VAXAPL.HLP
VAX APL Version 4.0 Help File

!Note that .us cannot be used in the primary level.
!
!
1 APL-applications
This section of the help facility describes four topics:

- o APL meta-functions (METAFNC)
- o APL meta-functions for .bxFMT (QWDFMT)
- o Definitions of SMG$ routines with .bxMAP (SMG)
- o Definitions of GKS$ routines with .bxMAP (GKS)
- o APL Workspace Interchange Standard (WSIS)

2 METAFNC
The installation kit for VAX APL contains an)INPUT file, METAFNC.AAS, that contains models of most of the APL primitive functions and operators, written in APL. These "meta-functions" are included for information only. The meta-functions are neither guaranteed to be accurate nor are a supported part of the VAX APL product.

The primitives in VAX APL for which meta-functions are provided, and the syntax used to invoke each meta-function, are given below:

APL Primitive	Syntax of APL Meta-function
A & B	A DAND B
A .ce B	A DMAXIMUM B
A .lo B	A DCIRCLE B
A , B	A DCATLAM (.io0) AXIS B
A ,[K] B	A DCATLAM K AXIS B
A / B	A DCOMPRESS (.io0) AXIS B
A /[K] B	A DCOMPRESS K AXIS B
A ? B	A DDEAL B (does error checking only)
A .de B	A DDECODE B
A % B	A DDIVIDE B
A .dq B	A DDOMINO B
A .da B	A DDROP B
A .da[K] B	A DDROP.usAXIS K AXIS B
A .en B	A DENCODE B
A = B	A DEQUALS B
A \ B	A DEXPAND (.io0) AXIS B
A \[K] B	A DEXPAND K AXIS B
A * B	A DPOWER B
A .fl B	A DMINIMUM B
A .ge B	A DFGE B
.gd B	MGRADEDOWN (.io0) AXIS B
.gd[K] B	MGRADEDOWN K AXIS B
.gu B	MGRADEUP (.io0) AXIS B
.gu[K] B	MGRADEUP K AXIS B
A .gd B	A DGRADEDOWN B
A .gu B	A DGRADEUP B
A > B	A DFGT B
A f.g B	A OINPROD 'f' LFN 'g' RFN B or A f INNER g B
A .io B	A DINDEXOF B
A .le B	A DFLE B
A .lg B	A DLOG B
A < B	A DFLT B
.ce B	MCEILING B
.lo B	MPITIMES B
, B	MRAVEL B
,[K]	MRAVEL.usAXIS K AXIS B
% B	MRECIPROCAL B

.dq B	MDOMINO B
A .ep B	A DMEMBER B
* B	MEXPON B
.fl B	MFLOOR B
A - B	A DMINUS B
.io B	MIOTA B
.lg B	MNATLOG B
- B	MNEGATIVE B
~ B	MNOT B
+ B	MCONJUGATE B
| B	MMAGNITUDE B
? B	MROLL B (does error checking only)
.rv B	MREVERSE (.io0) AXIS B
.rv[K] B	MREVERSE K AXIS B
! B	MFACTORIAL B
# B	MSIGNUM B
.tr B	MTRANSPOSE B
A .nn B	A DNAND B
A .ne B	A DFNE B
A .nr B	A DNOR B
A .or B	A DOR B
A .so.f B	A OOUTPROD 'f' RFN B or A OUTER f B
A + B	A DPLUS B
f / B	'f' OREDUCE (.io0) AXIS B or f REDUCE B
f /[K] B	'f' OREDUCE K AXIS B or f REDUCE[K] B
f.cs B	f CREDUCE B
f.cs[K] B	f CREDUCE[K] B
A | B	A DRESIDUE B
A .ro B	A DRHO B
A .rv B	A DROTATE (.io0) AXIS B
A .rv[K] B	A DROTATE K AXIS B
f \ B	'f' OSCAN (.io0) AXIS B or f SCAN B
f \[K] B	'f' OSCAN K AXIS B or f SCAN[K] B
f .cb B	f CSCAN B
f .cb[K] B	f CSCAN[K] B
f .dd B	f EACH B
A f .dd B	A f EACH B
A ! B	A DCOMBINATIONS B
A ^ B	A DTAKE B or A TAKE B
A ^[K] B	A DTAKE.usAXIS B
A # B	A DTIMES B
A .tr B	A DTRANSPOSE B
.fm B	MF B or THORN B
A .fm B	A DF B
.ru[K] B	DISCLOSE[K] B
.lu[K] B	ENCLOSE[K] B
.uu B	MUNIQUE B
A .uu B	A DUNION B
A .du B	A DINTERSECTION B
A .z~ B	A DDIFFERENCE B
A .ss B	A DSUBSET B
A .co B	A DCONTAINS B
A .mt B	A DMATCH B
A .bxSS B	A DQSS B
A .bxFMT B	A .ldFMT B
.bxENLIST B	ENLIST B

There is also a meta-function that illustrates APL value display; that is, the way APL formats default terminal output. This meta-function has the syntax: VALDPY B. The meta-function THORN illustrates how nested arrays are formatted according to .bxDC.

The meta-functions for the APL primitives call many special-purpose and utility meta-functions; these special-purpose and utility meta-functions are also in

METAFNC.APL. The meta-function OSCALPROD applies the implicit scalar product operator to a dyadic scalar function to perform scalar extension. The meta-function OSCALPROD.usAXIS applies the implicit scalar product operator to a dyadic scalar function applied to an axis.

Another)INPUT file, QWDFMT.AAS, contains the meta-function for .bxFMT. Note that the meta-functions in that file whose names begin with the characters .ldFMT describe the .bxFMT function.

The following VAX APL primitives do not have specific meta-functions:

```
     .ro B       .xq B
     .cc B       .cc[K] B
     A .cc B           A .cc[K] B
     A .cs B           A .cs[K] B
     A .cb B           A .cb[K] B
     .cr B       .cr[K] B
     A .cr B           A .cr[K] B
```

Implicit arguments to the APL primitives are passed to the meta-functions as the following APL variables:

Variable	Implicit Argument
BXIO	.bxIO
BXCT	.bxCT
.zp.zp	.bxPP
.zp.zw	.bxPW
.zaXIS	Axis for certain functions and operators
.zlFN	Left function to inner product
.zrFN	Right function to inner product

2 QWDFMT

The QWDFMT Model

This describes the QWDFMT.AAS)INPUT file which contains an APL model for the .bxFMT report formatter primitive function in VAX-11 APL v1.1, which is a superset of the $ primitive in APLSF-10/20. To use the model, type:

```
      Result <- Left_argument .ldFMT Right_argument
```

where the right argument may be either a simple array, or a quoted semi-colon list. The left argument must be a character vector domain. Inside QWDFMT, all functions begin with ".ldFMT.us" to avoid potential name conflicts. There are no global variables.

* * *   W A R N I N G   * * *

This note describes most of the differences between the APL model of Quad FMT and the implementation described in the VAX-11 APL Reference Manual (AA-P142B-TE).

The G (picture), T (absolute tab), and Y (byte) format phrases are not supported. The model's error checking is not complete so unexpected results may occur if any of these are included in the left argument format string. The use of G format phrase, in particular, will cause the model to suspend with an error. The W (exponent digits) qualifier is not supported, and causes an error to be signaled. The S (standard symbol substitution) qualifier also causes an error to be signaled.

2 SMG

The)INPUT file SMG.AAS contains the .bxMAP definitions for the SMG$ RTL routines. When you load SGM.AAS into a workspace, all of the definitions in the file are executed and you can call the routines just as you would a user-defined operation.

To load SGM.AAS into a workspace, use the following command:

```
      )INPUT SMG/TTY
```

(Note that you can also load a subset of these SMG$ routines by editing SMG.AAS before loading it.)

SMG.AAS contains definitions for the following routines:

```
     smg$add_key_def
     smg$begin_display_update
     smg$begin_pasteboard_update
     smg$cancel_input
     smg$change_pbd_characteristics
     smg$change_rendition
     smg$check_for_occlusion
     smg$change_virtual_display
     smg$control_mode
     smg$create_pasteboard
     smg$create_virtual_display
     smg$create_virtual_keyboard
     smg$create_key_table
     smg$cursor_column
     smg$cursor_row
     smg$del_term_table
     smg$delete_key_def
     smg$define_key
     smg$delete_chars
     smg$delete_line
     smg$delete_pasteboard
     smg$delete_virtual_display
     smg$delete_virtual_keyboard
     smg$disable_unsolicited_input
     smg$draw_line
     smg$draw_rectangle
     smg$enable_unsolicited_input
     smg$end_display_update
     smg$end_pasteboard_update
     smg$erase_chars
     smg$erase_display
     smg$erase_line
     smg$erase_pasteboard
     smg$find_cursor_display
     smg$flush_buffer
     smg$get_broadcast_message
     smg$get_char_at_physical_cursor
     smg$get_display_attr
     smg$get_key_def
     smg$get_pasteboard_attributes
     smg$get_term_data
     smg$home_cursor
     smg$init_term_table
     smg$init_term_table_by_type
     smg$insert_chars
     smg$insert_line
     smg$invalidate_display
     smg$label_border
     smg$load_key_def
     smg$move_virtual_display
     smg$paste_virtual_display
     smg$pop_virtual_display
     smg$put_chars
     smg$put_char_high_wide
     smg$put_char_wide
     smg$put_line
     smg$put_line_wide
     smg$put_with_scroll
     smg$read_composed_line
     smg$read_from_display
     smg$read_string
     smg$repaint_screen
     smg$repaste_virtual_display
     smg$restore_physical_screen
     smg$return_cursor_pos
     smg$ring_bell
     smg$save_physical_screen
     smg$scroll_display_area
     smg$set_cursor_abs
     smg$set_broadcast_trapping
     smg$set_cursor_rel
     smg$set_default_state
     smg$set_display_scroll_region
     smg$set_keypad_mode
     smg$set_out_of_band_asts
     smg$set_physical_cursor
     smg$snapshot
     smg$unpaste_virtual_display
```

2 GKS

The)INPUT file GKS.AAS contains the .bxMAP definitions for the GKS$ routines. When you load GKS.AAS into a workspace, all of the definitions in the file are executed and you can call the routines just as you would a user-defined operation. You must have the GKS full development license software in order to use these routines.

Before loading GKS.AAS into a workspace you must define the logical symbol GKSDEFS using the following command:

```
      $ DEFINE GKSDEFS SYS$SHARE:GKSRTLIB
```

To load GKS.AAS into a workspace, use the following command:

```
      )INPUT GKS/TTY
```

(Note that you can also load a subset of these GKS$ routines by editing GKS.AAS before loading it.)

GKS.AAS contains definitions for the following routines:

```
     gks$accum_xform_matrix        gks$inq_stroke_state
     gks$activate_ws               gks$inq_text_extent
     gks$assoc_seg_with_ws         gks$inq_text_fac
     gks$await_event               gks$inq_text_indexes
     gks$clear_ws                  gks$inq_text_rep
     gks$close_gks                 gks$inq_valuator_state
     gks$close_seg                 gks$inq_ws_category
     gks$close_ws                  gks$inq_ws_classification
     gks$copy_seg_to_ws            gks$inq_ws_defer_and_update
     gks$create_seg                gks$inq_ws_max_num
     gks$deactivate_ws             gks$inq_ws_state
     gks$delete_seg                gks$inq_ws_type
     gks$delete_seg_from_ws        gks$inq_ws_xform
     gks$emergency_close           gks$inq_wstype_list
     gks$error_handler             gks$inq_xform
     gks$escape                    gks$inq_xform_list
     gks$eval_xform_matrix         gks$insert_seg
```

```
gks$fill_area
gks$flush_device_events
gks$gdp
gks$get_choice
gks$get_item
gks$get_locator
gks$get_string
gks$get_stroke
gks$get_valuator
gks$init_choice
gks$init_locator
gks$init_pick
gks$init_string
gks$init_stroke
gks$init_valuator
gks$inq_active_ws
gks$inq_avail_gdp
gks$inq_choice_state
gks$inq_clip
gks$inq_color_fac
gks$inq_color_indexes
gks$inq_color_rep
gks$inq_current_xformno
gks$inq_def_choice_data
gks$inq_def_defer_state
gks$inq_def_locator_data
gks$inq_def_pick_data
gks$inq_def_string_data
gks$inq_def_stroke_data
gks$inq_def_valuator_data
gks$inq_dyn_mod_seg_attb
gks$inq_dyn_mod_ws_attb
gks$inq_fill_fac
gks$inq_fill_indexes
gks$inq_fill_rep
gks$inq_gdp
gks$inq_indiv_attb
gks$inq_input_dev
gks$inq_input_queue_overflow
gks$inq_level
gks$inq_locator_state
gks$inq_max_ds_size
gks$inq_max_ws_state_table
gks$inq_max_xform
gks$inq_more_simul_events
gks$inq_name_open_seg
gks$inq_open_ws
gks$inq_operating_state
gks$inq_pat_fac
gks$inq_pat_indexes
gks$inq_pick_id
gks$inq_pick_state
gks$inq_pixel
gks$inq_pixel_array_dim
gks$inq_pline_fac
gks$inq_pline_indexes
gks$inq_pline_rep
gks$inq_pmark_fac
gks$inq_pmark_indexes
gks$inq_pmark_rep
gks$inq_predef_color_rep
gks$inq_predef_fill_rep
gks$inq_predef_pline_rep
gks$inq_predef_pmark_rep
gks$inq_predef_text_rep
gks$inq_prim_attb
gks$inq_seg_attb
gks$log_error
gks$open_gks
gks$open_ws
gks$polyline
gks$polymarker
gks$redraw_seg_on_ws
gks$rename_seg
gks$request_choice
gks$request_locator
gks$request_pick
gks$request_string
gks$request_stroke
gks$request_valuator
gks$sample_choice
gks$sample_locator
gks$sample_pick
gks$sample_pick
gks$sample_string
gks$sample_stroke
gks$sample_valuator
gks$select_xform
gks$set_asf
gks$set_choice_mode
gks$set_clipping
gks$set_color_rep
gks$set_defer_state
gks$set_fill_color_index
gks$set_fill_index
gks$set_fill_int_style
gks$set_fill_rep
gks$set_fill_style_index
gks$set_locator_mode
gks$set_pat_ref_pt
gks$set_pat_size
gks$set_pick_id
gks$set_pick_mode
gks$set_pline_color_index
gks$set_pline_index
gks$set_pline_linetype
gks$set_pline_linewidth
gks$set_pline_rep
gks$set_pmark_color_index
gks$set_pmark_index
gks$set_pmark_rep
gks$set_pmark_size
gks$set_pmark_type
gks$set_seg_detectability
gks$set_seg_highlighting
gks$set_seg_priority
gks$set_seg_visibility
gks$set_seg_xform
gks$set_string_mode
gks$set_stroke_mode
gks$set_text_align
gks$set_text_color_index
gks$set_text_expfac
gks$set_text_fontprec
gks$set_text_height
gks$set_text_index
gks$set_text_path
gks$set_text_rep
gks$set_text_spacing
gks$set_text_upvec
gks$set_valuator_mode
gks$set_viewport
gks$set_viewport_priority
gks$set_window
```

```
gks$inq_seg_names            gks$set_ws_viewport
gks$inq_seg_names_on_ws      gks$set_ws_window
gks$inq_seg_priority         gks$text
gks$inq_set_assoc_ws         gks$update_ws
gks$inq_string_state
```

2 WSIS

The APL Workspace Interchange Standard (WSIS) describes a method for transferring workspaces from one APL implementation to another. The WSIS allows a workspace to be transferred regardless of its internal APL format or the size and content of the particular implementation. (Note that you cannot transfer nested arrays.)

The WSIS has been agreed to by implementors of APL and documented in the article "Workspace Interchange Convention," APL Quote-Quad, Vol. 9, No. 3, March 1979.

A workspace to be transferred is converted into a standard format and written to a magnetic tape (or optionally, to a disk file). Then, the workspace can be read from the tape and converted from the standard format to a particular implementation's format.

If you want to use the WSIS, you must install the optional WSIS software when you install VAX APL (for details, see the VAX APL Installation Guide). The optional WSIS software consists of the following:

APLTAP.EXE	A VMS program that copies WSIS-formatted files from disk to tape and from tape to disk.
WSOUT.APL	A VAX APL workspace that contains the function .zq.zqWSOUT, which converts VAX APL workspaces to WSIS-formatted workspaces.
WSIN.APL	A VAX APL workspace that contains the function .zq.zqWSIN, which converts WSIS-formatted workspaces to VAX APL workspaces.

3 Converting VAX APL Workspaces to WSIS-Formatted Workspaces

To create a tape file containing VAX APL workspaces that are to be transferred to a different APL implementation, follow these steps:

Invoke VAX APL, load the workspace that is to be transferred, copy the VAX APL workspace WSOUT from SYS$LIBRARY, and execute the APL function .zq.zqWSOUT. For example:

```
$ APL/TERM=VT340/SILENT
   )LOAD wsname
   )COPY SYS$LIBRARY:WSOUT
   .zq.zqWSOUT 'file-name'
```

This writes the workspace identified by wsname as a disk file with the name file-name. (The default file type of file-name is .AIS.) The disk file includes the workspace functions, operators, and variables, except for those whose names begin with .zq.zq. Certain VAX APL system variables are also copied. The WSIS software does not provide a way to copy the state indicator stack, groups, or channel assignments.

Repeat step 1 for each workspace to be transferred. Use a different file name for each workspace written to disk.

Execute APLTAP.EXE (from SYS$LIBRARY) to write the disk files to a tape (note that you can put multiple workspaces on a single tape). You will need to use the following APLTAP commands:

INITIALIZE	Opens a tape file and writes initial interchange information, which prepares the tape to receive the workspace named by the WRITE command. You will be prompted for the name of the tape file. The default file type of the tape file is .AXF.
WRITE	Copies a disk file to tape (the tape must have been initialized). You will be prompted for the name of the disk file that contains the WSIS-formatted workspace. Records in the disk file may contain a maximum of 512 bytes. The default file type of the disk file is .AIS.
TERMINATE	Closes the tape file.
EXIT	Exits from the APLTAP program. Closes any tape files that were initialized but not terminated.
<CTRL/Z>	Closes the tape file and exits from the APLTAP program (just as if you had executed the TERMINATE and EXIT commands).

For example:

```
$ RUN SYS$LIBRARY:APLTAP
APLTAP!INITIALIZE
Enter tape file specification:  tape
APLTAP!WRITE
Enter file name:  file-name-1
APLTAP!WRITE
Enter file name:  file-name-2
.
.
.
APLTAP!TERMINATE
APLTAP!EXIT
$
```

Note that APLTAP prompts for commands with APLTAP!. You may enter APLTAP commands in either upper- or lowercase, and you may abbreviate them to the shortest unique spelling.

APLTAP requires that the tape have a standard ANSI label. APLTAP writes fixed-length 1892-byte (8-bit bytes) records (it pads the last record with spaces). Other characteristics of the tape, such as density and parity, are not specified by the WSIS; APLTAP will execute successfully only if the sender and receiver have agreed on these characteristics.

You can use APLTAP to copy a WSIS-formatted file to a device other than tape (such as a disk file). If you respond to the tape file specification prompt with a disk file specification, APLTAP prints a warning but continues processing. Thus, although APLTAP will not write to an unlabeled tape, you could copy the WSIS-formatted files to an unlabeled tape by first using APLTAP to create a disk file, and then using some other mechanism to write the disk file to an unlabeled tape.

3 Converting WSIS-Formatted Workspaces to VAX APL Workspaces

To convert workspaces from WSIS format to VAX APL format, follow these steps:

Execute APLTAP.EXE (from SYS$LIBRARY) to copy the WSIS-formatted tape files to disk and to create a command file that will be used to convert the disk files to workspaces.

You will need to use the following APLTAP commands:

READ	Reads one or more tape files and creates disk files for input to the APL function .zq.zqWSIN. The default file type for these tape files is .AXF. Also creates a command file that contains the APL statements needed to execute .zq.zqWSIN. The default file type for the command file is .AAS. If you choose a different file type, then you will have to specify it when you use the)INPUT command (see Step 2).
EXIT or <CTRL/Z>	Exits from the APLTAP program.

For example:

```
$ RUN SYS$LIBRARY:APLTAP
APLTAP!READ
Enter name of command file:  command-file
Enter next tape file name (DONE to exit):  tape-file-1
Total number of errors =
Enter next tape file name (DONE to exit):  tape-file-2
Total number of errors =
.
.
.
Enter next tape file name (DONE to exit):  DONE
APLTAP!EXIT
$
```

APLTAP first prompts for the name of the command file to be created, then it successively prompts for tape files to process until you type DONE.

Note that APLTAP prompts for commands with APLTAP!. You may enter APLTAP commands in either upper- or lowercase, and you may abbreviate them to the shortest unique spelling.

APLTAP creates a disk file named WSINnnnn.AIS for each tape file entered (nnnn is a 4-digit decimal number; the first file is assigned 0000). The names will be used by the command file created for step 2.

A tape record may not exceed 4096 8-bit bytes in length. If the specification you supply as the tape file is not actually a tape device, APLTAP prints a warning but continues processing. Thus, although APLTAP will read only labeled tapes, you can copy a WSIS-formatted workspace from an unlabeled tape by first using some other mechanism to create a disk file from the unlabeled tape, and by then using APLTAP to process the disk file.

Invoke VAX APL and use the)INPUT command to execute the command file created in step 1. For example:

```
$ APL/TERM=VT340/SILENT
      )INPUT command-file/TTY
```

Note that the command file is created in TTY character set.

This procedure creates workspaces with file names taken from the WSIS tape; each workspace has a file type of .APL. If the name of any of the new workspaces is already in use in your default directory, WSIN changes the file type of the new workspace to .Wnn, where nn is a 2-digit decimal number.

WSIN lists the function, operator, and variable names on the terminal as it copies them to the new workspace. When WSIN has processed the entire file, it deletes the WSINnnnn.AIS file produced by APLTAP, but does not delete the command file.

3 Using WSIS to transfer files

The .zq.zqWSOUT and .zq.zqWSIN functions can also be used to transfer files. To convert a file to WSIS output form, load the WSOUT workspace from SYS$LIBRARY: and execute the .zq.zqWSOUT function with a left argument which is the file specification of the file to be converted. (The default file type is .AIX; for other file types include the appropriate qualifier: For instance, /AS for ASCII sequential, .AAS types.) The right argument is the output tape specification, the same as when transferring workspaces.

3 Error messages and Warnings generated by WSIS software

When you use the WSIS software, some error and warning messages may be displayed. In the messages, xx is a hexadecimal number, typically the error code from VMS Record Management Services.

4 WSOUT messages

.zq.zqWSOUT IS DONE
.zq.zqWSOUT has completed processing.

UNABLE TO ASSIGN THE OUTPUT FILE
.zq.zqWSOUT was unable to assign the output file to channel 1.

CREATING OUTPUT FILE: filespec
Informational.

OPERATION LOCKED: name
WSOUT cannot transfer locked function or operator.

UNABLE TO ASSIGN INPUT FILE
It is not possible to open the file to be transferred.

INVALID WORKSPACE IDENTIFIER
The workspace to be transferred has an invalid identifier.

VARIABLE'S VALUE IS NESTED OR HETEROGENEOUS: name
WSOUT cannot transfer nested or heterogeneous variable.

4 WSIN messages

INPUT FILE file NOT FOUND
.zq.zqWSIN was unable to find the specified input file.

FILE IN INCORRECT FORMAT
The specified input file is not in the expected format.

UNKNOWN PSEUDOVARIABLE name IGNORED
The named pseudovariable is unknown.

UNEXPECTED END OF FILE

WS FULL. YOU MUST START OVER.

UNEXPECTED ERROR NUMBER n

WARNING, IDENTIFIER: xxxx BEING TRUNCATED TO: xxx
If an incoming identifier has more than 31 characters, it is truncated to the first 31 characters.

FIX OF OPERATION name FAILED AT LINE n
The function or operator could not be created for some reason. The operation is left as an operation with all its lines turned into comments.

UNABLE TO FIX OPERATION name AT LINE n
A function or operator with all its lines commented out cannot be fixed. The operation is left as a character array.

UNABLE TO ASSIGN THE OUTPUT FILE
A file that is being transferred cannot be created.

EXECUTABLE EXPRESSION: expression SIGNALED THE
 FOLLOWING ERROR error message
An executable expression received an error.

SCALARS OF TYPE type ARE NOT ALLOWED. IGNORED.
An invalid pseudovariable was found.

TOO MANY VERSIONS OF THE SAME NAME--RAN OUT OF SUFFIXES
There are more than 99 files with the same name.

*****ERROR, BAD RANK IN name
The rank information for a transferred object is invalid.

*****ERROR, BAD SHAPE IN name
The shape information for a transferred object is invalid or inconsistent with the rank information.

*****ERROR CREATING NUMERIC VARIABLE name
An attempt was made to create an invalid numeric array. The variable is left in character form for possible repair. Too large an exponent is a possible cause of this message.

*****ERROR CREATING NUMERIC COMPONENT: name
An invalid numeric array was found when transferring a file.

CREATED CHARACTER VARIABLE: name
Successful transfer of a variable.

CREATED NUMERIC VARIABLE: name
Successful transfer of a variable.

CREATED OPERATION: name
Successful transfer of a function or operator.

EXECUTED EXECUTABLE EXPRESSION: expression
Performed the execution of the expression passed in the executable expression pseudovariable.

DONE WITH INPUT FILE filespec
A file has been successfully transferred and created.

4 APLTAP messages

Command Syntax Errors

Illegal command: Not one of READ, WRITE, INIT, TERM, or EXIT.

Tape has already been initialized.
An INITIALIZE command was entered for a tape that was
initialized for writing.

Tape initialized for writing.
A READ command was entered for a tape that was
initialized for writing.

Tape not initialized.
A WRITE or TERMINATE command was entered for a tape that
has not been initialized.

I/O Errors

Unable to open <SYS$INPUT | tape-file | source-file> (xx).
An error occurred when APLTAP tried to open the indicated file.

Unable to create <tape-file | wsin-file | log-file> (xx).
An error occurred when APLTAP tried to create the indicated file
(log-file refers to the command file).

Unable to connect to <SYS$INPUT | log-file | tape-file | wsin-file
| source-file> (xx).
An error occurred when APLTAP tried to connect to the indicated
file (log-file refers to the command file).

Error closing <input-file | tape-file>.
An error occurred when APLTAP tried to close the indicated file.

Unable to write out prologue (xx).
An error occurred when APLTAP tried to write the WSIS
prologue to the tape. The initialization is aborted.

Unable to write END pseudovariable (xx).
An error occurred when APLTAP tried to write the END
pseudovariable to the tape. The tape file is closed.

Unable to write to log file (xx).
An error occurred while APLTAP was writing to the
command file. READ processing is terminated.

Error writing to tape file (xx).
An error occurred while APLTAP was writing a data block
to the tape. Writing of this file is terminated.

Cannot write to WSIN file (xx).
An error occurred while APLTAP was writing to the
WSINnnnn file. READING of that tape file is stopped,
but READ processing continues.

Error reading from <input-file | SYS$INPUT | tape-file> (xx).
An error occurred while APLTAP was reading from the
specified file. Processing of that file is stopped.

Unexpected end of file reading from tape.
End of file occurred while APLTAP was reading the WSIS or
TRANSLATE pseudovariables.

Unexpected error reading from tape (xx).
An error occurred while APLTAP was reading from the
tape file. READING of that file is stopped, but READ

```
 processing continues.

        Tape Format Errors

 Number in tape input too large.
 The WSIS or TRANSLATE pseudovariable contained a numeric
 string that was too long to translate to a 32-bit integer.

 No WSINnnnn name available for use
  (nnnn is a 4-digit decimal number).
 All names of the form WSINnnnn are in use.

 Second vector is not TRANSLATE pseudovariable.

 TRANSLATE pseudovariable is not a matrix.

 TRANSLATE contains too many rows.

 TRANSLATE contains too few columns.

 First vector is not WSIS pseudovariable.

 Format on tape is not convention 0.
 The WSIS software supports version 0 of the
 workspace convention.

        Warnings from APLTAP

 Target device is not tape.

 The file specification given for the tape file does not
 correspond to a magnetic tape device.

 Incorrect length for WSIS pseudovariable.

 Nonblank padding at end of WSIS.

 Nonblank padding at end of TRANSLATE.

 TRANSLATE contains a character with more than two overstrikes
 at entry (xx).

 TRANSLATE contains character not found in .bxAV at entry (xx).

 Illegal reference to undefined character (xxx).

 Total number of errors = (nn).
 Note that if no errors occur, nn in this message is blank.


1 APL-initialization-stream
 Invokes the VAX APL interpreter.  When APL first starts executing,
 it looks for qualifiers and parameters in two places, first in the
 initialization file, and then on the DCL command line that called
 APL.  The initialization file and the DCL command line are called
 initialization streams.

 The APL command, initialization streams, qualifiers and parameters
 are described in detail in the VAX APL User's Guide.

 Format

    APL [/qualifiers] [parameters] [/qualifiers]

2 Parameters
 wsname
```

Specifies the name of a workspace that you want to load instead of the CONTINUE or CLEAR workspace that APL usually provides after startup. The default directory is the user's default area and the default file extension is APL.

2 Qualifiers

Modifies the action taken by the command. APL qualifiers can do the following:

- Specify the APL interface.
- Invoke the APL run-time system.
- Display an informational file.
- Execute a file of APL statements.
- Suppress the printing of start-up messages.
- Identify your terminal type.

The qualifiers marked with (D) are the default settings used by APL if a qualifier is not explicitly included by the user.

/EDIT=(TPU_values)
/NOEDIT=(TPU_values) (D)

The /EDIT qualifier, valid only with the /INTERFACE=CHARACTER_CELL qualifier specifies the TPU values be used with the Character-cell interface. The /EDIT qualifier may not be used in the initialization file. Every qualifier acceptable to EDIT/TPU can be used. For example:

```
       $APL/INTERFACE=CHARACTER_CELL-
     -$EDIT=(COMMAND=APL.TPU,SECTION=APL.TPU$SECTION)
```

/EXECUTE_ONLY

The /EXECUTE_ONLY qualifier directs APL to use the runtime support version of the APL interpreter, which can execute APL applications but cannot be used to develop APL applications. /EXECUTE_ONLY cannot be specified in an initialization file. The /EDIT and /INTERFACE qualifiers may not be used with the /EXECUTE_ONLY qualifier.

/HI = (FILE=filespec, CHSET=charset)
/NOHI (D)

The /HI qualifier specifies the name of a file to be printed when the APL session begins. At most two HI files can be specified, one in each initialization stream.

The /NOHI qualifier prevents the printing of a /HI file that was specified by a /HI qualifier earlier in the same initialization stream.

The file specification must include at least a file name. The default directory is the user's default area and the default file extension is .AAS.

The charset specification indicates the character set of the file and must be one of: BIT, COMPOSITE, KEY, or TTY.

/INPUT = (FILE=filespec, CHSET=charset)
/NOINPUT (D)

The /INPUT qualifier specifies the name of a file to be executed automatically via the APL)INPUT system command when the APL session begins. At most one INPUT file can be specified; APL uses the last one seen when parsing the initialization streams.

The /NOINPUT qualifier negates all previous /INPUT qualifiers. Thus if /NOINPUT appears in the initialization stream and is not followed by a /INPUT qualifier, no)INPUT file is processed.

The file specification must include at least a file name. The default

```
 directory is the user's default area and the default file extension is
 .AAS.

 The charset specification indicates the character set of the file
 and must be one of:  BIT, COMPOSITE, KEY, or TTY.

/INTERFACE={LINE | CHARACTER_CELL | DECwindows}
/NOINTERFACE

 /INTERFACE selects the type of interface. LINE is the default
 interface-type.  This qualifier is not available for use with the
 /EXECUTE_ONLY qualifier and may not be used in initialization files.

 The DECwindows value invokes full DECwindows support of the APL product.
 In addition to the initial APL DECwindow, you can open one or more sessions
 to edit user-defined operations and variables. If the /INTERFACE=DECwindows
 qualifier is used, APL ignores any value assigned to the /TERMINAL
qualifier.

 The CHARACTER_CELL value invokes the VAXTPU-based interface which makes
 windows available to edit user-defined operations and variables.  The /EDIT
 qualifier should be used to specify TPU options.

/SILENT [= BANNER | HI | WSMESSAGE | ALL | NONE]
/NOSILENT (D)

 The /SILENT qualifier controls whether the APL banner line, HI files, and
 initial workspace identification are printed.

 The /NOSILENT qualifier undoes the effect of all previous /SILENT
qualifiers.
 For example, if /NOSILENT is the last qualifier in the command line, the
 banner line, HI files (if any), and initial workspace identification are
 printed.

 Specifying /SILENT=BANNER suppresses printing of the APL banner line.
 Specifying /SILENT=HI suppresses printing of any HI files.  Specifying
 /SILENT=WSMESSAGE suppresses the identification of the initial workspace
 to be loaded.  Parameters to /SILENT may be combined in parentheses,
 for example, /SILENT=(BANNER,HI).  /SILENT=ALL is equivalent to
 /SILENT=(BANNER,HI,WSMESSAGE).  /SILENT=NONE is equivalent to /NOSILENT.
 If /SILENT is specified without a parameter, ALL is assumed.

/TERMINAL = termspec
/NOTERMINAL (D)

 The /TERMINAL qualifier specifies the terminal type.

 The /NOTERMINAL qualifier undoes the effect of all previous /TERMINAL
 qualifiers.

 If /TERMINAL is not specified, or if /NOTERMINAL is not followed by
 /TERMINAL, the user is prompted for the terminal type.

 Type )HELP APL-COMMAND-LINE TERMSPEC  for the list of supported terminals.

/VECTOR{=threshold}
/NOVECTOR

 /VECTOR invokes the vector processor; the non-negative, non-zero integer
 value for threshold specifies the point at which vector processing
 hardware is used in place of the normal scalar processing hardware. A value
 of 0 indicates that the vector processor will never be used. A value of 1
 indicates that the vector processor will always be used.


2 charset
```

The character set specification indicates the kind of character set
being used. Supported character sets are:

BIT	- Bit-paired ASCII/APL character set
COMPOSITE	- Composite APL character set
KEY	- Key-paired ASCII/APL character set
TTY	- VAX APL TTY mnemonics for the APL characters

2 termspec
The terminal specification indicates the kind of terminal being used.
Supported terminals are:

APL	- APL terminal (same as KEY)
BIT	- Bit-paired ASCII/APL terminal
COMPOSITE	- Composite APL character set
HDSAVT	- Human Designed Systems HDSAVT and HDSGVT
HDS201	- Human Designed Systems HDS201 and HDS201G
HDS221	- Human Designed Systems HDS221 and HDS221G
GIGI	- GIGI terminal (VK100)
KEY	- Key-paired ASCII/APL terminal
LA	- LA12, LA34, LA36, LA37, LA38, LA120, LS120, LA100
TTY	- Any terminal without APL character set
4013	- Tektronix 4013
4015	- Tektronix 4015
VS	- Digital VAXstation running VAX Workstation Software
VT102	- Digital VT102-PA/RA video terminal with APL
VT220	- Digital VT220 video terminal
VT240	- Digital VT240 video terminal
VT320	- Digital VT320 video terminal
VT330	- Digital VT330 video terminal
VT340	- Digital VT340 video terminal
DECTERM	- The DECwindows terminal emulator

1 Arithmetic-Functions
The arithmetic functions perform well-known mathematical
operations. All of them take numeric scalar arguments and
return numeric scalar results.

2 Add
The dyadic + function returns the sum of its arguments.

2 Ceiling
The monadic .ce function returns the smallest integer not
less than its argument, within a tolerance defined by .bxCT.

2 Circle
The dyadic .lo function performs trigonometric
functions. Only certain combinations of arguments are valid
for the circle function. The left argument of the circle
function specifies which trigonometric function is to be
performed. For arguments A and B, the table below lists
the possible values of A (near-integer argument), and
indicates the operation associated with each value.

