER STRUCTURE [ DTMB ] \VWORD SELECT NTH WORD IN 
CHARACTER STRUCTURE [ SEDIT ] \VEDIT EDIT LATENT EXPRESSION OR 
CHARACTER VERT ] \VCAT CATEGENET STRUCTURES BY ROWS [ COLFORM 
CHARACTER ] \VFRAME FRAME AN ARRAY [ MATRIX 
CHARACTER ] \VLOGICAL MISCELLANEOUS [ INTEGER FLOATING EMPTY 
CHARACTERISTICS OR CONTENTS OF VARS SELECTIVELY \VVARS DISPLAY 
CHARACTERS OR VALUES \VTC SYSTEM INDEPENDENT TERMINAL CONTROL 
CHECK TERMINAL ENTRY OR DEFAULT \VPROMPT PROMPT AND 
CHECKS A MATRIX FOR FRAMING \VFRAMETEST COMPUTE 
CHI SQUARE PROBABILITY FUNCTION \VQPROBF EXTRACT CITED STRINGS FROM CHARACTER ARRAYS
CITED STRINGS FROM CHARACTER ARRAYS  
CMATRIX ROWFORM ]  VCCAT  Catenate by columns [ VERTAB CFORMAT  
CNTR DM2 NEXTA ] YOUTPUT  CENTER HEADINGS OVER FORMATTED COLUMNS  
CODES  VCOLLECT  COLLECT AND SUMMARIZE COEFFICIENTS OF COMMON  
COEFFICIENTS OF COMMON CODES  VCOLLECT  COLLECT AND SUMMARIZE  
COLFORM CHARACTER VERT ]  VRCAT  Catenate structures by rows [  
VCOLLECT  COLLECT AND SUMMARIZE COEFFICIENTS OF COMMON CODES  
COLLECT AND SUMMARIZE COEFFICIENTS OF COMMON CODES  VCOLLECT  
COLNO ] YTABAS  COMPARE REQUIRED TAB SETTINGS TO EXISTING ONES  
COLUMN INDEX IN MATRIX B WHOSE MEMBERS ALL BELONG TO A VINDEX  
COLUMNS AT SIDES OF STRUCTURE  \NUMBLANKCOLS  COUNTS BLANK  
COLUMNS TO A MATRIX VECTOR OR SCALAR  \ADDCOLS  ADD  
COLUMNS [ CNTR DM2 NEXTA ] YOUTPUT  CENTER HEADINGS OVER FORMATTED  
COLUMNS [ VERTAB CFORMAT CMATRIX ROWFORM ]  VCCAT  Catenate by  
\COMB  ALL COMBINATIONS OF ELEMENTS  \DEBLANK UNIQ  \COMB  ALL  
COMMON CODES \COLNO  COMPARE REQUIRED TAB SETTINGS TO EXISTING ONES  
COMPARE \HEADERON  PUTS A HEADING ON A REPORT  \COMB  ALL COMBINATIONS OF ELEMENTS  \DEBLANK UNIQ  \COMB  ALL  
COMMON CODES \COLNO  COMPARE REQUIRED TAB SETTINGS TO EXISTING ONES  
COMPARE \HEADERON  PUTS A HEADING ON A REPORT  \COMB  ALL COMBINATIONS OF ELEMENTS  \DEBLANK UNIQ  \COMB  ALL  
COMMON CODES \COLNO  COMPARE REQUIRED TAB SETTINGS TO EXISTING ONES  
COMPARE \HEADERON  PUTS A HEADING ON A REPORT  \COMB  ALL COMBINATIONS OF ELEMENTS  \DEBLANK UNIQ  \COMB  ALL  
COMMON CODES \COLNO  COMPARE REQUIRED TAB SETTINGS TO EXISTING ONES  
COMPARE \HEADERON  PUTS A HEADING ON A REPORT  
COMPRESS CHARACTER MATRIX  EXPAND RESULT  \V2M  \V2M  \M2V  
COMPRESSION OR LOGICAL VECTOR  \CVEC  BUILD  
COMPUTE CHI SQUARE PROBABILITY FUNCTION  \QPROBF  
COMPUTE NORMAL DATE FROM ASTRONOMERS' DAY NUMBER  \DATE  
COMPUTE PI TO ARBITRARY PRECISION  \PI  
COMPUTE THE DATE OF EASTER  \EASTER  
CONFORM AND CATEGORIZE ANY STRUCTURES  \ON  
CONFORM STRUCT \ ]  \FILLS  REPLACE VACANT ELEMENTS  \CFORMAT  
CONFRAC ] \CONV  CONVERT DECIMAL VALUES TO ANY BASE  \DIGITS  
CONTENTS OF VARS SELECTIVELY  \VARS  DISPLAY CHARACTERISTICS OR  
CONTROL CHARACTERS  \TCC  SYSTEM INDEPENDENT TERMINAL  
CONTROLLED FORMAT [ HANG ]  \TABULATE  NUMERIC STRUCTURES IN  
\CONV  CONVERT DECIMAL VALUES TO ANY BASE  \DIGITS CONFRAC  \CONV  
CONVERT DECIMAL VALUES TO ANY BASE  \DIGITS CONFRAC  \CONV  
CONVERT INTEGER TO ROMAN NUMERALS  \ROMAN  
CONVERT MULTIPRECISION NUMBER TO CHARACTER STRING  \FORMAT  
CONVERT TO DECIMAL  \DEC  
CONVERT TO MULTIPRECISION FLOATING POINT  \SCALE  \FLOAT  
CONVERT TO MULTIPRECISION INTEGER  \FIX  
COUNTS BLANK COLUMNS AT SIDES OF STRUCTURE  \NUMBLANKCOLS  
CURRENT SESSION AND WORKSPACE STATUS  \NOW  \STATUS  
\CVEC  BUILD COMPRESSION OR LOGICAL VECTOR
ELEMENT IN ARRAY  \$REPL  REPLACE ALL OCCURRENCES OF
ELEMENTS [ DEBLANK UNIQ ]  \$COMB  ALL COMBINATIONS OF
ELEMENTS [ SORTDA KFORM ]  \$FREQ  FREQUENCY DISTRIBUTION OF
ELEMENTS [ UNSCAN ]  \$DIFF  DIFFERENCES BETWEEN ADJACENT
ELEMENTS FROM ANY STRUCTURE  \$REPL  SELECT UNIQUE
ELEMENTS [ CFTEST CFCONFORM STRUCT \$ ]  \$FILLS  REPLACE VACANT
EMPTY CHARACTER ]  \$LOGICAL  MISCELLANEOUS [ INTEGER FLOATING
\$ENC  GENERATE SUFFICIENT ENCODING POSITIONS
ENCODING POSITIONS  \$ENC  GENERATE SUFFICIENT
ENTRY OR DEFAULT  \$PROMPT  PROMPT AND CHECK TERMINAL
EQUAL INCREMENTS BETWEEN LIMITS  \$THRU  GENERATE INDICES OR OTHER
EQUAL INCREMENTS [ BY IN FROM ]  \$TO  NUMERIC VECTORS IN
\$RECT  ERECT WORD MATRIX FROM CHARACTER STRUCTURE [ DLTMB ]
EREECT WORD MATRIX FROM CHARACTER STRUCTURE [ DLTMB ]  \$RECT
ESCAPE ESCAPEX ]  \$VFORM  VARIABLE FORMAT BY ROW OF A MATRIX [ EXISTING ONES [ COLNO ]  \$TAB  COMPARE REQUIRED TAB SETTINGS TO
EXPAND LOGICAL VECTOR  \$VEC
EXPAND RESULT [ V2M ]  \$M2V  COMPRESS CHARACTER MATRIX
EXPRESSION OR CHARACTER STRUCTURE [ SEDIT ]  \$EDIT  EDIT LATENT
\$EXTEND  EXTEND VECTOR WITH LAST VALUE
EXTEND THE ']' IN REPORT FORMATTING [ ROWINDICES ]  \$ADJUSTDOWN
EXTEND VECTOR WITH LAST VALUE  \$EXTEND
EXTENDS ']' IN REPORT FORMATTING  \$ADJUSTUP
EXTRACT CITED STRINGS FROM CHARACTER ARRAYS  \$CITED
\$FADD  MULTIPRECISION FLOATING POINT ADDITION
\$FDIV  MULTIPRECISION FLOATING POINT DIVISION
\$FEXP  MULTIPRECISION FLOATING POINT EXPONENTIAL FUNCTION
\$FILLS  REPLACE VACANT ELEMENTS [ CFTEST CFCONFORM STRUCT \$ ]
FIRST OR ONLY APPEARANCE IN MATRIX [ FIRSTV ]  \$FIRSTM  SELECT
\$FIRSTM  SELECT FIRST OR ONLY APPEARANCE IN MATRIX [ FIRSTV ]
FIRSTV ]  \$FIRSTM  SELECT FIRST OR ONLY APPEARANCE IN MATRIX [ \$FIX  CONVERT TO MULTIPRECISION INTEGER
\$FLOAT  CONVERT TO MULTIPRECISION FLOATING POINT [ SCALE ]
FLOATING EMPTY CHARACTER ]  \$LOGICAL  MISCELLANEOUS [ INTEGER
FLOATING POINT ADDITION  \$FADD  MULTIPRECISION
FLOATING POINT DIVISION  \$FDIV  MULTIPRECISION
FLOATING POINT EXPONENTIAL FUNCTION  \$FEXP  MULTIPRECISION
FLOATING POINT MULTIPLICATION  \$FMUL  MULTIPRECISION
FLOATING POINT SQUARE ROOT  \$FSQRT  MULTIPRECISION
HORIZONTAL INTEGER BARGRAPHS  VBARGRAPH  PLOT
IF ]  VPREPARE  STANDARDIZE STRUCTURE FOR REPORT FORMATTING [ INCREMENTS BETWEEN LIMITS VTHRU  GENERATE INDICES OR OTHER EQUAL INCREMENTS [ BY IN FROM ]  VTO  NUMERIC VECTORS IN EQUAL INDEPENDENT TERMINAL CONTROL CHARACTERS  VTCC  SYSTEM
INDEX  COLUMN INDEX IN MATRIX B WHOSE MEMBERS ALL BELONG TO A INDEX IN MATRIX B WHOSE MEMBERS ALL BELONG TO A  VINDEX  COLUMN INDICES  [ AV NFORM LJNFORM ]  VGRADEUP  GENERATE ASCENDING ROW INDICES OR OTHER EQUAL INCREMENTS BETWEEN LIMITS VTHRU  GENERATE INSENSITIVE CHARACTERS OR VALUES  VDT  DELETE TRAILING INTEGER  VFIX  CONVERT TO MULTIPRECISION
INTEGER ADDITION  VADD  MULTIPRECISION
INTEGER BARGRAPHS  VBARGRAPH  PLOT HORIZONTAL
INTEGER DIVISION  VDIV  MULTIPRECISION
INTEGER FLOATING EMPTY CHARACTER  VLOGICAL  MISCELLANEOUS
INTEGER MULTIPLICATION  VMUL  MULTIPRECISION
INTEGER SQUARE ROOT  VSQRT  MULTIPRECISION
INTEGER SUBTRACTION  VSUB  MULTIPRECISION
INTEGER TO ROMAN NUMERALS  VRoman  CONVERT
INTEGER INTO CANONICAL FORMAT  VCan  EDIT MULTIPRECISION
INTO CANONICAL FORMAT  VCan  EDIT MULTIPRECISION INTEGERS
IOTA  VRIOTA  MATRIX ROW
IS ]  VT卢  TABLE LOOK-UP OF STRUCTURED ARGUMENTS [ JUSTIFY ANY ARRAY  [ DLB RJUST DL ]  VJUST  LEFT
KFORM ]  VFREQ  FREQUENCY DISTRIBUTION OF ELEMENTS  [ SORTDA LAST VALUE  VEXTEND  EXTEND VECTOR WITH LATENT EXPRESSION OR CHARACTER STRUCTURE  [ SEVIT ]  VEDIT  EDIT LEFT JUSTIFY ANY ARRAY  [ DLB RJUST DL ]  VJUST
LIM ]  VBLANK  DELETE SPECIFIC STRING FROM STRUCTURE [ LIMITS VTHRU  GENERATE INDICES OR OTHER EQUAL INCREMENTS BETWEEN
LISTFN  LISTS A FUNCTION IN STANDARD FORM
LISTS A FUNCTION IN STANDARD FORM  VLISTFN
LJNFORM ]  VGRADEUP  GENERATE ASCENDING ROW INDICES  [ AV NFORM
VJUST  LEFT JUSTIFY ANY ARRAY  [ DLB RJUST DL ]  VJUST
VLOC  LOCATE STRUCTURED DATA
LOCATE ONES IN NUMERIC STRUCTURE  VONESIN
LOCATE STRUCTURED DATA  VLOC
VLOGICAL  MISCELLANEOUS  INTEGER FLOATING EMPTY CHARACTER
LOGICAL VECTOR  VXVEC  BUILD COMPRESSION OR
LOGICAL VECTOR  VXVEC  EXPAND
LOWER ORDER DIGITS  VTRUNC  TRUNCATE HIGHER AND
MATRIX EXPAND RESULT [ V2M ] \width \measure formatted
MATRIX FOR FRAMING \width \measure character
MATRIX FOR FUNCTION-LIKE EDITING [ POSTEDIT ] \preedit \prepare
MATRIX FROM CHARACTER STRING \shape \shape
MATRIX FROM CHARACTER STRUCTURE [ DLTMB ] \vrect \vrect word
MATRIX VECTOR OR SCALAR \vaddcols add columns to a matrix
MATRIX VECTOR OR SCALAR \vaddrrows add rows to a matrix
MATRIX [ ESCAPE ESCAPEX ] \vform \vform
MATRIX [ FIRSTV ] \vfirstm select first or only appearance in a matrix
MATRIX [ USCORE ] \vuline underline specified rows of character
MEASURE FORMATTED MATRIX \vwidth
\vmedit edit matrix
MEMBERS ALL BELONG TO A \vindex \vindex column index in matrix b whose
MISCELLANEOUS [ INTEGER FLOATING EMPTY CHARACTER ] \vlogical
MONTHS \vamortize mortgage calculation by months
MOONPHASE ] \vdayno day number for astronomers [\vamortize
\vmul multiprecision integer multiplication
MULTIPLICATION \vmul \vmul multiprecision integer multiplication
MULTIPLICATION \vfmul \vfmul floating point multiplication
MULTIPLICATION FLOATING POINT ADDITION \vadd\add
MULTIPLICATION FLOATING POINT DIVISION \vdiv\div
MULTIPLICATION FLOATING POINT EXPONENTIAL FUNCTION \vexp\exp
MULTIPLICATION FLOATING POINT [ SCALE ] \vfloat\float convert to
MULTIPLICATION FLOATING POINT MULTIPLICATION \vfmul\fmul
MULTIPLICATION FLOATING POINT SQUARE ROOT \vsqrt\sqrt
MULTIPLICATION FLOATING POINT SUBTRACTION \vsub\sub
MULTIPLICATION INTEGER ADDITION \vadd\add convert to
MULTIPLICATION INTEGER DIVISION \vdiv\div
MULTIPLICATION INTEGER MULTIPLICATION \vmul\mul
MULTIPLICATION INTEGER SQUARE ROOT \vqrt\qrt
MULTIPLICATION INTEGER SUBTRACTION \vsub\sub
MULTIPLICATION INTEGERS INTO CANONICAL FORMAT \vcan\can edit
MULTIPLICATION NUMBER \valprec alter precision of a scalar or
MULTIPLICATION NUMBER TO CHARACTER STRING \vformat\format convert
VM2V   COMPRESS CHARACTER MATRIX   EXPAND RESULT [ V2M ]
NDATES PAYDAY   VDAYS   DATE CALCULATIONS [ DATES NEW SPACE FOR AN APL STATEMENT [ ALT ]   VTIME   RUNNING TIME AND NFORM LFNFORM ]   VGRADEUP   GENERATE ASCENDING ROW INDICES [ AV NORMAL DATE FROM ASTRONOMERS' DAY NUMBER   VDATE   COMPUTE NOW ]   VSTATUS   CURRENT SESSION AND WORKSPACE STATUS [ NTH WORD IN CHARACTER STRUCTURE [ DZMB ]   VWORD   SELECT NUMBER   VDATE   COMPUTE NORMAL DATE FROM ASTRONOMERS' DAY NUMBER   VALPRES   ALTER PRECISION OF A SCALAR OR MULTIPRECISION NUMBER FOR ASTRONOMERS [ MOONPHASE ]   VDAYNO   DAY NUMBER TO CHARACTER STRING   VFORMAT   CONVERT MULTIPRECISION VNUMBLANKCOLS COUNTS BLANK COLUMNS AT SIDES OF STRUCTURE NUMERALS   VRONAN   CONVERT INTEGER TO ROMAN NUMERIC PATTERN   VCHAR   BUILD CHARACTER ARRAY TO NUMERIC STRUCTURE   VONESIN   LOCATE ONES IN NUMERIC STRUCTURES IN CONTROLLED FORMAT [ HANG ]   VTABULATE NUMERIC VECTORS IN EQUAL INCREMENTS [ BY IN FROM ]   VTO OCCURRENCES OF ELEMENT IN ARRAY   VREPL   REPLACE ALL ONES [ COLNO ]   VTABLES   COMPARE REQUIRED TAB SETTINGS TO EXISTING ONES IN NUMERIC STRUCTURE   VONESIN   LOCATE ONES IN NUMERIC STRUCTURE ONLY APPEARANCE IN MATRIX [ FIRSTV ]   