Functions Performed by .lo

A	Function (Z_A.loB)	Domain	Range
.ng7	arc tanh B	1.ge|B	
.ng6	arc cosh B	B.ge1	Z.ge0
.ng5	arc sinh B		
.ng4	(.ng1+B*2)*0.5	1.le|B	Z.ge0
.ng3	arc tan B		(|Z).le.lg0.5
.ng2	arc cos B	1.ge|B	(0.leZ) & Z.le.lg1
.ng1	arc sin B	1.ge|B	(|Z).le.lg0.5
0	(1-B*2)*0.5	1.ge|B	(Z.ge0) & Z.le1
1	sin B		(1Z).le1
2	cos B		(1Z).le1
3	tan B	B.ne2|B%.lg0.5	
4	(1+B*2)*0.5		Z.ge1
5	sinh B		
6	cosh B		Z.ge1
7	tanh B		(|Z).le1

2 Combinations
For integer arguments A and B, the dyadic ! function returns
the number of combinations of B things taken A at a
time. For arguments A and B, the function's domain
is described as follows:

```
.nt(B<0)&(.nt INTEGER B)&.nt INTEGER A
```

where INTEGER is a function that returns 1 if all
the items in its argument are integers, and 0 otherwise.

Notice that the dyadic ! function is related to the
mathematical function BETA as follows:

```
   BETA(A,B) == %B#(A-1)!A+B-1 == %A#(B-1)!A+B-1
```

In the table below the value 1 for A, B, or B-A means that
the argument or the difference between the arguments is a negative
integer; the value 0 means that the argument or the difference
is not a negative integer.

Determining Result for Dyadic !

A	B	B-A	Result
0	0	0	(!B)%(!A)#!B-A
0	0	1	0
0	1	0	APL signals DOMAIN ERROR
0	1	1	(.ng1*A)#A!A-B+1
1	0	0	0
1	0	1	Not a possible case
1	1	0	(.ng1*B-A)#(|B+1)!(|A-1)
1	1	1	0

The ! symbol is formed by overstriking the ' and . symbols.

2 Conjugate
The monadic + function returns a result that is the same as

its argument; thus, +B is identical to B.

2 Divide
The dyadic % function returns the quotient of its arguments.
The right argument for the division function may not be 0,
unless the left argument is also 0; 0 % 0 is 1.

2 Exponential
The monadic * function raises the value of e
(2.71828182845904523536...) to the power specified by its
argument; thus, *B is e to the Bth power.

2 Factorial
The monadic ! of B (for integer arguments) is the product of
the first B positive integers. If the argument to the
factorial function is 0, the result is 1. If the argument
is a negative integer, a DOMAIN ERROR occurs. If
the argument is not an integer, !B is defined in terms
of the mathematical function GAMMA:

```
  !B == GAMMA(B+1)
```

The ! symbol is formed by overstriking the ' and . symbols.

2 Floor
The monadic .fl function returns the greatest integer not
greater than its argument, within a tolerance defined by
.bxCT. The meta-function for .fl is defined as follows:

```
      .dlZ_FLOOR B ;.bxCT ;BXCT ;N
[1]   BXCT_.bxCT  .dm  .bxCT_0
[2]   N_(#B)#.fl0.5 + |B
[3]   Z_N-(N-B)>BXCT # 1.ce|N
[4]   .dl
```

2 Logarithm
The dyadic .lg function returns the logarithm of its right
argument in the base of its left argument; thus, A.lgB is
the logarithm of B in base A. Both arguments must be greater
than zero. The left argument may not be 1 unless the right
argument is also 1; 1 .lg 1 is 1.

The .lg symbol is formed by overstriking the .lo and * symbols.

2 Magnitude
The monadic | function returns the absolute value of its
argument; thus, |B is the absolute value of B (that
is, B=|B, if B.ge0 and (-B)=|B, if B<0).

2 Maximum
The dyadic .ce function returns the greater of its two
arguments.

2 Minimum
The dyadic .fl function returns the smaller of its two
arguments.

2 Multiply
The dyadic # function returns the product of its arguments.

2 Natural-Logarithm
The monadic .lg function returns the natural logarithm of
its argument; thus, .lgB is the natural logarithm (base e)
of B.

The .lg symbol is formed by overstriking the .lo and * symbols.

2 Negative
The monadic - function returns the negative of its argument; thus -B is the negative of B. Be careful not to confuse the negative function with the high minus sign (.ng) used to denote a negative number.

2 Pi-Times
The monadic .lo function returns the product of its argument and the value of pi (3.14159265358979323846264...).

2 Power
The dyadic * raises the value of its left argument to the power specified by its right argument. The power function's domain is restricted to the following combinations of arguments:

Left	Right
Any	0
0	.ge0
>0	Any
<0	Integer

Note that 0*0 is 1.

If the right argument of the * function is exactly 0.5, APL returns the square root of the left argument.

2 Reciprocal
The monadic % function returns the reciprocal of its argument; thus, %B is the reciprocal of B.

2 Residue
The dyadic | function returns the residue of the right argument with respect to the left argument. The residue is obtained by adding or subtracting multiples of the left argument from the right argument. The result of a residue operation takes the sign of the left argument.

If the left and right arguments are equal, the residue is 0. (Note that the residue function is .bxCT-dependent.)
If the left argument is 0, then the residue equals the value of the right argument. If the left argument is not 0, then the residue is in the range of the left argument through 0; it may equal 0 but may not equal the value of the left argument.

2 Roll
When applied to an argument B, the monadic ? function generates an integer randomly selected from the integers .ioB (for a near-integer argument).

Note that the roll function is .bxIO-dependent: ?B when .bxIO is 1 is equivalent to (for the same random link) 1+?B when .bxIO is 0. .bxRL is an implicit argument and result.

Roll may generate duplicate values; thus, it differs from the dyadic deal function which generates a set of unique random numbers.

2 Signum
The monadic # function identifies the sign of its argument; thus, #B is the sign of B. The signum function returns .ng1 if the argument is less than 0, 1 if the argument is greater than 0, and 0 if the argument is equal to 0.

2 Subtract

```
 The dyadic - function returns the difference of its arguments.


1 Axis
 Form:   f[K]B    Af[K]B
 Argument Domain:
        Left
                Type:  Monadic or dyadic function
                Shape: --
                Depth: --
        Right
                Type:  Near-integer (floating for Laminate and Ravel),
                       any for user-defined operations
                Shape: Singleton (Vector for Take, Drop, Ravel, enclose,
                              disclose, and all dyadic scalar functions),
                       any for user-defined operations
                Depth: 0 or 1 (simple),
                       any for user-defined operations
 Result Domain:
                Type:  --
                Rank:  --
                Shape: --
                Depth: --
 Implicit Arguments:  .bxIO

 Axis makes the function to its left apply to the axis specified
 by the near-integer enclosed within brackets.  The following
 functions may be affected by axis:

 catenate (, and .cc)   derived reduction (/ and .cs)
 laminate (, and .cc)   derived scan (\ and .cb)
 ravel    (, and .cc)   reverse (.rv and .cr)
 compress (/ and .cs)   grade up (.gu)
 expand (\ and .cb)     grade down (.gd)
 rotate (.rv and .cr)     drop (.da)
 enclose  (.lu)         disclose (.ru)
 take (^)             all the dyadic scalar functions
 .bxEXP                 user-defined operations
 .bxREP
2 Errors
 AXIS RANK ERROR (NOT VECTOR DOMAIN)
 K is not a singleton and its rank is greater than 1.

 AXIS LENGTH ERROR (NOT SINGLETON)
 K does not have exactly one item (except
 for Take, Drop, Ravel, and all dyadic scalar functions).

 AXIS DOMAIN ERROR (LEFT ARGUMENT HAS WRONG LENGTH)
 the length of a does not equal the length of K.

 AXIS DOMAIN ERROR (INCORRECT TYPE)
 K is not empty and is not numeric.

 AXIS DOMAIN ERROR (NOT AN INTEGER)
 K is not a near-integer (except laminate and ravel).

 AXIS DOMAIN ERROR (INCORRECT OPERATION)
 The function f is not in the domain of axis.

 AXIS DOMAIN ERROR (AXIS LESS THAN QUAD IO)
 The value of K is less than the value of .bxIO.

 AXIS DOMAIN ERROR (RIGHT ARGUMENT HAS WRONG RANK)
 K is greater than the rank of the right argument.

 AXIS DOMAIN ERROR (ARGUMENT RANK AND AXIS INCOMPATIBLE)
 K is greater than the rank of the argument with the largest rank.
```

AXIS DOMAIN ERROR (DUPLICATE AXIS NUMBER)
K contains duplicate values.

AXIS DOMAIN ERROR (AXES NOT IN CONTIGUOUS ASCENDING ORDER)

LIMIT ERROR (INTEGER TOO LARGE)
K is greater than the largest allowable integer.

1 Comments

You can include comments in a user-defined operation definition at the end of the operation header, at the end of lines containing APL statements, or on separate lines. Comments can also appear in immediate mode. The first character in a comment must be a lamp character ("). The text of the comment can consist of any combination of valid APL characters and has no effect on the execution of the operation.

Note that a comment cannot extend across line boundaries. If you want a multi-line comment, you must repeat the lamp character at the beginning of each line of the comment.

1 Editor

APL provides four tools to allow you to define and edit operations.

The DECwindows interface invokes full DECwindows support to more easily edit operators, functions and variables. Enter)HELP EDITOR DECWINDOWS for more information.

The Character-cell interface provides a TPU-based window environment to facilitate the development of operators, functions and variables. Enter)HELP EDITOR CHARACTER-CELL for more information.

The)EDIT system command allows you to edit global APL objects with the VAX TPU editor. Enter)HELP SYSTEM-COMMANDS)EDIT for more information on this editor.

Line-edit mode can be used in any APL session. For more information, enter)HELP EDITOR LINE-EDIT.

2 Character-cell

The Character-cell interface provides a TPU/EVE-based window environment for APL sessions on the Digital VT220, VT240, VT320, VT330, VT340 and DECterm terminals. This environment inserts the text of the operation or variable you are editing into a temporary holding area, a buffer. You can display more than one buffer on the screen at one time and edit more than one operations during an APL session.

To start an edit session, press the Do or the PF4 key to display the Command: prompt. Enter the following command, Substituting the name of the operation or variable for object.

```
        GET object _ object-type
```

Substitute the appropriate APL character for object-type, according to the following table.

operation	.dl
numeric variable	.bx
character variable	.qq

For example, to start editing the user-defined operation STAT, enter the following command:

```
     GET STAT _ .dl
```

You can view operations that are suspended or pendent, but you cannot modify them.

For more information about the TPU/EVE editing environment, press the Do or the PF4 key and enter HELP to access the online Help Utility.

2 DECwindows

In addition to the interactive area in the initial APL DECwindow, the transcript session, you can open one or more sessions to edit user-defined operations and variables.

Follow these steps to start an edit session:

Click on the Commands option located on the Menu Bar in the transcript session to expose the Commands menu.

Select the Edit Existing or Edit New option. If you select the Edit New option, you must also select the object type.

Click on the input area of the dialog box and enter the name of the operation. If you are editing a new variable, you must also specify the type and rank of the variable.

The Title Bar in the edit session is the name of the operation or variable you are editing. You can enter text or you can use the mouse to copy text from the transcript session or another edit session.

Select the Exit, Update Workspace or Quit option from the Commands option in the edit session. Exit updates the workspace and closes the edit session. The Update Workspace option updates the workspace but does not close the edit session. The Quit option does not
update
the workspace and closes the edit session.

2 Line-edit

.dl operation name	Open operation for editing.
.dl	End function editing.
.pd	End function editing and lock operation.
[n]	Define or change line n.
[.ldn]	Delete line n.
[.ldn,l]	Delete all lines in the range beginning with line n and continuing for l lines.
[.ld]	Delete the current line.
[n.bx]	Display line n.
[.bxn]	Display all lines, beginning with line n.
[.bxn,l]	Display all lines in the range beginning with line n and continuing for l lines
[.bx]	Display all lines of the operation.
[n$/s1//s2/]	Beginning at line n, search for string s1 and replace it with string s2.

[m.bxn]	Do character editing of line m; begin at character position n.
[.so.bxn]	Open last entered line for character editing, beginning at character position n. This command can be used in either immediate mode or function-definition mode.
[.so.bx]	List last entered line. This command can be used in either immediate mode or function-definition mode.
/	In character editing, type beneath each character you want to delete.
n	In character editing, insert n (a numeric digit) spaces before current character.
letter	In character editing, insert multiples of 5 spaces -- A inserts 5 spaces, B inserts 10 spaces, and so on.
, text	In character editing, insert text before the current character.
line feed	Delete everything on the line to the right of the line feed.

1 Operation-header-format

The type of a user-defined function or operator depends on the number of arguments it takes. The formats of the headers for the function types are as follows:

Type	Format
niladic	.dl name
monadic	.dl name arg
dyadic	.dl arg2 name arg1
ambivalent	.dl {arg2} name arg1

The formats for the headers for the operator types are as follows:

```
Monadic operator, monadic derived function
   .dl (left-op op-name) arg

Monadic operator, dyadic derived function
   .dl arg2 (left-op op-name) arg1

Monadic operator, ambivalent derived function
   .dl {arg2} (left-op op-name) arg1

Dyadic operator, monadic derived function
   .dl (left-op op-name right-op) arg

Dyadic operator, dyadic derived function
   .dl arg2 (left-op op-name right-op) arg1

Dyadic operator, ambivalent derived function
   .dl {arg2} (left-op op-name right-op) arg1
```

All types of operations may return a result. This would be indicated in the header by an assignment statement following the .dl symbol. For example:

```
                .dl result _ name
```

All types of operations, except niladic functions, may accept an axis argument ({[K]}). This would be indicated in the header after the name of the operation.
For example, for a function:

```
                .dl name {[K]}
```

For example, for an operator

```
                .dl (left-op op-name {[K]}) arg
```

All types of operations may include local symbols. These would be specified at the end of the header in a list separated by semi-colons. The list begins with a semi-colon. For example:

```
                .dl name arg ;local-symbol; local-symbol
```

1 Error-Numbers

If an error is detected during the evaluation of an expression, APL displays:

- o An appropriate primary error message.
- o The text of the line in which the error occurred.
- o A caret approximately underneath the particular point in the line at which the error was discovered.

Often the primary error message is followed on the same line by a secondary error message that offers a more specific explanation of what caused the error. Secondary error messages are enclosed within parentheses. (If you do not want to see secondary error messages, set .bxTERSE to 1.)

When an expression that produces an error is executed by the .bxXQ function, the result returned is an empty array with the shape 0 n, where n is an ERROR NUMBER.

2 0001 FILE NOT FOUND
The requested workspace or file was not found in the specified disk area.

Secondary error message:

(FILE NOT FOUND)

2 0002 SYSTEM ERROR
An internal inconsistency was detected. Please report this error to your DIGITAL software specialist.

2 0003 WORKSPACE FULL
The active workspace could not retain all the information requested, nor could it expand further. Erase unneeded objects, issue a)MAXCORE command to enlarge the workspace, or do a)SAVE,)CLEAR, and)COPY sequence on the needed information.

Secondary error messages:

(EXCESSIVE FRAGMENTATION)

(MAXCORE EXCEEDED)

(VIRTUAL MEMORY EXHAUSTED)

2 0004 NOT A VALID SYSTEM IDENTIFIER
An attempt was made to use a system identifier that is not supported by this APL implementation.

2 0005 DEFN ERROR
Invalid syntax was detected in a line or command entered in function-definition mode.

Secondary error messages:

(CANNOT DELETE HEADER)

(EDIT COMMAND ILLEGAL IN QUAD FX ARGUMENT)

(EXPECTING A DOLLAR SIGN)

(EXPECTING A NUMBER)

(EXPECTING A NUMBER, OR RIGHT BRACKET)

(EXPECTING A NUMBER, QUAD, DELTA, OR JOT)

(EXPECTING A QUAD, OR RIGHT BRACKET)

(EXPECTING A QUAD)

(EXPECTING A RIGHT BRACKET)

(EXPECTING A STRING DELIMITER)

Did not find a delimiter for one of the search or replace strings for dollar sign editing.

(ILL FORMED LINE NUMBER)

(ILL FORMED NUMERIC CONSTANT)

(LEFT BRACKET EXPECTED)

(LINE NUMBER OUT OF RANGE)

A line number greater than 9,999 was specified.

(LINE NUMBER TRUNCATED)

More than five decimal digits were specified in a line number.

(LOCAL SYMBOL EXPECTED)

(NAME IN USE)

An attempt was made to use the same identifier for both arguments of an operation, or for both a label and a local symbol or argument.

(NEGATIVE INTEGER NOT ALLOWED)

(NO PREVIOUS SEARCH STRING)

The search string is empty and there was no previous use of dollar sign editing during this activation of the Del editor.

(NO SYMBOL AFTER OPENING DEL)

The operation name was missing from the line entered.

(NO SYMBOL AFTER RESULT ARROW)

(NOT A SYSTEM VARIABLE)

An attempt was made to localize a system function.

(NOT AN INTEGER)

A print position parameter that is not an integer was entered in
superedit mode.

(NOT IN FUNCTION DEFINITION MODE)

An edit command was entered outside of function-definition mode. Edit
commands are illegal in immediate mode except when used to display or
edit the last executed input line.

(OPERATION LOCKED)

An attempt was made to list or change a locked operation.

(OPERATION SUSPENDED, PENDENT, OR MONITORED)

An attempt was made to edit a pendent or monitored operation, or an
attempt was made to change the number of lines in a suspended operation
or the definition of a local symbol in a suspended operation.

(OPERATION SUSPENDED OR PENDENT)

For)EDIT, an attempt was made to end the VAXTPU session with an EXIT
command when you are not allowed to modify the function.

(RIGHT BRACE EXPECTED)

An error was discovered while the function editor scanned an operation
header and found a left brace that was not balanced with a right brace.

(RIGHT PARENTHESIS EXPECTED)

(RIGHT PARENTHESIS OR SYMBOL EXPECTED)

(SEMICOLON EXPECTED)

(SYMBOL EXPECTED)

(TOO MANY LINES IN OPERATION)

An attempt was made to close an operation that has more than 10,000 lines.

(UNEXPECTED CHARACTER IN HEADER)

2 0006 LABEL ERROR
Improper use of a colon was detected, or an improper variable
name was entered as a label.

Secondary error messages:

(DUPLICATE LABEL)

(NAME IN USE)

An attempt was made to use the same identifier for both a label and
a local symbol or argument.

(OPERATION SUSPENDED, PENDENT, OR MONITORED)

An attempt was made to change a label definition in a suspended, pendent,
or monitored operation.

2 0007 SYNTAX ERROR

Invalid syntax was detected, such as two variables without an intervening operation, an operation call with missing arguments, or an unmatched parenthesis.

Secondary error messages:

(BRANCH NOT ALLOWED IN MIDDLE OR AN EXPRESSION)

The branch (.go) function was used when it was not the principal function of a statement.

(DEPTH ERROR)

Either there are too many nested parentheses or brackets, or the expression is too complex for APL to parse.

(ILL FORMED NUMERIC CONSTANT)

(ILLEGAL CHARACTER IN EXPRESSION)

An internal .bxAV code appeared outside of a literal or comment.

(MISMATCHED DELIMITERS)

(MISSING ARGUMENT)

(MISSING LEFT ARGUMENT TO ASSIGNMENT)

There is no left argument to the specification function. For example, _2 is incorrect.

(MISSING OPERAND)

(NO DYADIC FORM OF DERIVED FUNCTION)

Scan, reduction, expansion, compression, and replication are monadic.

(NO DYADIC FORM OF FUNCTION)

(NO MONADIC FORM OF DERIVED FUNCTION)

Inner and outer product both derive dyadic functions.

(NO MONADIC FORM OF FUNCTION)

(NON-NILADIC FUNCTION HAS NO ARGUMENTS)

An ambivalent, dyadic, or monadic user operation was invoked without any arguments.

(NOT IN FUNCTION DEFINITION MODE)

An editing command was entered at the beginning of a line in immediate mode.

(OPERATOR HAS NO OPERANDS)

(SUBSCRIPT NOT ALLOWED)

An attempt was made to index something that does not have a value.

(UNBALANCED DELIMITER)

(WRONG NUMBER OF ARGUMENTS TO USER FUNCTION)

A monadic user function was invoked with two arguments.

2 0008 ERROR RETURNING FROM EXTERNAL ROUTINE
Secondary error messages:

(DOMAIN ERROR)

A conversion failed when data returned to the workspace.

(ILLEGAL ASCII CHARACTER)

A conversion to ASCII failed as character data returned
to the workspace.

(LENGTH ERROR)

A Varying sTring (/TYPE:VT) returned to the WS is bigger than
it was when it was passed to the external routine. (It is
allowed to be smaller or the same size.)

2 0009 RANK ERROR
The ranks of two operands did not conform.

Secondary error messages:

(ITEMS NOT SCALAR OR ALL THE SAME RANK)

The items of the right argument of disclose (.ru) are
not either scalars or of matching rank.

(ITEMS NOT SINGLETON OR ALL THE SAME RANK)

The items of the right argument must be either singletons or of
matching rank.

(LEFT ITEM NOT VECTOR DOMAIN)

Either the left argument or an item in the left argument to
pick (.ru) is not a singleton and its rank is greater than 1.

(MUST BE VECTOR)

B, in the argument to .bxPACK, is not a vector.

(NOT A SCALAR, VECTOR, OR MATRIX)

.dq, .gu, and .gd only accept arrays with a maximum rank of two.

(NOT MATRIX DOMAIN)

(NOT SINGLETON)

Deal, and the .bxWAIT and .bxDL functions only
accept single numbers as an argument.

(NOT VECTOR DOMAIN)

An argument or value is not a singleton and its rank
is greater than 1.

(NUMERIC PRIMARY KEY MUST BE SINGLETON)

A numeric key for a keyed file must be a singleton.

(RANKS DIFFER BY MORE THAN ONE)

The arguments, after singleton extension to catenate or rotate, differ in rank by more than one.

2 0010 LENGTH ERROR
The shapes of two operands did not conform.

Secondary error messages:

(ARGUMENT MUST BE 1 OR 2 ELEMENTS)

.iq, .oq, .bxCIQ, and .bxCOQ may have at most two items
in their right argument.

(ARGUMENT STRING IS TOO LONG)

The left argument to dyadic .gu or .gd is greater than 256 characters
along any one axis.

(DATA TYPE EXCEEDS DATA LENGTH)

The data type specified for .iq file input or the .bxCIQ function
is incompatible with the length of the left argument.

(DATA TYPE MISSING)

The data type parameter in the right argument to .bxCIQ
is required in this case.

(DISPLAY CONTROL ITEM WRONG LENGTH)

The first item must have length 4. The second item can either be
empty or have length 8.

(DISPLAY CONTROL VECTOR MUST BE TWO ITEMS)

The value must have length 2.

(ILLEGAL EMPTY ARGUMENT)

An empty argument was used with .bxFMT, .bxMAP, .bxQCO, .bxQLD,
.bxQPC, or .bxSIGNAL.

(INDEX LESS THAN INDEX ORIGIN)

An index is less than the current setting of .bxIO.

(INDEX OUT OF RANGE)

For pick (&), an element of the left argument exceeds the
length of the corresponding axis of an item of the right argument.

(ITEM COUNT MISMATCH)

The number of variable names specified in B, in the argument to
.bxPACK, is not equal to the number of packets contained in A.

(KEY VALUE TOO LARGE FOR KEY SIZE)

For /KY files.

(LEFT ARGUMENT LENGTH GREATER THAN RIGHT ARGUMENT DEPTH)

For pick (.ru), the length of the left argument is greater
than the depth of the right argument.

(LEFT ITEM LENGTH NOT EQUAL TO SELECTED ITEM RANK)

For pick (.ru), the length of an item of the left argument

does not match the rank of the selected item at the
corresponding depth of the right argument.

(LEFT LENGTH NOT EQUAL TO RIGHT RANK)

For ^, .da, or dyadic .tr.

(LENGTHS OF INNER AXES DO NOT MATCH)

For Base and Inner product, after singleton extension, the left
argument last axis length must equal the right argument first
axis length.

(NOT SINGLETON)

Dyadic .oq (for /AS files), ?, and the numeric system
variables require a single item for their argument or value.

(NUMBER OF ROWS MUST MATCH)

The number of rows in the arguments to dyadic .dq must match.

(SHAPES OF AXIS DO NOT MATCH)

For Catenate and Rotate, after singleton extension, the shape of
the left argument must match the shape of the right argument
except along the specified axis.

(THERE ARE FEWER ROWS THAN COLUMNS)

The number of rows in the right argument to .dq must be
greater than or equal to the number of columns.

(TOO MANY ELEMENTS IN KEY SPECIFICATION)

For /KY files.

2 0011 VALUE ERROR
A variable name was used and had not been assigned a
value, or a user operation that should return a value was
executed and it did not return a value.

Secondary error message:

(BRANCH HAS NO RESULT)
A branch (.go) expression was used as a response to .bx input.

(FUNCTION DOES NOT RETURN A RESULT)

(FUNCTION RESULT UNDEFINED)

(NO VALUE TO ASSIGN)

(REQUIRED VALUE NOT SUPPLIED)

(SUBSCRIPTED NAME IS UNDEFINED)
In the form A[K]_B, A is not a defined name.

2 0014 DEPTH ERROR
Too many right brackets or parentheses were on the line.

Secondary error message:

(LEFT ARGUMENT DEPTH GREATER THAN 2)
The items in A must be simple (vectors or singletons).

(TOO MANY DIVERTED INPUTS)

Files were nested to a depth greater than 10 with)INPUT.

2 0015 DOMAIN ERROR
The values given for the arguments were outside of the
function domain.

Secondary error messages:

(BUFFER OVERFLOW)

(CANNOT MODIFY SELECTIVE ASSIGNMENT TARGET)

The variable being assigned to cannot be modified by the expression
forming the left argument of the selective assignment. For example:
((.roA_.io2).rvA)_'AB' is incorrect.

(CANNOT SIGNAL EOF)

.bxSIGNAL does not accept 75 as a right argument.

(CHANNEL NOT ASSIGNED)

An attempt was made to use .bxWAIT on an unassigned channel.

(CHANNEL NOT ASSIGNED TO A KEYED FILE)

The file associated with the channel number is not a /KY file.
An attempt was made to use .bxWAIT on an unassigned channel.

(CHARACTER KEY TOO LONG OR NOT IN VECTOR DOMAIN)

For /KY files.

(CONFLICTING QUALIFIERS SPECIFIED)

More than one of the following qualifiers was specified in the
argument to .bxASS:/READONLY, /WRITEONLY, or /UPDATE.

(DATA TYPE MUST BE UNSPECIFIED OR ZERO)

For .bxCIQ or .bxCOQ. In the case of .bxCOQ, APL cannot
create a header and perform a conversion when packing an
enclosed array. This means that for X, an enclosed array,
and N, a non-zero number, the following expressions signal
an error: X .bxCOQ 2 N and X .bxCOQ 4 N

(DELETION NOT ALLOWED)

A sequential delete was attempted for a /KY or /AS file.

(DIVISION BY ZERO)

(DUPLICATE FMT QUALIFIER)

(DUPLICATE FMT STANDARD SUBSTITUTION CHARACTER)

(EMPTY FMT STRING PARAMETER NOT ALLOWED)

(ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)

The argument is not a simple, homogeneous array.

(ENCLOSED VALUE REQUIRED)

The value must be an enclosed array.

(ERROR ACTIVATING IMAGE)

For .bxMAP, if the shared image named by B does not exist.

For .bxTT,)EDIT, or the initialization stream, if there
is an attempt to enter VT220, VT240, VT320, VT330, VT340 or
DECterm mode when SYS$SYSTEM:APLSHR is not accessible.

APL can signal this error when you invoke APL with the /TERMINAL
qualifier, when you use)EDIT with an HDS201 or HDS221 terminal, or
when you set .bxTT.

(ERROR PARSING ARGUMENT TO BLOCK SIZE)

An error was discovered when parsing the /BLOCKSIZE qualifier in the
argument to .bxASS.

(ERROR PARSING ARGUMENT TO BUFFER COUNT)

An error was discovered when parsing the /BUFFERCOUNT qualifier in the
argument to .bxASS.

(ERROR PARSING ARGUMENT TO CCONTROL)

An invalid value was specified for the /CCONTROL qualifier in the
argument to .bxASS.

(ERROR PARSING ARGUMENT TO DEFAULT FILE SPECIFICATION)

An error was discovered when parsing the /DEFAULTFILE qualifier in the
argument to .bxASS.

(ERROR PARSING ARGUMENT TO DISPOSE)

An error was discovered when parsing the /DISPOSE qualifier in the
argument to .bxASS.

(ERROR PARSING ARGUMENT TO EVENT FLAG)

An error was discovered when parsing the /EFN qualifier in the
argument to .bxASS.

(ERROR PARSING ARGUMENT TO KEY SPECIFICATION)

An error was discovered when parsing the /KY qualifier in the
argument to .bxASS.

(ERROR PARSING ARGUMENT TO MAXLEN)

An error was discovered when parsing the /MAXLEN qualifier in the
argument to .bxASS.

(ERROR PARSING ARGUMENT TO PROTECTION)

An error was discovered when parsing the /PROTECTION qualifier in the
argument to .bxASS.

(ERROR PARSING ARGUMENT TO RECORD TYPE)

An error was discovered when parsing the /RECORDTYPE qualifier in the
argument to .bxASS.

(EXTRANEOUS CHARACTERS AFTER COMMAND)

For .bxMAP, if there are any characters other than spaces following B

(FILE IS ASSIGNED WRITE ONLY)

The file associated with the channel number cannot be rewound
because it was assigned with the /WRITEONLY qualifier.

(FILE SPECIFICATION IS MISSING)

There is no file specification or default file specification in the
argument to .bxASS

(FMT DECORATION OR LITERAL STRING TOO LONG)

(FMT RIGHT ARGUMENT DOES NOT MATCH FORMAT PHRASE)

(FONT FILE COULD NOT BE OPENED)

For .bxTT or the initialization stream, if there is an attempt to
enter VT220, VT240, VT320, VT330 or VT340 mode when the APL font
file is not accessible.

(FUNCTION HAS NO FILL ITEM)

Either each (.dd) or outer product (.so.f) was applied with a
user-defined function to an empty argument.

(FUNCTION HAS NO IDENTITY ELEMENT)

The inner axes of an inner product or the reduction axis is empty.

(FUNCTION MISSING)

For .bxMAP, if function-name is not present or if it is followed
by any attributes.

(ILL FORMED FMT PARAMETER)

(ILL FORMED NAME)

Either A has a formal parameter that contains illegal characters,
or B has a value for the /ENTRY or /VALUE qualifier that
contains illegal characters.

(ILLEGAL ASCII CHARACTER)

(ILLEGAL CHARACTER IN FMT LEFT ARGUMENT)

(ILLEGAL COMPOSITE CHARACTER)

(ILLEGAL DATA TYPE CONVERSION)

(ILLEGAL DEC MULTINATIONAL CHARACTER)

(ILLEGAL EMPTY ARGUMENT)

(ILLEGAL FMT FORMAT PHRASE)

(ILLEGAL FMT G FORMAT PHRASE PATTERN CHARACTER)

(ILLEGAL FMT LITERAL STRING DELIMITER)

(ILLEGAL FMT S QUALIFIER SYMBOL)

(ILLEGAL ISO 8BIT CHARACTER)

(ILLEGAL LEFT ARGUMENT TO ASSIGNMENT)

An element of A is not an undefined or variable name.

(ILLEGAL MODE)

(ILLEGAL NAME CLASS)

For .bxPACK, the right argument is not a variable. For assignment (_), the left argument is not either a variable or an undefined name.

(ILLEGAL SELECTIVE ASSIGNMENT FUNCTION)

The function f is not one of the allowed selection functions.

(ILLEGAL USE OF FMT QUALIFIER)

(INCORRECT PARAMETER)

A parameter in the left argument to .bxMAP is incorrect.

(INCORRECT TYPE)
For .bxMAP, if either .ya or .yb is non-empty and numeric

(INDEX LESS THAN INDEX ORIGIN)

An element of an argument is less than the current setting of .bxIO.

(INDEX OUT OF RANGE)

An element of the left argument exceeds the length of the corresponding axis of an item of the right argument.

(INTEGER OVERFLOW)

(INVALID CHANNEL NUMBER)

(INVALID CIQ HEADER)

(INVALID EXTERNAL DATA TYPE)

(INVALID FILE SPECIFICATION)

There is an error in the shared image file specification in the right argument of .bxMAP.

(INVALID FUNCTION IN SELECTIVE ASSIGNMENT)

The principal function or functions in the left argument is inelegible for use with the selective assignment. For example: (A+B)_3 is incorrect.

(INVALID HEADER TYPE)

An incorrect header type was specified for .bxCOQ or .bxCIQ.

(INVALID KEYED FILE PURE DATA TYPE)

For /KY files.

(INVALID LENGTH IN PACK HEADER)

A[1] .ne .roA in the argument to .bxPACK.

(INVALID OBJECT IN INDEXED ASSIGNMENT)

(INVALID OBJECT IN SELECTIVE ASSIGNMENT)

The first object inside the parentheses of selective assignment must be a variable name. For example: (1^2)_3 is incorrect.

(INVALID OBJECT IN STRAND ASSIGNMENT)

(INVALID PACK HEADER)

(.roA) < 8 in the argument to .bxPACK. The shortest possible
packed data has 4 items for the .bxPACK header and 4 items for the
shortest .bxCOQ header.

(INVALID RANK IN PACK HEADER)

A[3] .ne 1 in the argument to .bxPACK. (1 means the packed data is a
vector.)

(INVALID RHO VECTOR IN PACK HEADER)

(.roA) .ne 4 + A[4] in the argument to .bxPACK.

(INVALID TYPE IN PACK HEADER)

A[2] .ne 1 in the argument to .bxPACK. (1 means the type is integer.)

(INVALID WATCH MODE)

An incorrect mode was specified for .bxWATCH.

(KEY OF REFERENCE OUT OF RANGE OR NOT A NUMERIC SINGLETON)

For /KY files.

(KEY NOT FOUND IN TREE)
A specifies an entry point that does not exist in
the shared image.

(LEFT ARGUMENT NOT DENSE FROM INDEX ORIGIN)

For dyadic .tr.

(MISSING FMT FORMAT PHRASE SEPARATOR)

(MISSING FMT FORMAT PHRASE/QUALIFIER CHARACTER)

(MISSING FMT FORMAT PHRASE/QUALIFIER PARAMETER)

(MISSING LITERAL STRING IN FMT LEFT ARGUMENT)

(NAME IN USE)

For .bxMAP, if the name specified for function-name is
already defined as an object other than a function.

(NEGATIVE INTEGER NOT ALLOWED)

Shriek (!) is not defined for negative integers.

(NEGATIVE NUMBER NOT ALLOWED)

(NO DIGIT SELECTOR IN FMT G FORMAT PHRASE PATTERN)

(NO FMT EDITING FORMAT PHRASE)

(NOT A LETTER)

A nonletter was used as the left argument to .bxNL.

(NOT A VALID SYSTEM IDENTIFIER)

(NOT AN EXTERNAL FUNCTION)

For .bxMAP, if the argument names an illegal identifier, a system identifier, a name with no value, or a name that is not an external function.

(NOT AN INTEGER)

An argument is not near-integer.

(OPERATION SUSPENDED, PENDENT, OR MONITORED)

(PARAMETER OUT OF RANGE)

An attempt was made to use an unavailable value as the argument.

(REDUNDANT KEYWORD OR QUALIFIER)

A keyword or qualifier was repeated in the argument to .bxASS.

(RIGHT ARGUMENT IS LESS THAN LEFT)

For dyadic ?.

(RIGHT ARGUMENT IS TOO DEEPLY NESTED)

The right argument to .bxFMT is not a vector domain of simple arrays.

(SEMICOLON LIST NOT ALLOWED)

A semicolon list was used as an argument to a primitive function.

(SINGULAR MATRIX)

For .dq.

(SYSTEM VARIABLE MUST BE 0 OR 1 OR 2 OR 3)

The value of .bxGAG must be 0, 1, 2, or 3.

(SYSTEM VARIABLE VALUE MAY ONLY BE 0 OR 1)

.bxIO, .bxNG, .bxTERSE, .bxTIMEOUT, or .bxTLE only accept 0 or 1.

(TIMEOUT READ UNSUPPORTED FOR CURRENT VALUE OF QUAD TT)

An attempt was made to set .bxTIMELIMIT while the current value of .bxTT indicates a VT220 or VT240 terminal.

(UNBALANCED PARENS IN FMT LEFT ARGUMENT)

(UNBALANCED TEXT DELIMITER IN FMT LEFT ARGUMENT)

(UNPAIRED SYMBOL IN FMT S QUALIFIER)

(UNRECOGNIZED SEARCH MODE)

For /KY files.

(UNSUCCESSFUL TRAP IN LOCKED FUNCTION)

An error occurred while executing the trap expression in a locked function.

(VECTOR PROCESSOR NOT AVAILABLE)

(WIDTH TOO SMALL)

The width parameter for dyadic .fm is too small to accommodate the data.

(WILD CARDS NOT ALLOWED IN FILE SPECIFICATION)

Wildcards are not allowed in the right argument to .bxMAP.

2 0016 UNBALANCED DELIMITER
An input line had unbalanced parentheses, or the argument to the execute function contained unbalanced quotes or .dl characters.

2 0017 EDIT ERROR
An improper character editing request was entered.

Secondary error messages:

(COLUMN POSITION OUT OF RANGE)

The print position number that was entered for superedit was greater than the page width or was negative.

(EXPECTING A RIGHT BRACKET)

An attempt was made to delete the line number during line editing.

(ILL FORMED LINE NUMBER)

(ILLEGAL CHARACTER IN LINE EDIT COMMAND)

The command that was entered included a character other than a letter, digit, /, space, or backspace.

(LEFT BRACKET MISSING)

(LINE EDITING NOT ALLOWED IN EXECUTE)

(NONEXISTENT LINE)

(PREVIOUS INPUT LINE EMPTY)

(OPERATION SUSPENDED, PENDENT, OR MONITORED)

An attempt was made to make an illegal change to a suspended, pendent, or monitored operation.

2 0018 ATTENTION SIGNALED
The attention signal was detected during operation execution.

2 0019 DEVICE DOES NOT EXIST
An invalid device specification was entered.

2 0020 DEVICE NOT AVAILABLE
The requested device has already been assigned to another process.

2 0021 INCORRECT COMMAND
A system command was entered improperly.

Secondary error messages:

(AMBIGUOUS ABBREVIATION)

Not enough characters of a system command were typed to distinguish it from other commands.

(MISSING SYSTEM COMMAND)

A right parenthesis was entered at the beginning of a line and was not followed by a known system command.

(NO SUCH SYSTEM COMMAND)

2 0022 INCORRECT PARAMETER
Invalid syntax was specified for a recognized system command.

Secondary error messages:

(ARGUMENT STRING IS TOO LONG)

The argument entered for)DO or)PUSH was more than 2096 keystrokes.

(CURRENT WORKSPACE CLEARED)

APL failed to load the requested workspace.

(EXTRANEOUS CHARACTERS AFTER COMMAND)

Extra characters were entered after all the required
parameters for a system command.

(ILL FORMED NAME)

In the argument to)ERASE or)GROUP.

(ILL FORMED NUMERIC CONSTANT)

A numeric argument to a system command was entered
improperly.

(ILLEGAL ASCII CHARACTER)

An illegal character was used in the argument to)PUSH.

(ILLEGAL NAME CLASS)

A label or system object was used in the argument to)GROUP.

(INVALID CHARACTER SET QUALIFIER)

An invalid qualifier was used in the argument to)INPUT or)OUTPUT.

(INVALID KEYWORD OR QUALIFIER)

An invalid keyword or qualifier was used in the argument to)INPUT,)OUTPUT,
)SAVE, or)STEP.

(LINE TOO LONG TO TRANSLATE)

The argument entered for)DROP or)LIB was greater than
approximately 2048 keystrokes.

(LOWERCASE QUALIFIER REPEATED)

An invalid repetition of /LOWERCASE was used in the argument to
)DO or)PUSH.

(MISSING ARGUMENT)

An argument was not supplied for a system command that should have one.

(NOKEYPAD QUALIFIER REPEATED)

An invalid repetition of /NOKEYPAD was used in the argument to
)DO or)PUSH.

(NOLOGICALS QUALIFIER REPEATED)

An invalid repetition of /NOLOGICALS was used in the argument to
)DO or)PUSH.

(NOSYMBOLS QUALIFIER REPEATED)

An invalid repetition of /NOSYMBOLS was used in the argument to
)DO or)PUSH.

(NOT A GROUP)

An attempt was made to display the contents of a nongroup.

(NOT A LETTER)

The argument to)NMS,)VARS,)FNS, or)GRPS was not a letter.

(NOTIFY QUALIFIER REPEATED)

An invalid repetition of /NOTIFY was used in the argument to
the)PUSH command.

(NOWAIT QUALIFIER REPEATED)

An invalid repetition of /NOWAIT was used in the argument to
the)PUSH command.

(PARAMETER OUT OF RANGE)

The numeric argument entered for a system command was
outside the legal range of values for the command. The
ranges are:

For)DIGITS, 1 to 16

For)WIDTH, 35 to 2048

For)MAXCORE, the)MINCORE value to 1048576

For)MINCORE, 0 to the)MAXCORE value

For)SAVE/MAXLEN, 512 to 2048

(PARENT QUALIFIER REPEATED)

In the)ATTACH command.

(PROCESS NAME QUALIFIER REPEATED)

In the)PUSH command.

(REDUNDANT KEYWORD OR QUALIFIER)

More than one keyword or qualifier was used in the argument to)OUTPUT or
)STEP.

(SYSTEM VARIABLE VALUE MAY ONLY BE 0 OR 1)

In the)ORIGIN command.

(UNRECOGNIZED QUALIFIER KEYWORD)

(WILD CARDS NOT ALLOWED IN FILE SPECIFICATION)

A wildcard was used in the name of a workspace identifier.

2 0023 WORKSPACE LOCKED
Secondary error messages:

(INCORRECT PASSWORD)

(WORKSPACE HAS NO PASSWORD)

An incorrect password (or none at all) was given to access a workspace that was saved with a password.

2 0024 NOT GROUPED, NAME IN USE

2 0025 EXECUTE ERROR

APL signaled an error while executing the argument to the .xq execute function.

2 0027 LIMIT ERROR
The result of the operation exceeded some implementation limit; for example, if the argument array to .bxFX had more than (.ng1 + 2*16) columns.

Secondary error messages:

(ARGUMENT STRING IS TOO LONG)

The length of an argument cannot be greater than 255 keystrokes.

(ARGUMENT TOO LARGE)

The argument to ! was larger than 100000.

(ARGUMENT TOO LONG)

A contains more than 255 formal parameters (including the result).

(AXIS TOO LONG)

(DELAY VALUE TOO LARGE)

The delay specified for .bxDL was larger than approximately 3.4E11 milliseconds.

(FLOATING OVERFLOW)

(INPUT LINE TOO LONG)

(INTEGER TOO LARGE)

A value is greater than the largest allowable integer.

(PARAMETER OUT OF RANGE)

One of the parameters in the left argument of dyadic .fm is out of range.

(RANK TOO LARGE)

(VOLUME TOO LARGE)

2 0028 AXIS RANK ERROR (NOT VECTOR DOMAIN)
The specified axis number argument was not in a vector domain.

2 0029 AXIS LENGTH ERROR
The specified axis number argument had more than one item.

Secondary error messages:

(ARGUMENT RANK AND AXIS INCOMPATIBLE)

(LEFT ARGUMENT HAS WRONG LENGTH)

The length of the axis argument to ^ or .da does not match the length of the left argument.

(NOT SINGLETON)

2 0030 AXIS DOMAIN ERROR
The specified axis number argument was not a nonnegative integer (except in the case of laminate, which accepts floating-point numbers greater than .ng1), or the specified function was not in the domain of the axis operator.

Secondary error messages:

(ARGUMENT RANK AND AXIS INCOMPATIBLE)

(AXES NOT IN CONTIGUOUS ASCENDING ORDER)

Axis numbers must be in contiguous ascending order for Ravel.

(AXIS LESS THAN INDEX ORIGIN)

(DUPLICATE AXIS NUMBER)

An axis number for ^ or .da was duplicated.

(ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)

K must be a simple homogeneous array.

(INCORRECT OPERATION)

A operation was specified that was not one of the following: Ravel, Catenate/Laminate, Reverse, Rotate, Expand, Scan, Replicate/Compress, Reduce, Monadic Grade up/down, Take, Drop.

(INCORRECT TYPE)

(NOT AN INTEGER)

(RIGHT ARGUMENT HAS WRONG RANK)

An axis number was specified that is greater than the rank of the right argument.

(SEMICOLON LIST NOT ALLOWED)

2 0031 PROTECTION VIOLATION
The protection assigned to the workspace you specified prohibits the access you requested.

Secondary error message:

(INSUFFICIENT PRIVILEGE OR FILE PROTECTION VIOLATION)

2 0032 INVALID SIMULTANEOUS ACCESS
More than one user tried to save the same workspace simultaneously, or a user tried to access a nonshared file that is already in use.

Secondary error message:

(FILE CURRENTLY LOCKED BY ANOTHER USER)

2 0033 IO ERROR
Secondary error messages:

(INVALID WILDCARD OPERATION)

For)OUTPUT, a wildcard was specified in place of a value for the file specification.

(NULL PRIMARY KEY)

(SEQUENTIAL DELETE OPERATION IS NOT ALLOWED FOR KY FILES)

2 0034 COMPONENT ERROR
An attempt was made to read a component that cannot be read.

Secondary error messages:

(COMPONENT CROSSES CELL BOUNDARY)

(COMPONENT IS DAMAGED)

(RECORD NOT A COMPONENT)

2 0035 INVALID FILE SPECIFICATION
Secondary error message:

(WILD CARDS NOT ALLOWED IN FILE SPECIFICATIONS)

Wild cards are invalid in the file specifications for)INPUT and)OUTPUT.

2 0036 INDEX RANK ERROR
Secondary error message:

(CANNOT INDEX A SCALAR)

2 0037 INDEX LENGTH ERROR
Secondary error message:

(INDEX OUT OF RANGE)

2 0038 INDEX DOMAIN ERROR
Secondary error messages:

(INCORRECT TYPE)

An index array that does not consist of nonnegative integers was entered.

(INDEX LESS THAN INDEX ORIGIN)

(NOT AN INTEGER)

2 0039 NO SUCH DIRECTORY

2 0040 OPERATOR DOMAIN ERROR (ARRAY OPERAND NOT ALLOWED)

An array was specified as an operand to an each (.dd) or a dot (.) operator.

2 0041 NO ROOM ON FILE STRUCTURE OR QUOTA EXCEEDED
The specified file structure was full, or the disk allocation was exceeded. In the latter case, files must be deleted from the user's disk area before more files can be added.

2 0042 DEVICE IS WRITE-LOCKED
The specified device (usually a magnetic tape) was physically

write-protected.

2 0043 SYSTEM RESOURCES EXHAUSTED
The system ran out of space to perform certain functions for the user. See the system manager at your installation.

2 0044 ERROR INVOKING EXTERNAL ROUTINE
An error occurred while trying to map an external routine or process the actual arguments before executing the external routine.

Secondary error messages:

(DOMAIN ERROR)

One of the following has occurred:

The data leaving the workspace cannot be converted to the data type expected by the external routine (for example, numbers could not be converted to /TYPE:T).

A conversion failed as data passed from the workspace to the external routine.

(EXTRANEOUS CHARACTERS AFTER COMMAND)

Unrecognized input, such as an undefined or repeated qualifier, appeared at the end of the command.

(ILL FORMED NAME)

The actual parameter specified for either the /ACCESS:OUT or /ACCESS:INOUT qualifier is not a valid APL name.

(ILLEGAL ASCII CHARACTER)

A conversion to ASCII failed as character data /TYPE:T or /TYPE:VT left the workspace.

(ILLEGAL NAME CLASS)

The actual parameter specified for either /ACCESS:OUT or /ACCESS:INOUT qualifier is defined, but is not a variable.

(INCORRECT PARAMETER)

One of the following situations has occurred:

The actual parameter specified for either the /ACCESS:OUT or /ACCESS:INOUT qualifier is currently undefined and is /TYPE:Z. The parameter must be either defined so an unconverted value can be passed or undefined with a known data type, not /TYPE:Z.

The actual argument is missing when the formal parameter was specified with the /MECHANISM:IMMEDIATE qualifier.

(INCORRECT TYPE)

The actual parameter specified for either the /ACCESS:OUT or /ACCESS:INOUT qualifier is not a character.

(LENGTH ERROR)

One of the following situations has occurred:

The actual argument has a length greater than 4 bytes when .bxMAP was specified with the /MECHANISM:IMMEDIATE qualifier.

The actual argument has a length greater that 2+2*16 when dyadic .bxMAP was specified with the /MECHANISM:DESCRIPTOR qualifier.

A complex data type is being passed an odd number of items (APL requires two numbers to form each complex number).

The length of a Varying sTring (/TYPE:VT) is greater than .ng1+2*16.

(NOT VECTOR DOMAIN)

The actual parameter specified for either the /ACCESS:OUT or /ACCESS:INOUT qualifier is not in the vector domain.

(NOT SINGLETON)

The actual argument is not a singleton (as it should be) when dyadic .bxMAP is specified with the /MECHANISM:IMMEDIATE qualifier.

(WRONG NUMBER OF ARGUMENTS TO USER FUNCTION)

More actual arguments were specified than there are formal parameters defined in the formal parameters of the external routine.

2 0045 SIGNAL FROM EXTERNAL ROUTINE
An external routine signaled the error that is the secondary error message.

2 0046 OPERATION INVALID IN THIS CONTEXT
An attempt was made to use)STEP when there was no suspended operation.

2 0047 OUTPUT LINE TOO LONG
Secondary error messages:

(BUFFER OVERFLOW)

A line editing sequence created a line that was too long to fit in the I/O buffer.

(PAGE WIDTH EXCEEDED)

A line editing sequence created a line longer than the page width limit.

2 0048 INPUT LINE TOO LONG
Secondary error message:

(ARGUMENT STRING IS TOO LONG)

The argument to)HELP was longer than APL's input buffer.

2 0049 FILE CONTAINS A DAMAGED WORKSPACE
The file specified by)LOAD,)COPY, or)PCOPY contains a damaged workspace.

Secondary error message:

(CURRENT WORKSPACE CLEARED)

An attempt was made to load a file that contains a damaged workspace. The current workspace is cleared.

2 0050 CHARACTER ERROR
The user entered an illegal overstruck character.

Secondary error messages:

(ILLEGAL CHARACTER IN EXPRESSION)

An internal .bxAV code was included outside of a literal or comment.

(ILLEGAL OVERSTRIKE)

2 0051 INPUT ABORTED
The user typed the abort input signal to escape from quad, quote quad, or del quad input.

2 0052 FUNCTION EDITING ABORTED
The user typed the abort input signal to escape from the function editor.

2 0053 LINE EDITING ABORTED
The user typed the abort input signal to escape from character editing mode.

2 0054 INTERNAL ERROR SAVING WORKSPACE
An internal inconsistency was detected. Please notify your DIGITAL software specialist.

2 0055 NOT A RANDOM ACCESS DEVICE

2 0056 INCORRECT MODE FOR DEVICE
The I/O mode for the action requested was improper for the chosen device.

2 0057 FILE DOES NOT CONTAIN A WORKSPACE
An attempt was made to load or copy a file that does not contain an APL workspace.

Secondary error message:

(CURRENT WORKSPACE CLEARED)

2 0058 DATA TRANSMISSION ERROR
A data transmission error was detected during input or output. This message is usually associated with a nonrecoverable device error.

2 0059 FILE ALREADY EXISTS WITH GIVEN NAME
An attempt was made to save a workspace with the same file name as an existing file that is not a workspace.

2 0060 WS NOT SAVED, THIS WS IS wsname
An attempt was made to save a workspace with the same file name as an existing workspace, without first making that same name the workspace identification (returned by)WSID). This error message is to prevent inadvertent overwriting of previously saved workspaces.

2 0062 NOT A DIRECTORY STRUCTURED DEVICE

2 0063 FILE ASSIGNED READ ONLY

2 0064 CHANNEL NOT ASSIGNED
The channel specified in a file operation was not previously associated with a file via a .bxASS system function.

2 0065 CHANNEL CANNOT DO BOTH INPUT AND OUTPUT
An attempt was made to do both input and output to a channel assigned to a sequentially organized file.

2 0066 NOT AN INPUT DEVICE
The user tried to perform input from an output-only device, such as a line printer.

2 0067 NOT AN OUTPUT DEVICE
The user tried to perform output from an input-only device, such as a card reader.

2 0068 END OF FILE ENCOUNTERED
A sequential read operation was attempted when there was no next record or component and when the channel was assigned with /SIGNAL.

2 0069 RECORD NOT FOUND
A random read operation was attempted on a non-existent record or component when the channel was assigned with /SIGNAL.

2 0071 DEVICE ERROR
A file operation attempted to use a mode that is improper for the device specified in the associated .bxASS function.

2 0072 SYSTEM SERVICE FAILURE

2 0073 SUBPROCESS ERROR
Secondary error message:

(COMMAND BUFFER OVERFLOW - SHORTEN EXPRESSION OR COMMAND LINE)

2 0074 BLOCK TOO BIG
A data-transfer error occurred during I/O. Specifically, the last read attempted to read a block of data that was too large.

2 0075
The end of the file was reached when /SIGNAL was not being used. No message is printed and execution continues.

2 0076 RESULT ERROR (BRANCH HAS NO RESULT)
Branch was used with .bx input.

2 0077 STOPSET
The operation was suspended because a stop bit was set for the current line.

2 0078 END OF TAPE
The end of a reel of magnetic tape was reached.

2 0079 SYSTEM FUNCTION ILLEGAL IN EXECUTE
The .bxBREAK system function was used in the argument to the execute function.

2 0080 RETURN TO CALLER OF THIS IMAGE
The right argument to .bxSIGNAL was 80.

2 0081 BROADCAST RECEIVED
A broadcast was received when .bxGAG was set to 3.

2 0082 CHANNEL NUMBER IS NOT AVAILABLE

2 0083 DAMAGED WORKSPACE HAS BEEN CORRECTED
Secondary error message:

(SOME SYMBOLS MAY HAVE BEEN ERASED)

A workspace, which previously contained corrupted data, was loaded with the /CHECK qualifier.

2 0086 FILE IS ASSIGNED WRITE ONLY

2 0100 HI FILE READ ERROR
An error occurred while reading the file specified by the /HI qualifier on an APL command line or in an initialization file.

2 0101 INITIAL WORKSPACE NOT FOUND
The workspace that was specified on the APL command line or in the initialization file was not found by APL.

2 0102 VECTOR PROCESSOR NOT AVAILABLE

2 0103 ERROR IN INITIALIZATION FILE
APL detected an error while processing the parameters in the initialization file identified by the logical name APL$INIT.

2 0104 NEGATIVE THRESHOLD WITH VECTOR QUALIFIER NOT ALLOWED

2 0105 ERROR INITIALIZING CONSOLE CHANNEL

2 0106 ERROR INITIALIZING WORKSPACE ENVIRONMENT

2 0108 FATAL INITIALIZATION ERROR

2 0109 FATAL ERROR SETTING UP CLEAR WORKSPACE

2 0110 ERROR READING INPUT FILE

2 0111 EDIT COMMAND ERROR
Secondary Error Messages:

(%% QUALIFIER REPEATED)

The same qualifier was specified more than once. %% is the name of the repeated qualifier.

(ARGUMENT TO %% IS OUT OF RANGE)

A numeric value that is outside the acceptable range was specified for a qualifier. %% is the name of the qualifier.

(BAD ARGUMENT TO %%)

An invalid value was specified for a qualifier. %% is the name of the qualifier.

(CANNOT EDIT SYSTEM SYMBOL)

(EDIT COMMAND UNAVAILABLE DURING FUNCTION DEFINITION)

(ENCLOSED ARRAY NOT ALLOWED)

(EXECUTE QUALIFIER ARGUMENT IS TOO LONG)

(ILL FORMED NUMERIC CONSTANT)

(ILL FORMED NUMERIC MATRIX)

(ILLEGAL ASCII CHARACTER)

(ILLEGAL NAME CLASS)

(INCORRECT PARAMETER)

(MISSING ARGUMENT)

(OPERATION LOCKED)

(OPERATION SUSPENDED, PENDENT, OR MONITORED)

(UNBALANCED DELIMITER)

(UNRECOGNIZED QUALIFIER KEYWORD)

(UNSUPPORTED TERMINAL TYPE)

```
 (VOLUME TOO LARGE)

2 0112  ERROR PROCESSING HELP
 Secondary Error Messages:

 (INVALID KEY)

 (TOO MANY HELP KEYS SPECIFIED)

 (ERROR OPENING AS INPUT)

 The file that was specified as the argument to
 the )HELP command did not exist.

2 0113  WATCH POINT ACTIVATED
 Secondary Error Messages:

 (VARIABLE HAS BEEN MODIFIED)

 (VARIABLE HAS BEEN MODIFIED BY INDEX)

 (VARIABLE HAS BEEN REFERENCED)

2 0114  ERROR PROCESSING ATTACH
 Secondary Error Messages:

 An error occurred when APL attempted to process the )ATTACH command.

 (ATTACH REQUEST REFUSED)

 (NONEXISTENT PROCESS)

 (INVALID LOGICAL NAME)

1 Execute-only
 QAPL is an 'execute only' version of APL. It does not support the
 interactive sessions or features necessary for program development
 and cannot use the Character-cell or DECwindows interface.

 You can invoke QAPL with the following DCL command:

 $ APL/EXECUTE_ONLY/TERMINAL=termspec apl-parameters

 The following features of VAX APL are not included in QAPL:

     Immediate mode
     Del editor
     Quad Input
     Certain system variables or functions, including:
        .bxAUS
        .bxCR, .bxFX, .bxVR
        .bxTRACE, .bxSTOP, .bxBREAK, .bxMONITOR, .bxWATCH
     Certain system commands, including:
        )CONTINUE, )SAVE
        )PASSWORD, )WSID (action form)
        )DIGITS, )ORIGIN, )WIDTH, )STEP
        )ERASE, )FNS, )GROUP, )GRP, )GRPS, )NMS,
        )SIV, )SINL, )SIS, )VARS
        )VER, )MON, )OWNER, )CLEAR, )CHARGE, )SIZE, )XLOAD

 )EDIT can edit variables but not functions.

 You can use .bxNL and .bxEX to manipulate workspace objects inside
 of QAPL, and you can use .bxNC to test ambivalent functions inside
 of QAPL.

 There are three system functions and three system commands that allow
```

you to bring objects into a QAPL workspace. These are .bxQPC, .bxQCO, .bxQLD,)COPY,)PCOPY, and)LOAD.

1 File-System

VAX APL provides an extensive file system that allows you to process external data files with the types of file organization that are supported by RMS, the file processing system used by the VAX/VMS operating system.

VAX APL supports the following types of file organization:

- o ASCII sequential organization -- standard ASCII files in which each record (except the last) is logically adjacent to the next record.

- o Internal sequential organization -- files stored in internal APL format. Such files can be accessed faster than ASCII files. Each record (except the last) is logically adjacent to the next record.

- o Direct-access organization -- shareable, random-access files containing records, called components, that are identified by a unique index called a component number. The VAX RMS name for these files is single-key indexed files; APL uses the component number as the key value.

- o Relative organization -- shareable, random-access files containing records identified by a relative record number.

- o Keyed organization -- shareable, random-access files containing records identified by primary and/or secondary keys.

Using the APL file system to process data files is essentially a 3-step process:

- o Associate a file specification and related file information with a channel number. The file can be an existing file or one you want to create.

  ```
  Type )HELP .bxASS  for more information.
  ```

- o Open the file and read or write records until there are no more records to be processed.

  ```
  Type )HELP FILE-SYSTEM INPUT-QUAD   for more information.
       )HELP FILE-SYSTEM OUTPUT-QUAD
  ```

- o Close the file and disassociate it from the channel to which it was assigned.

  ```
  Type )HELP .bxDAS  for more information.
       )HELP .bxCLS
  ```

2 AS-Input-and-Output-Modes

The input or output mode you use to read from or write to an ASCII sequential file. The modes and their meanings are as follows:

Mode	Character Set	Input or Output Type
1	TTY	.bx
2	TTY	.qq
3	TTY	.qd
4	APL	.bx
5	APL	.qq

6	APL	.qd
7	KEY	.bx
8	KEY	.qq
9	KEY	.qd
10	BIT	.bx
11	BIT	.qq
12	BIT	.qd
13	COMPOSITE	.bx
14	COMPOSITE	.qq
15	COMPOSITE	.qd

2 File-Organization-Qualifiers

Qualifier	Default File Extension	Type of File
/AS	.AAS	ASCII sequential; can open for either read or write, but not both.
/AS*	.AAS	ASCII sequential; file is positioned at end of file to allow appending.
/IS	.AIS	Internal sequential; can open for either read or write, but not both.
/IS*	.AIS	Internal sequential; file is positioned at end of file to allow appending.
/DA	.AIX	Direct-access; can do both read and write (this is the default).
/RF	.ARF	Relative; can do both read and write.
/KY	.AKY	Keyed; can do both read and write.

2 Input-Quad

The monadic .iq function is for reading files. It works in much the same way as the basic quad input variable.

The file input-quad function is formed by overstriking the .bx and the _ symbols.

Input form:

```
       .iq [[mode | index]] chan
```

where

mode is an integer representing one of the modes used when accessing files with ASCII sequential organization only. Type)HELP FILE-SYSTEM AS-INPUT-AND-OUTPUT-MODES for more information.

index is the number of a component, record or key specification in a direct-access, relative file or keyed file.

When you specify a mode or index, it must be enclosed in brackets.

chan is a positive integer scalar whose value is a channel number in the range 1 through 999.

The value of the .iq function is the record read from the specified file.

When a .iq function references a channel associated with a file that is not open, APL opens the file, then executes the function.

2 Key-specification

The /KY qualifier has the following form:

```
/KY:k1, k2, k3...
```

The values for /KY represent optional key specifications:
k1 specifies the primary key, k2 specifies the first alternate key,
and so on.

Each key specification is a character string in the following form:

```
e1:e2 :INW | :INL | :INQ | :CHARACTER
```

where

e1 is a numeric character value that specifies the location of the
first byte of the key. (The first byte of the record is location
number 1.)

e2 is a numeric character value that specifies the length, in bytes,
of the key.

INW, INL, INQ, or CHARACTER specify the data type of the key: INW
specifies a 16-bit integer word; INL specifies a 32-bit integer
longword; INQ specifies a 64-bit integer quadword; and CHARACTER
specifies a character key with a length of 1 to 255 bytes.

To read records randomly from a /KY file, use the .iq function.

```
     Form: .iq [value;key-number;technique;key-type] channel data-type
```

where

value is optional and specifies the key for the record you want to
read. It can be a near-integer singleton or a character vector.

key-number is optional and is a near-integer singleton. It specifies
which key of reference you want to read. Use 0 for the primary key,
1 for the first alternate key, and so on. The default is the primary key.

technique is optional and specifies the search technique that APL uses
to retrieve the record you want to read. It belongs to the character
vector domain, and has three possible values: 'EQL', 'GTR', and 'GEQ'.
'EQL' is the default value.

channel is required and specifies the channel number currently assigned
to the /KY file.

data-type is optional and specifies the data type of the record you
want to read.

To read records sequentially from a /KY file, use the .iq function.

```
     Form:    .iq channel data-type
```

2 Output-Quad

The ambivalent .oq function is for writing files. It works in much
the same way as the basic quad output variable. The file output-quad
function is formed by overstriking the .bx and the .go symbols.

Output form:

```
       data     .oq [[mode | index]] chan [[data-type]]
```

where

data is the data that is to be written to the file.

mode is an integer representing one of the modes used when accessing files with ASCII sequential organization only. Type)HELP FILE-SYSTEM AS-INPUT-AND-OUTPUT-MODES for more information.

index is the number of a component, record or key specification in a direct-access, relative file or keyed file.

When you specify a mode or index, it must be enclosed in brackets.

chan is a positive integer scalar whose value is a channel number in the range 1 through 999.

The .oq function is quiet; it does not display a result if it is the leftmost function in a statement. When it is not the leftmost function, .oq returns the value of its left argument.

The .oq function in its monadic form deletes a component or record from a direct-access or relative file. APL signals DOMAIN ERROR if you use monadic .oq with a sequential file. When monadic .oq is not the leftmost function in the statement, it returns the value 0 75.ro0.

When a .oq function references a channel associated with a file that is not open, APL opens the file, then executes the function.

The .oq symbol is formed by overstriking the .bx and .go symbols.

2 Pure-Data-IO

If you want to create files for use by programs written in other languages, you may want to write records containing 'pure' data; that is, records that are vectors of values with no embedded format information.

Similarly, if you want to read files created by non-APL programs, you need a way to tell APL not to look for the internal formatting information that would be in the records if APL had written them.

APL interprets a data record as a vector of pure data if you use the type option with the .iq or .oq function. The formats are:

```
   data .oq [[index]] channel type
        .iq [[index]] channel type
```

where

type specifies the data type to be used to interpret the data. The values and meanings for the type parameter are listed under)HELP GLOSSARY PURE-DATA-TYPE. type is invalid with ASCII sequential files. Also, you can only use the following type parameters with keyed files: 0, 1, 5, 6, and 11-15.

index is the number of a component or record in a direct-access or relative file.

channel is a positive integer scalar whose value is a channel number in the range 1 through 999.

For .iq, the type parameter has the same effect for internal sequential, direct-access, and relative files. It

```
 tells APL to assume that the record is a vector of pure data, and
 to assign the indicated data type to it.

 For .oq, the type parameter tells APL to reformat the data
 in the specified data type, and write it as a single,
 unsegmented vector of values.


1 Function-Names

 Below is a list of the various APL primitive function names.  For more
 information about any of these functions, type )HELP FUNCTION-NAMES
 followed by the desired function's name.

 Note that Possible Errors Generated excludes VALUE ERROR, WORKSPACE FULL
 ERROR, and AXIS errors.

2 Base
 Form:  A.deB
 Argument Domain:
        Left
                Type:   Numeric
                Shape:  Any
                Depth:  0 or 1 (simple)
        Right
                Type:   Numeric
                Shape:  Any
                Depth:  0 or 1 (simple)
 Result Domain:
                Type:   Numeric
                Rank:   0.ce.ng2+(.ro.roA)+.ro.roB
                Shape:  (.ng1.da.roA),1.da.roB
                Depth:  0 or 1 (simple)
 Implicit Arguments:  None

 The dyadic .de function reduces a representation in a number
 system to a value.  More specifically, it converts to
 decimal those vectors along the first axis of the right
 argument that are expressed in the positional number bases
 of radices given by vectors along the last axis of the left
 argument.

3 Errors
 LENGTH ERROR (LENGTH OF INNER AXES DO NOT MATCH)
 the length of last axis of A is not equal to the length of the
 first axis of B, and neither axis is 1.

 DOMAIN ERROR (INCORRECT TYPE)
 Either A or B is not numeric and not empty.

 LIMIT ERROR (VOLUME TOO LARGE)
 (#/.roA)#1^.roB overflows when 1=.ng1^.roA.

 LIMIT ERROR (FLOATING OVERFLOW)
 Arithmetic overflow occurred


2 Branch
 Form:  .goB
 Argument Domain:
                Type:   Near-Integer
                Shape:  Any
                Depth:  0 or 1 (simple)
 Result Domain:  None
 Implicit Arguments:  None

 The monadic .go function (known as branch) modifies
```

the standard order of execution in a user-defined operation.

Normally, lines in functions are executed in the order of their line numbers. You can modify this standard order of execution by using the branch function (.go), which changes the sequence of execution by transferring control to another line in the function.

The branch function is monadic; its argument array must be in a vector domain and its first value must be a near-integer. When the argument array is not a singleton, APL takes the array's first item as the object of the branch.

The possible line number specifications that can be the arguments of a branch function, and the effects of each, are summarized below.

Line Number Specifications

Kind of Line Number Specification	Effect of the Branch
A line number within the operation	Execution continues at that line
Zero or a line number NOT within the operation	Operation execution ends and control returns to immediate mode or the calling operation
An empty array	The branch is ignored; execution continues at the next statement
No argument (bare branch)	Terminates a suspended operation and all preceding pendent operations

Branch must be the principal function in a statement.

3 Errors
SYNTAX ERROR (BRANCH NOT ALLOWED IN MIDDLE OF AN EXPRESSION)
The branch function was used when it was not the principal function of a statement.

VALUE ERROR (BRANCH HAS NO RESULT)
A branch (.go) expression was used as a response to .bx input.

DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
B must be a simple, homogeneous array.

DOMAIN ERROR (INCORRECT TYPE)
B is non-empty and is character when it should be numeric.

DOMAIN ERROR (NOT AN INTEGER)
B is not a near-integer.

LIMIT ERROR (INTEGER TOO LARGE)
B is greater than the largest allowable integer.

2 Catenate-Laminate
```
 Forms:  A,B    A,[K]B    A.ccB    A.cc[K]B
 Argument Domain:
        Left
                Type:   Any
                Shape:  --
                Depth:  Any
        Right
                Type:   Any
                Shape:  --
                Depth:  Any
 Result Domain:
```

```
                Type:   --
                Rank:   1.ce(.ro.roA).ce.ro.roB (for catenate) or
                        1+(.ro.roA).ce.ro.roB (for laminate)
                Shape:  --
                Depth:  Same as deepest argument
 Implicit arguments:  None
```

The dyadic , function joins together the specified axis of two arrays. If for A,[K]B or A.cc[K]B, K is a near-integer, the functions is called catenation and A and B are joined along the Kth axis. If K is not a near-integer, the function is called lamination and A and B are joined along a new axis lying between the axes named by .flK and .ceK.

If you do not specify an axis in the square brackets, , catenates the items along the last axis, and .cc catenates the items along the first axis.

The .cc symbol is formed by overstriking the , and - symbols.

3 Errors

AXIS RANK ERROR (NOT VECTOR DOMAIN)
K is not in the vector domain.

AXIS LENGTH ERROR (NOT SINGLETON)
K is not a singleton.

AXIS DOMAIN ERROR (SEMICOLON LIST NOT ALLOWED)
There is a semicolon inside of the brackets that surround K.

AXIS DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
K must be a simple homogeneous array.

AXIS DOMAIN ERROR (INCORRECT TYPE)
K is not numeric.

AXIS DOMAIN ERROR (AXIS LESS THAN INDEX ORIGIN)
K is less than .bxIO.

AXIS DOMAIN ERROR (ARGUMENT RANK AND AXIS INCOMPATIBLE)
K is greater than the rank of the argument with the largest rank.

RANK ERROR (RANKS DIFFER BY MORE THAN ONE)
A and B are not scalars and their ranks differ by more than 1.

RANK ERROR
For laminate, A and B are not scalars and their ranks do not match.

LENGTH ERROR (SHAPES OFF AXIS DO NOT MATCH)
The shapes of A and B do not match (except along the Kth axis).

DOMAIN ERROR (INCORRECT TYPE)
A and B are not empty and not the same type.

LIMIT ERROR (INTEGER TOO LARGE)
A is greater than the largest allowable integer.