VFIRSTM   SELECT FIRST OR ORDER DIGITS   VTRUNC   TRUNCATE HIGHER AND LOWER OTHER EQUAL INCREMENTS BETWEEN LIMITS   VTHRU   GENERATE INDICES OR VOUTPUT CENTER HEADINGS OVER FORMATTED COLUMNS [ CNTR DMZ NEXTA ] OVER FORMATTED COLUMNS [ CNTR DMZ NEXTA ]   VOUTPUT   CENTER HEADINGS VPAD   PADS ARRAYS WITH BLANKS OR ZEROS PADS ARRAYS WITH BLANKS OR ZEROS   VPAD PATTERN   VCHAR   BUILD CHARACTER ARRAY TO NUMERIC PAYDAY ]   VDAYS   DATE CALCULATIONS [ DATES NDATES VPI   COMPUTE PI TO ARBITRARY PRECISION PI TO ARBITRARY PRECISION   VPI   COMPUTE PLOT HORIZONTAL INTEGER BARGRAPHS   VBARGRAPH POINT ADDITION   VFADD   MULTIPRECISION FLOATING POINT DIVISION   VFDIV   MULTIPRECISION FLOATING POINT EXPONENTIAL FUNCTION   VFEXP   MULTIPRECISION FLOATING POINT MULTIPLICATION   VFMLUL   MULTIPRECISION FLOATING POINT SQUARE ROOT   VFSQRT   MULTIPRECISION FLOATING POINT SUBTRACTION   VFSUB   MULTIPRECISION FLOATING POINT [ SCALE ]   VFLOAT   CONVERT TO MULTIPRECISION FLOATING POSITIONS   VENC   GENERATE SUFFICIENT ENCODING
POSTEDIT ] ▼PREDICT PREPARE MATRIX FOR FUNCTION-LIKE EDITING [
PRECISION ▼PI COMPUTE PI TO ARBITRARY
PRECISION OF A SCALAR OR MULTIPRECISION NUMBER ▼VALPREC ALTER
▼PREDICT PREPARE MATRIX FOR FUNCTION-LIKE EDITING [ POSTEDIT ] ▼PREDICT
PREPARE MATRIX FOR FUNCTION-LIKE EDITING [ IF ]
PREPARE MATRIX FOR FUNCTION-LIKE EDITING [ POSTEDIT ] ▼PREDICT
PRESENTS TWO STRUCTURES SIDE BY SIDE IN REPORT FORMAT ▼BESIDE
PROBABILITY FUNCTION ▼QPROBF COMPUTE CHI SQUARE
▼PROMPT PROMPT AND CHECK TERMINAL ENTRY OR DEFAULT
▼PROMPT PROMPT AND CHECK TERMINAL ENTRY OR DEFAULT ▼PROMPT
PUTS A HEADING ON A REPORT [ ▼COMPARE ] ▼HEADERON
▼QPROBF COMPUTE CHI SQUARE PROBABILITY FUNCTION
▼RCAT CATENATE STRUCTURES BY ROWS [ COLFORM CHARACTER VERT ]
▼REPL REPLACE ALL OCCURRENCES OF ELEMENT IN ARRAY
REPLACE ALL OCCURRENCES OF ELEMENT IN ARRAY ▼REPL
REPLACE VACANT ELEMENTS [ ▼CFORMAT CONFORM STRUCT △ ] ▼FILLS
REPORT [ ▼COMPARE ] ▼HEADERON PUTS A HEADING ON A
REPORT FORMAT ▼BESIDE PRESENTS TWO STRUCTURES SIDE BY SIDE IN
REPORT FORMATTING ▼ADJUSTUP EXTENDS '|' IN
REPORT FORMATTING [ IF ] ▼PREPARE STANDARDIZE STRUCTURE FOR
REPORT FORMATTING [ ▼ROWINDICES ] ▼ADJUSTDOWN EXTEND THE '|' IN
REQUIRED TAB SETTINGS TO EXISTING ONES [ ▼COLNO ] ▼TABS COMPARE
▼RIOTA MATRIX ROW IOTA
RJUST DL ] ▼LJUST LEFT JUSTIFY ANY ARRAY [ DLB
▼ROMAN CONVERT INTEGER TO ROMAN NUMERALS
ROMAN NUMERALS ▼ROMAN CONVERT INTEGER TO
ROOT ▼SQRT MULTIPRECISION INTEGER SQUARE
ROOT ▼FSQRT MULTIPRECISION FLOATING POINT SQUARE
ROUNDS ▼ROUNDS SELECTIVE SYMMETRICAL
ROUNDS SELECTIVE SYMMETRICAL ROUNDS ▼DROUNDS DISTRIBUTIVE
▼ROWINDICES [ ▼AV NFORM ▼LJNFORM ] ▼GRADEUP GENERATE ASCENDING
ROW IOTA ▼RIOTA MATRIX
ROW OF A MATRIX [ ▼ESCAPE ▼ESCAPEX ] ▼VFORM VARIABLE FORMAT BY
ROWFORM ] ▼VCAT CATENATE BY COLUMNS [ ▼VERTAB ▼CFORMAT ▼CMATRIX
ROWINDICES ] ▼ADJUSTDOWN EXTEND THE '|' IN REPORT FORMATTING[
ROWS OF CHARACTER MATRIX [ ▼USCORE ] ▼ULINE UNDERLINE SPECIFIED
ROWS TO A MATRIX VECTOR OR SCALAR ▼ADDROWS ADD
ROWS [ COLFORM CHARACTER VERT ] ▼RCAT CATENATE STRUCTURES BY
RUNNING TIME AND NEW SPACE FOR AN APL STATEMENT [ ▼ALT ] ▼TIME
SCALAR ▼ADDROWS ADD ROWS TO A MATRIX VECTOR OR
SCALAR ADDDCOLS ADD COLUMNS TO A MATRIX
SCALAR OR MULTIPRECISION NUMBER ALPREC ALTER PRECISION OF A
SCALAR UNITVADDCOLS DISTRIBUTIVE RONDING OF A VECTOR TO ARBITRARY
SCALE ] VFLOAT CONVERT TO MULTIPRECISION FLOATING POINT [
SEDIT ] VEDIT EDIT LATENT EXPRESSION OR CHARACTER STRUCTURE [
SELECT FIRST OR ONLY APPEARANCE IN MATRIX [ FIRSTV ] VFIRSTV
SELECT NTH WORD IN CHARACTER STRUCTURE [ DTMB ] VWORD
SELECT UNIQUE ELEMENTS FROM ANY STRUCTURE VDREP
SELECTIVELY VVARS DISPLAY CHARACTERISTICS OR CONTENTS OF VARS
SESSION AND WORKSPACE STATUS [ NOW ] VSTATUS CURRENT
SETTINGS TO EXISTING ONES [ COLNO ] VTABS COMPARE REQUIRED TAB
\SHAPE SHAPE MATRIX FROM CHARACTER STRING
\SHAPE MATRIX FROM CHARACTER STRING \SHAPE
SIDE BY SIDE IN REPORT FORMAT \VBESIDE PRESENTS TWO STRUCTURES
SIDE IN REPORT FORMAT \VBESIDE PRESENTS TWO STRUCTURES SIDE BY
SIDES OF STRUCTURE \VBESIDE \NUMBLANKCOLS COUNTS BLANK COLUMNS AT
SORTDA KFORM ] VREQ FREQUENCY DISTRIBUTION OF ELEMENTS [
SPACE FOR AN APL STATEMENT [ ALT ] \VTIME RUNNING TIME AND NEW
SPECIFIC STRING FROM STRUCTURE [ LIM ] \VBLANK DELETE
SPECIFIED ROWS OF CHARACTER MATRIX [ USCORE ] \VLINE UNDERLINE
\VSQRT MULTIPRECISION INTEGER SQUARE ROOT
SQUARE PROBABILITY FUNCTION \VPROBF COMPUTE CHI
SQUARE ROOT \VSQRT MULTIPRECISION INTEGER SQUARE ROOT \VFSQRT MULTIPRECISION FLOATING POINT
STANDARD FORM \LISTFN LISTS A FUNCTION IN
STANDARDIZE STRUCTURE FOR REPORT FORMATTING [ IF ] \VPREPARE
STATEMENT [ ALT ] \VTIME RUNNING TIME AND NEW SPACE FOR AN APL
\VSTATUS CURRENT SESSION AND WORKSPACE STATUS [ NOW ]
\VSTATUS CURRENT SESSION AND WORKSPACE
\STRING SHAPE SHAPE MATRIX FROM CHARACTER STRING
\STRING \FORMAT CONVERT MULTIPRECISION NUMBER TO CHARACTER
\STRING FROM STRUCTURE [ LIM ] \VBLANK DELETE SPECIFIC
\STRING FROM CHARACTER ARRAYS \VCITED EXTRACT CITED
\STRUCT \VFILLS REPLACE VACANT ELEMENTS [ CFORMAT CONFORM
\STRUCT \ONESIN LOCATE ONES IN NUMERIC
\STRUCT \VDREP SELECT UNIQUE ELEMENTS FROM ANY
\STRUCT \NUMBLANKCOLS COUNTS BLANK COLUMNS AT SIDES OF
\STRUCT FOR REPORT FORMATTING [ IF ] \VPREPARE STANDARDIZE
\STRUCT [ DLTMB ] \VERECT ERECT WORD MATRIX FROM CHARACTER
\STRUCT [ DTMB ] \VWORD SELECT NTH WORD IN CHARACTER