2 Compress-Replicate

```
 Forms:  A/B    A/[K]B    A.csB    A.cs[K]B
 Argument Domain:
        Left
                Type:   Near-integer
                Shape:  Vector domain
                Depth:  0 or 1 (simple)
        Right
                Type:   Any
```

```
               Shape:  Any
               Depth:  Any
Result Domain:
               Type:   Same as right argument
               Rank:   1.ce.ro.roB
               Shape:  ((K-1)^.roB),(+/|A),K.da.roB (when &/A<0 is true)
                       ((K-1)^.roB),(.roB)[K],K.da.roB (when &/A<0 is false)
               Depth:  Any
Implicit Arguments:  Fill item (Type )HELP GLOSSARY FILL-ITEM for
                                 more information.)
```

Compression and replication are monadic functions derived from the slash (/) operator. They build arrays by specifying the items to be deleted, preserved, or duplicated from an existing array, and by indicating where fill items are to be added in the new array. When items are preserved or deleted, this is known as compression (the left operand is Boolean). When items are duplicated, deleted, or filled, this is known as replication (the left operand is integer).

For compression, each Boolean item in A corresponds to the position of an item in B. When A is 1, the item in B is preserved in the result array. When A is 0, the item in B is deleted from the result array.

For replication, each positive scalar and each zero in A corresponds to the position of an item in B. Negative integers, which specify fill items, are not associated with explicit positions in B. When A is Boolean, the effects are the same as for compression (items are either preserved or deleted in the result array). When 1<A, the item in B is repeated A times in the result array. When A is negative, APL builds .abA occurrences of the fill item into the new array.

If you do not specify an axis in the square brackets, / compresses the items along the last axis, and .cs compresses the items along the first axis.

See also .bxREP (Type)HELP .bxREP for more information.)

The .cs symbol is formed by overstriking the / and - symbols.

3 Errors

AXIS RANK ERROR (NOT VECTOR DOMAIN)
K is not in the vector domain.

AXIS LENGTH ERROR (NOT SINGLETON)
K is not a singleton.

AXIS DOMAIN ERROR (SEMICOLON LIST NOT ALLOWED)
There is a semicolon inside of the brackets that surround K.

AXIS DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
K must be a simple homogeneous array.

AXIS DOMAIN ERROR (INCORRECT TYPE)
K is not numeric.

AXIS DOMAIN ERROR (NOT AN INTEGER)
K is not near-integer.

AXIS DOMAIN ERROR (AXIS LESS THAN INDEX ORIGIN)
K is less than .bxIO.

AXIS DOMAIN ERROR (RIGHT ARGUMENT HAS WRONG RANK)
K is greater than the rank of B.

RANK ERROR (NOT VECTOR DOMAIN)

```
 A is not a singleton and its rank is greater than 1.

 LENGTH ERROR
 B is not a singleton and its length along the Kth axis
 is not equal to the number of nonnegative integers in A.

 DOMAIN ERROR (INCORRECT TYPE)
 A is not empty and not numeric.

 DOMAIN ERROR (NOT AN INTEGER)
 A is not a near-integer.

 LIMIT ERROR (INTEGER TOO LARGE)
 Either A or K is greater than the largest allowable integer.

 SYNTAX ERROR (NO DYADIC FORM OF DERIVED FUNCTION)
 Compression and replication are monadic.

2 Contains
 Form:  A.coB
 Argument Domain:
        Left
                Type:   Any
                Shape:  Any
                Depth:  Any
        Right
                Type:   Any
                Shape:  Any
                Depth:  Any
 Result Domain:
                Type:   Boolean
                Rank:   0 (scalar)
                Shape:  Scalar
                Depth:  0 (simple scalar)
 Implicit Arguments:  .bxCT (determines comparison precision)

 Possible Errors Generated:  None

 The dyadic .co function determines whether the left
 argument contains all of the items found in the right
 argument. The result is a Boolean scalar: true, if the
 left argument is a superset of the right argument, and
 false if it is not. Duplicate items in either argument
 do not affect the result.

 The .co symbol is formed by overstriking the .ru and .us
 symbols.

3 Errors
 No errors generated

2 Deal
 Form:  A?B
 Argument Domain:
        Left
                Type:   Nonnegative near-integer
                Shape:  Singleton
                Depth:  0 or 1 (simple)
        Right
                Type:   Nonnegative near-integer
                Shape:  Singleton
                Depth:  1 (simple)
 Result Domain:
                Type:   Nonnegative integer
                Rank:   1 (vector)
                Shape:  A
                Depth:  1 (simple vector)
 Implicit Arguments:  .bxIO
                      .bxRL (random number generator seed)

 For A?B, the dyadic ? function generates a vector
 of integers randomly selected from .ioB; no number is
 selected more than once.  The length of the result vector is
 specified by A.

 Note that the deal function is .bxIO-dependent; A?B when .bxIO
 is 1 is equivalent to 1+A?B when .bxIO is 0.

3 Errors
 RANK ERROR (NOT SINGLETON)
 A or B is not a singleton and its rank is greater than 1.

 LENGTH ERROR (NOT SINGLETON)
 A or B does not have exactly 1 item.

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 A and B must be simple homogeneous arrays.

 DOMAIN ERROR (INCORRECT TYPE)
 A or B is not empty and not numeric.

 DOMAIN ERROR (NOT AN INTEGER)
 A or B is not a near-integer.

 DOMAIN ERROR (NEGATIVE NUMBER NOT ALLOWED)
 A or B is less than 0.

 DOMAIN ERROR (RIGHT ARGUMENT IS LESS THAN LEFT)
 B is less than A.

 LIMIT ERROR (INTEGER TOO LARGE)
 A is greater than the largest allowable integer.

2 Depth
 Form:  .mtB
 Argument Domain:
                Type:   Any
                Shape:  Any
                Depth:  Any
 Result Domain:
                Type:   Integer (non-negative)
                Rank:   0 (scalar)
                Shape:  Scalar
                Depth:  0 (simple scalar)
 Implicit Arguments:  None

 Possible Errors Generated:  None

 The monadic .mt function (known as depth) indicates the maximum
 level of nesting in an array. A simple array has 1 level of
 nesting (0 if the array is scalar). An enclosed array has a depth
 of at least 2.

3 Errors
 No errors generated

2 Disclose
 Forms:  .ruB   .ru[K]B
 Argument Domain:
                Type:   Any
                Shape:  Any
                Depth:  Any
 Result Domain:
                Type:   Same as constituent items in B
```

```
               Rank:   (.ro.roB)+^.ce/.ro.dd.ro.dd(,B),.lu^B
               Shape:  (.roB),^.ce/(.ro.dd(,B),.lu^B)~.lu.io0
                       (.roZ)[,K] .sa ^.ce/(.ro.dd(,B)~.lu.io0
               Depth:  0.ce.ng1+.mtB
Implicit Arguments:  None
```

The monadic .ru function reduces the depth of an array. It
reverses the building action of the monadic enclose (.lu)
function.

Disclose reduces one level of enclosure in the argument array.

3 Errors

AXIS RANK ERROR (NOT VECTOR DOMAIN)
K is not a singleton and its rank is greater than 1.

AXIS LENGTH ERROR (ARGUMENT RANK AND AXIS INCOMPATIBLE)
The length of K is not equal to the rank of the items of B.

AXIS DOMAIN ERROR (SEMICOLON LIST NOT ALLOWED)
There is a semicolon inside of the brackets that surround K.

AXIS DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
K is not a simple homogeneous array.

AXIS DOMAIN ERROR (INCORRECT TYPE)
K is not empty and not numeric.

AXIS DOMAIN ERROR (NOT AN INTEGER)
K is not a near-integer.

LIMIT ERROR (INTEGER TOO LARGE)
K is greater than the largest allowable integer.

AXIS DOMAIN ERROR (AXIS LESS THAN INDEX ORIGIN)
K is less than the current setting of .bxIO.

AXIS DOMAIN ERROR (ARGUMENT RANK AND AXIS INCOMPATIBLE)
An element of K is greater than the rank of the items
in B plus the rank of B.

AXIS DOMAIN ERROR (DUPLICATE AXIS NUMBER)
K contains duplicate values.

RANK ERROR (ITEMS NOT SINGLETON OR ALL THE SAME RANK)
The items of B must be either singletons or of matching rank.

2 Drop

```
Forms:  A.daB  A.da[K]B
Argument Domain:
       Left
               Type:   Near-integer
               Shape:  Vector domain
               Depth:  0 or 1 (simple)
       Right
               Type:   Any
               Shape:  Any
               Depth:  Any
Result Domain:
               Type:   Same as right argument
               Rank:   Same as right argument
               Shape:  0.ce(.roB)-|A (if no explicit axis)
               Depth:  Same as right argument
Implicit Arguments:  None
```

The dyadic .da function builds an array by dropping a
specified number of items from an existing array.

The left argument specifies how many items are to be
dropped from each axis in the right argument array. Thus,
for A.daB, item A[K] is used to drop values along axis
K of B. If the value of the argument is positive, APL drops
the specified number of item from the beginning of the
vector; if the value is negative, APL drops items from
the end of the vector.

3 Errors

AXIS DOMAIN ERROR (SEMICOLON LIST NOT ALLOWED)
There is a semicolon inside of the brackets that surround K.

AXIS DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
K must be a simple homogeneous array.

AXIS RANK ERROR (NOT VECTOR DOMAIN)
K is not in the vector domain.

AXIS LENGTH ERROR (LEFT ARGUMENT HAS WRONG LENGTH)
The length of A does not equal the length of K.

AXIS DOMAIN ERROR (ARGUMENT RANK AND AXIS INCOMPATIBLE)
The length of K is greater than the rank of the right argument.

AXIS DOMAIN ERROR (INCORRECT TYPE)
K is not numeric.

AXIS DOMAIN ERROR (NOT AN INTEGER)
K is not near-integer.

AXIS DOMAIN ERROR (AXIS LESS THAN INDEX ORIGIN)
K is less than .bxIO.

AXIS DOMAIN ERROR (DUPLICATE AXIS NUMBER)
K contains duplicate values.

RANK ERROR (NOT VECTOR DOMAIN)
A is not a singleton and its rank is greater than 1.

LENGTH ERROR (LEFT LENGTH NOT EQUAL TO RIGHT RANK)
No axis is specified and B is not a scalar and its rank is
not equal to the length of A.

DOMAIN ERROR (INCORRECT TYPE)
A is not empty and not numeric.

DOMAIN ERROR (NOT AN INTEGER)
A is not a near-integer.

LIMIT ERROR (INTEGER TOO LARGE)
A or K is greater than the largest allowable integer.

2 Enclose

```
 Forms:  .luB    .lu[K]B
 Argument Domain:
        Left
                Type:   Any
                Shape:  Any
                Depth:  Any
 Result Domain:
                Type:   Same as constituent items
                Rank:   (.ro.roB)-.ro,K
                Shape:  (.roB)[(.io.ro.roB).ntK]
                Depth:  (0.ne.mtB)+.mtB
 Implicit Arguments:  None
```

The monadic .lu function builds enclosed arrays. For a

non-simple scalar argument, the result of the form .luB
is always an enclosed scalar item. If the argument is a
simple scalar, the depth remains the same: B.sa.luB when
B is a simple scalar. The result of the form .lu[K]B is
an array of enclosed scalars.

Each time you use monadic .lu, you increase the depth of the
argument by one (unless the argument is a simple scalar).

You can also enclose arrays that are already enclosed. The
only limit to the depth you create is the memory available
to the workspace.

Using the catenate function (,) with .lu allows you to create
arrays with multiple items. In such an expression, you must use
parentheses to prevent the scope of .lu from extending to the
rightmost end of the expression. For example:

```
      B_4
      C_.io5
      D_2 2 .ro 'ABCD'
      .bx_E_A , (.luB) , .luC      "Note use of parentheses
4 +---------+ +--+
  |1 2 3 4 5| |AB|
  +---------+ |CD|
              +--+
      .roE
3
      .mtE
2
```

The result of the form .lu[K]B is in an array of items
formed by enclosing subarrays along the axes given by K.

3 Errors

AXIS RANK ERROR (NOT VECTOR DOMAIN)
K is not a singleton and its rank is greater than 1.

AXIS LENGTH ERROR (ARGUMENT RANK AND AXIS INCOMPATIBLE)
The length of K is greater than the rank of B.

AXIS DOMAIN ERROR (SEMICOLON LIST NOT ALLOWED)
There is a semicolon inside of the brackets that surround K.

AXIS DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
K is not a simple homogeneous array.

AXIS DOMAIN ERROR (INCORRECT TYPE)
K is not empty and not numeric.

AXIS DOMAIN ERROR (NOT AN INTEGER)
K is not a near-integer.

AXIS DOMAIN ERROR (AXIS LESS THAN INDEX ORIGIN)
K is less than the current setting of .bxIO.

AXIS DOMAIN ERROR (ARGUMENT RANK AND AXIS INCOMPATIBLE)
An element of K is greater than the rank of B.

AXIS DOMAIN ERROR (DUPLICATE AXIS NUMBER)
K contains duplicate values.

LIMIT ERROR (INTEGER TOO LARGE)
K is greater than the largest allowable integer.

2 Enlist

Type: Monadic System Function

```
 Form:  simple-vector _ .ep array
 Argument Domain:
                Type:   Any
                Shape:  Any
                Depth:  Any
 Result Domain:
                Type:   Same as argument
                Rank:   1
                Shape:  Vector
                Depth:  1 (simple vector)
 Implicit Arguments:  None

 Enlist builds a simple vector with all of the simple scalars
 in its argument.  For example:  .ep (2 (2 2.ro5 6 7 8)) 'ABC'
 is 2 5 6 7 8 ABC.

 Enlist is in effect a pervasive ravel.

3 Errors
 No errors generated

2 Execute
 Form:  .xqB
 Argument Domain:
                Type:   Any
                Shape:  Any (Vector domain for characters)
                Depth:  0 or 1 (simple)
 Result Domain:
                Type:   Any
                Rank:   Any
                Shape:  Any
                Depth:  Any
 Implicit Arguments:  None

 The monadic .xq function executes the expression represented by
 a character-string or numeric argument as if that expression were
 entered in immediate mode or included in a user-defined
 function.

 The .xq symbol is formed by overstriking the .de and .so symbols.

3 Errors
 RANK ERROR (NOT VECTOR DOMAIN)
 B is not a singleton and its rank is greater than 1.

 EXECUTE ERROR
 An error occurred while the expression in B was being executed.

2 Expand
 Forms:  A\B    A\[K]B    A.cbB    A.cb[K]B
 Left Operand Domain:
                Array:  Simple, homogeneous
                Type:   Boolean
                Shape:  Vector domain
                Depth:  0 or 1 (simple)
 Right Argument Domain:
                Type:   Any
                Shape:  Any
                Depth:  Any
 Result Domain:
                Type:   Same as right argument
                Rank:   1.ce.ro.roB
                Shape:  --
                Depth:  1.ce.mtB
 Implicit Arguments:  Fill item (Type )HELP GLOSSARY FILL-ITEM for
                                 more information.)
```

Expansion is a monadic function derived from the backslash (\) operator. It builds an array by combining the items of an existing array with fill items.

Each item in the operand (A in the form) is a Boolean scalar that corresponds to the position of an item in the right argument (B). When A is 1, APL inserts the corresponding item along the relevant axis of B into the result array. When A is 0, APL inserts a fill item into that position in the result array.

There must be a 1 for each item along the relevant axis in the right argument, so that all the items in B appear in the result array. Any number of fill items may be included.

If you do not specify an axis in the square brackets, \ expands the items along the last axis, and .cb expands the items along the first axis.

See also .bxEXP (Type)HELP .bxEXP for more information.)

The .cb symbol is formed by overstriking the \ and - symbols.

3 Errors

SYNTAX ERROR (NO DYADIC FORM OF DERIVED FUNCTION)
Expansion is monadic.

AXIS RANK ERROR (NOT VECTOR DOMAIN)
K is not in the vector domain.

AXIS LENGTH ERROR (NOT SINGLETON)
K is not a singleton.

AXIS DOMAIN ERROR (SEMICOLON LIST NOT ALLOWED)
There is a semicolon inside of the brackets that surround K.

AXIS DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
K must be a simple homogeneous array.

AXIS DOMAIN ERROR (INCORRECT TYPE)
K is not numeric.

AXIS DOMAIN ERROR (NOT AN INTEGER)
K is not near-integer.

AXIS DOMAIN ERROR (AXIS LESS THAN INDEX ORIGIN)
K is less than .bxIO.

AXIS DOMAIN ERROR (RIGHT ARGUMENT HAS WRONG RANK)
K is greater than the rank of B.

LIMIT ERROR (INTEGER TOO LARGE)
A or K is greater than the largest allowable integer.

RANK ERROR (NOT VECTOR DOMAIN)
A is not a singleton and its rank is greater than 1.

LENGTH ERROR
B is not a singleton and its length along the Kth axis is not equal to the number of nonnegative integers in A.

DOMAIN ERROR
A is not Boolean.

DOMAIN ERROR (INCORRECT TYPE)
A is not empty and not numeric.

```
 DOMAIN ERROR (NOT AN INTEGER)
 A is not a near-integer.

2 First
 Form:  ^B
 Argument Domain:
               Type:   Any
               Shape:  Any
               Depth:  Any
 Result Domain:
               Type:   Same as selected item
               Rank:   Same as selected item
               Shape:  Same as selected item
               Depth:  0.ce.ng1+.mtB
 Implicit Arguments:  None

 The monadic ^ function (known as first) builds an array by
 disclosing the first item from an existing array.

 The ^ function only selects items from the top nesting
 level (to select deeper items, see information on the Pick
 function).

 If B is empty, the ^ returns the prototype of B.

3 Errors
 No errors generated

2 Format-Dyadic
 Form:  A.fmB
 Argument Domain:
        Left
               Type:   Near-integer
               Shape:  Vector domain
               Depth:  0 or 1 (simple)
        Right
               Type:   Numeric
               Shape:  Any
               Depth:  0 or 1 (simple)
 Result Domain:
               Type:   Character
               Rank:   1.ce.ro.roB
               Shape:  (.ng1.da.roB),+/1 0.cs(2,.ce0.5#.ro,A).roA
                       (provided no widths are 0)
               Depth:  1 (simple)
 Implicit Arguments:  .bxNG (controls display of negative numbers)


 The dyadic .fm function formats its right argument according
 to the width and precision information supplied by its left
 argument.

 The .fm symbol is formed by overstriking the .en and .so
 symbols.

3 Errors
 RANK ERROR (NOT VECTOR DOMAIN)
 A is not a singleton and its rank is greater than 1.

 LENGTH ERROR
 A's length is not 1, 2, or twice the number of columns in B.

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 A and B must be simple homogeneous arrays.

 DOMAIN ERROR (INCORRECT TYPE)
 Either A or B is not numeric and not empty.
 DOMAIN ERROR (NOT AN INTEGER)
 A is not a near-integer.

 DOMAIN ERROR (NEGATIVE NUMBER NOT ALLOWED)
 A width parameter is negative.

 DOMAIN ERROR (WIDTH TOO SMALL)
 The width parameters are too small to accommodate the precision
 parameters.

 LIMIT ERROR (INTEGER TOO LARGE)
 A is greater than the largest allowable integer.

 LIMIT ERROR (PARAMETER OUT OF RANGE)
 A width parameter is greater than 255, or a precision parameter
 is less than .ng127 or greater than 127.

2 Format-Monadic
 Form:  .fmB
 Argument Domain:
               Type:   Any
               Shape:  Any
               Depth:  Any
 Result Domain:
               Type:   Character
               Rank:   1.ce.ro.roB for numeric B,
                       .ro.roB for character B
               Shape:  (.ng1.da.roB) == (.ng1.da.roResult)
               Depth:  1 (simple)
 Implicit Arguments:  .bxPP (controls print precision)
                      .bxNG (controls display of negative numbers)
                      .bxDC (controls display of enclosed arrays)

 Possible Errors Generated:   None

 The monadic .fm function formats its argument array as a
 character array, making it look as it would appear when
 displayed by APL.

 The .fm symbol is formed by overstriking the .en and .so
 symbols.

3 Errors
 No errors generated

2 Grade-Down-Dyadic
 Form:  A.gdB
 Argument Domain:
       Left:
               Type:   Character
               Shape:  Any
               Depth:  0 or 1 (simple)
       Right:
               Type:   Character
               Shape:  Any
               Depth:  0 or 1 (simple)
 Result Domain:
               Type:   Nonnegative integer
               Rank:   1 (vector)
               Shape:  1^.roB
               Depth:  1 (simple vector)
 Implicit Arguments:  .bxIO (.gdB when .bxIO.mt1 is
                             identical to 1+.gdB when .bxIO.mt0)

 The dyadic .gd function returns a numeric vector whose
 items can be used to sort the items along the first
```

axis of the right argument in descending order. (The sort
is performed according to the collating sequence defined
in A.)

Grade down does not actually sort arrays; it creates a
permutation vector of the index numbers of the argument
array's items, and this vector can then be used to sort
the array.

Sorting an array requires two steps: First, the array is
used as the right argument to the grade function while the
left argument determines the order in which APL collates
the array items. Then, the result is used to index the
array.

If two or more items of the right argument
have the same value, the order of the items is
determined by their relative positions in the original
array.

The .gd symbol is formed by overstriking the .dl and .ab symbols.

3 Errors
AXIS DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
K must be a simple homogeneous array.

AXIS DOMAIN ERROR (INCORRECT OPERATION)
An attempt was made to specify an axis.

LENGTH ERROR (ARGUMENT STRING IS TOO LONG)
A has more than 65535 items or one of its axes has
more than 256 items.

DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
A and B must be simple homogeneous arrays.

DOMAIN ERROR (INCORRECT TYPE)
A or B is not of type character.

2 Grade-Down-Monadic
```
 Forms:  .gdB  .gd[K]B
 Argument Domain:
               Type:   Any
               Shape:  Matrix, vector, or scalar (not nonscalar singletons)
               Depth:  0 or 1 (simple)
 Result Range:
               Type:   Nonnegative integer
               Rank:   1 (vector)
               Shape:  (.rv.roB)[K]
               Depth:  1 (simple vector)
 Implicit Arguments:  .bxIO (.gdB when .bxIO.mt1 is
                             identical to 1+.gdB when .bxIO.mt0)
```

The monadic .gd function returns a numeric vector whose
items can be used to sort the items of the argument in
descending order. Thus, grade down does not actually sort
arrays; it creates a permutation vector of the index
numbers of the argument array's items, and this vector
can then be used to sort the array.

The .gd symbol is formed by overstriking the .dl and
| symbols.

3 Errors
AXIS RANK ERROR (NOT VECTOR DOMAIN)
K is not in the vector domain.

AXIS LENGTH ERROR (NOT SINGLETON)
K is not a singleton.

AXIS DOMAIN ERROR (SEMICOLON LIST NOT ALLOWED)
There is a semicolon inside of the brackets that surround K.

AXIS DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
K must be a simple homogeneous array.

AXIS DOMAIN ERROR (INCORRECT TYPE)
K is not numeric.

AXIS DOMAIN ERROR (NOT AN INTEGER)
K is not near-integer.

AXIS DOMAIN ERROR (AXIS LESS THAN INDEX ORIGIN)
K is less than .bxIO.

AXIS DOMAIN ERROR (ARGUMENT RANK AND AXIS INCOMPATIBLE)
K is greater than the rank of B.

LIMIT ERROR (INTEGER TOO LARGE)
An item of K is bigger than the largest allowable integer.

RANK ERROR (NOT A SCALAR, VECTOR, OR MATRIX)
The rank of B is greater than 2.

DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
B must be simple homogeneous arrays.

2 Grade-Up-Dyadic

```
 Form:  A.guB
 Argument Domain:
        Left:
                Type:   Character
                Shape:  Any
                Depth:  0 or 1 (simple)
        Right:
                Type:   Character
                Shape:  Any
                Depth:  0 or 1 (simple)
 Result Domain:
                Type:   Nonnegative integer
                Rank:   1 (vector)
                Shape:  1^.roB
                Depth:  1 (simple vector)
 Implicit Arguments:  .bxIO (.guB when .bxIO.mt1 is
                              identical to 1+.guB when .bxIO.mt0)
```

The dyadic .gu function returns a numeric vector whose items can be used to sort the items along the first axis of the right argument in ascending order. (The sort is performed according to the collating sequence defined in A.) Grade up does not actually sort arrays; it creates a permutation vector of the index numbers of the argument array's items, and this vector can then be used to sort the array.

Sorting an array requires two steps: First, the array is used as the right argument to the grade function while the left argument determines the order in which APL collates the array items. Then, the result is used to index the array.

If two or more items of the right argument have the same value, the order of the items is determined by their relative positions in the original array.

The .gu symbol is formed by overstriking the .ld and .ab symbols.

3 Errors

AXIS DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
K must be a simple homogeneous array.

AXIS DOMAIN ERROR (INCORRECT OPERATION)
An attempt was made to specify an axis.

LENGTH ERROR (ARGUMENT STRING IS TOO LONG)
A has more than 65535 items or one of its axes
has more than 256 items.

DOMAIN ERROR (INCORRECT TYPE)
Either A or B is not of type character

DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
A and B must be simple homogeneous arrays.

2 Grade-Up-Monadic

```
Forms:  .guB   .gu[K]B
Argument Domain:
               Type:   Any
               Shape:  Matrix, vector, or scalar (not nonscalar singletons)
               Depth:  0 or 1 (simple)
Result Domain:
               Type:   Nonnegative integer
               Rank:   1 (vector)
               Shape:  (.rv.roB)[K]
               Depth:  1 (simple vector)
Implicit Arguments:  .bxIO (.guB when .bxIO.mt1 is
                            identical to 1+.guB when .bxIO.mt0)
```

The monadic .gu function returns a numeric vector whose
items can be used to sort the items of the argument in
ascending order. Thus, grade up does not actually sort
arrays; it creates a permutation vector of the index
numbers of the argument array's items, and this vector
can then be used to sort the array.

The .gu symbol is formed by overstriking the .ld
and | symbols.

3 Errors

AXIS RANK ERROR (NOT VECTOR DOMAIN)
K is not in the vector domain.

AXIS LENGTH ERROR (NOT SINGLETON)
K is not a singleton.

AXIS DOMAIN ERROR (SEMICOLON LIST NOT ALLOWED)
There is a semicolon inside of the brackets that surround K.

AXIS DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
K must be a simple homogeneous array.

AXIS DOMAIN ERROR (INCORRECT TYPE)
K is not numeric.

AXIS DOMAIN ERROR (NOT AN INTEGER)
K is not near-integer.

AXIS DOMAIN ERROR (AXIS LESS THAN INDEX ORIGIN)
K is less than .bxIO.

AXIS DOMAIN ERROR (ARGUMENT RANK AND AXIS INCOMPATIBLE)

K is greater than the rank of B.

LIMIT ERROR (INTEGER TOO LARGE)
An item of K is bigger than the largest allowable integer.

RANK ERROR (NOT A SCALAR, VECTOR, OR MATRIX)
the rank of B is greater than 2.

2 Index-Generator

```
 Form:  .ioB
 Argument Domain:
               Type:   Nonnegative near-integer
               Shape:  Singleton
               Depth:  0 or 1 (simple)
 Result Domain:
               Type:   Nonnegative integer
               Rank:   1 (vector)
               Shape:  ,B
               Depth:  1 (simple vector)
 Implicit Arguments:  .bxIO (.ioB when .bxIO.mt1 is
                             identical to 1+.ioB when .bxIO.mt0)
```

For an argument B, the monadic .io function generates a
vector of B consecutive, ascending integers starting with
the value of the index origin.

If the index origin is 1, the integers have values 1 through
B; if the index origin is 0, the integers have values 0
through B - 1:

3 Errors

RANK ERROR (NOT VECTOR DOMAIN)
B is not a singleton and its rank is greater than 1.

LENGTH ERROR (NOT SINGLETON)
B does not have exactly one item.

DOMAIN ERROR (INCORRECT TYPE)
B is not empty and not numeric.

DOMAIN ERROR (NOT AN INTEGER)
B is not a near-integer.

LIMIT ERROR (INTEGER TOO LARGE)
B is greater than the largest allowable integer.

DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
B must be a simple homogeneous array.

DOMAIN ERROR (NEGATIVE NUMBER NOT ALLOWED)
B is less than 0.

2 Index-Of

```
 Form:  A.ioB
 Argument Domain:
        Left
               Type:   Any
               Shape:  Vector domain
               Depth:  Any
        Right
               Type:   Any
               Shape:  Any
               Depth:  Any
 Result Domain:
               Type:   Nonnegative integer
               Rank:   Same as right argument
               Shape:  Same as right argument
               Depth:  0 or 1 (simple)
 Implicit Arguments:  .bxCT (determines comparison precision)
                      .bxIO (A.ioB when .bxIO.mt1 is
                             identical to 1+A.ioB when .bxIO.mt0)

 The dyadic .io function returns the position of the first
 occurrence in the left argument of the corresponding item
 in the right argument.

3 Errors
 RANK ERROR (NOT VECTOR DOMAIN)
 A is not a singleton and its rank is greater than 1.

2 Inner-Product
 Form:  Af.gB
 Operand Domain:
        Left
                Type:  Dyadic function
        Right
                Type:  Dyadic function
 Argument Domain:
        Left
                Type:   Any
                Shape:  Any, inner axes of A and B must conform
                Depth:  Any
        Right
                Type:   Any
                Shape:  Any, inner axes of A and B must conform
                Depth:  Any
 Result Domain:
                Type:   Any
                Rank:   0.ce.ng2+(.ro.roA)+.ro.ro.B
                Shape:  (.ng1.da.roA),1.da.roB
                Depth:  --
 Implicit Arguments:  None

 The derived function inner product operator produces the common
 algebraic matrix product of two arrays.

 The name, inner product, comes from the application of the function
 g along the inner axes of the two arguments. f can be a primitive
 dyadic function, a dyadic system function, a dyadic user-defined
 function, or a dyadic derived function from an arbitrary operator
 sequence. (The inner axes are the last axis of the left argument and
 the first axis of the right argument.)

3 Errors
 SYNTAX ERROR (NO MONADIC FORM OF DERIVED FUNCTION)
 Inner product is dyadic.

 OPERATOR DOMAIN ERROR (NOT A DYADIC SCALAR FUNCTION)
 Either f or g is not a dyadic scalar function.

 LENGTH ERROR (LENGTH OF INNER AXES DO NOT MATCH)
 The last axis of A, or the first axis of B is not equal to
 one, and the length of the last axis of A does not equal the
 length of the first axis of B.

 DOMAIN ERROR (INCORRECT TYPE)
 A or B is not empty and its type is not in the domain of g.

 DOMAIN ERROR (FUNCTION HAS NO IDENTITY ELEMENT)
 Only the inner axes of A and B ((0=.ng1^.roA)&0=1^.roB)
 are empty and the function g has no identity item.

 DOMAIN ERROR
 A or B is not empty and its value is not in the domain of g.
```

VALUE ERROR
f and g must be functions that return values.

2 Intersection

```
 Form:  A .du B
 Argument Domain:
        Left
                Type:   Any
                Shape:  Any
                Depth:  Any
        Right
                Type:   Any
                Shape:  Any
                Depth:  Any
 Result Domain:
                Type:   See explanation below
                Rank:   1 (vector)
                Shape:  .ro.uu((,A).epB)/,A
                Depth:  --
 Implicit Arguments:  .bxCT (determines comparison precision)
```

Possible Errors Generated: None

The dyadic .du function returns the common items found in both arguments. The result is the intersection of the arguments with duplicate items removed. Note that the order of the items in the result is not predictable.

The type of the result depends on the types of the arguments, as shown below:

Arguments	Resulting Type
Neither empty	Same as left argument
One empty	Same as nonempty argument
Both empty	Same as left argument

3 Errors

No errors generated

2 Match

```
 Form:  A.mtB
 Argument Domain:
        Left
                Type:   Any
                Shape:  Any
                Depth:  Any
        Right
                Type:   Any
                Shape:  Any
                Depth:  Any
 Result Domain:
                Type:   Boolean
                Rank:   0 (scalar)
                Shape:  Scalar
                Depth:  0 (simple scalar)
 Implicit Arguments:  .bxCT (determines comparison precision)
```

Possible Errors Generated: None

The dyadic .mt function determines whether the two arguments are identical in rank, shape, and value. The result is a Boolean scalar: true, if the arguments are identical, and false if they are not.

The .mt symbol is formed by overstriking the = and .us symbols.

```
3 Errors
 No errors generated

2 Matrix-Divide
 Form:  A.dqB
 Argument Domain:
        Left
                Type:   Numeric
                Shape:  Matrix, vector, or scalar (not
                        nonscalar singletons)
                Depth:  0 or 1 (simple)
        Right
                Type:   Numeric
                Shape:  Matrix, vector, or scalar (not
                        nonscalar singletons)
                Depth:  0 or 1 (simple)
 Result Domain:
                Type:   Numeric
                Rank:   Same as left argument
                Shape:  (1.da.roB),1.da.roA
                Depth:  0 or 1 (simple)
 Implicit Arguments:  .bxCT (used in the test for singularity)

 For arguments A and B, the dyadic .dq function determines
 the generalized solution R to the linear system
 A=B+.#R.  If B  has more rows than columns, then dyadic
 .dq returns the least-squares solution to the linear system.

 The .dq symbol is formed by overstriking the .bx and %
 symbols.

3 Errors
 RANK ERROR (NOT A SCALAR, VECTOR, OR MATRIX)
 A or B is not a scalar, vector, or matrix.

 LENGTH ERROR (NUMBER OF ROWS MUST MATCH)
 The number of rows in A and B are not equal.

 LENGTH ERROR (FEWER ROWS THAN COLUMNS)
 The number of rows in B is less than the number of
 columns in B.

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 A and B must be simple homogeneous arrays.

 DOMAIN ERROR (INCORRECT TYPE)
 Either A or B is not numeric and not empty.

 DOMAIN ERROR (SINGULAR MATRIX)
 The columns of B are not linearly independent.

 DOMAIN ERROR (DIVISION BY ZERO)
 Division by 0 was attempted.

 LIMIT ERROR (FLOATING OVERFLOW)
 Arithmetic overflow occurred.

2 Matrix-Inverse
 Form:  .dqB
 Argument Domain:
                Type:   Numeric
                Shape:  Matrix, vector, or scalar (not nonscalar singletons)
                Depth:  0 or 1 (simple)
 Result Domain:
                Type:   Numeric
                Rank:   Same as argument
                Shape:  .rv.roB
```

```
               Depth:  0 or 1 (simple)
 Implicit Arguments:  .bxCT (used in the test for singularity)

 The monadic .dq function inverts a matrix to facilitate
 matrix division and a variety of other matrix operations.

 The .dq symbol is formed by overstriking the .bx and
 the % symbols.

3 Errors
 RANK ERROR (NOT A SCALAR, VECTOR, OR MATRIX)
 The rank of B is greater than 2.

 LENGTH ERROR (THERE ARE FEWER ROWS THAN COLUMNS)
 B has fewer rows than columns.

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 B must be a simple homogeneous array.

 DOMAIN ERROR (INCORRECT TYPE)
 B is not empty and not numeric.

 DOMAIN ERROR (SINGULAR MATRIX)
 B's columns are not linearly independent.

 DOMAIN ERROR (DIVISION BY ZERO)
 Division by 0 was attempted.

 LIMIT ERROR (FLOATING OVERFLOW)
 Arithmetic overflow occurred.

2 Membership
 Form:  A.epB
 Argument Domain:
        Left
               Type:   Any
               Shape:  Any
               Depth:  Any
        Right
               Type:   Any
               Shape:  Any
               Depth:  Any
 Result Domain:
               Type:   Boolean
               Rank:   Same as left argument
               Shape:  Same as left argument
               Depth:  0 or 1 (simple)
 Implicit Arguments:  .bxCT (determines comparison precision)

 Possible Errors Generated:  None

 The dyadic .ep function determines whether particular
 items of the left argument array occur as items of the
 right argument array.

3 Errors
 No errors generated

2 Outer-Product
 Form:  A.so.fB
 Operand Domain:
        Left
               Type:  Always jot (.so)
        Right
               Type:  Dyadic function
 Argument Domain:
        Left
```

```
                Type:   Any
                Shape:  Any
                Depth:  Any
        Right
                Type:   Any
                Shape:  Any
                Depth:  Any
 Result Domain:
                Type:   Depends on right operand result domain
                Rank:   (.ro.roA)+.ro.roB
                Shape:  (.roA),.roB
                Depth:  --
 Implicit Arguments:  None
```

Outer product is a derived function that specifies an operation to be performed between every item of one array and every item of another array.