STRUCTURE [ LIM ] ▽BLANK DELETE SPECIFIC STRING FROM
STRUCTURE [ SEDIT ] ▽EDIT EDIT LATENT EXPRESSION OR CHARACTER
STRUCTURED ARGUMENTS [ IS ] ▽TLU TABLE LOOK-UP OF
STRUCTURED DATA ▽LOC LOCATE
STRUCTURES ▽ON CONFORM AND CATENATE ANY
STRUCTURES BY ROWS [ COLFORM CHARACTER VERT ] ▽RCAT CATENATE
STRUCTURES IN CONTROLLED FORMAT [ HANG ] ▽TABULATE NUMERIC
STRUCTURES SIDE BY SIDE IN REPORT FORMAT ▽BESIDE PRESENTS TWO
STRUCTURES [ CENTER ] ▽CENTERON CENTERS AND CATENATES TWO
▽SUB MULTIPRECISION INTEGER SUBTRACTION
SUBTRACTION ▽SUB MULTIPRECISION INTEGER
SUBTRACTION ▽FSUB MULTIPRECISION FLOATING POINT
SUFFICIENT ENCODING POSITIONS ▽ENC GENERATE
SUMMARIZE COEFFICIENTS OF COMMON CODES ▽COLLECT COLLECT AND
SYMmetrical Rounding ▽ROUNDS SELECTIVE
SYSTEM INDEPENDENT TERMINAL CONTROL CHARACTERS ▽TCC
TABLE LOOK-UP OF STRUCTURED ARGUMENTS [ IS ] ▽TLU
▽TABLS COMPARE REQUIRED TAB SETTINGS TO EXISTING ONES [ COLNO ]
▽TABULATE NUMERIC STRUCTURES IN CONTROLLED FORMAT [ HANG ]
▽TCC SYSTEM INDEPENDENT TERMINAL CONTROL CHARACTERS
TERMINAL CONTROL CHARACTERS ▽TCC SYSTEM INDEPENDENT
TERMINAL ENTRY OR DEFAULT ▽PROMPT PROMPT AND CHECK
THE DATE OF EASTER ▽VEASTER COMPUTE
THE ' | ' IN REPORT FORMATTING [ ROWINDICES ] ▽ADJUSTDOWN EXTEND
▽THRU GENERATE INDICES OR OTHER EQUAL INCREMENTS BETWEEN LIMITS
▽TIME RUNNING TIME AND NEW SPACE FOR AN APL STATEMENT [ ALT ]
TIME AND NEW SPACE FOR AN APL STATEMENT [ ALT ] ▽TIME RUNNING
▽TLU TABLE LOOK-UP OF STRUCTURED ARGUMENTS [ IS ]
▽TO NUMERIC VECTORS IN EQUAL INCREMENTS [ BY PROM ]
TO A ▽INDEX COLUMN INDEX IN MATRIX B WHOSE MEMBERS ALL BELONG
TO A MATRIX VECTOR OR SCALAR ▽ADDCOLS ADD COLUMNS
TO A MATRIX VECTOR OR SCALAR ▽ADDDROWS ADD ROWS
TO ANY BASE [ DIGITS CONFRAC ] ▽CONV CONVERT DECIMAL VALUES
TO ARBITRARY PRECISION ▽PI COMPUTE PI
TO ARBITRARY SCALAR UNIT▽DROUNDS DISTRIBUTIVE ROUNding OF A VECTOR
TO CHARACTER STRING ▽FORMAT CONVERT MULTIPRECISION NUMBER
TO DECIMAL ▽DEC CONVERT
TO EXISTING ONES [ COLNO ] ▽TABLS COMPARE REQUIRED TAB SETTINGS
TO MULTIPRECISION FLOATING POINT [ SCALE ] ▽FLOAT CONVERT
TO MULTIPRECISION INTEGER ▽FIX CONVert
TO NUMERIC PATTERN \( \textbf{\texttt{\textbackslash char}} \) BUILD CHARACTER ARRAY
TO ROMAN NUMERALS \( \textbf{\texttt{\textbackslash roman}} \) CONVERT INTEGER
TOLERANT DIVISION \( [ \textbf{\texttt{\textbackslash cdiv}} ] \) \( \textbf{\texttt{\textbackslash zdv}} \) ZERO
TRAILING INSIGNIFICANT CHARACTERS OR VALUES \( \textbf{\texttt{\textbackslash dt}} \) DELETE
\( \textbf{\texttt{\textbackslash trunc}} \) TRUNCATE HIGHER AND LOWER ORDER DIGITS
\( \textbf{\texttt{\textbackslash uline}} \) UNDERLINE SPECIFIED ROWS OF CHARACTER MATRIX \( [ \textbf{\texttt{\textbackslash uscore}} ] \)
UNDERLINE SPECIFIED ROWS OF CHARACTER MATRIX \( [ \textbf{\texttt{\textbackslash uscore}} ] \) \( \textbf{\texttt{\textbackslash uline}} \)
UNIQ \( \textbf{\texttt{\textbackslash vcomb}} \) ALL COMBINATIONS OF ELEMENTS \( [ \textbf{\texttt{\textbackslash deblank}} \) SELECT
UNIQUE ELEMENTS FROM ANY STRUCTURE \( \textbf{\texttt{\textbackslash vrep}} \)
UNITY\$D\$ROUNDS DISTRIBUTIVE Rounding Of A VECTOR TO ARBITRARY SCALAR
UNSCAN \( \textbf{\texttt{\textbackslash vdiff}} \) DIFFERENCES BETWEEN ADJACENT ELEMENTS \( [ \textbf{\texttt{\textbackslash uscore}} ] \) \( \textbf{\texttt{\textbackslash uline}} \)
UNDERLINE SPECIFIED ROWS OF CHARACTER MATRIX \( [ \textbf{\texttt{\textbackslash uscore}} ] \) \( \textbf{\texttt{\textbackslash uline}} \)
VACANT ELEMENTS \( [ \textbf{\texttt{\textbackslash cformat conform struct \& \textbackslash deblank}} \) SELECT
VALUE \( \textbf{\texttt{\textbackslash vextend}} \) EXTEND VECTOR WITH LAST VALUES \( \textbf{\texttt{\textbackslash vd}} \) DELETE TRAILING INSIGNIFICANT CHARACTERS OR VALUES TO ANY BASE \( [ \textbf{\texttt{\textbackslash digits conform}} ] \) \( \textbf{\texttt{\textbackslash conv}} \) CONVERT DECIMAL VARIABLE FORMAT BY ROW OF A MATRIX \( [ \textbf{\texttt{\textbackslash escape escapex}} ] \) \( \textbf{\texttt{\textbackslash vform}} \)
VARS DISPLAY CHARACTERISTICS OR CONTENTS OF VARS SELECTIVELY
VARS SELECTIVELY \( \textbf{\texttt{\textbackslash vars}} \) DISPLAY CHARACTERISTICS OR CONTENTS OF VECTOR \( \textbf{\texttt{\textbackslash vvec}} \) BUILD COMPRESSION OR LOGICAL VECTOR \( \textbf{\texttt{\textbackslash vxvec}} \) EXPAND LOGICAL VECTOR OR SCALAR \( \textbf{\texttt{\textbackslash vaddcols}} \) ADD COLUMNS TO A MATRIX
VECTOR OR SCALAR \( \textbf{\texttt{\textbackslash vaddrows}} \) ADD ROWS TO A MATRIX
VECTOR TO ARBITRARY SCALAR UNITY\$D\$ROUNDS DISTRIBUTIVE Rounding Of A VECTOR WITH LAST VALUE \( \textbf{\texttt{\textbackslash vextend}} \) EXTEND VECTORS IN EQUAL INCREMENTS \( [ \textbf{\texttt{\textbackslash by \textbackslash in \textbackslash from}} ] \) \( \textbf{\texttt{\textbackslash vto}} \) NUMERIC VERT \( \textbf{\texttt{\textbackslash vrcat}} \) CATENATE STRUCTURES BY ROWS \( [ \textbf{\texttt{\textbackslash colform character matrix rowform}} ] \) \( \textbf{\texttt{\textbackslash vccat}} \) CATENATE BY COLUMNS \( [ \textbf{\texttt{\textbackslash vformat conform structure rowform}} ] \) \( \textbf{\texttt{\textbackslash vformat conform structure}} \)
\( \textbf{\texttt{\textbackslash v2m}} \) \( \textbf{\texttt{\textbackslash vm2v}} \) COMPRESS CHARACTER MATRIX EXPAND RESULT \( [ \textbf{\texttt{\textbackslash escape escapex}} ] \) \( \textbf{\texttt{\textbackslash vform}} \) \( \textbf{\texttt{\textbackslash v2m}} \) \( \textbf{\texttt{\textbackslash vm2v}} \)
WHOSE MEMBERS ALL BELONG TO A \( \textbf{\texttt{\textbackslash vindex}} \) COLUMN INDEX IN MATRIX B
WIDTH MEASURE FORMATTED MATRIX
WITH BLANKS OR ZEROS \( \textbf{\texttt{\textbackslash vpad}} \) PADS ARRAYS WITH LAST VALUE \( \textbf{\texttt{\textbackslash vextend}} \) EXTEND VECTOR
WORD SELECT NTH WORD IN CHARACTER STRUCTURE \( [ \textbf{\texttt{\textbackslash dtmb}} ] \) \( \textbf{\texttt{\textbackslash word}} \)
WORD IN CHARACTER STRUCTURE \( [ \textbf{\texttt{\textbackslash dtmb}} ] \) \( \textbf{\texttt{\textbackslash word}} \)
WORD MATRIX FROM CHARACTER STRUCTURE \( [ \textbf{\texttt{\textbackslash dtmb}} ] \) \( \textbf{\texttt{\textbackslash vrect}} \) ERECT WORKSPACE STATUS \( [ \textbf{\texttt{\textbackslash now}} ] \) \( \textbf{\texttt{\textbackslash vstatus}} \) CURRENT SESSION AND \( \textbf{\texttt{\textbackslash vxvec}} \) EXPAND LOGICAL VECTOR \( \textbf{\texttt{\textbackslash zdv}} \) ZERO TOLERANT DIVISION \( [ \textbf{\texttt{\textbackslash cdiv}} ] \) \( \textbf{\texttt{\textbackslash zdv}} \)
ZEROS \( \textbf{\texttt{\textbackslash vfill}} \) REPLACE VACANT ELEMENTS \( [ \textbf{\texttt{\textbackslash cformat conform struct}} \) SELECT
\( \textbf{\texttt{\textbackslash deblank}} \) SELECT
\( \textbf{\texttt{\textbackslash deblank}} \) SELECT
\( \textbf{\texttt{\textbackslash escape escapex}} \) \( \textbf{\texttt{\textbackslash escape escapex}} \)
\( \textbf{\texttt{\textbackslash escape escapex}} \) \( \textbf{\texttt{\textbackslash escape escapex}} \)}
<table>
<thead>
<tr>
<th>Primary Function Names vs. Abstract</th>
<th>Sorted by Abstract</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADDCOLUMNS</td>
<td>ADD COLUMNS TO A MATRIX VECTOR OR SCALAR</td>
</tr>
<tr>
<td>ADDROWS</td>
<td>ADD ROWS TO A MATRIX VECTOR OR SCALAR</td>
</tr>
<tr>
<td>COMB</td>
<td>ALL COMBINATIONS OF ELEMENTS [DEBLANK UNIQ]</td>
</tr>
<tr>
<td>ALPREC</td>
<td>ALTER PRECISION OF A SCALAR OR MULTIPRECISION NUMBER</td>
</tr>
<tr>
<td>CHAR</td>
<td>BUILD CHARACTER ARRAY TO NUMERIC PATTERN</td>
</tr>
<tr>
<td>CVEC</td>
<td>BUILD COMPRESSION OR LOGICAL VECTOR</td>
</tr>
<tr>
<td>CCAT</td>
<td>CATENATE BY COLUMNS [VERTAB CFORMAT CMATRIX ROWFORM]</td>
</tr>
<tr>
<td>RCAT</td>
<td>CATENATE STRUCTURES BY ROWS [COLFORM CHARACTER VERT]</td>
</tr>
<tr>
<td>OUTPUT</td>
<td>CENTER HEADINGS OVER FORMATTED COLUMNS [CNTR DMZ NEXTA]</td>
</tr>
<tr>
<td>CENTERON</td>
<td>CENTERS AND CATENATES TWO STRUCTURES [CENTER]</td>
</tr>
<tr>
<td>FRAMETEST</td>
<td>CHECKS A MATRIX FOR FRAMING</td>
</tr>
<tr>
<td>COLLECT</td>
<td>COLLECT AND SUMMARIZE COEFFICIENTS OF COMMON CODES</td>
</tr>
<tr>
<td>INDEX</td>
<td>COLUMN INDEX IN MATRIX B WHOSE MEMBERS ALL BELONG TO A</td>
</tr>
<tr>
<td>TABS</td>
<td>COMPARE REQUIRED TAB SETTINGS TO EXISTING ONES [COLNO]</td>
</tr>
<tr>
<td>N2V</td>
<td>COMPRESS CHARACTER MATRIX EXPAND RESULT [V2M]</td>
</tr>
<tr>
<td>QPROBF</td>
<td>COMPUTE CHI SQUARE PROBABILITY FUNCTION</td>
</tr>
<tr>
<td>DATE</td>
<td>COMPUTE NORMAL DATE FROM ASTRONOMERS' DAY NUMBER</td>
</tr>
<tr>
<td>PI</td>
<td>COMPUTE PI TO ARBITRARY PRECISION</td>
</tr>
<tr>
<td>EASTER</td>
<td>COMPUTE THE DATE OF EASTER</td>
</tr>
<tr>
<td>ON</td>
<td>CONFORM AND CATENATE ANY STRUCTURES</td>
</tr>
<tr>
<td>CONV</td>
<td>CONVERT DECIMAL VALUES TO ANY BASE [DIGITS CONFRAC]</td>
</tr>
<tr>
<td>ROMAN</td>
<td>CONVERT INTEGER TO ROMAN NUMERALS</td>
</tr>
<tr>
<td>FORMAT</td>
<td>CONVERT MULTIPRECISION NUMBER TO CHARACTER STRING</td>
</tr>
<tr>
<td>DEC</td>
<td>CONVERT TO DECIMAL</td>
</tr>
<tr>
<td>FIX</td>
<td>CONVERT TO MULTIPRECISION INTEGER</td>
</tr>
<tr>
<td>FLOAT</td>
<td>CONVERT TO MULTIPRECISION FLOATING POINT [SCALE]</td>
</tr>
<tr>
<td>NUMBLANKCOLS</td>
<td>COUNTS BLANK COLUMNS AT SIDES OF STRUCTURE</td>
</tr>
<tr>
<td>STATUS</td>
<td>CURRENT SESSION AND WORKSPACE STATUS [NOW]</td>
</tr>
<tr>
<td>DAYS</td>
<td>DATE CALCULATIONS [DATES NDATES PAYDAY]</td>
</tr>
<tr>
<td>DAYNO</td>
<td>DAY NUMBER FOR ASTRONOMERS [MOONPHASE]</td>
</tr>
<tr>
<td>BLANK</td>
<td>DELETE SPECIFIC STRING FROM STRUCTURE [LIM]</td>
</tr>
<tr>
<td>DT</td>
<td>DELETE TRAILING INSIGNIFICANT CHARACTERS OR VALUES</td>
</tr>
<tr>
<td>DIFF</td>
<td>DIFFERENCES BETWEEN ADJACENT ELEMENTS [UNSCAN]</td>
</tr>
<tr>
<td>VARS</td>
<td>DISPLAY CHARACTERISTICS OR CONTENTS OF VARS SELECTIVELY</td>
</tr>
<tr>
<td>DROUNDS</td>
<td>DISTRIBUTIVE ROUNDING OF A VECTOR TO ARBITRARY SCALAR UNIT</td>
</tr>
<tr>
<td>EDIT</td>
<td>EDIT LATENT EXPRESSION OR CHARACTER STRUCTURE [SEEDIT]</td>
</tr>
<tr>
<td>MEDIT</td>
<td>EDIT MATRIX</td>
</tr>
<tr>
<td>CAN</td>
<td>EDIT MULTIPRECISION INTEGERS INTO CANONICAL FORMAT</td>
</tr>
<tr>
<td>Erect</td>
<td>ERECT WORD MATRIX FROM CHARACTER STRUCTURE [DLTMB]</td>
</tr>
<tr>
<td>XVEC</td>
<td>EXPAND LOGICAL VECTOR</td>
</tr>
<tr>
<td>ADJUSTDOWN</td>
<td>EXTEND THE '{' IN REPORT FORMATTING [ROWINDICES]</td>
</tr>
<tr>
<td>EXTEND</td>
<td>EXTEND VECTOR WITH LAST VALUE</td>
</tr>
<tr>
<td>ADJUSTUP</td>
<td>EXTENDS '{' IN REPORT FORMATTING</td>
</tr>
<tr>
<td>CITED</td>
<td>EXTRACT CITED STRINGS FROM CHARACTER ARRAYS</td>
</tr>
<tr>
<td>FRAME</td>
<td>FRAME AN ARRAY [MATRIX CHARACTER]</td>