3 Errors

OPERATOR DOMAIN ERROR (NOT A DYADIC SCALAR FUNCTION)
The function f is not a dyadic scalar function.

DOMAIN ERROR (INCORRECT TYPE)
A or B is not empty and its type is not in the domain of f.

DOMAIN ERROR
A or B is not empty and its value is not in the domain of f.

SYNTAX ERROR (NO MONADIC FORM OF DERIVED FUNCTION)
Outer product is dyadic.

2 Pick

```
 Form:  A.ruB
 Argument Domain:
        Left
                Type:   Nonnegative near-integer
                Shape:  Vector domain (same length as .mtB)
                Depth:  Less than or equal to 2
        Right
                Type:   Any
                Shape:  Any
                Depth:  Any
 Result Domain:
                Type:   Any
                Rank:   Any
                Shape:  Any
                Depth:  (.mtB)-.roA (provided A is along
                                    the deepest path)
 Implicit Arguments:  .bxIO (A.ruB when .bxIO.mt1 is
                             identical to (1+A).ruB when .bxIO.mt0)
```

The dyadic .ru function selects and discloses an item from an existing array. The items in A specify the coordinates of items in B. For example:

```
     V_21 22 23 24 25 26
     2.ruV      "Select second item in V
22
     V[2]       "Note similarity to indexing
22
```

You can select an item from any depth in an enclosed array. The length of A determines the depth of the selected item: When A has one item, the selection is from the top level of B; when A has two items, the selection is from the second level; and so on.

When an item in B is a scalar, the corresponding item in
A must be empty.

3 Errors
DEPTH ERROR (LEFT ARGUMENT DEPTH GREATER THAN 2)
The items in A must be simple (vectors or singletons).

RANK ERROR (NOT VECTOR DOMAIN)
A is not a singleton and its rank is greater than 1.

RANK ERROR (LEFT ITEM NOT VECTOR DOMAIN)
An item of A is not a singleton and its rank is greater than 1.

LENGTH ERROR (LEFT ARGUMENT LENGTH GREATER THAN RIGHT ARGUMENT DEPTH)
The length of A is greater than the depth of B.

LENGTH ERROR (LEFT ITEM LENGTH NOT EQUAL TO SELECTED ITEM RANK)
The length of an item of A does not match the rank of the
selected item at the corresponding depth of B.

DOMAIN ERROR (INCORRECT TYPE)
A is not empty and not numeric.

DOMAIN ERROR (NOT AN INTEGER)
A is not a near-integer.

DOMAIN ERROR (INDEX OUT OF RANGE)
An element of A exceeds the length of the corresponding axis
of an item of B.

DOMAIN ERROR (INDEX LESS THAN INDEX ORIGIN)
An element of A is less than the current setting of .bxIO.

LIMIT ERROR (INTEGER TOO LARGE)
A is greater than the largest allowable integer.

2 Ravel
```
 Forms:  ,B   ,[K]B
 Argument Domain:
                Type:   Any
                Shape:  Any
                Depth:  Any
 Result Domain:
                Type:   Same as argument
                Rank:   1 (vector)
                Shape:  #/.roB
                Depth:  Same as argument
 Implicit Arguments:  None
```

The monadic , function returns a vector made up of the
items of the argument array, stored in row-major order
(by increasing index position).

3 Errors
AXIS RANK ERROR (NOT VECTOR DOMAIN)
K is not a singleton and its rank is greater than 1.

AXIS LENGTH ERROR (NOT SINGLETON)
K is noninteger and is not a singleton.

AXIS LENGTH ERROR (ARGUMENT RANK AND AXIS INCOMPATIBLE)
The length of K is greater than the rank of B.

AXIS DOMAIN ERROR (SEMICOLON LIST NOT ALLOWED)
There is a semicolon inside the brackets that surround K.

AXIS DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)

```
 K must be a simple homogeneous array.

 AXIS DOMAIN ERROR (INCORRECT TYPE)
 K is not empty and not numeric.

 AXIS DOMAIN ERROR (AXIS LESS THAN INDEX ORIGIN)
 K is less than or equal to .bxIO-1.

 AXIS DOMAIN ERROR (ARGUMENT RANK AND AXIS INCOMPATIBLE)
 K is greater than the rank of B.

 AXIS DOMAIN ERROR (AXES NOT IN CONTIGUOUS ASCENDING ORDER)

 LIMIT ERROR (INTEGER TOO LARGE)
 An item of K is bigger than the largest allowable integer.

2 Reduce
 Forms:  f/A    f/[K]A   f.csA   f.cs[K]A
 Left Operand Domain:
               Type:  Dyadic function
 Right Argument Domain:
               Type:   Any
               Shape:  Any
               Depth:  Any
 Result Domain:
               Type:   Any
               Rank:   0.ce.ng1+.ro.roA
               Shape:  (.roB)[(.io.ro.roB).ntK]
               Depth:  Any
 Implicit Arguments:  None

 Reduction is a monadic function derived from the slash (/)
 operator. Use any dyadic function as the operand (f in the
 form) to slash. The result operates as if f were applied
 between successive items along a specified axis of an array
 B.

 If you do not specify an axis in the square brackets, /
 reduces the items along the last axis, and .cs reduces
 the items along the first axis. Type )HELP GLOSSARY
 IDENTITY-ITEMS  for more information.

 The .cs symbol is formed by overstriking the / and -
 symbols.

3 Errors
 SYNTAX ERROR (NO DYADIC FORM OF DERIVED FUNCTION)
 Reduction is monadic.

 SYNTAX ERROR (NO DYADIC FORM OF FUNCTION)
 The function f is not a dyadic function.

 AXIS RANK ERROR (NOT VECTOR DOMAIN)
 K is not a singleton and its rank is greater than 1.

 AXIS LENGTH ERROR (NOT SINGLETON)
 K is not a singleton.

 AXIS DOMAIN ERROR (SEMICOLON LIST NOT ALLOWED)
 There is a semicolon inside of the brackets that surround K.

 AXIS DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 K must be a simple homogeneous array.

 AXIS DOMAIN ERROR (INCORRECT TYPE)
 K is not numeric.
 AXIS DOMAIN ERROR (NOT AN INTEGER)
 K is not near-integer.

 AXIS DOMAIN ERROR (AXIS LESS THAN INDEX ORIGIN)
 K is less than .bxIO.

 AXIS DOMAIN ERROR (ARGUMENT RANK AND AXIS INCOMPATIBLE)
 K is greater than the rank of B.

 LIMIT ERROR (INTEGER TOO LARGE)
 K is greater than the largest allowable integer.

 DOMAIN ERROR (NOT A DYADIC SCALAR FUNCTION)
 The function f is not a dyadic function.

 DOMAIN ERROR (INCORRECT TYPE)
 A is not empty and its type is not in the domain of f.

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 f requires a simple or homogeneous array as its argument.

 DOMAIN ERROR (FUNCTION HAS NO IDENTITY ELEMENT)
 The K axis of A is length 0 and all other axes are nonzero
 length and the function f has no identity item.

 DOMAIN ERROR
 B is not empty and its value is not in the domain of f.

 VALUE ERROR
 f must be a function that returns a value.

2 Represent
 Form:  A.enB
 Argument Domain:
        Left
                Type:   Numeric
                Shape:  Any
                Depth:  0 or 1 (simple)
        Right
                Type:   Numeric
                Shape:  Any
                Depth:  0 or 1 (simple)
 Result Domain:
                Type:   Numeric
                Rank:   (.ro.roA)+.ro.roB
                Shape:  (.roA),.roB
                Depth:  0 or 1 (simple)
 Implicit Arguments:  None

 The dyadic .en function represents an array in any number
 system: the left argument specifies the number system; the
 right argument specifies the array to be represented.

3 Errors
 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 A and B must be simple homogeneous arrays.

 DOMAIN ERROR (INCORRECT TYPE)
 Either A or B is not numeric and not empty.

 LIMIT ERROR (FLOATING OVERFLOW)
 Arithmetic overflow occurred.

2 Reshape
 Form:  A.roB
 Argument Domain:
        Left
```

```
                Type:   Nonnegative near-integer
                Shape:  Vector domain
                Depth:  0 or 1 (simple)
        Right
                Type:   Any
                Shape:  Any
                Depth:  Any
 Result Domain:
                Type:   Same as right argument
                Rank:   .roA
                Shape:  ,A
                Depth:  --
 Implicit Arguments:  None

 The dyadic .ro function creates an array of data items
 from the right argument taken in row-major order and
 arranged in the shape specified by the left argument.

3 Errors
 RANK ERROR (NOT VECTOR DOMAIN)
 A is not a singleton and its rank is greater than 1.

 LENGTH ERROR
 B is empty and A does not contain at least one 0 value.

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 B must be a simple homogeneous array.

 DOMAIN ERROR (INCORRECT TYPE)
 A is not empty and not numeric.

 DOMAIN ERROR (NOT AN INTEGER)
 A is not a near-integer.

 DOMAIN ERROR (NEGATIVE NUMBER NOT ALLOWED)
 A is less than 0.

 LIMIT ERROR (INTEGER TOO LARGE)
 A is greater than the largest allowable integer.

2 Reverse
 Forms:  .rvB   .rv[K]B   .crB   .cr[K]B
 Argument Domain:
                Type:   Any
                Shape:  Any
                Depth:  Any
 Result Domain:
                Type:   Same as argument
                Rank:   Same as argument
                Shape:  Same as argument
                Depth:  Same as argument
 Implicit Arguments:  None

 The monadic reverse function returns the items of the
 argument array in reverse order along the relevant axis.
 If you do not specify an axis in the square brackets, .rv
 reverses the items along the last axis, and .cr reverses
 the items along the first axis.

 The .rv symbol is formed by overstriking the .lo and |
 symbols.  The .cr symbol is formed by overstriking the
 .lo and - symbols.

3 Errors
 AXIS RANK ERROR (NOT VECTOR DOMAIN)
 K is not a singleton and its rank is greater than 1.
```

AXIS LENGTH ERROR (NOT SINGLETON)
K is not a singleton.

AXIS DOMAIN ERROR (SEMICOLON LIST NOT ALLOWED)
There is a semicolon inside of the brackets that surround K.

AXIS DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
K must be a simple homogeneous array.

AXIS DOMAIN ERROR (INCORRECT TYPE)
K is not numeric.

AXIS DOMAIN ERROR (NOT AN INTEGER)
K is not near-integer.

AXIS DOMAIN ERROR (AXIS LESS THAN INDEX ORIGIN)
K is less than .bxIO.

AXIS DOMAIN ERROR (ARGUMENT RANK AND AXIS INCOMPATIBLE)
K is greater than the rank of B.

LIMIT ERROR (INTEGER TOO LARGE)
An item of K is bigger than the largest allowable integer.

2 Rotate

```
 Forms:  A.rvB   A.rv[K]B   A.crB   A.cr[K]B
 Argument Domain:
        Left
                Type:   Near-integer
                Shape:  Conforms to right argument
                Depth:  0 or 1 (simple)
        Right
                Type:   Any
                Shape:  Any
                Depth:  Any
 Result Domain:
                Type:   Same as right argument
                Rank:   Same as right argument
                Shape:  Same as right argument
                Depth:  Same as right argument
 Implicit Arguments:  None
```

The dyadic .rv or .cr function rotates items along
the relevant axis of the right argument in a way specified
by the left argument. If the left argument is positive,
the shift is to the left; if it is negative, the shift is
to the right.

If you do not specify an axis in the square brackets, .rv
rotates the items along the last axis, and .cr rotates
the items along the first axis.

The .rv symbol is formed by overstriking the .lo and |
symbols. The .cr symbol is formed by overstriking the .lo
and - symbols.

3 Errors

AXIS RANK ERROR (NOT VECTOR DOMAIN)
K is not in the vector domain.

AXIS LENGTH ERROR (NOT SINGLETON)
K is not a singleton.

AXIS DOMAIN ERROR (SEMICOLON LIST NOT ALLOWED)
There is a semicolon inside of the brackets that surround K.

AXIS DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)

K must be a simple homogeneous array.

AXIS DOMAIN ERROR (INCORRECT TYPE)
K is not numeric.

AXIS DOMAIN ERROR (NOT AN INTEGER)
K is not near-integer.

AXIS DOMAIN ERROR (AXIS LESS THAN INDEX ORIGIN)
K is less than .bxIO.

AXIS DOMAIN ERROR (RIGHT ARGUMENT HAS WRONG RANK)
K is greater than the rank of B.

RANK ERROR (RANKS DIFFER BY MORE THAN ONE)
A is not a singleton and its rank is not equal to one
less than the rank of B.

LENGTH ERROR (SHAPES OFF AXIS DO NOT MATCH)
A is not a singleton and its shape does not equal the
shape of B, excluding axis K.

DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
A must be a simple homogeneous array.

DOMAIN ERROR (INCORRECT TYPE)
B is not a singleton and A is not empty and not numeric.

DOMAIN ERROR (NOT AN INTEGER)
B is not a singleton and A is not a near-integer.

LIMIT ERROR (INTEGER TOO LARGE)
A is greater than the largest allowable integer.

2 Scan

```
 Forms:  f\A   f\[K]A   f.cbA   f.cb[K]A
 Left Operand Domain:
                Type:  Dyadic function
 Argument Domain:
                Type:   Any
                Rank:   Any
                Shape:  Any
                Depth:  Any
 Result Domain:
                Type:   Depends on f
                Rank:   Same as B
                Shape:  Same as B
                Depth:  --
 Implicit Arguments:  None
```

Scan is a monadic function derived from the backslash (\)
operator. Use any dyadic function as the operand (f in the
form) to slash. f can be a primitive dyadic function, a dyadic
system function, a dyadic user-defined function, or a dyadic
derived function from an arbitrary operator sequence.

The result operates as if f were applied between successive items
along a specified axis of an array (B). Thus, a scan of an array
works the same as a reduction does, except that the scan returns
the results as the function is applied to each successive group
of items.

If you do not specify an axis in the square brackets, \
scans the items along the last axis, and .cb scans
the items along the first axis.

The .cb symbol is formed by overstriking the \ and -

symbols.

3 Errors
SYNTAX ERROR (NO DYADIC FORM OF DERIVED FUNCTION)
Scan is monadic.

SYNTAX ERROR (NO DYADIC FORM OF FUNCTION)
The function f is not a dyadic function.

AXIS RANK ERROR (NOT VECTOR DOMAIN)
K is not a singleton and its rank is greater than 1.

AXIS LENGTH ERROR (NOT SINGLETON)
K is not a singleton.

AXIS DOMAIN ERROR (SEMICOLON LIST NOT ALLOWED)
There is a semicolon inside of the brackets that surround K.

AXIS DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
K must be a simple homogeneous array.

AXIS DOMAIN ERROR (INCORRECT TYPE)
K is not numeric.

AXIS DOMAIN ERROR (NOT AN INTEGER)
K is not near-integer.

AXIS DOMAIN ERROR (AXIS LESS THAN INDEX ORIGIN)
K is less than .bxIO.

AXIS DOMAIN ERROR (ARGUMENT RANK AND AXIS INCOMPATIBLE)
K is greater than the rank of B.

LIMIT ERROR (INTEGER TOO LARGE)
K is greater than the largest allowable integer.

OPERATOR DOMAIN ERROR (NOT A DYADIC SCALAR FUNCTION)
The function f is not a dyadic scalar function.

DOMAIN ERROR (INCORRECT TYPE)
A is not empty and its type is not in the domain of f.

DOMAIN ERROR
B is not empty and its value is not in the domain of f.

VALUE ERROR
f must be a function that returns a value.

2 Shape
```
Form:  .roB
Argument Domain:
               Type:   Any
               Shape:  Any
               Depth:  Any
Result Domain:
               Type:   Nonnegative integer
               Rank:   1 (vector)
               Shape:  the rank of B
               Depth:  1 (simple vector)
Implicit Arguments:  None
```

Possible Errors Generated: None

The monadic .ro function returns a vector of nonnegative integers that represent the lengths of each of the axes of the argument array.

3 Errors
No errors generated

2 Subset

```
 Form:  A.ssB
 Argument Domain:
        Left
                Type:   Any
                Shape:  Any
                Depth:  Any
        Right
                Type:   Any
                Shape:  Any
                Depth:  Any
 Result Domain:
                Type:   Boolean
                Rank:   0 (scalar)
                Shape:  .io0
                Depth:  0 (simple scalar)
 Implicit Arguments:  .bxCT (determines comparison precision)

 Possible Errors Generated:  None
```

The dyadic .ss function determines whether the right
argument contains all of the items in the left
argument. The result is a Boolean scalar: true, if the
left argument is a subset of the right argument, and false
if it is not. Duplicate items in either argument do not
affect the result.

The .ss symbol is formed by overstriking the .lu and .us symbols.

3 Errors
No errors generated

2 Take

```
 Forms:  A^B   A^[K]B
 Argument Domain:
        Left
                Type:   Near-integer
                Shape:  Vector domain
                Depth:  0 or 1 (simple)
        Right
                Type:   Any
                Shape:  Any
                Depth:  Any
 Result Domain:
                Type:   Same as right argument
                Rank:   Same as right argument
                Shape:  |A (if no explicit axis)
                Depth:  --
 Implicit Arguments:  Fill item (Type )HELP GLOSSARY FILL-ITEM for
                                    more information.)
```

The dyadic ^ function builds an array by taking a specified
number of items from an existing array. Each item in A
corresponds to an axis in B. The value of each item in A
specifies how many items to take from the axis. Thus, for
A^B, item A[K] is used to take values along axis K of B.

If an item of the left argument is a positive integer n, APL
takes the first n data items from the appropriate axis of the
right-argument; if an item of the left argument is negative,
APL takes the last n data items from the appropriate axis of
the right-argument array.

If the value of an item in A is greater than the length of the

corresponding axis of B, APL pads the result array with fill
items. This operation is known as overtake.

When you use ^ with an axis argument, K is a vector of axis numbers
whose lengths are determined by corresponding items of the left
argument, A.

3 Errors
AXIS LENGTH ERROR (LEFT ARGUMENT HAS WRONG LENGTH)
The length of A does not equal the length of K.

AXIS DOMAIN ERROR (SEMICOLON LIST NOT ALLOWED)
There is a semicolon inside of the brackets that surround K.

AXIS DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
K must be a simple homogeneous array.

AXIS RANK ERROR (NOT VECTOR DOMAIN)
K is not in the vector domain.

AXIS DOMAIN ERROR (ARGUMENT RANK AND AXIS INCOMPATIBLE)
The length of K is greater than the rank of the right argument.

AXIS DOMAIN ERROR (DUPLICATE AXIS NUMBER)
K contains duplicate values.

AXIS DOMAIN ERROR (INCORRECT TYPE)
K is not numeric.

AXIS DOMAIN ERROR (NOT AN INTEGER)
K is not near-integer.

AXIS DOMAIN ERROR (AXIS LESS THAN INDEX ORIGIN)
K is less than .bxIO.

RANK ERROR (NOT VECTOR DOMAIN)
A is not in the vector domain.

LENGTH ERROR (LEFT LENGTH NOT EQUAL TO RIGHT RANK)
No axis is specified and B is not a scalar and its rank
is not equal to the length of A.

DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
A must be a simple homogeneous array.

DOMAIN ERROR (INCORRECT TYPE)
A is not empty and not numeric.

DOMAIN ERROR (NOT AN INTEGER)
A is not a near-integer.

LIMIT ERROR (INTEGER TOO LARGE)
A or K is greater than the largest allowable integer.

LIMIT ERROR (VOLUME TOO LARGE)
An attempt was made to take an amount of data that
exceeds the limits of APL.

2 Transpose-Dyadic
Form: A.trB
Argument Domain:
```
       Left
               Type:   Nonnegative near-integer
               Shape:  Vector domain
               Depth:  0 or 1 (simple)
       Right
               Type:   Any
               Shape:  Any
               Depth:  Any
 Result Domain:
               Type:   Same as right argument
               Rank:   RANK_.ce/A+.nt.bxIO
               Shape:  .fl/((.ioRANK).so.=A)x(RANK,.roA).ro.roB
               Depth:  Same as right argument
 Implicit Arguments:  .bxIO (A.trB when .bxIO.mt1 is
                             identical to (1+A).trB when .bxIO.mt0)

 The dyadic .tr function permutes the axes of the right
 argument in a way specified by the left argument.

 The .tr symbol is formed by overstriking the .lo and \
 symbols.

3 Errors
 RANK ERROR (NOT VECTOR DOMAIN)
 B is not a scalar and A is not in a vector domain.

 LENGTH ERROR (LEFT LENGTH NOT EQUAL TO RIGHT RANK)
 The length of A is not equal to the rank of B.

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 A must be a simple homogeneous array.

 DOMAIN ERROR (INCORRECT TYPE)
 A is not empty and not numeric.

 DOMAIN ERROR (NOT AN INTEGER)
 A is not a near-integer.

 DOMAIN ERROR (LEFT ARGUMENT NOT DENSE FROM QUAD IO)
 A is not a dense sequence beginning at .bxIO.

 LIMIT ERROR (INTEGER TOO LARGE)
 A is greater than the largest allowable integer.

2 Transpose-Monadic
 Form:  .trB
 Argument Domain:
               Type:   Any
               Shape:  Any
               Depth:  Any
 Result Domain:
               Type:   Same as argument
               Rank:   Same as argument
               Shape:  .rv.roB
               Depth:  Same as argument
 Implicit Arguments:  None

 Possible Errors Generated:  None

 The monadic .tr function transposes the axes of an
 array; thus, .trB is B with the order of the axes
 reversed.

 The .tr symbol is formed by overstriking the .lo and
 \ symbols.

3 Errors
 No errors generated

2 Union
 Form:  A.uuB
 Argument Domain:
        Left
```

```
               Type:   Any
               Shape:  Any
               Depth:  Any
       Right
               Type:   Any
               Shape:  Any
               Depth:  Any
Result Domain:
               Type:   Any
               Rank:   1 (vector)
               Shape:  .ro.uu(,A),,B
               Depth:  Any
Implicit Arguments:  .bxCT (determines comparison precision)
```

The dyadic .uu function joins the two arguments and
removes all duplicate items. The result is a vector
that includes all the items from both arguments.

The type of the result depends on the types of the
arguments, as shown below:

Arguments	Resulting Type
Neither empty	Same as left argument
One empty	Same as nonempty argument
Both empty	Same as left argument

3 Errors
No errors generated

2 Unique

```
Form:  .uuB
Argument Domain:
               Type:   Any
               Shape:  Any
               Depth:  Any
Result Domain:
               Type:   Same as argument
               Rank:   1 (vector)
               Shape:  Equal to number of unique items
               Depth:  Same as argument
Implicit Arguments:  .bxCT (determines comparison precision)
```

The monadic .uu function removes duplicate items form an array.
The result is a vector of the unique items in the argument.

3 Errors
No errors generated

2 Without

```
Form:  A.ntB
Argument Domain:
       Left
               Type:   Any
               Shape:  Any
               Depth:  Any
       Right
               Type:   Any
               Shape:  Any
               Depth:  Any
Result Domain:
               Type:   Any
               Rank:   1 (vector)
               Shape:  .ro.uu(.nt(,A).epB)/,A
               Depth:  --
Implicit Arguments:  .bxCT (determines comparison precision)
```

Possible Errors Generated: None

The dyadic .nt function returns all the items in the left argument that are not found in the right argument. Duplicate items in the right argument do not affect the result. Duplicates in the left argument are not removed, unless they are specified in the right argument.

If your data represent sets, and you want to remove duplicates from your result, you can use the Unique (.uu) function along with the .nt function.

3 Errors
No errors generated

1 Glossary

2 Abort-input-signal
A technique for escaping to immediate mode when APL is waiting for input. Different terminals form the abort input signal differently. Consult the index to find more information on this subject.

2 Ambivalent-Function
A function that may be monadic or dyadic, depending on how many arguments are supplied when it is invoked.

2 Ambivalent-System-Functions

Function	Shape	Type	Units
.bxBOX	Left: Vector domain	Character	Delimiter
	Right: Matrix domain	Character	Delimited lines
.bxMAP	Left: Vector domain	Character	Function name
	Right: Matrix domain	Character	Shared image info
.bxMONITOR	Left: Vector domain	Numeric	Line numbers
	Right: Matrix domain	Character	Operation names
.bxNL	Left: Vector domain	Character	Letter list
	Right: Vector domain	Integer	Name classes
.bxPACK	Left: Vector domain	Numeric	Data packets
	Right: Matrix domain	Character	Variable names
.bxREWIND	Left: Singleton	Near-int	Key of reference
	Right: Vector domain	Near-int	Channel numbers
.bxSIGNAL	Left: Vector domain	Character	Error message
	Right: Singleton	Integer	Error number
.bxSTOP	Left: Vector domain	Numeric	Line numbers
	Right: Matrix domain	Character	Operation names
.bxTRACE	Left: Vector domain	Numeric	Line numbers
	Right: Matrix domain	Character	Operation names
.bxWAIT	Left: Singleton	Near-int	Timelimit
	Right: Vector domain	Near-int	Channel numbers
.bxWATCH	Left: Singleton	Near-int	Watch mode
	Right: Matrix domain	Character	Variable names

2 APL-terminal
A terminal that has an APL keyboard, that is, a terminal that

can be set up to use the APL key-paired (typewriter-paired), APL bit-paired, or the COMPOSITE APL character set.

2 Argument
An array that is manipulated by a function. APL functions take zero, one, or two arguments.

2 Array
Any number (including 0 or 1) of items treated as a unit.

2 ASCII-Character-Set

	00	16	32	48	64	80	96	112
0	NUL	DLE	SP	0	@	P	`	p
1	SOH	DC1	!	1	A	Q	a	q
2	STX	DC2	.qu	2	B	R	b	r
3	ETX	DC3	.ps	3	C	S	c	s
4	EOT	DC4	$	4	D	T	d	t
5	ENQ	NAK	.pc	5	E	U	e	u
6	ACK	SYN	.ap	6	F	V	f	v
7	BEL	ETB	'	7	G	W	g	w
8	BS	CAM	(	8	H	X	h	x
9	HT	EM	)	9	I	Y	i	y
10	LF	SUB	*	:	J	Z	j	z
11	VT	ESC	+	;	K	[	k	{
12	FF	FS	,	<	L	\	l	|
13	CR	GS	-	=	M	]	m	}
14	SO	RS	.	>	N	.cf	n	~
15	SI	US	/	?	O	.us	o	DEL

2 ASCII-Control-Characters
.bxCTRL (.bxIO_0)

Index	Character Name	Octal Value	Hex Value
0	NUL (Null)	000	00
1	SOH (Start of Heading)	001	01
2	STX (Start of Text)	002	02
3	ETX (End of Text)	003	03
4	EOT (End of Transmission)	004	04
5	ENQ (Enquiry)	005	05
6	ACK (Acknowledge)	006	06
7	BEL (Bell)	007	07
8	BS (Backspace)	010	08
9	HT (Horizontal Tabulation)	011	09
10	LF (Line Feed)	012	0A
11	VT (Vertical Tabulation)	013	0B
12	FF (Form Feed)	014	0C
13	CR (Carriage Return)	015	0D
14	SO (Shift Out)	016	0E
15	SI (Shift In)	017	0F
16	DLE (Data Line Escape)	020	10
17	DC1 (Device Control 1)	021	11
18	DC2 (Device Control 2)	022	12
19	DC3 (Device Control 3)	023	13
20	DC4 (Device Control 4)	024	14
21	NAK (Negative Acknowledge)	025	15
22	SYN (Synchronous Idle)	026	16
23	ETB (End-of-Transmission Block)	027	17
24	CAN (Cancel)	030	18
25	EM (End of Medium)	031	19
26	SUB (Substitute)	032	1A
27	ESC (Escape)	033	1B
28	FS (File Separator)	034	1C
29	GS (Group Separator)	035	1D
30	RS (Record Separator)	036	1E
31	US (Unit Separator)	037	1F

```
  32      DEL (Delete)                    177                7F
```

2 Assignment
 A method for associating a name with an array.

2 Atomic-vector
 An array, returned by the system function .bxAV,
 that contains all the characters in the APL character set.

2 Attention-signal
 A technique for suspending the execution of an operation
 and escaping to immediate mode. The weak attention signal
 (formed by pressing <CTRL/C> once) means suspend execution
 of the current operation after executing the current
 statement, and return control to immediate mode. The
 strong attention signal (formed by pressing <CTRL/C> twice
 means suspend the current operation as soon as possible,
 even in the middle of the statement, and return control to
 immediate mode.

2 Axis
 A dimension along which data elements in an array are arranged.

2 Boolean
 A numeric item that has the value 0 or 1.

2 Branch
 Within a user-defined operation, a change in the normal
 order of statement execution.

2 Canonical-representation
 A character matrix with rows consisting of the original
 lines of a user-defined operation.

2 Channel
 The logical path through which the APL file system interacts with
 external files and mailboxes.

2 Character-editing-mode
 While in function-definition mode, a mode of editing in
 which you can edit individual characters in a line.

2 Command-Line
 The line that contains the DCL command APL. You enter the
 command line in response to the DCL prompt ($).

 Type)HELP APL-COMMAND-LINE for a description of the APL command line,

2 Comment
 Nonexecutable characters appearing to the right of (and on
 the same line as) the " symbol; you can place a comment
 at the end of a line containing APL statements or on a
 separate line.

2 Comparison-Tolerance
 An amount used by APL when it calculates how much two numbers
 can differ and still be considered equal. The system variable
 .bxCT contains the comparison tolerance used by APL.

 Below is the meta-function DFEQ which provides an exact
 definition of how .bxCT is applied to numeric scalars in A=B.

```
     .dlZ_A DFEQ B ;.bxCT;T
 [1]    .bxCT_0
 [2]    T_0.le(#A)##B
 [3]    A_A#T
 [4]    B_B#T
 [5]   Z_(|A-B).leBXCT#(|A).ce|B
 [6]   Z_Z#T
     .dl
```

2 Component
In an external file, a record that contains an APL object.

2 Constant
An item whose value is literally the constant itself.

2 Dense-Sequence
For some functions, APL requires that an argument of nonnegative integers must form a dense sequence, beginning at .bxIO. This means that the smallest element in the argument must be .bxIO, and that an integer N from the argument domain may be included only if N-1 is also included. For example, if the argument domain is the integers from 1 to 3, the arguments 2 1 3, 1 2 2, and 1 1 1 form dense sequences, but the arguments 1 3 1 and 3 2 3 do not.

2 Depth
The degree of nesting of an array.

2 Derived-function
A function that results from the combination of an operator and its argument operand or operands.

2 Domain
The permissible type, shape, and values of a function's argument arrays or the permissible objects of an operator's operands.

2 Dummy-argument
In the header of a user-defined operation, an identifier that serves as a placeholder for the actual argument, operand, or result that is supplied when the operation is called.

2 Dyadic-function
A function that takes both a left and a right argument.

2 Dyadic-System-Functions
A dyadic function takes both a left and a right argument.

Function		Shape	Type	Units
.bxCIQ	Left:	Vector domain	Integer	Packed data
	Right:	Vector domain	Integer	--
.bxCOQ	Left:	Array	Any	Data to be packed
	Right:	Vector domain	Integer	--
.bxEXP	Left:	Vector domain	Near-Bool	Expand information
	Right:	Any	Any	Array to be expanded
.bxFMT	Left:	Vector domain	Character	Format string
	Right:	Any	Any	Semicolon list
.bxREP	Left:	Vector domain	Near-Int	Replicate information
	Right:	Any	Any	Array to be replicated
.bxSS	Left:	Vector domain	Character	Target string
	Right:	Vector domain	Character	Search string

2 Enclosed-array
An array that includes one or more arrays.

2 Empty-array
An array that has a type and shape but no items. The length

```
 of the array along at least one axis is 0.

2 Error-trapping
 Techniques to find and react to errors that occur during
 the execution of user-defined operations.

2 Event-Flag
 A shareable qualifier, accessible through the APL file system,
 intended to aid in synchronizing access to shared files
 or mailboxes.

2 Execute-Only-APL
 The DCL command APL/EXECUTE_ONLY [parameters] invokes the runtime
 support version of VAX APL, called QAPL.  QAPL can execute
 applications written in VAX APL but does not contain the features
 to develop applications.  QAPL can be copied to any valid VAX/VMS
 system free of charge.

 Type )HELP EXECUTE-ONLY  for more information.

2 Expression
 An identifier or constant standing alone, or a function or
 operator and its arguments, or an expression enclosed in
 parentheses.

2 External-data
 Data created outside of APL.

2 External-Routine
 Any routine not written in APL that can be called from the
 APL environment. These include library routines and
 routines written in languages that support the VAX/VMS
 Calling Standard.

2 File-specification
 A standard VAX/VMS file specification in the following form:

   node::device:[directory]filename.filetype;version

 The node is the name of the computer you are using.  It may
 have up to six characters, plus an access control string
 enclosed within diereses (on APL terminals) or double
 quotation marks (on TTY terminals).

 The device is a device name of up to 15 characters.

 The directory is a directory name that includes up to seven
 subdirectory levels separated by periods (.).  Each level's
 name may have up to nine characters.  The directory must be
 enclosed in brackets (<dir> is also accepted).

 The file name and the file type have up to 31 characters,
 and the version is a decimal number not greater than 32767.

 The wild cards accepted in VAX/VMS file specifications include:
 * and % for the file name and file type and * for the version.

 The characters permitted in the file specification string
 include letters (upper- and lowercase), numbers, APL .us, .dd
 or ".

 The defaults are as follows:

   Component              Default

    node        The computer you are using.
```

device	Your default device.
directory	Your default directory.
filename	No default; must be specified.
filetype	AIX - file system AAS -)INPUT/)OUTPUT APL - Workspace Id
version	For input, the highest version number. For output, the highest version number plus one.

2 Fill-element
 A scalar data element (either a space or a 0) inside a fill item.

2 Fill-item
 An array (consisting of spaces, zeros, or a combination of the two)
 that APL inserts into another array. The shape and contents of a
 fill item is based on the prototype of the array that APL is using
 as a model for the array being built. Fill items are used by take,
 replicate, expand, disclose, and .bxBOX.

2 Function
 An operation that applies to arrays and produces an array
 as a result.

2 Function-definition-mode
 An operating mode in which the lines of APL you enter are
 not executed immediately but rather are stored for later
 execution. Function-definition mode begins when you type a
 .dl and ends when you type a second .dl or .pd. This mode
 is used when creating user-defined functions and
 operators.

2 Global-symbol
 A symbol that has the same value inside and outside of
 a user-defined operation.

2 Header
 The initial line of a user-defined operation. See
 Operation Header for more information.

2 Heterogeneous-array
 An array that contains both character and numeric data.

2 High-minus
 The symbol (.ng) used to represent the negative sign in APL.

2 Homogeneous-array
 An array that contains either character or numeric
 data, but not both.

2 Identifier
 A variable name, label name, group name, or user-defined
 operation name. See also System identifier.

2 Identity-element
 An argument (if one exists) to a dyadic function which, when
 used as one argument to the function, does not change the
 value of the other argument. For example, for any identity
 element i applied to a dyadic function f and an argument a,
 a does not change: i f a equals a.

2 Identity-function
 A function that APL applies to the prototype of an array
 when performing the reduction (f/B) of an axis that has

length zero. Note that the inner product (f.g) derived functions imply the use of reduction. The identity function is applied to the prototype of the argument array in place of the specified function.