</tr>
<tr>
<td>FREQ</td>
<td>FREQUENCY DISTRIBUTION OF ELEMENTS [SORTDA XFORM]</td>
</tr>
<tr>
<td>GRADEUP</td>
<td>GENERATE ASCENDING ROW INDICES [AV NFORM LJNFORM]</td>
</tr>
<tr>
<td>THRU</td>
<td>GENERATE INDICES OR OTHER EQUAL INCREMENTS BETWEEN LIMITS</td>
</tr>
<tr>
<td>ENC</td>
<td>GENERATE SUFFICIENT ENCODING POSITIONS</td>
</tr>
<tr>
<td>LJUST</td>
<td>LEFT JUSTIFY ANY ARRAY [DLB RJUST DL]</td>
</tr>
<tr>
<td>LISTFN</td>
<td>LISTS A FUNCTION IN STANDARD FORM</td>
</tr>
<tr>
<td>ONESIN</td>
<td>LOCATE ONES IN NUMERIC STRUCTURE</td>
</tr>
<tr>
<td>LOC</td>
<td>LOCATE STRUCTURED DATA</td>
</tr>
<tr>
<td>RIOTA</td>
<td>MATRIX ROW IOTA</td>
</tr>
<tr>
<td>WIDTH</td>
<td>MEASURE FORMATTED MATRIX</td>
</tr>
</tbody>
</table>
LOGICAL
AMORTIZE
ADD
DIV
FADD
FDIV
FEXP
FMUL
FSQRT
FSUB
MUL
SQRT
SUB
TABULATE
PAD
BARGRAPH
PREEDIT
BESIDE
PROMPT
HEADERON
REPL
FILLS
TIME
FIRSTM
WORD
DREP
ROUNDS
SHAPE
PREPARE
TCC
TRUNC
ULINE
VFORM
ZDIV

MISCELLANEOUS [ INTEGER FLOATING EMPTY CHARACTER ]
MORTGAGE CALCULATION BY MONTHS
MULTIPRECISION INTEGER ADDITION
MULTIPRECISION INTEGER DIVISION
MULTIPRECISION FLOATING POINT ADDITION
MULTIPRECISION FLOATING POINT DIVISION
MULTIPRECISION FLOATING POINT EXPONENTIAL FUNCTION
MULTIPRECISION FLOATING POINT MULTIPLICATION
MULTIPRECISION FLOATING POINT SQUARE ROOT
MULTIPRECISION FLOATING POINT SUBTRACTION
MULTIPRECISION INTEGER MULTIPLICATION
MULTIPRECISION INTEGER SQUARE ROOT
MULTIPRECISION INTEGER SUBTRACTION
NUMERIC STRUCTURES IN CONTROLLED FORMAT [ HANG ]
NUMERIC VECTORS IN EQUAL INCREMENTS [ BY IN FROM ]
PADS ARRAYS WITH BLANKS OR ZEROS
PLOT HORIZONTAL INTEGER BARGRAPHS
PREPARE MATRIX FOR FUNCTION-LIKE EDITING [ POSTEDIT ]
PRESIDENT TWO STRUCTURES SIDE BY SIDE IN REPORT FORMAT
PROMPT AND CHECK TERMINAL ENTRY OR DEFAULT
PUTS A HEADING ON A REPORT [ COMPARE ]
REPLACE ALL OCCURRENCES OF ELEMENT IN ARRAY
REPLACE VACANT ELEMENTS [ CFORMAT CONFORM STRUCT ]
RUNNING TIME AND NEW SPACE FOR AN APL STATEMENT [ ALT ]
SELECT FIRST OR ONLY APPEARANCE IN MATRIX [ FIRSTV ]
SELECT NTH WORD IN CHARACTER STRUCTURE [ DTMB ]
SELECT UNIQUE ELEMENTS FROM ANY STRUCTURE
SELECTIVE SYMMETRICAL ROUNDING
SHAPE MATRIX FROM CHARACTER STRING
STANDARDIZE STRUCTURE FOR REPORT FORMATTING [ IF ]
SYSTEM INDEPENDENT TERMINAL CONTROL CHARACTERS
TABLE LOOK-UP OF STRUCTURED ARGUMENTS [ IS ]
TRUNCATE HIGHER AND LOWER ORDER DIGITS
UNDERLINE SPECIFIED ROWS OF CHARACTER MATRIX [ USCORE ]
VARIABLE FORMAT BY ROW OF A MATRIX [ ESCAPE ESCAPEX ]
ZERO TOLERANT DIVISION [ CDIV ]
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADD</td>
<td>Multiprecision Integer Addition</td>
</tr>
<tr>
<td>ADDCOLS</td>
<td>Add Columns to a Matrix Vector or Scalar</td>
</tr>
<tr>
<td>ADDBROWS</td>
<td>Add Rows to a Matrix Vector or Scalar</td>
</tr>
<tr>
<td>ADJUSTDOWN</td>
<td>Extend the '</td>
</tr>
<tr>
<td>ADJUSTUP</td>
<td>Extends '</td>
</tr>
<tr>
<td>ALPREC</td>
<td>Alter Precision of a Scalar or Multiprecision Number</td>
</tr>
<tr>
<td>AMORTIZE</td>
<td>Mortgage Calculation by Months</td>
</tr>
<tr>
<td>BARGRAPH</td>
<td>Plot Horizontal Integer Bar Graphs</td>
</tr>
<tr>
<td>BESIDE</td>
<td>Presents Two Structures Side by Side in Report Format</td>
</tr>
<tr>
<td>BLANK</td>
<td>Delete Specific String From Structure [LIN]</td>
</tr>
<tr>
<td>CAN</td>
<td>Edit Multiprecision Integers Into Canonical Format</td>
</tr>
<tr>
<td>CCEAT</td>
<td>Catenate by Columns [VERTAB CFORMAT CMATRIX ROWFORM]</td>
</tr>
<tr>
<td>CENRON</td>
<td>Centers and Catenates Two Structures [CENTER]</td>
</tr>
<tr>
<td>CHAR</td>
<td>Build Character Array to Numeric Pattern</td>
</tr>
<tr>
<td>CITED</td>
<td>Extract Cited Strings From Character Arrays</td>
</tr>
<tr>
<td>COLLECT</td>
<td>Collect and Summarize Coefficients of Common Codes</td>
</tr>
<tr>
<td>COMB</td>
<td>All Combinations of Elements [DEBLANK UNIQ]</td>
</tr>
<tr>
<td>CONV</td>
<td>Convert Decimal Values To Any Base [DIGITS CONFRAC]</td>
</tr>
<tr>
<td>CVEC</td>
<td>Build Compression or Logical Vector</td>
</tr>
<tr>
<td>DATE</td>
<td>Compute Normal Date From Astronomers' Day Number</td>
</tr>
<tr>
<td>DAYNO</td>
<td>Day Number For Astronomers [MOONPHASE]</td>
</tr>
<tr>
<td>DAYS</td>
<td>Date Calculations [DATES NDATES PAYDAY]</td>
</tr>
<tr>
<td>DEC</td>
<td>Convert To Decimal</td>
</tr>
<tr>
<td>DIFF</td>
<td>Differences Between Adjacent Elements [UNSCAN]</td>
</tr>
<tr>
<td>DIV</td>
<td>Multiprecision Integer Division</td>
</tr>
<tr>
<td>DREP</td>
<td>Select Unique Elements From Any Structure</td>
</tr>
<tr>
<td>DROUNDS</td>
<td>Distributive Rounding Of A Vector To Arbitrary Scalar Unit</td>
</tr>
<tr>
<td>DT</td>
<td>Delete Trailing Insignificant Characters Or Values</td>
</tr>
<tr>
<td>EASTER</td>
<td>Compute The Date Of Easter</td>
</tr>
<tr>
<td>EDIT</td>
<td>Edit Latent Expression Or Character Structure [SEDIT]</td>
</tr>
<tr>
<td>ENC</td>
<td>Generate Sufficient Encoding Positions</td>
</tr>
<tr>
<td>ERERT</td>
<td>Erect Word Matrix From Character Structure [DLTMB]</td>
</tr>
<tr>
<td>EXTEND</td>
<td>Extend Vector With Last Value</td>
</tr>
<tr>
<td>FADD</td>
<td>Multiprecision Floating Point Addition</td>
</tr>
<tr>
<td>FDIV</td>
<td>Multiprecision Floating Point Division</td>
</tr>
<tr>
<td>FEXP</td>
<td>Multiprecision Floating Point Exponential Function</td>
</tr>
<tr>
<td>FILL</td>
<td>Replace Vacant Elements [CFORMAT CONFORM STRUCT Δ]</td>
</tr>
<tr>
<td>FIRSTM</td>
<td>Select First Or Only Appearance In Matrix [FIRSTV]</td>
</tr>
<tr>
<td>FIX</td>
<td>Convert To Multiprecision Integer</td>
</tr>
<tr>
<td>FLOAT</td>
<td>Convert To Multiprecision Floating Point [SCALE]</td>
</tr>
<tr>
<td>FMUL</td>
<td>Multiprecision Floating Point Multiplication</td>
</tr>
<tr>
<td>FORMAT</td>
<td>Convert Multiprecision Number To Character String</td>
</tr>
<tr>
<td>FRAME</td>
<td>Frame An Array [MATRIX CHARACTER]</td>
</tr>
<tr>
<td>FRAMETEST</td>
<td>Checks A Matrix For Framing</td>
</tr>
<tr>
<td>FREQ</td>
<td>Frequency Distribution Of Elements [SORTDA KFORM]</td>
</tr>
<tr>
<td>FSQRT</td>
<td>Multiprecision Floating Point Square Root</td>
</tr>
<tr>
<td>FSUB</td>
<td>Multiprecision Floating Point Subtraction</td>
</tr>
<tr>
<td>GRADEUP</td>
<td>Generate Ascending Row Indices [AV NFORM LINFORM]</td>
</tr>
<tr>
<td>HEADERON</td>
<td>Puts A Heading On A Report [COMPARE]</td>
</tr>
<tr>
<td>INDEX</td>
<td>Column Index In Matrix B Whose Members All Belong To A</td>
</tr>
<tr>
<td>LISTEN</td>
<td>Lists A Function In Standard Form</td>
</tr>
<tr>
<td>LJJUST</td>
<td>Left Justify Any Array [DLB RJUST DL]</td>
</tr>
<tr>
<td>LOC</td>
<td>Locate Structured Data</td>
</tr>
<tr>
<td>LOGICAL</td>
<td>Miscellaneous [Integer Floating Empty Character]</td>
</tr>
<tr>
<td>MEDIT</td>
<td>Edit Matrix</td>
</tr>
</tbody>
</table>
MUL  MULTIPRECISION INTEGER MULTIPLICATION
M2V  COMPRESS CHARACTER MATRIX EXPAND RESULT [ V2M ]
NUMBLANKCOLS COUNTS BLANK COLUMNS AT SIDES OF STRUCTURE
ON  CONFORM AND CATENATE ANY STRUCTURES
ONESIN  LOCATE ONES IN NUMERIC STRUCTURE
OUTPUT  CENTER HEADINGS OVER FORMATTED COLUMNS [ CNTR DMZ NEXTA ]
PAD  PADS ARRAYS WITH BLANKS OR ZEROS
PI  COMPUTE PI TO ARBITRARY PRECISION
PREEDIT  PREPARE MATRIX FOR FUNCTION-LIKE EDITING [ POSTEDIT ]
PREPARE  STANDARDIZE STRUCTURE FOR REPORT FORMATTING [ IF ]
PROMPT  PROMPT AND CHECK TERMINAL ENTRY OR DEFAULT
QPROBF  COMPUTE CHI SQUARE PROBABILITY FUNCTION
RCAT  CATENATE STRUCTURES BY ROWS [ COLFORM CHARACTER VERT ]
REPL  REPLACE ALL OCCURRENCES OF ELEMENT IN ARRAY
RIOTA  MATRIX ROW IOTA
ROMAN  CONVERT INTEGER TO ROMAN NUMERALS
ROUNDS  SELECTIVE SYMMETRICAL ROUNDING
SHAPE  SHAPE MATRIX FROM CHARACTER STRING
SQRAT  MULTIPRECISION INTEGER SQUARE ROOT
STATUS  CURRENT SESSION AND WORKSPACE STATUS [ NOW ]
SUB  MULTIPRECISION INTEGER SUBTRACTION
TABS  COMPARE REQUIRED TAB SETTINGS TO EXISTING ONES [ COLNO ]
TABULATE  NUMERIC STRUCTURES IN CONTROLLED FORMAT [ HANG ]
TCC  SYSTEM INDEPENDENT TERMINAL CONTROL CHARACTERS
THRU  GENERATE INDICES OR OTHER EQUAL INCREMENTS BETWEEN LIMITS
TIME  RUNNING TIME AND NEW SPACE FOR AN APL STATEMENT [ ALT ]
TLU  TABLE LOOK-UP OF STRUCTURED ARGUMENTS [ IS ]
TO  NUMERIC VECTORS IN EQUAL INCREMENTS [ BY IN FROM ]
TRUNC  TRUNCATE HIGHER AND LOWER ORDER DIGITS
ULINE  UNDERLINE SPECIFIED ROWS OF CHARACTER MATRIX [ USCORE ]
VARS  DISPLAY CHARACTERISTICS OR CONTENTS OF VARS SELECTIVELY
VFORM  VARIABLE FORMAT BY ROW OF A MATRIX [ ESCAPE ESCAPEX ]
WIDTH  MEASURE FORMATTED MATRIX
WORD  SELECT NTH WORD IN CHARACTER STRUCTURE [ DTMB ]
XVEC  EXPAND LOGICAL VECTOR
ZDIV  ZERO TOLERANT DIVISION [ CDIV ]
## INDEX

<table>
<thead>
<tr>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADD</td>
<td>47</td>
</tr>
<tr>
<td>ADDCOLS</td>
<td>1</td>
</tr>
<tr>
<td>ADDROWS</td>
<td>2</td>
</tr>
<tr>
<td>ADJUSTDOWN</td>
<td>24</td>
</tr>
<tr>
<td>ADJSTUP</td>
<td>25</td>
</tr>
<tr>
<td>ALPREC</td>
<td>48</td>
</tr>
<tr>
<td>ALT</td>
<td>45</td>
</tr>
<tr>
<td>AMORTIZE</td>
<td>63</td>
</tr>
<tr>
<td>AV</td>
<td>11</td>
</tr>
<tr>
<td>BARGRAPH</td>
<td>26</td>
</tr>
<tr>
<td>BESIDE</td>
<td>27</td>
</tr>
<tr>
<td>BLANK</td>
<td>28</td>
</tr>
<tr>
<td>BY</td>
<td>76</td>
</tr>
<tr>
<td>CAN</td>
<td>49</td>
</tr>
<tr>
<td>CCAT</td>
<td>3</td>
</tr>
<tr>
<td>CDIV</td>
<td>78</td>
</tr>
<tr>
<td>CENTER</td>
<td>29</td>
</tr>
<tr>
<td>CENTERON</td>
<td>29</td>
</tr>
<tr>
<td>CEFORMAT</td>
<td>84</td>
</tr>
<tr>
<td>CHAR</td>
<td>4</td>
</tr>
<tr>
<td>CHARACTER</td>
<td>9</td>
</tr>
<tr>
<td>CITPD</td>
<td>30</td>
</tr>
<tr>
<td>CMATRIX</td>
<td>3</td>
</tr>
<tr>
<td>CNTR</td>
<td>36</td>
</tr>
<tr>
<td>COLFORM</td>
<td>16</td>
</tr>
<tr>
<td>COLLECT</td>
<td>31</td>
</tr>
<tr>
<td>COLNO</td>
<td>43</td>
</tr>
<tr>
<td>COMB</td>
<td>79</td>
</tr>
<tr>
<td>COMPARE</td>
<td>33</td>
</tr>
<tr>
<td>CONFORM</td>
<td>33</td>
</tr>
<tr>
<td>CONFRAC</td>
<td>84</td>
</tr>
<tr>
<td>CONV</td>
<td>64</td>
</tr>
<tr>
<td>CVEC</td>
<td>80</td>
</tr>
<tr>
<td>DATE</td>
<td>65</td>
</tr>
<tr>
<td>DATES</td>
<td>67</td>
</tr>
<tr>
<td>DAYNO</td>
<td>66</td>
</tr>
<tr>
<td>DAYS</td>
<td>67</td>
</tr>
<tr>
<td>DEBLANK</td>
<td>79</td>
</tr>
<tr>
<td>DEC</td>
<td>68</td>
</tr>
<tr>
<td>DIFF</td>
<td>5</td>
</tr>
<tr>
<td>DIV</td>
<td>50</td>
</tr>
<tr>
<td>DL</td>
<td>34</td>
</tr>
<tr>
<td>DLB</td>
<td>34</td>
</tr>
<tr>
<td>DLTMB</td>
<td>7</td>
</tr>
<tr>
<td>DMZ</td>
<td>36</td>
</tr>
<tr>
<td>DREP</td>
<td>32</td>
</tr>
<tr>
<td>DROUNDS</td>
<td>69</td>
</tr>
<tr>
<td>DT</td>
<td>81</td>
</tr>
<tr>
<td>DTMB</td>
<td>23</td>
</tr>
<tr>
<td>FASTER</td>
<td>82</td>
</tr>
<tr>
<td>Function</td>
<td>Page</td>
</tr>
<tr>
<td>----------------</td>