2 Identity-item
Identity Items for the Scalar Dyadic Functions

Dyadic Function	Identity Item	Symbol
Plus	0	+
Minus	0	-
Times	1	#
Divide	1	%
Power	1	*
Residue	0	|
Maximum	Most negative representable number	.ce
Minimum	Largest representable positive number	.fl
Logarithm	None	.lg
Combination	1	!
Circle	None	.lo
And	1	&
Or	0	.or
Nand	None	.nn
Nor	None	.nr
Less	0	<
Not Greater	1	.le
Equal to	1	=
Not Less	1	.ge
Greater	0	>
Not Equal	0	.ne

Identity Items for the Non-Scalar Dyadic Functions

Dyadic Function	Symbol	Identity Item
Reshape	.ro	.roP
Catenate	,	((.ng1.da.roP),0).ro.lu((.ng1.da.roP),0).roP
Rotate	.rv	(.ng1.da.roP).ro0
Rotate	.cr	(1.da.roP).ro0
Transpose	.tr	.io.ro.roP
Pick	.ru	.io0
Drop	.da	(.ro.roP).ro0
Take	^	.roP
Without	.nt	.io0
Matrix Divide	.dq	(1^.roP).so.=.io^.roP

2 Immediate-mode
An APL operating mode in which lines are executed immediately after they are entered.

2 Index
A notation used to specify the position of items within an array that you want to reference. The index appears immediately to the right of an array and consists of two brackets enclosing values that correspond to axes in the array. Index is synonymous with subscript.

2 Index-origin
The starting point for the index values of an array. The index origin may be 0 or 1. The system variable .bxIO contains the current index origin value.

2 Indexed-assignment

```
 The assignment of values to selected items of a variable.
 The indexed variable is positioned to the left of the
 assignment arrow (_), and the index specifies the items in
 the array where the assignment is applied. Indexed
 assignment is synonymous with subscripted assignment or
 indexed specification.

2 Indexing
 The use of an index to access particular items from an array.

2 Initialization-File
 A file, referenced by the VAX/VMS logical name APL$INIT, which
 contains parameters that are processed when APL is initialized.

2 Initialization-Stream
 Either the DCL command line that invokes APL, or the
 initialization file referenced by the VMS logical name
 APL$INIT. Either or both of these streams may contain
 parameters to be processed when APL is initialized.

2 Integer
 Any of the positive and negative integers, or zero.

2 Internal-data
 Data stored in one of the four APL internal data
 type formats.

2 Key
 A field defined by its location and length within each
 record and used to sort the records. At least one key,
 called the primary key, must be defined for a keyed file.
 Optionally, additional keys, called alternate keys, may be
 defined.

2 Key-of-Reference
 The specific key used in a sequential or random read of a
 multikey file.

2 Keyed-file
 A file in which records are organized by fields, called keys,
 inside the records.  The VAX RMS term is indexed sequential file
 organization (ISAM).  The keys of the file define the order in
 which the records are retrieved; you can retrieve records
 sequentially by one of the sorted orders or randomly by one of the
 record's key values.  A keyed file must contain at least one key.

2 Label
 An identifier associated with a line in a user-defined operation.

2 Latent-expression
 A character vector representing an APL expression; the expression
 is associated with a workspace and is automatically executed when
 the workspace is loaded.  The system variable .bxLX contains
 the value of the workspace's latent expression.

2 Line
 The statement or statements you enter beginning after an APL input
 prompt and ending when you type <RETURN> to enter the line.

2 Local-symbol
 A symbol that has significance only during the execution of a
 particular user-defined operation.

2 Locked-operation
 An operation definition that cannot be changed or displayed.

2 Logical-Name
```

A symbolic name for any portion or all of a file specification.

2 Mailbox
A virtual device useful for sending messages to other processes.

2 Matrix
An array consisting of any number of items arranged along
two axes, commonly called rows and columns.

2 Matrix-Domain
A matrix, vector, or singleton.

2 Monadic-function
A function that takes one argument.

2 Monadic-System-Functions

Function	Shape	Type	Units
.bxARBOUT	Vector domain	Integer	Character codes
.bxASS	Vector domain	Character	File information
.bxASS	Vector domain	Near-int	Channel numbers
.bxBREAK	Any	Any	APL expression
.bxCHS	Vector domain	Near-int	Channel numbers
.bxCLS	Vector domain	Near-int	Channel numbers
.bxCR	Vector domain	Character	Operation name
.bxDAS	Vector domain	Near-int	Channel numbers
.bxDL	Singleton	Floating	Seconds
.bxDVC	Vector domain	Near-int	Channel numbers
.bxEFC	Vector domain	Near-int	Channel numbers
.bxEFR	Vector domain	Near-int	Channel numbers
.bxEFS	Vector domain	Near-int	Channel numbers
.bxEX	Matrix domain	Character	Name list
.bxFI	Vector domain	Character	Numeric strings
.bxFLS	Vector domain	Near-int	Channel numbers
.bxFX	Matrix domain	Character	Operation definition
.bxMBX	Vector domain	Near-int	Channel numbers
.bxNC	Matrix domain	Character	Name list
.bxOM	Vector domain	Boolean	Indexes of 1s
.bxQCO	Vector domain	Character	Workspace name, object names
.bxQLD	Vector domain	Character	Workspace name
.bxQPC	Vector domain	Character	Workspace name, object names
.bxRELEASE	Vector domain	Near-int	Channel numbers

.bxVI	Vector domain	Character	Numeric strings
.bxVR	Vector domain	Any	Value or object name
.bxXQ	Vector domain	Any	APL expression

2 Monitored-operation
 An operation whose lines are being monitored via .bxMONITOR.

2 Multikey-file
 A file in which records are organized by fields, called
 keys, inside the records. The RMS term is indexed
 sequential file organization (ISAM). The keys of the file
 define the order in which the records are retrieved: you
 can retrieve records sequentially by one of the sorted
 orders or randomly by one of the record's sort values. A
 multikey file must contain at least one key.

2 Near-Integer
 A numeric item whose floor is equal to its ceiling.
 Note that the floor and ceiling of a item are affected
 by the APL comparison tolerance.

 Below is the meta-function INTEGER and the function NUMERIC.

```
      .dlZ_INTEGER A ;.bxCT
 [1]    " RETURNS 1 IF A IS ONLY INTEGERS, 0 OTHERWISE
 [2]    .go(Z_0) IF 0.epNUMERIC A
 [3]    .bxCT_BXCT
 [4]   Z_&/,(.ceA)=.flA
      .dl

      .dlZ_NUMERIC A    " RETURNS 1 IF A IS NUMERIC, 0 OTHERWISE
 [1]   Z_1
 [2]   .go0 IF EMPTY A  " TRUE IF EMPTY
 [3]   Z_0.epA.ep.bxAV  " TRUE IF NON-CHARACTER
      .dl
```

2 Nested-array
 A synonym for enclosed array.

2 Next-record-pointer
 An internal mechanism that keeps track of the next record to
 be processed by a sequential input function.

2 Niladic-function
 A function that takes no arguments.

2 Niladic-System-Functions

Function	Description (value in clear workspace)
.bxAI	Account information as 4-integer vector
.bxALPHA	'.ldABCDEFGHIJKLMNOPQRSTUVWXYZ'
.bxALPHAL	'abcdefghijklmnopqrstuvwxyz'
.bxALPHAU	'.ud.za.zb.zc.zd.ze.zf.zg.zh.zi.zj.zk.zl.zm .zn.zo.zp.zq.zr.zs.zt.zu.zv.zw.zx.zy.zz'
.bxASCII	.bxAV subset; approximates ASCII characters
.bxAV	Atomic vector
.bxCHANS	Empty numeric vector

.bxCTRL	The first 32 ASCII characters and <DEL>
.bxLC	Line numbers in state indicator
.bxNUM	'0123456789'
.bxRESET	Clears the state indicator (no value)
.bxTS	Time stamp as 7-integer vector
.bxTT	Terminal type, 1 through 19
.bxUL	Process identification number (PID)
.bxVERSION	Interpreter and workspace versions
.bxWA	Workspace available in bytes

2 Non-APL-terminal
A terminal that does not have an APL keyboard. On such a terminal, APL characters must be represented by ASCII mnemonics.

2 Nonnegative-integer
Any of the positive integers or zero.

2 Operation
Either a user-defined function or user-defined operator. Occasionally, operation refers to a mathematical action (such as the addition operation) or to an action taken by the APL interpreter.

2 Operation-body
The executable lines of APL that appear in an operation definition.

2 Operation-header
The first line you enter when you define an operation. It names the function or operator; indicates whether the operation is niladic, monadic, dyadic, or ambivalent; indicates whether the operation returns a value; indicates the use of an axis argument; and identifies the operation's local symbols.

2 Operator
An operation that is applied to either arrays or functions or both and produces a derived function as a result. In VAX APL there are user-defined operators and primitive operators.

Type)HELP OPERATORS for more information.

2 Operator-sequence
A sequence of functions and operators whose result is a derived function.

2 Overstruck-characters
Some APL characters are formed by combining two other APL characters. For example, the .iq symbol combines the .bx and _ symbols.

Different terminal types form overstrikes in different ways:

Some terminals allow you to enter the first character, use a backspace, and then enter the second character on top of the first.

Other terminals allow you to use a COMPOSE key (or CTRL/D) and then to enter the two characters. On these terminals, only the resulting overstrike character gets displayed.

```
 Type )HELP TERMINAL-SUPPORT followed by a terminal type for
 more information.

2 Panic-exit
 A technique for immediately suspending the execution of an operation
 and giving control to the operating system. The panic exit is
 formed by pressing <CTRL/Y> once. After a panic exit, you can
 return to where you left off by executing the DCL command CONTINUE.
 If you enter the panic exit while an operation is executing, the
 operation is suspended; if you then enter CONTINUE, the operation
 resumes execution at the point where it was interrupted.

2 Pendent-operation
 An operation that has called another operation and is waiting for
 that operation to return.

2 Pervasive-operation
 An operation that occurs at all depths (levels of nesting) of
 an array.

2 PID
 Process Identification, an integer value that uniquely
 identifies a VAX/VMS process.

2 Positive-integer
 The integers greater than zero.

2 Primitive-Mixed-Functions
 The primitive mixed functions allow more extensive array manipulation
 than the scalar functions.  Depending on the values of their
 arguments, mixed functions may:

     o Take a scalar argument and return a vector result

     o Take vector arguments and return a scalar result

     o Take a matrix argument and return a vector result

 Type )HELP FUNCTION-NAMES  for more information.

2 Primitive-Scalar-Functions
 The primitive scalar functions include the arithmetic,
 relational, and logical functions that almost everyone is
 familiar with -- addition, subtraction, equality, AND, OR, and
 so on -- plus a few operations that are less familiar, such
 as residue and roll.  These functions are called SCALAR
 functions because they take scalar arguments and return
 scalar results.

 To obtain more help type )HELP ARITHMETIC-FUNCTIONS
                          )HELP LOGICAL-FUNCTIONS
                          )HELP RELATIONAL-FUNCTIONS

2 Print-precision
 The maximum number of significant digits displayed in
 floating-point output.  The system variable .bxPP contains
 the current print precision value.

2 Print-width
 The maximum number of characters that APL can display on
 a terminal output line.  The system variable .bxPW contains
 the current print width value.

2 Process
 The basic entity scheduled by the VAX/VMS software that provides
 the context in which an image executes.
```

2 Process-identification
An integer value that uniquely identifies a VMS process.
2 Prototype
An array that APL uses to determine the shape and contents of fill items. The prototype of an array B has the same shape as the first item of B, and has character blanks and zeros in positions corresponding to characters and numbers respectively in the first item of B.

2 Process-identification
An integer value that uniquely identifies a VAX/VMS process.

2 Pure-data-record
A record that is a vector of values, with none of the embedded format information that APL includes within component data records.

2 Pure-data-type

Type	External Data Type
0	No conversion; use type of 'data'
1	Convert to 32-bit integer
2	Convert to 1-bit Boolean
3	Convert to single-precision floating-point
4	Convert to D-floating double-precision
5	Convert to 8-bit .bxAV characters
6	Convert to 8-bit ASCII characters
7	Convert to 8-bit unsigned numeric bytes
8	Convert to G-floating double-precision floating-point
9	Convert to H-floating floating-point
10	Convert to 16-bit integer
11	Convert to 8-bit DEC multi-national characters
12	Convert to 8-bit .bxAV characters in TTY mnemonics
13	Convert to 8-bit .bxAV characters in KEY-paired APL
14	Convert to 8-bit .bxAV characters in BIT-paired APL
15	Convert to 8-bit .bxAV characters in COMPOSITE APL

2 Quiet-Functions
A function that does not return a value unless one is needed; that is, a value is returned only if it is not the leftmost function.

Below is a list of quiet functions.

.bxARBOUT	.oq
.bxQCO	.go
.bxQPC	_
.bxQLD	.bxDAS
.bxCLS	.bxRELEASE
.bxWAIT	

2 Random-link
The current value used by the APL random number generator. The system variable .bxRL contains the current random link value.

2 Range
The permissible type, shape, and values of a function's result array.

2 Rank
The number of axes along which an array's items are arranged.

2 Recursive-operation
A user-defined operation that calls itself.

2 Reshape
 A function used to change the number of an array's axes or
 to change the length of one or more of its axes.

2 Row-major-order
 An ordering of the items of an array so that the last subscript
 value varies most rapidly. For example, the row-major order of
 a 2 by 3 matrix would be [1;1], [1;2], [1;3], [2;1], [2;2], [2;3].

2 Scalar
 A rank 0 array (an array with no axes) containing a
 single numeric or character item.

2 Scalar-extension
 An implicit operator that is applied to a dyadic scalar
 function when one or both of the function's arguments are
 singletons. This implicit operator reshapes the singleton
 argument to match the shape of the non-singleton argument,
 allowing the single value from the singleton to be applied
 to each item of the other argument. When both arguments
 are singletons, the argument with the smaller rank is
 reshaped to match the rank of the other singleton.

2 Scalar-product
 An implicit operator that applies a dyadic scalar function
 over each corresponding pair of items in the two arguments.

2 Selective-assignment
 A method for replacing selected items of an array.

2 Shadow
 The act of localizing a name when a user-defined operation
 is activated so that the old value of the name is saved
 and the name becomes undefined in the context of the newly
 activated user-defined operation. The old value of the
 name is restored when the user-defined operation exits to
 its calling environment.

2 Shape
 The way an array's items are arranged; specifically, a numeric
 vector that describes the length of each of the array's axes.

2 Signal
 A term often used in the description of what APL does
 when it detects an error; APL "signals" an error.

2 Simple-array
 An non-enclosed array whose depth is less than 2.

2 Simple-scalar
 A scalar that contains only a single character or number.

2 Singleton
 A 1-item array of any rank.

2 Singleton-extension
 An implicit operation that is applied to a dyadic scalar function
 when one or both of the function's arguments are singletons. This
 implicit operator reshapes the singleton argument to match the
 shape of the nonsingleton argument, allowing the single value from
 the singleton to be applied to each item of the other argument.
 When both arguments are singletons, the argument with the smaller
 rank is reshaped to match the rank of the other singleton.

2 Specification
 A method for associating a name with an array.

```
2 State-indicator
 A vector that reports the status of user-defined operations,
 quad input requests, and execute functions.

2 Statement
 One or more expressions executed as a unit.

2 Stop-bit
 A setting associated with a line in an operation definition
 that causes the operation to be suspended before the line
 is executed.

2 Strand
 Two or more juxtaposed arrays (including scalars) which form a vector.

2 Strand assignment
 The process of associating a strand of values with a set of names.

2 Subprocess
 A process created by and subordinate to another process.
 The subprocess shares the resources of the creating process.

2 Subscript
 A notation used to specify the position of items within an
 array that you want to reference.  The subscript appears
 immediately to the right of an array and consists of two
 brackets enclosing values that correspond to axes in the
 array.  Subscript is synonymous with index.

2 Subscripted-assignment
 An assignment that modifies only the items that are specified
 by an index list.  Subscripted assignment is synonymous with
 indexed assignment or subscripted specification.

2 Suspended-operation
 An operation that has stopped executing, but still has lines
 of APL to be processed.

2 Symbol-table
 A data structure inside the APL interpreter.  The symbol
 table keeps track of the names of all objects in a workspace.

2 System-functions
 APL system functions supplement the primitive functions
 by providing additional processing capabilities.  You
 access a system function by stating its name and arguments
 (if any), just as you would access a primitive function or
 user-defined operation.

 Type )HELP QUAD-NAMES  for a list of system functions.

 For descriptions of type, shape, and units of functions
 within a particular category, type

    )HELP GLOSSARY NILADIC-SYSTEM-FUNCTIONS
    )HELP GLOSSARY MONADIC-SYSTEM-FUNCTIONS
    )HELP GLOSSARY AMBIVALENT-SYSTEM-FUNCTIONS
    )HELP GLOSSARY DYADIC-SYSTEM-FUNCTIONS

2 System-identifier
 Any system-provided name that always begins with the quad
 .bx symbol.  System identifier refers to system variables
 and functions.

2 System-variables
 APL system variables, like ordinary variables, can be used
 in any language expression or function.  Unlike ordinary
```

variables, system variables have special meaning to the system. They allow you to:

- o Set the index origin and comparison tolerance
- o Change the output precision and line width
- o Automatically save an active workspace after operation editing and data input

The following lists the system variables, the range of values you can specify for them, and their default values.

Variable	Value Range	Default
.bxAUS	0, 1, 2	0
.bxCT	0 to 2.328E.ng10	1E.ng15
.bxDC	Nested vector	(.ng1 .ng1 0 2) ''
.bxDML	512 to 2048	2048
.bxERROR	Error message	''
.bxGAG	0, 1, 2, 3	Terminal dependent
.bxIO	0, 1	1
.bxL	Any	.io0
.bxLX	Expression	''
.bxNG	0, 1, 2	1
.bxPP	1 to 16	10
.bxPW	35 to 2044	Terminal width
.bxR	Any	.io0
.bxRL	0 to .ng1+2*31	695197565
.bxSF	Prompt	.bx:<CR><LF><6 spaces>
.bxSINK	Any	Always .io0
.bxTERSE	0, 1	0
.bxTIMELIMIT	.ng1 to 255	0
.bxTIMEOUT	0, 1	0
.bxTLE	0,1	Terminal dependent
.bxTRAP	Expression	''
.bxTT	1 to 15	Terminal dependent

2 SYS$INPUT

The VAX/VMS logical name for your default input device (usually your terminal).

2 SYS$OUTPUT

The VAX/VMS logical name for your default output device (usually your terminal).

2 Terminal-designator

A character string that identifies the type of terminal you are using. If you do not supply a terminal designator in an initialization stream, APL will prompt you for it.

2 Trace-Bit

A setting associated with a line in an operation definition that causes the values of the statements on the line to be displayed each time the line is executed.

Type)HELP .bxTRACE for more information.

2 TTY-Character-Set

TTY mnemonics are combinations of ASCII characters used to represent APL characters on terminals that do not have an APL keyboard.

These TTY mnemonics are either single ASCII characters or keyword abbreviations of two letters preceded by a period.

In the table below, the TTY mnemonic keywords are not preceded by the required period.

TTY Set	Name	ASCII Set	APL Set	Characters to Strike Over	
AB	stile (ABsolute value)	|	|		
AG	Accent Grave	'	.ag	0	/
AL	ALpha		.al		
AP	AmPersand		.ap	3	/
BX	quad (BoX)		.bx		
CB	Column Backslash		.cb	\	-
CC	Column Catenate		.cc	,	-
CE	CEiling		.ce		
CF	CircumFlex		.cf	6	/
CO	COntains		.co	.ru	.us
CR	Column Reverse		.cr	.lo	-
CS	Column Slash		.cs	/	-
DA	Down Arrow		.da		
DD	Dieresis		.dd		
DE	base (DEcode)		.de		
DL	DeL		.dl		
DM	DiaMond		.dm		
DQ	Divide Quad		.dq	.bx	%
DU	Down U		.du		
EN	represent (ENcode)		.en		
EP	EPsilon		.ep		
FL	FLoor		.fl		
FM	thorn (ForMat)		.fm	.en	.so
GD	Grade Down		.gd	.dl	|
GE	Greater than or Equal		.ge		
GO	right arrow (GO to)		.go		
GU	Grade Up		.gu	.ld	|
IB	I-Beam		.ib	.en	.de
IO	IOta		.io		
IQ	Input Quad		.iq	.bx	_
JA-JZ	lowercase letters	a-z	ja-jz	A-Z	\
KA	CTRL/A (Start of Heading)	SOH	ka	A	\
KB	CTRL/B (Start of Text)	STX	kb	B	\
KC	CTRL/C (End of Text)	ETX	kc	C	\
KD	CTRL/D (End of Transmission)	EOT	kd	D	\
KE	CTRL/E (Enquiry)	ENQ	ke	E	\
KF	CTRL/F (Acknowledge)	ACK	kf	F	\
KG	CTRL/G (Bell)	BEL	kg	G	\
KH	CTRL/H (BackSpace)	BS	kh	H	\
KI	CTRL/I (Horizontal Tab)	HT	ki	I	\
KJ	CTRL/J (Line Feed)	LF	kj	J	\
KK	CTRL/K (Vertical Tab)	VT	kk	K	\
KL	CTRL/L (Form Feed)	FF	kl	L	\
KM	CTRL/M (Carriage Return)	CR	km	M	\
KN	CTRL/N (Shift Out)	SO	kn	N	\
KO	CTRL/O (Shift In)	SI	ko	O	\
KP	CTRL/P (Data Line Escape)	DLE	kp	P	\
KQ	CTRL/Q (Device Control 1)	DC1	kq	Q	\
KR	CTRL/R (Device Control 2)	DC2	kr	R	\
KS	CTRL/S (Device Control 3)	DC3	ks	S	\
KT	CTRL/T (Device Control 4)	DC4	kt	T	\
KU	CTRL/U (Negative Acknowledge)	NAK	ku	U	\
KV	CTRL/V (Synchronous Idle)	SYN	kv	V	\
KW	CTRL/W (End-of-Transmission Block)	ETB	kw	W	\
KX	CTRL/X (Cancel)	CAN	kx	X	\
KY	CTRL/Y (End of Medium)	EM	ky	Y	\
KZ	CTRL/Z (Substitute)	SUB	kz	Z	\
LB	Left Brace	{	{		
LD	delta (Lower Del)		.ld		
LE	Less than or Equal		.le		
LG	LoGarithm		.lg	.lo	*
LK	Left tacK		.lk		

LO	circle (Large O)		.lo		
LU	Left U		.lu		
MT	MaTch		.mt	=	.us
NE	Not Equal		.ne		
NG	high minus (NeGation)		.ng		
NN	NaNd		.nn	&	~
NR	NoR		.nr	.or	~
NT	tilde (NoT)	~	~		
OM	OMega		.om		
OQ	Output Quad		.oq	.bx	.go
OR	OR		.or		
PC	PerCent sign		.pc	:	/
PD	Protected Del		.pd	.dl	~
PS	Pound Sign		.ps	=	|
QD	Quad Del		.qd	.bx	.dl
QQ	Quote Quad		.qq	.bx	'
QU	double QUote		.qu	4	/
RB	Right Brace	}	}		
RK	Right tacK		.rk		
RO	RhO		.ro		
RU	Right U		.ru		
RV	ReVerse		.rv	.lo	|
SO	jot (Small O)		.so		
SQ	Squish Quad		.sq	[	]
SS	SubSet		.ss	.lu	.us
TR	TRanspose		.tr	.lo	\
UD	Underscored Delta		.ud	.ld	.us
US	UnderScore		.us		
UU	Up U		.uu		
WD	DEL (Delete)	DEL	wd	8	\
WE	CTRL/[(Escape)	ESC	we	3	\
WF	CTRL/\ (File Separator)	FS	wf	4	\
WG	CTRL/] (Group Separator)	GS	wg	5	\
WN	CTRL/@ (Null)	NUL	wn	0	\
WR	CTRL/^ (Record Separator)	RS	wr	6	\
WU	CTRL/underscore (Unit Separator)	US	wu	7	\
XQ	hydrant (eXecute)		.xq	.de	.so
ZA-ZZ	underscored letters		.za-.zz	A-Z	.us

Note that sharp sign, percent sign, ampersand, up arrow and underscore represent #, %, &, ^, and _ in TTY mnemonics.

2 TTY-mnemonics

Combinations of ASCII characters used to represent APL characters on terminals that do not have an APL keyboard.

2 Type

The data type of an array, either character or numeric.

2 UIC

User Identification Code, assigned by VAX/VMS to each user on the system. It is made up of a group identifier or name, and a member identifier or name.

2 Units

The arguments to some system functions and system commands have specific meanings when used as arguments. Not only does the argument to .bxCR have to be in the character vector domain, for example; it must also be the name of a user-defined operation. The term 'units' tries to describe the specific meaning of these arguments when possible.

2 User-defined-operation

Lines of APL statements (sometimes called programs) that are stored and executed as a unit. The operation can be a function or an operator. External routines become user-defined functions after you map them with .bxMAP.

```
2 Valence
 Ambivalent function
     A function that may be monadic or dyadic, depending on how
     many arguments are supplied when it is invoked.

 Dyadic function
     A function that takes both a left and a right argument.

 Dyadic operator
     An operator that takes both a left and a right operand and
     produces a derived function that is either monadic, dyadic,
     or ambivalent.

 Monadic function
     A function that takes only a right argument.

 Monadic operator
     An operator that takes only a left operand and produces a
     derived function that is either monadic, dyadic or ambivalent.

 Niladic function
     A functions that takes no arguments.

2 Variable
 An identifier whose value may change.

2 Vector
 An array consisting of any number of items arranged
 along one axis.

2 Vector-Domain
 A vector or a singleton.

2 Vector-notation
 See Strand.

2 Version-number
 The display of a version number has the form lv.u-edit, where
 l is the support letter, v is the version number, u is the update
 number, and edit is the edit number.

2 Whitespace
 A sequence of spaces or tabs.

2 Wildcard-character
 A star (*) or percent sign (%) used in a selection string to indicate that
 any characters are allowed in that position.  Wildcard characters can be
 used in file specifications and in identifier strings to commands like
 )VARS.

2 Workspace
 A block of storage in which you operate during an APL session.

2 Workspace-Interchange
 The APL Workspace Interchange Standard (WSIS) describes a method for
 transferring workspaces from one APL implementation to another.
 The WSIS allows a workspace to be transferred regardless of its
 internal APL format or the size and content of the particular
 implementation's .bxAV.


1 Help
 The )HELP command provides you with controlled access to the VMS
 HELP librarian to obtain help on various topics related to the
 VAX APL language.  APL looks for the file associated with the
 logical name APL$HELP:.  If that is not defined, it looks for
 SYS$HELP:VAXAPL.HLB. This system command accepts terms familiar to
 APL as keys into the APL help library and returns a character vector
 (help text) with embedded <CR><LF>s.

2 Character-Set
 Once APL determines a primary key, it translates the key and all related
 subkeys from .bxAV characters to TTY mnemonics using .qq mode; this
 produces keys in a format understood by the help facility, which then
 locates the appropriate text. This text is then translated from TTY
 mnemonics to .bxAV characters, made into uppercase, and then sent to the
 appropriate output destination by APL. (The text is not made into
 uppercase in two instances: When your terminal is a VT102 or is in TTY
 mode; and when you execute )HELP with .bxXQ or .xq.)

2 How-to-build
      To create help file:

      $LIBRARY/CREATE=(KEYSIZE=nn)/HELP output.HLB input.HLP

      how to determine the keysize :

      $LIBRARY/LIST/FULL output.HLB

      Currently the maximum keysize value is 48.

      maximum buffer length is 3500 characters
      maximum number of keys is 10
      maximum keylength is 128 characters

2 Keys
 The VAX APL Help facility contains the actual text of the topics
 (in TTY mnemonics) and is organized into two or three levels.
 For example, .BXASS is a secondary level topic under QUAD-NAMES
 which is a primary level topic. To enable easy access to the
 secondary level, the APL help facility will do some amount of
 translation to generate a primary key from an argument that is a
 secondary key, assuming they have the following characteristics:

             INPUT               TRANSLATION

         Secondary key           Primary key

      <null>                   Help
      Numeric                  Error-numbers
      <.BXAV Char.>            Symbols
      /<string>          Qualifiers
      .BX<string>        Quad-names
      )<string>          System-commands

 This translation allows you to type )HELP .BXASS and receive
 information on that function without having to type )HELP
 QUAD-NAMES .BXASS.

 All other input is assumed to be primary key with optional
 trailing subkeys separated by spaces.  For example, )HELP
 ARITHMETIC-FUNCTIONS would provide a description of arithmetic
 functions and a list of subtopics on which you could obtain help.
 Typing )HELP ARITHMETIC-FUNCTIONS FACTORIAL would provide you with
 information on the factorial function.

 Note that the keys can take unique abbreviations, with uniqueness
 ranging from one to four characters.

2 Output-Format
 The )HELP command returns a character vector with embedded <CR><LF>s.
 If you have requested information on a topic that exists within
 the VAX APL Help facility, the output will be the following:
    Key1
       Key2
          Key3 ...

         help text

         additional help text if any
```

If you have requested information on a topic that does not exist within the VAX APL Help facility, the output will be the following:

```
     Sorry, no documentation on XXX

     Additional information available on ....
```

You have the option to capture the text retrieved by the)HELP command and modify it to any format you desire.

2 Special-Cases

The character '*' is a valid symbol for VAX APL, but it is also treated specially by the VMS HELP utility. Ordinarily, the HELP utility would return the following:

```
    HELP *            returns text on all 1 level keys.
    HELP *...         returns all text from all keys.
```

You may obtain help on the APL * character and also use the HELP utility's 'return all' text functionality by typing one of the following:

```
      HELP *                  returns help text on symbol '*'
      HELP **                 returns text on all 1 level keys.
      HELP **...              returns all text on all keys.
```

Other special translations performed by)HELP if the character is the first non-blank character:

INPUT	TRANSLATION
Secondary key	Generated Keys
'.'	symbols period
'$'	symbols dollar
'%'	symbols divide
'!'	symbols shriek
''''	symbols '
'@'	symbols atsign
'"'	symbols lamp
'?'	symbols questionmark
'('	symbols leftparenthesis

1 Indexing

To access an individual item stored in an array, you must know its position, or index value, within the array. The number of index values, or indexes, needed for an array depends on the array's rank. In general, the number of indexes must match the number of axes of the array; thus, a vector requires one index, a matrix requires two indexes, an array with three axes requires three indexes, and so forth. Scalars may not be indexed.

To index an array, specify the array name, followed by the indexes enclosed in square brackets and separated with semicolons. Note that the number of semicolons in an index specification is equal to one less than the rank of the array being indexed. Each index must be a near-integer constant array (which may be empty), or an expression that evaluates to a near-integer array.

Note that indexing is .bxIO dependent.

2 Errors

describes APL errors that may occur:

SYNTAX ERROR (ILLEGAL NAME CLASS)
A is not a variable.

INDEX RANK ERROR (CANNOT INDEX A SCALAR)
A is a scalar.

INDEX RANK ERROR
The number of axes of A does not equal one more than the number of semicolons in K.

INDEX DOMAIN ERROR (INCORRECT TYPE)
K is not empty and is of type character.

DOMAIN ERROR (NOT AN INTEGER)
K is not a near-integer.

LIMIT ERROR (INTEGER TOO LARGE)
K is greater than the largest allowable integer.

INDEX LENGTH ERROR (SUBSCRIPT OUT OF RANGE)
A value of K is out of the range of the corresponding axis length.

1 Logical-Functions

The monadic .nt and the dyadic &, .or, .nn, and .nr functions are commonly called logical functions. The domain and range of logical functions are restricted to the Boolean values 0 and 1. The results of logical operations for arguments A and B are as follows:

Arguments		Functions				
		And	Or	Nand	Nor	Not
A	B	A&B	A.orB	A.nnB	A.nrB	.ntB
0	0	0	0	1	1	-
0	1	0	1	1	0	-
1	0	0	1	1	0	-
1	1	1	1	0	0	-
-	0	-	-	-	-	1
-	1	-	-	-	-	0

The .nn symbol is formed by overstriking the & and ~ symbols.
The .nr symbol is formed by overstriking the .or and ~ symbols.

1 Operators

APL operators take either functions or values as operands, and produce functions (known as derived functions) as results.