<td>------</td>
</tr>
<tr>
<td>FDIT</td>
<td>6</td>
</tr>
<tr>
<td>RMPTY</td>
<td>86</td>
</tr>
<tr>
<td>ENC</td>
<td>70</td>
</tr>
<tr>
<td>FRECT</td>
<td>7</td>
</tr>
<tr>
<td>ESCAPE</td>
<td>21</td>
</tr>
<tr>
<td>ESCAPEX</td>
<td>21</td>
</tr>
<tr>
<td>EXTEND</td>
<td>83</td>
</tr>
<tr>
<td>FADD</td>
<td>51</td>
</tr>
<tr>
<td>FDIV</td>
<td>52</td>
</tr>
<tr>
<td>FEXP</td>
<td>53</td>
</tr>
<tr>
<td>FILLS</td>
<td>84</td>
</tr>
<tr>
<td>FIRSTM</td>
<td>8</td>
</tr>
<tr>
<td>FIRSTV</td>
<td>8</td>
</tr>
<tr>
<td>FIX</td>
<td>54</td>
</tr>
<tr>
<td>FLOAT</td>
<td>55</td>
</tr>
<tr>
<td>FLOATING</td>
<td>86</td>
</tr>
<tr>
<td>FMUL</td>
<td>56</td>
</tr>
<tr>
<td>FORMAT</td>
<td>57</td>
</tr>
<tr>
<td>FRAME</td>
<td>9</td>
</tr>
<tr>
<td>FRAMETEST</td>
<td>10</td>
</tr>
<tr>
<td>FREQ</td>
<td>71</td>
</tr>
<tr>
<td>FROM</td>
<td>76</td>
</tr>
<tr>
<td>FSQRT</td>
<td>58</td>
</tr>
<tr>
<td>FSUB</td>
<td>59</td>
</tr>
<tr>
<td>GRADEUP</td>
<td>11</td>
</tr>
<tr>
<td>HANG</td>
<td>39</td>
</tr>
<tr>
<td>HEADERON</td>
<td>33</td>
</tr>
<tr>
<td>IF</td>
<td>38</td>
</tr>
<tr>
<td>IN</td>
<td>76</td>
</tr>
<tr>
<td>INDEX</td>
<td>12</td>
</tr>
<tr>
<td>INTEGER</td>
<td>86</td>
</tr>
<tr>
<td>IS</td>
<td>90</td>
</tr>
<tr>
<td>KFORM</td>
<td>71</td>
</tr>
<tr>
<td>LIM</td>
<td>28</td>
</tr>
<tr>
<td>LISTFN</td>
<td>40</td>
</tr>
<tr>
<td>LJNFORM</td>
<td>11</td>
</tr>
<tr>
<td>LJUST</td>
<td>34</td>
</tr>
<tr>
<td>LOC</td>
<td>85</td>
</tr>
<tr>
<td>LOGICAL</td>
<td>86</td>
</tr>
<tr>
<td>MATRIX</td>
<td>9</td>
</tr>
<tr>
<td>MEDIT</td>
<td>13</td>
</tr>
<tr>
<td>MOONPHASE</td>
<td>66</td>
</tr>
<tr>
<td>MUL</td>
<td>60</td>
</tr>
<tr>
<td>M2V</td>
<td>14</td>
</tr>
<tr>
<td>NDATES</td>
<td>67</td>
</tr>
<tr>
<td>NEXTA</td>
<td>36</td>
</tr>
<tr>
<td>NFORM</td>
<td>11</td>
</tr>
<tr>
<td>NOW</td>
<td>42</td>
</tr>
<tr>
<td>NUMBLANKCOLS</td>
<td>87</td>
</tr>
<tr>
<td>ON</td>
<td>35</td>
</tr>
<tr>
<td>Function</td>
<td>Page</td>
</tr>
<tr>
<td>-------------</td>
<td>------</td>
</tr>
<tr>
<td>ONESIN</td>
<td>88</td>
</tr>
<tr>
<td>OUTPUT</td>
<td>36</td>
</tr>
<tr>
<td>PAD</td>
<td>37</td>
</tr>
<tr>
<td>PAYDAY</td>
<td>67</td>
</tr>
<tr>
<td>PI</td>
<td>72</td>
</tr>
<tr>
<td>POSTEDIT</td>
<td>15</td>
</tr>
<tr>
<td>PREEDIT</td>
<td>15</td>
</tr>
<tr>
<td>PREPARE</td>
<td>38</td>
</tr>
<tr>
<td>PROMPT</td>
<td>41</td>
</tr>
<tr>
<td>QPROPF</td>
<td>73</td>
</tr>
<tr>
<td>RCAT</td>
<td>16</td>
</tr>
<tr>
<td>REPL</td>
<td>17</td>
</tr>
<tr>
<td>RIOTA</td>
<td>18</td>
</tr>
<tr>
<td>RJUST</td>
<td>34</td>
</tr>
<tr>
<td>ROMAN</td>
<td>74</td>
</tr>
<tr>
<td>ROUNDS</td>
<td>75</td>
</tr>
<tr>
<td>ROWFORM</td>
<td>3</td>
</tr>
<tr>
<td>ROWINDICES</td>
<td>24</td>
</tr>
<tr>
<td>SCALE</td>
<td>55</td>
</tr>
<tr>
<td>SEDIT</td>
<td>6</td>
</tr>
<tr>
<td>SHAPE</td>
<td>19</td>
</tr>
<tr>
<td>SORTDA</td>
<td>71</td>
</tr>
<tr>
<td>SQRT</td>
<td>61</td>
</tr>
<tr>
<td>STATUS</td>
<td>42</td>
</tr>
<tr>
<td>STRUCT</td>
<td>84</td>
</tr>
<tr>
<td>SUB</td>
<td>62</td>
</tr>
<tr>
<td>TABS</td>
<td>43</td>
</tr>
<tr>
<td>TABULATE</td>
<td>39</td>
</tr>
<tr>
<td>TCC</td>
<td>44</td>
</tr>
<tr>
<td>THRU</td>
<td>89</td>
</tr>
<tr>
<td>TIME</td>
<td>45</td>
</tr>
<tr>
<td>TLU</td>
<td>90</td>
</tr>
<tr>
<td>TO</td>
<td>76</td>
</tr>
<tr>
<td>TRUNC</td>
<td>77</td>
</tr>
<tr>
<td>ULINE</td>
<td>20</td>
</tr>
<tr>
<td>UNIQ</td>
<td>79</td>
</tr>
<tr>
<td>UNSCAN</td>
<td>5</td>
</tr>
<tr>
<td>USCORE</td>
<td>20</td>
</tr>
<tr>
<td>VARS</td>
<td>46</td>
</tr>
<tr>
<td>VERT</td>
<td>16</td>
</tr>
<tr>
<td>VERTAB</td>
<td>3</td>
</tr>
<tr>
<td>VFORM</td>
<td>21</td>
</tr>
<tr>
<td>V2M</td>
<td>14</td>
</tr>
<tr>
<td>WIDTH</td>
<td>22</td>
</tr>
<tr>
<td>WORD</td>
<td>23</td>
</tr>
<tr>
<td>XVEC</td>
<td>91</td>
</tr>
<tr>
<td>ZDIV</td>
<td>78</td>
</tr>
<tr>
<td>D</td>
<td>84</td>
</tr>
<tr>
<td>DIGITS</td>
<td>64</td>
</tr>
</tbody>
</table>
This form may be used to communicate your views about this publication. They will be sent to the author's department for whatever review and action, if any, is deemed appropriate. Comments may be written in your own language; use of English is not required.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation whatever. You may, of course, continue to use the information you supply.

Note: Copies of IBM publications are not stocked at the location to which this form is addressed. Please direct any requests for copies of publications, or for assistance in using your IBM system, to your IBM representative or to the IBM branch office serving your locality.

Possible topics for comment are:

- Clarity
- Accuracy
- Completeness
- Organization
- Coding
- Retrieval
- Legibility

If you wish a reply, give your name and mailing address:

________________________________________________________________________

________________________________________________________________________

What is your occupation? ____________________________________________________

Number of latest Newsletter associated with this publication: ____________________

Thank you for your cooperation. No postage stamp necessary if mailed in the U.S.A. (Elsewhere, an IBM office or representative will be happy to forward your comments or you may mail directly to the address in the Edition Notice on the back of the title page.)