Operators are either monadic or dyadic, but not ambivalent. Monadic operators bind to the left; that is, they take a left operand and not a right operand. Dyadic operators take a left and a right operand. Derived functions are either monadic, dyadic, or ambivalent (their classification depends on the operands and not on the valence of the operator).

```
 You can specify an axis when you use some of the operators. Since axis
 binds to the left, it must appear to the right of the operator.

 There are four APL operators: slash (/)
                               backslash (\)
                               each (.dd)
                               dot (.).

 The slash operator can produce the following derived functions:
                               Compress
                               Replicate
                               Reduce

 The backslash operator can produce the following derived functions:
                               Expand
                               Scan

 The each operator can produce the following derived function:
                               Itemwise application

 The dot operator can produce the following derived functions:
                               Inner product
                               Outer product

 For information on a desired operator type )HELP OPERATORS  followed by the
 operator name.

 For information on a desired derived function type )HELP FUNCTION-NAMES
 followed by the appropriate derived function name.

2 Dot
 The dyadic dot (.) operator takes a left and right operand and
 produces a dyadic derived function.

 When the left operand is a jot (.so), the derived function is
 outer product. Type )HELP FUNCTION-NAMES OUTER-PRODUCT  for more
 information.

 When the left operand is a function, the derived function is
 inner product. Type )HELP FUNCTION-NAMES INNER-PRODUCT  for more
 information.

 The right operand is always a dyadic function.

2 Backslash
 The monadic backslash (\) operator takes a left operand and
 produces a monadic derived function.

 When the operand is a value, the derived function is expansion.
 Type )HELP FUNCTION-NAMES EXPAND for more information.

 When the operand is a function, the result is scan.
 Type )HELP FUNCTION-NAMES SCAN for more information.

2 Each
 The monadic each (.dd) operator takes a function f as the left
 operand. The result is either a monadic or dyadic derived function
 (depending on the valence of f). The function f can be a primitive
 dyadic function, a dyadic system function, a dyadic user-defined
 function, or a dyadic derived function from an arbitrary operator
 sequence.

 When you use .dd, the action of f is applied between successive
 items of an array (B in the form). The action of f is only applied
 to the top level of nesting in an enclosed array (.dd is not
 pervasive).
```

```
3 Errors
 OPERATOR DOMAIN ERROR (OPERAND TO EACH NOT A FUNCTION)

 RANK ERROR
 In the case of a dyadic function f, the ranks must match.

 LENGTH ERROR
 In the case of a dyadic function f, the shapes must match.

2 Slash
 The monadic slash (/) operator takes a left operand and produces
 a monadic derived function.

 When the operand is a value, the derived function is either
 compression or replication. Type )HELP FUNCTION-NAMES
 COMPRESS-REPLICATE  for more information.

 When the operand is a function, the derived function is reduction.
 Type )HELP FUNCTION-NAMES REDUCE for more information.


1 Quad-Names
 Quad names include all names that start with a .bx.
 This includes system variables, niladic system functions,
 monadic system functions, ambivalent system functions,
 dyadic system functions, and basic file system functions.

 For more information about a specific quad name type the
 )HELP QUAD-NAMES  followed by the name of the desired
 function or variable.

2 .bxAI
 .bxAI - Accounting information
 Type:  Niladic System Function
 Form:  uic/cpu-time/connect-time _ .bxAI
 Result Domain:
               Type:   Numeric
               Rank:   1 (vector)
               Shape:  4
               Depth:  1 (simple vector)

 .bxAI returns a vector of the following information:

  o The user identification number; for the user identification
    code (UIC) GROUP,MEMBER, this is MEMBER+(GROUP#2*16).

  o Computer time (CPU time) used during the current APL session.

  o Connect time; time elapsed since the beginning of the
    current APL session.

 All times are expressed in milliseconds.
 The vector has a fourth element, which is always 0.

3 Errors
 No errors generated

2 .bxALPHA
 .bxALPHA - Alphabetics
 Type:  Niladic System Function
 Form:  .ldABCDEFGHIJKLMNOPQRSTUVWXYZ _ .bxALPHA
 Result Domain:
               Type:   Character
               Rank:   1 (vector)
               Shape:  27
               Depth:  1 (simple vector)
 Vector of 27 characters: .ld and A through Z.
 .bxALPHA is a subset of .bxAV.

3 Errors
 No errors generated

2 .bxALPHAL
 .bxALPHAL - Lowercase Alphabetics
 Type:  Niladic System Function
 Form:  lowercase-alphabet _ .bxALPHAL
 Result Domain:
               Type:   Character
               Rank:   1 (vector)
               Shape:  26
               Depth:  1 (simple vector)

 Vector of 26 lowercase characters: a through z.
 .bxALPHAL is a subset of .bxAV.

3 Errors
 No errors generated

2 .bxALPHAU
 .bxALPHAU - Underscored Alphabetics
 Type:  Niladic System Function
 Form:  underscored alphabet _ .bxALPHAU
 Result Domain:
               Type:   Character
               Rank:   1 (vector)
               Shape:  27
               Depth:  1 (simple vector)

 Vector of 27 underscored characters: .ud and .za through .zz.
 .bxALPHAU is a subset of .bxAV.

3 Errors
 No errors generated

2 .bxARBOUT
 .bxARBOUT - Arbitrary Output
 Type:  Monadic System Function (quiet)
 Form:  .io0 _ .bxARBOUT B
 Argument Domain:
               Type:   Near-integer
               Shape:  Vector domain
               Depth:  1 (simple)
               Range:  0 to 225
 Result Domain:
               Type:   Numeric
               Rank:   1 (vector)
               Shape:  0 (empty)
               Depth:  1 (simple vector)
 Implicit Arguments:  None

 .bxARBOUT allows you to send untranslated output to the terminal
 (actually, to the default output device).  .bxARBOUT outputs the
 argument's items as if they were character codes.

 Type )HELP GLOSSARY ASCII-CHARACTER-SET  for more information.

3 Errors
 RANK ERROR (NOT VECTOR DOMAIN)
 The argument is not a singleton and has a rank greater than 1.

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 The argument is not a simple, homogeneous array.
 DOMAIN ERROR (INCORRECT TYPE)
 The argument is non-empty and numeric.

 DOMAIN ERROR (NOT AN INTEGER)
 The argument is not near-integer.

 LIMIT ERROR (INTEGER TOO LARGE)
 The argument is greater than the largest allowable integer.

 DOMAIN ERROR
 The argument is not an integer in the range 0-255.

2 .bxASCII
 Type:  Niladic System Function
 Form:  ASCII-characters _ .bxASCII
 Result Domain:
               Type:   Character
               Rank:   1 (vector)
               Shape:  128
               Depth:  1 (simple vector)

 .bxASCII is a subset of .bxAV; it returns a vector of 128
 characters which approximate the 7-bit ASCII character set.
 .bxASCII contains the ASCII control characters (.bxCTRL) and
 the lowercase letters (.bxALPHAL).

3 Errors
 No errors generated

2 .bxASS
 .bxASS - Associating Files with Channels (two forms)

 The query form returns the current value of assignments made
 previously with the action form.

 Type:  Monadic System Function (query form)
 Form:  current-assignments _ .bxASS chans
 Argument Domain:
               Type:   Near-integer
               Shape:  Vector domain
               Depth:  1 (simple)
               Range:  .ng999 to 999 (not 0)
 Result Domain:
               Type:   Character
               Rank:   1 or 2
               Shape:  Vector or matrix
               Depth:  1 (simple)
 Implicit Arguments:  None

 The action form associates a file with a channel.

 Type:  Monadic System Function (action form)
 Form:  chan _ .bxASS file-attributes
 Argument Domain:
               Type:   Character
               Shape:  Vector domain
               Depth:  1 (simple)
 Result Domain:
               Type:   Integer
               Rank:   0 (scalar)
               Shape:  .io0 (scalar)
               Depth:  0 (simple scalar)
 Implicit Arguments:  None

 The details of the file attributes for the action form of
 .bxASS are as follows:
      .bxASS chan fspec /forg
          /BLOCKSIZE : blocksize
          /BUFFERCOUNT : n
          /CCONTROL : {FORTRAN | LIST | NONE}
          /DEFAULTFILE : defaultspec
          /DISPOSE : {KEEP | DELETE | PRINT |
                 PRINTDELETE | SUBMIT | SUBMITDELETE}
          /EFN : n
          /MAXLEN : length
          /MBX
          /NFS
          /NOSHARE
          /NOWRITERS
          /OPEN : {NEW | OLD}
          /PROTECTION : protection
          /READONLY : NOLOCKS
          /RECORDTYPE : {FIXED | STREAM | STREAMCR |
                 STREAMLF | VARIABLE}
          /SHARE
          /SIGNAL
          /UPDATE
          /WRITEONLY
```

If .bxASS fails, it returns 0 and sets .bxERROR.

Type)HELP FILE-SYSTEM FILE-ORGANIZATION-QUALIFIERS for information on /forg. Type)HELP GLOSSARY FILE-SPEC for information on file-spec.

3 Errors

For the action form of .bxASS:

DOMAIN ERROR (ERROR PARSING ARGUMENT TO CCONTROL)
An invalid value was specified for /CCONTROL.

DOMAIN ERROR (REDUNDANT KEYWORD OR QUALIFIER)
A keyword or qualifier was repeated.

DOMAIN ERROR (CONFLICTING QUALIFIERS SPECIFIED)
More than one of the following qualifiers was specified in the argument: /READONLY, /WRITEONLY, or /UPDATE.

FILE PROTECTION VIOLATION
A delete value was specified for /DISPOSE when VMS delete privileges were not enabled.

EOF ENCOUNTERED
While the /SIGNAL qualifier was enabled, a sequential read operation was attempted on a nonexistent record.

RECORD NOT FOUND
While the /SIGNAL qualifier was enabled, a random read operation was attempted on a nonexistent record.

BLOCK TOO BIG
An attempt was made to use a pure data record size that exceeds the current /MAXLEN value.

IO ERROR (INVALID RECORD SIZE)

IO ERROR (FILE CURRENTLY LOCKED BY ANOTHER USER)
The file specified in the argument is locked by a user outside the APL environment (VAX/RMS is denying access).

For the query form:

RANK ERROR (NOT VECTOR DOMAIN)

The argument is not a singleton and its rank is
greater than 1.

DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
The value specified for chans must be a simple
homogeneous array.

DOMAIN ERROR (NOT AN INTEGER)
The channel number is not an integer.

LIMIT ERROR (INTEGER TOO LARGE)
The argument is greater than the largest allowable integer.

DOMAIN ERROR (INVALID CHANNEL NUMBER)
The value specified for chans is outside the argument domain.

2 .bxAUS

```
 .bxAUS - Automatic Save
 Type:   System Variable
 Forms:  .bxAUS _ near-integer-singleton
         integer-scalar _ .bxAUS
 Value Domain:
                Type:     Near-integer
                Shape:    Singleton
                Depth:    0 or 1 (simple)
                Value:    0, 1, or 2
                Default:  0
 Result Domain:
                Type:   Integer
                Rank:   0 (scalar)
                Shape:  .io0 (scalar)
                Depth:  0 (simple scalar)
```

.bxAUS periodically backs up workspace.

.bxAUS _ 0 means the automatic save feature is not activated.
.bxAUS _ 1 or 2 means the feature is activated

The workspace is saved in your default directory as follows:

Value of .bxAUS	File Name of Saved Workspace
1	APLxxxxxxxx.TMP, where xxxxxxxx is the value of .bxUL represented in hexadecimal.
2	File name = CONTINUE. File type = APL.

3 Errors

RANK ERROR (NOT VECTOR DOMAIN)
The value is not a singleton and its rank is greater than 1.
For example: .bxAUS_2 2.ro4 is incorrect.

LENGTH ERROR (NOT SINGLETON)
The value is not a single item.
For example: .bxAUS_.io3 is incorrect.

DOMAIN ERROR (INCORRECT TYPE)
The value is non-empty and character. For example: .bxAUS_'A'
is incorrect.

DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
The argument is not a simple, homogeneous array.

DOMAIN ERROR (NOT AN INTEGER)
The value is not near-integer. For example: .bxAUS_2.5
is incorrect.

```
 LIMIT ERROR (INTEGER TOO LARGE)
 The value is greater than the largest allowable integer.
 For example: .bxAUS_.fl2*33 is incorrect.

 DOMAIN ERROR (PARAMETER OUT OF RANGE)
 An attempt was made to use an unavailable value as the value.
 For example: .bxAUS_10 or .bxAUS_3 is incorrect.

2 .bxAV
 .bxAV - Atomic Vector
 Type:  Niladic System Function
 Form:  all-known-chars _ .bxAV
 Result Domain:
               Type:   Character
               Rank:   1 (vector)
               Shape:  256
               Depth:  1 (simple vector)

 .bxAV contains a vector of the 256 characters known to APL.

3 Errors
 No errors generated

2 .bxBOX
 .bxBOX - Forming Character Matrices and Vectors
 Type:  Ambivalent System Function
 Form:  boxed-text _ delimiter .bxBOX text
 Argument Domain:
     Left
               Type:   Character
               Shape:  Vector domain
               Depth:  0 or 1 (simple)
     Right
               Type:   Character
               Shape:  Matrix domain
               Depth:  0 or 1 (simple)
 Result Domain:
               Type:   Character
               Rank:   1 or 2
               Shape:  Matrix or Vector
               Depth:  1 (simple)
 Implicit Arguments:  None

 Returns a matrix from a character vector whose rows
 are delineated by <CR><LF> and vice versa.

 When the right argument is in the vector domain, .bxBOX
 forms a matrix. When the argument is a matrix, .bxBOX forms
 a vector. If the argument is empty, the result is an empty
 character matrix with the shape 0 0.

 When producing a matrix, APL uses a delimiter to determine where
 to form rows. The left argument optionally specifies a delimiting
 string. The default delimiter is <CR><LF>. The number of columns
 in the resulting matrix is equal to the longest string contained
 between any pair of delimiters. Shorter strings are padded with
 trailing blanks.

 When producing a vector with the monadic form, APL removes any
 trailing blanks and inserts the <CR><LF> delimiter at the end
 of each row.

 When producing a vector with the dyadic form, APL does not remove
 trailing blanks from the rows of the matrix argument. It does insert
 the specified delimiter at the end of each row.
```

```
3 Errors
 RANK ERROR (NOT MATRIX DOMAIN)
 B is not in the matrix domain.

 RANK ERROR (NOT VECTOR DOMAIN)
 The left argument is not a singleton and its rank
 is greater than 1.

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 The argument is not a simple, homogeneous array.

 DOMAIN ERROR (INCORRECT TYPE)
 An argument is a numeric array.

2 .bxBREAK
 .bxBREAK - Suspending Execution
 Type:  Monadic System Function (no result)
 Form:  .bxBREAK apl-expression
 Argument Domain:
               Type:   Any
               Shape:  Any
               Depth:  0 or 1 (simple)
 Result Domain:
               Type:   None
               Rank:   None
               Shape:  None
               Depth:  None
 Implicit Arguments:  None

 Suspends operation execution and returns control
 to immediate mode. Usually, .bxBREAK is placed inside
 an operation causing the operation to end at a given point.
 The break to immediate mode is not trappable with .bxTRAP.

3 Errors
 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 The argument is not a simple, homogeneous array.

2 .bxCHANS
 .bxCHANS - Returning Channel Numbers
 Type:  Niladic System Function
 Form:  current-channels _ .bxCHANS
 Result Domain:
               Type:   Integer
               Rank:   1 (vector)
               Shape:  Vector
               Depth:  1 (simple vector)

 Identifies channel numbers associated with files.

3 Errors
 No errors generated

2 .bxCHS
 .bxCHS - Returning File Organization and Open Status
 Type:  Monadic System Function
 Form:  file-org/status _ .bxCHANS chans
 Argument Domain:
               Type:   Near-integer
               Shape:  Vector domain
               Range:  .ng999 to 999 (but not 0)
               Depth:  0 or 1 (simple)
 Result Domain:
               Type:   Integer
               Rank:   1 or 2
               Shape:  Vector or matrix
               Depth:  1 (simple)
```

Implicit Arguments: None

Returns file organization and open status.

Below are the possible .bxCHS codes

First Element		Second Element	
Code	Organization	Code	Status
0	Not applicable	0	Channel free
1	/AS	1	Assigned but not open
2	/IS	2	Open for output
3	Not applicable	3	Open for input
4	/DA	4	Open for input and output
7	/RF		
8	/KY		

3 Errors

RANK ERROR (NOT VECTOR DOMAIN)
The argument is not a singleton and its rank
is greater than 1.

DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
The value specified for chans must be a simple array.

DOMAIN ERROR (NOT AN INTEGER)
The channel number is not an integer.

DOMAIN ERROR (INVALID CHANNEL NUMBER)
The value specified for chans is outside the argument domain.

LIMIT ERROR (INTEGER TOO LARGE)
The argument is outside the range of valid integer values.

2 .bxCIQ

```
.bxCIQ - Unpacking Data
Type:  Dyadic System Function
Form:  unpacked-data _ packed-data .bxCIQ header {type}
Argument Domain:
       Left
               Type:   Integer
               Shape:  Vector domain
               Depth:  0 or 1 (simple)
       Right
               Type:   Near-integer
               Shape:  Vector domain
               Depth:  0 or 1 (simple)
Result Domain:
               Type:   Any
               Rank:   Any
               Shape:  Any
               Depth:  Any
Implicit Arguments:  None
```

Unpacks data packed by .bxCOQ.

The following form unpacks data:

packed-data .bxCIQ header {type}

where

packed-data is the packed data; it must be in the format
of the result of .bxCOQ, with or without a header. It may be
empty only if header is 0.

header is 2, if a header exists, or 0, if no header exists.
If you specify 0 and a header does exist, the header is
treated as part of the data to be unpacked.

type, if specified, indicates the type that you want to convert
the packed data into. If header is 0, type must be specified and
must not be 0. If header is 2, type must be unspecified or 0.

The result of .bxCIQ may be an enclosed array if its left argument
was created by applying .bxCOQ to an enclosed array without type
conversion. In this case, the right argument to .bxCIQ must specify
only a header (ie, be the value 2) with no type conversion.

Type)HELP GLOSSARY PURE-DATA-TYPE for more information on the
possible values for the type parameter.

3 Errors
RANK ERROR (NOT VECTOR DOMAIN)
An argument is not a singleton and its rank
is greater than 1.

LENGTH ERROR (ARGUMENT MUST BE 1 OR 2 ELEMENTS)
.bxCIQ may have at most two items in the right argument.

DOMAIN ERROR (INCORRECT TYPE)
An argument is non-empty and of type character.

DOMAIN ERROR (NOT AN INTEGER)
An argument is not a near-integer.

LIMIT ERROR (INTEGER TOO LARGE)
A or B is greater than the largest allowable integer.

DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
An argument is not a simple, homogeneous array.

DOMAIN ERROR (INVALID HEADER TYPE)
An incorrect header type was specified for .bxCIQ.
The first element of B must be 0 or 2.

LENGTH ERROR (DATA TYPE MISSING)
The data type parameter in the right argument to .bxCIQ
is required in this case.

LENGTH ERROR (DATA TYPE EXCEEDS DATA LENGTH)
The data type specified for .bxCIQ is incompatible with the
length of the left argument.

DOMAIN ERROR (INVALID EXTERNAL DATA TYPE)
The second element of B is not a valid external data type.

DOMAIN ERROR (DATA TYPE MUST BE UNSPECIFIED OR ZERO)
If the first element of B is 2, then the second element of B
(if present)must be 0 (to specify the use of the data type
packed in A).

DOMAIN ERROR (INVALID CIQ HEADER)
The first element of B specifies that A contains a packed
header but it is not in the correct form.

DOMAIN ERROR
The data in A cannot be converted to the external data
type specified by the second element of B.

2 .bxCLS
.bxCLS - Closing Files
Type: Monadic System Function (quiet)

```
 Form:  .io0 _ .bxCLS chans
 Argument Domain:
                Type:   Near-integer
                Shape:  Vector domain
                Depth:  0 or 1 (simple)
                Value:  .ng999 to 999 (not 0)
 Result Domain:
                Type:   Numeric
                Rank:   1 (vector)
                Shape:  0 (empty)
                Depth:  1 (simple vector)
 Implicit Arguments:  None
```

Closes the files on one or more channels.

3 Errors

RANK ERROR (NOT VECTOR DOMAIN)
The argument is not a singleton and its rank
is greater than 1.

DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
The value specified for chans must be a simple array.

DOMAIN ERROR (NOT AN INTEGER)
The channel number is not an integer.

DOMAIN ERROR (INVALID CHANNEL NUMBER)
The value specified for chans is outside the argument domain.

LIMIT ERROR (INTEGER TOO LARGE)
The argument is outside the range of valid integer values.

2 .bxCOQ

```
 .bxCOQ - Packing Data
 Type:  Dyadic System Function
 Form:  packed-data _ data .bxCOQ header {type}
 Argument Domain:
        Left
                Type:   Any
                Shape:  Any
                Depth:  Any
        Right
                Type:   Near-integer
                Shape:  Vector domain
                Depth:  0 or 1 (simple)
 Result Domain:
                Type:   Integer
                Rank:   1 (vector)
                Shape:  Vector
                Depth:  1 (simple vector)
 Implicit Arguments:  None
```

Packs data of different types for storage as one
record.

To pack data, use the following form:

data .bxCOQ header type

where

data is any array you want to pack into an integer vector.

header is 0, 2, or 4. Use 0 if you do not want a header; 2
if you do want a header; and 4 if you want only a header.

type, if specified, indicates whether the data is to be converted to another data type before being packed.

The left argument to .bxCOQ may be an enclosed array. If
the "type" parameter in the right argument to .bxCOQ is equal
to 0 or is omitted, the enclosed data is packed into an integer
vector with enough information to recreate its structure (depth)
as well as its shape. In this case, the "type" field in the
COQ header of the packed data has a value of 17. 17 is not a
valid value for the "type" parameter in the right argument to
.bxCOQ or .bxCIQ.

If the "type" parameter in the right argument to .bxCOQ is
non-zero, all of the scalar elements from the items of the
enclosed data are converted to the specified "type" and packed
in ravel order into the result. The enclosed data must be
homogeneous (but not necessarily simple) for the conversion
to be successful. No structure (depth) information about the
enclosed data is retained in the result.

Type)HELP GLOSSARY PURE-DATA-TYPE for more information
on the possible values for the type parameter.

3 Errors

RANK ERROR (NOT VECTOR DOMAIN)
B is not a singleton and its rank is greater than 1 or
A is not a singleton and its rank is greater than 1 when
B specifies an external data type of 11 through 15.

LENGTH ERROR (ARGUMENT MUST BE 1 OR 2 ELEMENTS)
.bxCOQ may have at most two items in the right argument.

DOMAIN ERROR (DATA TYPE MISSING)
The left argument to .bxCOQ has an enclosed array
as its value, which is invalid when the right argument
is 0 0 (specifying that the result should be the
same type as the left argument with no header).

DOMAIN ERROR (DATA TYPE MUST BE UNSPECIFIED OR ZERO)
For .bxCOQ, APL cannot create a header and perform a conversion
when packing an enclosed array. This means that for X, an
enclosed array, and N, a non-zero number, the following
expressions signal an error: X .bxCOQ 2 N and X .bxCOQ 4 N

DOMAIN ERROR (INCORRECT TYPE)
An argument is non-empty and of type character.

DOMAIN ERROR (NOT AN INTEGER)
B argument is not a near-integer.

LIMIT ERROR (INTEGER TOO LARGE)
B is greater than the largest allowable integer.

DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
B is not a simple, homogeneous array.

DOMAIN ERROR (INVALID HEADER TYPE)
An incorrect header type was specified for .bxCOQ.
The first element of B must be 0, 2, or 4.

DOMAIN ERROR (INVALID EXTERNAL DATA TYPE)
The second element of B does not specify a valid
external data type.

DOMAIN ERROR
The data in A cannot be converted to the external
data type specified by B.

```
2 .bxCR
 .bxCR - Obtaining a Canonical Representation
 Type:  Monadic System Function
 Form:  canonical-rep _ .bxCR operation-name
 Argument Domain:
                Type:   Character
                Shape:  Vector domain
                Depth:  0 or 1 (simple)
 Result Domain:
                Type:   Character
                Rank:   2
                Shape:  Matrix
                Depth:  1 (simple matrix)
 Implicit Arguments:  .bxPP (controls print precision)

 Returns a canonical representation of a user-defined
 operation whose name is the character string specified.

3 Errors
 RANK ERROR (NOT VECTOR DOMAIN)
 The argument is not a singleton and its rank
 is greater than 1.

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 The argument is not a simple, homogeneous array.

 DOMAIN ERROR (INCORRECT TYPE)
 The argument is numeric.

2 .bxCT
 .bxCT - Comparison Tolerance
 Type:   System Variable
 Forms:  .bxCT _ tolerance-value
         floating-scalar _ .bxCT
 Value Domain:
                Type:     Numeric
                Shape:    Singleton
                Depth:    0 or 1 (simple)
                Value:    0 through 2.328E.ng10
                Default:  1E.ng15
 Result Domain:
                Type:   Non-negative numeric
                Rank:   0 (scalar)
                Shape:  .io0 (scalar)
                Depth:  0 (simple scalar)

 Determines the degree of tolerance applied in
 numeric comparisons.

 The value of .bxCT affects the following primitive functions:

 Ceiling                            .ce
 Floor                              .fl
 Less than                          <
 Less than or equal to              .le
 Equal to                           =
 Greater than or equal to           .ge
 Greater than                       >
 Not equal to                       .ne
 Residue                            |
 Matrix inverse and divide          .dq
 Index of                           .io
 Set membership                     .ep
 Set union and unique               .uu
 Set intersection                   .du
 Without                            .nt
 Subset                             .ss
 Contains                           .co
 Match                              .mt
 Matrix inverse and divide          .dq

3 Errors
 RANK ERROR (NOT VECTOR DOMAIN)
 The value is not a singleton and its rank is greater
 than 1. For example: .bxCT_2 2.ro4 is incorrect.

 LENGTH ERROR (NOT SINGLETON)
 The value is not a single item. For example:
 .bxCT.io3 is incorrect.

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 The value is not a simple, homogeneous array.

 DOMAIN ERROR (INCORRECT TYPE)
 The value is non-empty and character. For example: .bxCT_'A'
 is incorrect.

 DOMAIN ERROR (PARAMETER OUT OF RANGE)
 An attempt was made to use an unavailable value as the value.
 For example: .bxCT_2.5 is incorrect.

2 .bxCTRL
 .bxCTRL - Control Characters
 Type:  Niladic System Function
 Form:  control-chars _ .bxCTRL
 Result Domain:
               Type:   Character
               Rank:   1 (vector)
               Shape:  33
               Depth:  1 (simple vector)

 Vector of 33 ASCII control characters.
 .bxCTRL is a subset of .bxAV

 Type )HELP GLOSSARY ASCII-CONTROL-CHARACTERS  for more information.

3 Errors
 No errors generated

2 .bxDAS
 .bxDAS - Deassigning Files
 Type:  Monadic System Function (quiet)
 Form:  .io0 _ .bxDAS chans
 Argument Domain:
               Type:   Near-integer
               Shape:  Vector domain
               Depth:  0 or 1 (simple)
               Value:  .ng999 to 999 (but not 0)
 Result Domain:
               Type:   Numeric
               Rank:   1 (vector)
               Shape:  0 (empty)
               Depth:  1 (simple vector)
 Implicit Arguments:  None

 Disassociates files from one or more channels.

3 Errors
 RANK ERROR (NOT VECTOR DOMAIN)
 The argument is not a singleton and its rank is
 greater than 1.

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 The value specified for chans must be a simple array.
```

DOMAIN ERROR (NOT AN INTEGER)
The channel number is not an integer.

DOMAIN ERROR (INVALID CHANNEL NUMBER)
The value specified for chans is outside the argument domain.

LIMIT ERROR (INTEGER TOO LARGE)
The argument is outside the range of valid integer values.

2 .bxDC

```
Type:   System Variable
Forms:  .io0 _ .bxDC _ display-area box-characters
        current-setting _ .bxDC
Value Domain:
               Type:     Enclosed, heterogeneous
               Shape:    2 (vector)
               Depth:    2 or 3 (enclosed)
               Default:  (.ng1 1 0 2) ''
Result Domain:
               Type:   Enclosed, heterogeneous
               Rank:   1 (vector)
               Shape:  2 (vector)
               Depth:  2 or 3 (enclosed vector)
```

.bxDC specifies how APL displays enclosed arrays. You can set .bxDC to draw boxes around enclosed items of an array, and the resulting display can help to visualize the nested structure of the array. You can also increase the amount of blank space that APL uses to surround an enclosed item.

The value you assign to .bxDC is a 2-item enclosed vector. The second item is a character vector that is either empty (''), if you do not want boxes, or has length 8. The vector specifies the characters for APL to use when it draws the boxes. The first four items specify the symbols for the corners of boxes (upper left, upper right, lower left, lower right), the next two items specify the left and right sides, and the last two specify the top and bottom.

For example, if you specify '++++||==' as the second item of the .bxDC value, APL draws boxes that look like the following:

```
     +======+
     |      |
     |      |
     +======+
```

The first item of the .bxDC value is a simple numeric vector of length 4. Data elements 1 and 2 of this item specify where an item is displayed when a display area is larger than the structure of the item itself. The first data element controls the vertical placement; the item can be at the top, center, or bottom of the display area. The second data element controls the horizontal placement; the item can be at the left, center, or right of the display area. The following list describes the meaning of the values you can specify for these two data elements.

Positioning Items in Display Areas

First Element	Location	Second Element	Location
.ng1	Top	.ng1	Left
0	Center	0	Center
1	Bottom	1	Right

Data elements 3 and 4 of the first item of the .bxDC value allow you to change the size of the display areas. The third

data element controls the vertical space between rows of items;
the integer you specify indicates how many blank rows you want to add.
The fourth element controls the horizontal space between columns; the
integer you specify indicates how many blank columns you want to add.
(When you display boxes, the minimum value you can specify for the
third and fourth elements is 2.)

3 Errors

RANK ERROR (MUST BE VECTOR)
The value, and each item in the value, must be vectors.

LENGTH ERROR (DISPLAY CONTROL VECTOR MUST BE TWO ITEMS)
The value must have length 2.

LENGTH ERROR (DISPLAY CONTROL ITEM WRONG LENGTH)
The first item must have length 4. The second item can
either be empty or have length 8.

DOMAIN ERROR (ENCLOSED VALUE REQUIRED)
The value must be an enclosed array.

DOMAIN ERROR (ENCLOSED ARRAY NOT ALLOWED)
The first item must be a simple homogeneous array.
The second item, if empty, must be simple.

DOMAIN ERROR
The first item must be near-integer.

DOMAIN ERROR (PARAMETER OUT OF RANGE)
Elements 1 and 2 of the first item can only be .ng1, 0, or 1.

DOMAIN ERROR (NEGATIVE INTEGER NOT ALLOWED)
Elements 3 and 4 of the first item must be non-negative.

DOMAIN ERROR (INCORRECT TYPE)
The first item must be character. The second item, if simple
and non-empty, must be character. The first 4 elements of the
second item must be character. The last 4 elements of the second
item must be empty or character.

RANK ERROR (NOT SINGLETON)
The first 4 elements of the second item must be non-empty
singletons.

LENGTH ERROR (NOT SINGLETON)
Each of the first 4 elements of the second item must have
length 1.

RANK ERROR (NOT VECTOR DOMAIN)
The last 4 elements of the second item must be singletons
or vectors.

2 .bxDL

```
 .bxDL - Delaying the Execution of an Operation
 Type:  Monadic System Function
 Form:  actual-delay _ .bxDL seconds
 Argument Domain:
                Type:   Numeric
                Shape:  Singleton
                Depth:  0 or 1 (simple)
                Value:  seconds <.ng1+2*18
 Result Domain:
                Type:   Non-negative numeric
                Rank:   0 (scalar)
                Shape:  .io0 (scalar)
                Depth:  0 (simple scalar)
 Implicit Arguments:  None
 Delays execution by the number of seconds (range is
 0 to .ng1+2*18) specified in its argument.

 A weak attention signal stops the wait, but does not
 suspend execution.

3 Errors
 RANK ERROR (NOT VECTOR DOMAIN)
 The argument has a rank greater than 1.

 LENGTH ERROR (NOT SINGLETON)
 The argument is not a single item.

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 The argument is not a simple, homogeneous array.

 DOMAIN ERROR (INCORRECT TYPE)
 The argument is non-numeric.

 LIMIT ERROR (DELAY VALUE TOO LARGE)
 The argument to .bxDL is greater than .ng1+2*18.

2 .bxDML
 .bxDML - Workspace Maximum Record Length
 Type:   System Variable
 Forms:  .bxDML _ default-length
         integer-scalar _ .bxDML
 Value Domain:
                Type:     Near-integer
                Shape:    Singleton
                Depth:    0 or 1 (simple)
                Value:    512 through 2048 (bytes)
                Default:  2044
 Result Domain:
                Type:   Integer
                Rank:   0 (scalar)
                Shape:  .io0 (scalar)
                Depth:  0 (simple scalar)

 Controls default maximum record length used to save
 the workspace or to create a file.

3 Errors
 RANK ERROR (NOT VECTOR DOMAIN)
 The value is not a singleton and its rank is greater than
 1. For example: .bxDML_2 2.ro4 is incorrect.

 LENGTH ERROR (NOT SINGLETON)
 The value is not a single item. For example:
 .bxDML_.io3 is incorrect.

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 The value is not a simple, homogeneous array.

 DOMAIN ERROR (INCORRECT TYPE)
 The value is non-empty and character. For example: .bxDML_'A'
 is incorrect.

 DOMAIN ERROR (NOT AN INTEGER)
 The value is not near-integer. For example: .bxDML_2.5
 is incorrect.

 LIMIT ERROR (INTEGER TOO LARGE)
 The value is greater than the largest allowable integer.
 For example: .bxDML_.fl2*33 is incorrect.
```

DOMAIN ERROR (PARAMETER OUT OF RANGE)
An attempt was made to use an unavailable value as the value.
For example: .bxDML_2099 is incorrect.

2 .bxDVC
.bxDVC - Returning Device Characteristics
Type: Monadic System Function (query)
Form: characteristics _ .bxDVC chans
Argument Domain:

```
               Type:   Near-integer
               Shape:  Vector domain
               Depth:  0 or 1 (simple)
               Value:  .ng999 through 999 (but not 0)
```

Result Domain:

```
               Type:   Integer
               Rank:   1 or 2
               Shape:  Vector or matrix (n by 2)
               Depth:  1 (simple)
```

Implicit Arguments: None

Returns the VMS device characteristics longword
for the device where one or more files are located.
The files are specified by their channel numbers.

For each channel specified in the argument, .bxDVC
returns one row containing two values: the first value
is the VAX/VMS device-characteristics longword, and
the second value is always 0.

It is usually helpful to convert the device-characteristics
longword to binary format before examining it. You can compare
the binary value of the longword with the device characteristics
listed below. The first element in the list is associated with
the rightmost bit in the longword, the second element is associated
with the next rightmost bit, and so forth.

Bit	Type or Condition of Device
0	Record-oriented
1	Carriage-control
2	Terminal
3	Directory-structured
4	Single directory-structured
5	Sequential, block-oriented
6	Being spooled
7	Operator console
8	RA50,RA81,RA82,RH60
9-12	(Bits reserved)
13	Network
14	File-oriented
15	(Bit reserved)
16	Shareable
17	Generic
18	Available for use
19	Mounted
20	Mailbox
21	Marked for dismount
22	Error logging enabled
23	Allocated
24	Non-file-structured
25	Software write-locked
26	Capable of providing input
27	Capable of providing output
28	Allows random access
29	Real-time
30	Read-checking enabled
31	Write-checking enabled

```
3 Errors
 RANK ERROR (NOT VECTOR DOMAIN)
 The argument is not a singleton and its rank is greater
 than 1.

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 The value specified for chans must be a simple array.

 DOMAIN ERROR (NOT AN INTEGER)
 The channel number is not an integer.

 DOMAIN ERROR (INVALID CHANNEL NUMBER)
 The value specified for chans is outside the argument
 domain.

 LIMIT ERROR (INTEGER TOO LARGE)
 The argument is outside the range of valid integer values.


2 .bxEFC
 .bxEFC - Event Flag Clear
 Type:  Monadic System Function
 Form:  previous-values _ .bxEFC chans
 Argument Domain:
               Type:   Near-integer
               Shape:  Vector domain
               Depth:  0 or 1 (simple)
               Value:  .ng999 through 999 (but not 0)
 Result Domain:
               Type:   Numeric
               Rank:   1 or 2
               Shape:  Vector or matrix (n by 2)
               Depth:  1 (simple)
 Implicit Arguments:  None

 Clears event flags associated with one or more
 channels. Returns the previous settings of the event flags.

3 Errors
 RANK ERROR (NOT VECTOR DOMAIN)
 The argument is not a singleton and its rank is greater
 than 1.

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 The value specified for chans must be a simple array.

 DOMAIN ERROR (NOT AN INTEGER)
 The channel number is not an integer.

 DOMAIN ERROR (INVALID CHANNEL NUMBER)
 The value specified for chans is outside the argument domain.

 LIMIT ERROR (INTEGER TOO LARGE)
 The argument is outside the range of valid integer values.

2 .bxEFR
 .bxEFR - Event Flag Read
 Type:  Monadic System Function
 Form:  event-flag-values _ .bxEFR chans
 Argument Domain:
               Type:   Near-integer
               Shape:  Vector domain
               Depth:  0 or 1 (simple)
               Value:  .ng999 through 999 (but not 0)
 Result Domain:
               Type:   Numeric
               Rank:   1 or 2
               Shape:  Vector or matrix (n by 2)
               Depth:  1 (simple)
 Implicit Arguments:  None

 Returns the setting for event flags on one or
 more channels.

3 Errors
 RANK ERROR (NOT VECTOR DOMAIN)
 The argument is not a singleton and its rank is greater
 than 1.

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 The value specified for chans must be a simple array.

 DOMAIN ERROR (NOT AN INTEGER)
 The channel number is not an integer.

 DOMAIN ERROR (INVALID CHANNEL NUMBER)
 The value specified for chans is outside the argument domain.

 LIMIT ERROR (INTEGER TOO LARGE)
 The argument is outside the range of valid integer values.

2 .bxEFS
 .bxEFS - Event Flag Set
 Type:  Monadic System Function
 Form:  previous-values _ .bxEFS chans
 Argument Domain:
               Type:   Near-integer
               Shape:  Vector domain
               Depth:  0 or 1 (simple)
               Value:  .ng999 through 999 (but not 0)
 Result Domain:
               Type:   Numeric
               Rank:   1 or 2
               Shape:  Vector or matrix (n by 2)
               Depth:  1 (simple)
 Implicit Arguments:  None

 Sets event flags associated with one or more channels.
 Returns the previous settings of the event flags.

3 Errors
 RANK ERROR (NOT VECTOR DOMAIN)
 The argument is not a singleton and its rank is greater
 than 1.

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 The value specified for chans must be a simple array.

 DOMAIN ERROR (NOT AN INTEGER)
 The channel number is not an integer.

 DOMAIN ERROR (INVALID CHANNEL NUMBER)
 The value specified for chans is outside the argument domain.

 LIMIT ERROR (INTEGER TOO LARGE)
 The argument is outside the range of valid integer values.

2 .bxERROR
 .bxERROR - Error Message
 Type:   System Variable
 Forms:  most-recent-error _ .bxERROR
         .bxERROR _ error-text
 Value Domain:
```

```
               Type:     Character
               Shape:    Vector domain
               Depth:    0 or 1 (simple)
               Default:  ''
Result Domain:
               Type:   Character
               Rank:   1 (vector)
               Shape:  Vector
               Depth:  1 (simple vector)

Character vector that describes the last error to
occur. .bxERROR is set implicitly by the system when an
error occurs, but can also be set by the user.

3 Errors
RANK ERROR (NOT VECTOR DOMAIN)
The value is not a singleton and its rank is greater than
1. For example: .bxERROR_2 2.ro4 or .bxERROR_2 2.ro'A' is
incorrect.

DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
The value is not a simple, homogeneous array.

DOMAIN ERROR (INCORRECT TYPE)
The value is non-empty and numeric. For example:
.bxERROR_10 is incorrect.

2 .bxEX
.bxEX - Erasing a Named Object
Type:  Monadic System Function
Form:  erased/not-erased _ .bxEX name-list
Argument Domain:
               Type:   Character
               Shape:  Matrix domain
               Depth:  1 (simple)
Result Domain:
               Type:   Boolean
               Rank:   1 (vector)
               Shape:  .roname-list
               Depth:  1 (simple vector)
Implicit Arguments:  None

Expunges existing use of a name in the workspace.
Returns a vector of Booleans that indicate which objects
were erased: a 1 means that the object was erased; a
0 means that the object cannot be erased.

The argument name-list is in the character matrix domain
with 1 name per row.

3 Errors
RANK ERROR (NOT MATRIX DOMAIN)
The argument is not in the matrix domain.

DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
The argument is not a simple, homogeneous array.

DOMAIN ERROR (INCORRECT TYPE)
The argument is non-empty and numeric.

2 .bxEXP
.bxEXP - Expansion
Type:  Dyadic System Function
Form:  A .bxEXP B   A .bxEXP[K] B
Argument Domain:
      Left
               Type:   Near-Boolean
```

```
                Shape:  Vector domain
                Depth:  0 or 1 (simple)
       Right
                Type:   Any
                Shape:  Any
                Depth:  Any
 Result Domain:
                Type:   Same as argument
                Rank:   1.ce.ro.roB
                Shape:  (K-1)^.roB),(.ro,A),K.da.roB
                        (for .bxIO 1)
                Depth:  1.ce.mtB
 Implicit Arguments:  None
```

.bxEXP builds an array by combining the items of an
existing array with fill items.

.bxEXP works the same as the expansion derived function. See Expand
function (Type)HELP FUNCTION-NAMES EXPAND for more information.)

3 Errors

AXIS RANK ERROR (NOT VECTOR DOMAIN)
K is not a singleton and its rank is greater than 1.

AXIS LENGTH ERROR (NOT SINGLETON)
K is not a singleton.

AXIS DOMAIN ERROR (SEMICOLON LIST NOT ALLOWED)
There is a semicolon inside of the brackets that surround
K.

AXIS DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
K must be a simple array.

AXIS DOMAIN ERROR (INCORRECT TYPE)
K is not numeric.

AXIS DOMAIN ERROR (NOT AN INTEGER)
K is not a near-integer.

AXIS DOMAIN ERROR (AXIS LESS THAN INDEX ORIGIN)
K is less than .bxIO.

AXIS DOMAIN ERROR (RIGHT ARGUMENT HAS WRONG RANK)
K is greater than the rank of B.

RANK ERROR (NOT VECTOR DOMAIN)
A is not a singleton and its rank is greater than 1.

LENGTH ERROR
B is not a singleton and its length along the Kth axis
is not equal to the number of 1's in A.

DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
A is not a simple, homogeneous array.

DOMAIN ERROR (INCORRECT TYPE)
A is not empty and not numeric.

DOMAIN ERROR
A is not Boolean.

LIMIT ERROR (INTEGER TOO LARGE)
A or K is greater than the largest allowable integer.

2 .bxFI

.bxFI - Converting Characters to Numerics

```
 Type:  Monadic System Function
 Form:  numeric-values _ .bxFI numeric-character-string
 Argument Domain:
                Type:   Character
                Shape:  Vector domain
                Depth:  0 or 1 (simple)
 Result Domain:
                Type:   Numeric
                Rank:   1 (vector)
                Shape:  Vector
                Depth:  1 (simple vector)
 Implicit Arguments:  .bxNG (determines minus sign placement)

 Converts character argument to numeric, placing
 0s in each position not corresponding to
 a valid number.

3 Errors
 RANK ERROR (NOT MATRIX DOMAIN)
 The argument is not in the matrix domain.

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 The argument is not a simple, homogeneous array.

 DOMAIN ERROR (INCORRECT TYPE)
 The argument is non-empty and numeric.

2 .bxFLS
 .bxFLS - Returning File Information
 Type:  Monadic System Function (query)
 Form:  file-info _ .bxFLS chans
 Argument Domain:
                Type:   Near-integer
                Shape:  Vector domain
                Depth:  0 or 1 (simple)
                Value:  .ng999 through 999 (but not 0)
 Result Domain:
                Type:   Integer
                Rank:   1 or 2
                Shape:  Vector or matrix (n by 5)
                Depth:  1 (simple)
 Implicit Arguments:  None

 Returns information about files on one or more
 channels. The result contains one row of five values
 for each channel specified in the argument. The meanings
 of the values differ according to each file's organization.

 The values returned by .bxFLS have the following meanings
 (from left to right):

 First value:  Share bit: 1 means that you specified /SHARE
               in the argument for the associated .bxASS
               function; 0 means that you did not.

 Second value: For sequential files, the second value is
               the number of records read or written since
               the file was opened. For direct-access and
               relative files, it is the value of the last
               record or component number used for a successful
               read or write. For keyed files, the second value
               is always 0.

 Third value:  The maximum record size of the file (0 means
               there is no user limit on record size).

 Fourth value: The /BLOCKSIZE setting for the file.
 Fifth value:  The type of the most recent I/O operation. You can
               use this information in determining the location of
               the next record pointer. There are six possible I/O
               operations:

               Value Returned     I/O Operation

                    1             Sequential read
                    2             Random read
                    3             Sequential write
                    4             Random write
                    5             Sequential delete
                    6             Random delete

3 Errors
 RANK ERROR (NOT VECTOR DOMAIN)
 The argument is not a singleton and its rank is greater
 than 1.

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 The value specified for chans must be a simple array.

 DOMAIN ERROR (NOT AN INTEGER)
 The channel number is not an integer.

 DOMAIN ERROR (INVALID CHANNEL NUMBER)
 The value specified for chans is outside the argument
 domain.

 LIMIT ERROR (INTEGER TOO LARGE)
 The argument is outside the range of valid integer
 values.

2 .bxFMT
 .bxFMT - The Report Formatter
 Type:  Dyadic System Function
 Form:  report _ format-phrases .bxFMT {array | (array ; array ;...)}
 Argument Domain:
        Left
                Type:   Character
                Shape:  Vector domain
                Depth:  1 (simple)
        Right
                Type:   Any
                Shape:  Any
                Depth:  .le2 (vector of arrays or a simple array)
 Result Domain:
                Type:   Character
                Rank:   2
                Shape:  Matrix
                Depth:  1 (simple matrix)
 Implicit Arguments:  .bxNG (determines minus sign placement)

 Converts argument to character matrix in designated format.

 To obtain more help type )HELP QUAD-NAMES .bxFMT PARAMETERS
                          )HELP QUAD-NAMES .bxFMT QUALIFIERS-DECORATORS
                          )HELP QUAD-NAMES .bxFMT SYNTAX
                          )HELP QUAD-NAMES .bxFMT VALID-COMBINATIONS
3 Parameters

 Summary of Format Phrase Parameters

 Parameter                     Meaning

 rep           The number of consecutive target columns to be
```

	affected by the format phrase, or the number of times to repeat a parenthesized group of format phrases (to a maximum of (2*16) - 2).
quals	One or more of the format phrase qualifiers or decorators specified under Summary of Format Phrase Qualifiers and Decorators.
width	The width in the result array of the formatted value from the target column in the right argument. The width must be an integer in the range 1 through 255.
dig	The number of decimal places (F, or fixed-point, format) or significant digits (E, or floating-point with exponent, format) to be included in the result array. The digit parameter's value must be an integer in the range 0 through 127.
col	For the T (absolute tab) format, an integer in the range 1 through 255 that identifies the leftmost column that the next formatted value is to occupy in the result array. For the X (relative tab) format, an integer in the range .ng255 through 255 that identifies the number of columns to shift before outputting the next formatted value.

3 Qualifier-Decorators

Summary of Format Phrase Qualifiers and Decorators

Qualifiers	Meaning
B	For types I, E, F, G, and Y, if the value of the item in the target column is zero, make the fields in the target column blank in the result array.
C	For types I and F, insert commas between each group of three digits in the integer part of the formatted value.
L	For types I, F, A, E and Y, left-justify the fields in the target column.
Kn	For types I, F, G, and E, before formatting the fields in the target column, multiply the fields by the scale factor 10*n.
S"symbol pairs"	For types I, E, F, G, and Y, replace, in the formatted output, all occurrences of the first character in each symbol pair with the corresponding second character of the symbol pair.
Wn	For type E, use n exponent digits in the formatted output.
Z	For types I, F, and Y, fill leading blanks in the formatted output with zeros.

Decorators

M"text"	For types I, F, and G, replace the sign of negative formatted values with 'text' placed to the left of the value.
N"text"	For types I, F, and G, place 'text' to the right of negative formatted values.

O"text"	For types I, F, E, G, and Y, replace formatted zero values with 'text'.
P"text"	For types I, F, and G, place 'text' to the left of positive formatted values.
Q"text"	For types I, F, and G, place 'text' to the right of positive formatted values.
R"text"	For types I, F, E, A, G, and Y, fill unused columns in the formatted output with 'text'.

Note that the delimiting pair " " may also be any of the following pairs:

```
.qq .qq
.dd .dd
.bx .bx
<  >
.lu .ru
```

3 Syntax

Summary of Format Phrase Syntax

Phrase	Type of Data
[rep][quals]Awidth	Character
[rep][quals]Ewidth.dig	Floating-point with Exponent
[rep][quals]Fwidth.dig	Fixed-point
[rep][quals]G"pattern"	Picture
[rep][quals]Iwidth	Integer
[rep]T[col]	Absolute Tab
[rep]X[col]	Relative Tab
[rep][quals]Ywidth	Byte (Hex)
[rep]"text"	Literal

Note that the delimiting pair " " may also be any of the following pairs:

```
.qq .qq
.dd .dd
.bx .bx
<  >
.lu .ru
```

3 Valid-Combinations

Format phrase	Qualifiers							Decorators						Parameters			
	B	C	L	Kn	Wn	S	Z	M	N	O	P	Q	R	w	d	r	c
A			x										x	+		0+	
Y	x		x			x	x			x			x	+		0+	
I	x	x	x	x		x	x	x	x	x	x	x	x	+		0+	
F	x	x	x	x		x	x	x	x	x	x	x	x	+	+	0+	
E	x		x	x	x	x	x			x			x	+	+	0+	
G	x			x		x		x	x	x	x	x	x			0+	
literal																0+	
T																0+	0+
X																0+	-0+

x	means the qualifier or decorator is permitted with the format phrase type.
+	means a value must be specified and must be a positive integer.

0+   means a value is optional but if specified must be
      a positive integer or zero.

-0+   means a value is optional but if specified must be
      an integer.

3 Errors

RANK ERROR (NOT VECTOR DOMAIN)
A is not a singleton and its rank is greater than 1.

LENGTH ERROR
A is empty.

DEPTH ERROR
There are more than eight nested parentheses in A.

DOMAIN ERROR (RIGHT ARG TOO DEEPLY NESTED)
The right argument is not a vector domain of simple arrays.

DOMAIN ERROR (ENCLOSED ARRAY IS NOT ALLOWED)
The left argument is enclosed.

DOMAIN ERROR (INCORRECT TYPE)
A is not type character.

DOMAIN ERROR (PARAMETER OUT OF RANGE)
The repetition count, field width, number of decimal places or significant digits, column position, scale factor, or exponent size is out of range.

DOMAIN ERROR (ILLEGAL USE OF FMT QUALIFIER)
The specified qualifier and format phrase are incompatible.

DOMAIN ERROR (DUPLICATE FMT QUALIFIER)
A qualifier is used more than once with a particular format phrase.

DOMAIN ERROR (ILLEGAL CHARACTER IN FMT LEFT ARGUMENT)
An invalid character appears in A.

DOMAIN ERROR (ILL FORMED FMT PARAMETER)
An invalid numeric parameter (such as a negative sign with no number) was found.

DOMAIN ERROR (ILLEGAL FMT FORMAT PHRASE)
A letter in A does not represent a valid format phrase or qualifier.

DOMAIN ERROR (ILLEGAL FMT LITERAL STRING DELIMITER)
A decorator or literal string delimiter was invalid (see Table 5-1 for a list of the valid delimiters).

DOMAIN ERROR (MISSING FMT FORMAT PHRASE/QUALIFIER CHARACTER)
A format phrase or qualifier was expected but not supplied.

DOMAIN ERROR (MISSING FMT FORMAT PHRASE/QUALIFIER PARAMETER)
No string was included with a decorator or an S format phrase; no number was included where a width or decimal parameter was required; or no number was included with a K or W qualifier.

DOMAIN ERROR (UNPAIRED SYMBOL IN FMT S QUALIFIER)
The length of the standard symbol substitution string is not even.

DOMAIN ERROR (EMPTY FMT STRING PARAMETER NOT ALLOWED)

```
 The O, R, or S qualifier string is empty.

 DOMAIN ERROR (ILLEGAL FMT S QUALIFIER SYMBOL)
 The first symbol of a substitution pair is
 not * . , 0 9 Z or @

 DOMAIN ERROR (MISSING FMT FORMAT PHRASE SEPARATOR)
 A format phrase separator (such as a comma or parenthesis)
 was expected but not supplied.

 DOMAIN ERROR (MISSING LITERAL STRING IN FMT LEFT ARGUMENT)
 The text string parameter was missing from a decorator.

 DOMAIN ERROR (UNBALANCED TEXT DELIMITER IN FMT LEFT ARGUMENT)
 The closing delimiter for a text string was not
 compatible with the opening delimiter.

 DOMAIN ERROR (UNBALANCED PARENS IN FMT LEFT ARGUMENT)
 The parentheses in A are not nested properly.

 DOMAIN ERROR (FMT DECORATION OR LITERAL STRING TOO LONG)
 A text string in A consists of more than 255 characters.

 DOMAIN ERROR (DUPLICATE FMT STANDARD SUBSTITUTION CHARACTER)
 A substitute for a standard symbol character was
 specified more than once.

 DOMAIN ERROR (NO FMT EDITING FORMAT PHRASE)
 A does not contain at least one value editing format
 phrase, that is, at least one of type A, I, E, F, G, or Y.

 DOMAIN ERROR (ILLEGAL FMT G FORMAT PHRASE PATTERN CHARACTER)
 An invalid character was found in a type G format phrase pattern string.

 DOMAIN ERROR (FMT RIGHT ARGUMENT DOES NOT MATCH FORMAT PHRASE)
 The data type of a value in B does not match the type
 called for by the format phrase specification in A.

 DOMAIN ERROR (NO DIGIT SELECTOR IN FMT G FORMAT PHRASE PATTERN)
 A type G format phrase pattern does not contain at least one
 9 or one Z, or a character which is substituted for a 9 or a Z.

 LIMIT ERROR (FLOATING OVERFLOW)
 The K scaling factor causes a value to exceed the
 largest representable number

2 .bxFX
 .bxFX - Establishing an Operation
 Type:  Monadic System Function
 Form:  operation-name _ .bxFX operation-definition
 Argument Domain:
               Type:   Character
               Shape:  Matrix domain
               Depth:  1 (simple)
 Result Domain:
               Type:   Character (Numeric if error is detected)
               Rank:   0 or 1
               Shape:  Vector (Scalar if error is detected)
               Depth:  0 or 1 (simple)
 Implicit Arguments:  .bxIO (controls origin of line number in error)

 Establishes an operation from its canonical representation.
 If the operation cannot be established, .bxFX returns the
 line number where the error occurred and puts a message in
 .bxERROR.

3 Errors
```

```
 RANK ERROR (NOT MATRIX DOMAIN)
 The shape of the argument is not in the matrix domain.

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 The argument is not a simple, homogeneous array.

 DOMAIN ERROR (INCORRECT TYPE)
 The argument is nonempty and numeric.

2 .bxGAG
 .bxGAG - Preventing Interruptions
 Type:   System Variable
 Forms:  .bxGAG _ near-integer-singleton
         integer-scalar _ .bxGAG
 Value Domain:
                Type:     Integer
                Shape:    Singleton
                Depth:    0 or 1 (simple)
                Value:    0, 1, 2, or 3
                Default:  Determined when APL is invoked
 Result Domain:
                Type:   Integer
                Rank:   0 (scalar)
                Shape:  .io0 (scalar)
                Depth:  0 (simple scalar)

 Indicates whether to accept messages sent from
 other users:

     .bxGAG _ 0   means accept (default)
     .bxGAG _ 1   means refuse
     .bxGAG _ 2   means trap, translate, and display messages
     .bxGAG _ 3   means signal BROADCAST RECEIVED. The text of
                  the broadcast is the secondary error message
                  in .bxERROR.

 If your terminal is an APL terminal, then .bxGAG_2 when APL
 is invoked.

3 Errors
 RANK ERROR (NOT VECTOR DOMAIN)
 The value is not a singleton and its rank is greater than
 1. For example: .bxGAG_2 2.ro4 is incorrect.

 LENGTH ERROR (NOT SINGLETON)
 The value is not a single item. For example:
 .bxGAG_.io3 is incorrect.

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 The value is not a simple, homogeneous array.

 DOMAIN ERROR (INCORRECT TYPE)
 The value is non-empty and character. For example:
 .bxGAG_'A' is incorrect.

 DOMAIN ERROR (NOT AN INTEGER)
 The value is not near-integer. For example: .bxGAG_2.5
 is incorrect.

 LIMIT ERROR (INTEGER TOO LARGE)
 The value is greater than the largest allowable integer.
 For example: .bxGAG_.fl2*33 is incorrect.

 DOMAIN ERROR (SYSTEM VARIABLE MUST BE 0 OR 1 OR 2 OR 3)

2 .bxIO
 .bxIO - Index Origin
 Type:   System Variable
 Forms:  .bxIO _ near-integer-singleton
         integer-scalar _ .bxIO
 Value Domain:
                Type:     Integer
                Shape:    Singleton
                Depth:    0 or 1 (simple)
                Value:    0 or 1
                Default:  1
 Result Domain:
                Type:   Integer
                Rank:   0 (scalar)
                Shape:  .io0 (scalar)
                Depth:  0 (simple scalar)
```

Sets index origin for arrays and axis.

3 Errors

RANK ERROR (NOT VECTOR DOMAIN)
The value is not a singleton and its rank is greater than 1. For example: .bxIO_2 2.ro4 is incorrect.

LENGTH ERROR (NOT SINGLETON)
The value is not a single item. For example: .bxIO_.io3 is incorrect.

DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
The value is not a simple, homogeneous array.

DOMAIN ERROR (INCORRECT TYPE)
The value is non-empty and character. For example: .bxIO_'G' is incorrect.

DOMAIN ERROR (NOT AN INTEGER)
The value is not near-integer. For example: .bxIO_.5 is incorrect.

LIMIT ERROR (INTEGER TOO LARGE)
The value is greater than the largest allowable integer. For example: .bxIO_.fl2*33 is incorrect.

DOMAIN ERROR (SYSTEM VARIABLE VALUE MAY ONLY BE 0 OR 1)

2 .bxL

```
 .bxL - Watched Variable Name
 Type:   System Variable
 Forms:  .bxL _ any-value
         variable-name _ .bxL
 Value Domain:
                Type:   Any
                Shape:  Any
                Depth:  Any
 Result Domain:
                Type:     Character (any when set by user)
                Rank:     1 (vector) (any when set by user)
                Shape:    Vector (any when set by user)
                Depth:    1 (simple vector) (any when set by user)
                Default:  ''
```

A variable that is used implicitly by .bxWATCH. .bxL contains a character vector showing the name of a watched variable that has changed. .bxL is set implicitly by the system when a variable changes, but can also be set by the user.

3 Errors

No errors generated

```
2 .bxLC
 .bxLC - Line Counter
 Type:  Niladic System Function
 Form:  current-line-number _ .bxLC
 Result Domain:
                Type:   Integer
                Rank:   1 (vector)
                Shape:  Vector
                Depth:  1 (simple vector)
 Default Value: Empty

 Vector of line numbers in the state indicator;
 most recently suspended operation appears first.

 Typing .go.bxLC restarts the most recently suspended
 operation at the beginning of the line where execution
 was stopped.

3 Errors
 No errors generated

2 .bxLX
 .bxLX - Latent Expression
 Type:   System Variable
 Forms:  .bxLX _ character-vector
         current-value _ .bxLX
 Value Domain:
                Type:     Character
                Shape:    Vector domain
                Depth:    0 or 1 (simple)
                Default:  ''
 Result Domain:
                Type:   Character
                Rank:   1 (vector)
                Shape:  Vector
                Depth:  1 (simple vector)

 Causes expression to be executed automatically
 when workspace is loaded.

 The expression is not executed when you load the
 workspace with the )XLOAD system command.

3 Errors
 RANK ERROR (NOT VECTOR DOMAIN)
 The value is not a singleton and its rank is greater than 1.
 For example: .bxLX_2 2.ro4 or .bxLX_2 2.ro'A' is incorrect.

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 The value is not a simple, homogeneous array.

 DOMAIN ERROR (INCORRECT TYPE)
 The value is non-empty and numeric. For example: .bxLX_10
 is incorrect.

2 .bxMAP
 .bxMAP - Defining External Routines to APL
 Type:   Ambivalent System Function
 Forms:  description _ .bxMAP external-routine-name
         defined-function _ function-header .bxMAP shared-image-info
 Monadic Argument Domain:
                Type:   Character
                Shape:  Vector domain
                Depth:  0 or 1 (simple)
 Dyadic Argument Domain:
        Left
                Type:   Character
                Shape:  Vector domain
                Depth:  0 or 1 (simple)
        Right
                Type:   Character
                Shape:  Vector domain
                Depth:  0 or 1 (simple)
 Result Domain:
                Type:   Character
                Rank:   1 (vector)
                Shape:  Vector
                Depth:  1 (simple vector)
 Implicit Arguments:  None


 The dyadic form associates an external routine with a
 user-defined function.

 The left argument is a function header describing the
 external routine. Its form is:

       result/att _ function-name arg1/att arg2/att...

       where the possible attributes specified for /att are:

           /ACCESS: {IN | INOUT | OUT}

           /TYPE:vms-data-type

           /MECHANISM: {IMMEDIATE | REFERENCE | DESCRIPTOR}

 The right argument is the name of a shared image and its entry point.
 Its form is:

       {vms-filename | vms-logical-name} {/ENTRY | /VALUE}:symbol

 Use /ENTRY to specify the name of the entry point in the shared image.
 Use /VALUE to specify the name of a global constant in the shared image.
 Use symbol as the name of the entry point or the global constant.

 Type )HELP /TYPE  to see the values of vms-data-type.

3 Errors
 ERROR INVOKING EXTERNAL ROUTINE (WRONG NUMBER OF ARGUMENTS
 TO USER FUNCTION) More actual arguments were specified
 than there are formal parameters defined in the formal
 parameters of the external routine.

 ERROR INVOKING EXTERNAL ROUTINE (ERROR ACTIVATING IMAGE)
 The shared image that contains the external routine no
 longer exists or is unavailable for some reason.

 ERROR INVOKING EXTERNAL ROUTINE (KEY NOT FOUND IN TREE)
 The symbol that was specified for either the /ENTRY or
 /VALUE qualifier does not exist in the shared image.

 ERROR INVOKING EXTERNAL ROUTINE (NOT VECTOR DOMAIN)
 The actual parameter specified for either the /ACCESS:OUT
 or /ACCESS:INOUT qualifier is not in the vector domain.

 ERROR INVOKING EXTERNAL ROUTINE (INCORRECT TYPE)
 The actual parameter specified for either the /ACCESS:OUT
 or /ACCESS:INOUT qualifier is not character.

 ERROR INVOKING EXTERNAL ROUTINE (ILL FORMED NAME)
 The actual parameter specified for either the /ACCESS:OUT
 or /ACCESS:INOUT qualifier is not a valid APL name.
```

ERROR INVOKING EXTERNAL ROUTINE (EXTRANEOUS CHARACTERS AFTER COMMAND) The actual parameter specified for either the /ACCESS:OUT or /ACCESS:INOUT qualifier is followed by nonwhite space.

ERROR INVOKING EXTERNAL ROUTINE (INCORRECT PARAMETER) One of the following situations has occurred: The actual parameter specified for either the /ACCESS:OUT or /ACCESS:INOUT qualifier is currently undefined and is /TYPE:Z (the parameter must either be defined so an unconverted value can be passed or undefined with a known data type, not /TYPE:Z); or the actual argument is missing when the formal parameter was specified with the /MECHANISM:IMMEDIATE qualifier.

ERROR INVOKING EXTERNAL ROUTINE (ILLEGAL NAME CLASS) The actual parameter specified for either the /ACCESS:OUT or /ACCESS:INOUT qualifier is defined, but is not a variable.

ERROR INVOKING EXTERNAL ROUTINE (NOT SINGLETON) The actual argument is not a singleton (as it should be) when dyadic .bxMAP is specified with the /MECHANISM:IMMEDIATE qualifier.

ERROR INVOKING EXTERNAL ROUTINE (LENGTH ERROR) One of the following situations has occurred: The actual argument has a length greater than 4 bytes when dyadic .bxMAP was specified with the /MECHANISM:IMMEDIATE qualifier; The actual argument has a length greater than 2*16 when dyadic .bxMAP was specified with the /MECHANISM:DESCRIPTOR qualifier; A complex data type is being passed as an even number of items (APL requires two numbers to form each complex number); or the length of a Varying sTring (/TYPE:VT) is greater than .ng1+2*16.

ERROR INVOKING EXTERNAL ROUTINE (DOMAIN ERROR) One of the following situations has occurred: The data leaving the workspace cannot be converted to the data type expected by the external routine (for example, numbers could not be converted to /TYPE:T); or a conversion failed as data passed from the workspace to the external routine.

ERROR INVOKING EXTERNAL ROUTINE (ILLEGAL ASCII CHARACTER) A conversion to ASCII failed as character data (/TYPE:T or /TYPE:VT) left the workspace.

SIGNAL FROM EXTERNAL ROUTINE (xxx) An error occurred in the external routine. The value for xxx is the error signaled.

ERROR RETURNING FROM EXTERNAL ROUTINE (LENGTH ERROR) A Varying sTring (/TYPE:VT) returned to the WS is bigger than it was when it was passed to the external routine. (It is allowed to be smaller or the same size.)

ERROR RETURNING FROM EXTERNAL ROUTINE (ILLEGAL ASCII CHARACTER) A conversion to ASCII failed as character data returned to the workspace.

ERROR RETURNING FROM EXTERNAL ROUTINE (DOMAIN ERROR) A conversion failed when data returned to the workspace.

2 .bxMBX

.bxMBX - Mailbox
Type: Monadic System Function
Form: mailbox-info _ .bxMBX chans
Argument Domain:

```
               Type:   Near-integer
               Shape:  Vector domain
               Depth:  0 or 1 (simple)
               Value:  .ng999 through 999 (but not 0)
Result Domain:
               Type:   Integer
               Rank:   1 or 2
               Shape:  Vector or matrix (n by 3)
               Depth:  1 (simple)
Implicit Arguments:  None

Returns information about mailboxes on one or
more channels.

For each channel specified, .bxMBX returns a row of
three elements denoting (from left to right):

    1. The physical device number assigned to the mailbox
       (or 0 if the mailbox is remote, and .ng1 if the channel
       is not associated with a mailbox).

    2. The process identification number (PID) of the last user
       to receive a message you sent to the mailbox (or .ng1 if
       no messages have been sent).

    3. The PID of the last user from which you received a message
       in the mailbox (or .ng1 if no messages have been received).

3 Errors
 RANK ERROR (NOT VECTOR DOMAIN)
 The argument is not a singleton and its rank is greater
 than 1.

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 The value specified for chans must be a simple array.

 DOMAIN ERROR (NOT AN INTEGER)
 The channel number is not an integer.

 DOMAIN ERROR (INVALID CHANNEL NUMBER)
 The value specified for chans is outside the argument
 domain.

 LIMIT ERROR (INTEGER TOO LARGE)
 The argument is outside the range of valid integer
 values.

2 .bxMONITOR
 .bxMONITOR - Gathering Data on Operations
 Type:   Ambivalent System Function
 Forms:  success/failure _ line-numbers .bxMONITOR operation-names
         monitor-database _ .bxMONITOR operation-name
 Argument Domain:
        Left
                Type:   Near-integer
                Shape:  Vector domain
                Depth:  0 or 1 (simple)
        Right
                Type:   Character
                Rank:   1 or 2
                Shape:  Matrix domain (n by 3)
                Depth:  0 or 1 (simple)
 Result Domain:
                Type:   Integer (dyadic) or Boolean (monadic)
                Rank:   1 or 2
                Shape:  Vector or matrix
                Depth:  1 (simple)
```

Implicit Arguments: None

Gathers information about operation execution
counts and CPU times. The dyadic form enables and
disables monitoring. The monadic form queries for
any collected data.

The result of the dyadic form is a Boolean vector where
each position corresponds to a row of the right argument:
a 1 indicates successful enabling; a 0 indicates unsuccessful.

An empty left argument disables monitoring on the operations
listed in the right argument. A value of 0 in the left argument
means to monitor the entire operation.

The monadic form queries for any collected data. You can
only query for information on one operation at a time. The
result is an n-by-3 matrix where n is the number of
monitored lines and the 3 columns represent the following:

```
      Line-number   Execution-count   CPU-time-in-milliseconds
```

3 Errors
For the dyadic form:

RANK ERROR (NOT MATRIX DOMAIN)
The right argument is not in the matrix domain.

RANK ERROR (NOT VECTOR DOMAIN)
The left argument is not a singleton and its rank is
greater than 1.

DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
The arguments must be simple, homogeneous arrays.

DOMAIN ERROR (INCORRECT TYPE)
The right argument must be of type character, and the
left argument must be near-integer.

DOMAIN ERROR (NOT AN INTEGER)
The left argument is not near-integer.

LIMIT ERROR (INTEGER TOO LARGE)
The argument is greater than the largest allowable integer.

For the monadic form:

RANK ERROR (NOT MATRIX DOMAIN)
The argument is not in the matrix domain.

LENGTH ERROR
There is more than one row specified in the argument.

DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
The argument is not a simple, homogeneous array.

DOMAIN ERROR (INCORRECT TYPE)
The argument is not of type character.

2 .bxNC
.bxNC - Returning a Name Classification
Type: Monadic System Function
Form: name-class-list _ .bxNC name-list
Argument Domain:
```
               Type:   Character
               Shape:  Matrix domain
               Depth:  0 or 1 (simple)
```

```
 Result Domain:
                Type:   Integer
                Rank:   1 (vector)
                Shape:  1^.roname-list
                Depth:  1 (simple vector)
 Implicit Arguments:  None

 Returns the classification of one or more names.
 The argument name-list is in the character matrix domain
 with 1 row per name.

 The possible name classes are shown below.

 .bxNC Name Classes

 Value            Meaning

  .ng20         Derived function
  .ng5            Niladic system function
  .ng4            Group
  .ng3            Monadic, Dyadic, or Ambivalent system function
  .ng2            System variable
  .ng1            Ill-formed identifier
   0        Name not in use
   1        Label name
   2        Variable name
   3        User-defined function name
   4        User-defined operation name

3 Errors
 RANK ERROR (NOT MATRIX DOMAIN)
 The argument is not in the matrix domain.

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 The argument is not a simple, homogeneous array.

 DOMAIN ERROR (INCORRECT TYPE)
 The argument is non-empty and numeric.

2 .bxNG
 .bxNG - Print High Minus
 Type:   System Variable
 Forms:  .bxNG _ near-integer-singleton
         integer-scalar _ .bxNG
 Value Domain:
                Type:     Integer
                Shape:    Singleton
                Depth:    0 or 1 (simple)
                Value:    0, 1, or 2
                Default:  1 (high minus sign)
 Result Domain:
                Type:   Integer
                Rank:   0 (scalar)
                Shape:  .io0 (scalar)
                Depth:  0 (simple scalar)

 Controls the printing of the negative sign in .fm and .bxFMT
 and the recognition of negative numbers in .bxVI and .bxFI.

     .bxNG _ 0   means use the APL minus sign.
     .bxNG _ 1   means use the APL high minus sign. (default)
     .bxNG _ 2   means use APL '+' which is ASCII '-'.

 When .bxNG _ 2, negative numbers are preceded by an APL '+' symbol
 when formatted by .fm and .bxFMT. APL '+' prints as an ASCII '-' so
 .bxNG _ 2 can be used to handle negative numbers in strings that
 will be read or written in ASCII. Note that .bxFI and .bxVI
```

recognize negative numbers that are preceded by an APL '+' symbol
as negative numbers (when .bxNG = 2).

3 Errors

RANK ERROR (NOT VECTOR DOMAIN)
The value is not a singleton and its rank is greater than
1. For example: .bxNG_2 2.ro4 is incorrect.

LENGTH ERROR (NOT SINGLETON)
The value is not a single item. For example:
.bxNG_.io3 is incorrect.

DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
The value is not a simple, homogeneous array.

DOMAIN ERROR (INCORRECT TYPE)
The value is non-empty and character. For example:
.bxNG_'A' is incorrect.

DOMAIN ERROR (NOT AN INTEGER)
The value is not near-integer. For example: .bxNG_2.5
is incorrect.

LIMIT ERROR (INTEGER TOO LARGE)
The value is greater than the largest allowable integer.
For example: .bxNG_.fl2*33 is incorrect.

DOMAIN ERROR (PARAMETER OUT OF RANGE)
An attempt was made to use an unavailable value as the value.
For example: .bxNG_10 is incorrect.

2 .bxNL

```
.bxNL - Constructing a List of Names
Type:  Ambivalent System Function
Form:  name-list _ letter-list .bxNL name-classes
Argument Domain:
       Left
               Type:   Character
               Shape:  Vector domain
               Depth:  0 or 1 (simple)
       Right
               Type:   Near-integer
               Shape:  Vector domain
               Depth:  0 or 1 (simple)
Result Domain:
               Type:   Character
               Rank:   2 (matrix)
               Shape:  Matrix
               Depth:  1 (simple matrix)
Implicit Arguments:  None
```

Lists the names of all existing APL objects belonging to
the name classes specified in the right argument. The
possible values for the right argument and the classes
represented are as follows:

Value	Names Returned
.ng5	Niladic system functions
.ng4	Groups
.ng3	Monadic, Dyadic, and Ambivalent system functions
.ng2	System variables
1	Labels
2	User-defined variables
3	User-defined functions
4	User-defined operators

```
 The left argument is optional; it allows you to restrict
 the name list to names beginning with the specified characters.

3 Errors
 RANK ERROR (NOT VECTOR DOMAIN)
 An argument is not in the vector domain.

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 The arguments must be simple, homogeneous arrays.

 DOMAIN ERROR (NOT AN INTEGER)
 The right argument is not near-integer.

 LIMIT ERROR (INTEGER TOO LARGE)
 The right argument is greater than the largest allowable
 integer.

 DOMAIN ERROR (PARAMETER OUT OF RANGE)
 An attempt was made to use an unavailable value for the
 right argument.

 DOMAIN ERROR (INCORRECT TYPE)
 The left argument is non-empty and numeric.

 DOMAIN ERROR (NOT A LETTER)
 The left argument contain only the letters A...Z .ld .ud

2 .bxNUM
 .bxNUM - Digits
 Type:  Niladic System Function
 Form:  '0123456789' _ .bxNUM
 Result Domain:
               Type:   Character
               Rank:   1 (vector)
               Shape:  10
               Depth:  1 (simple vector)

 Vector of 10 characters; 0 through 9.
 .bxNUM is a subset of .bxAV.

3 Errors
 No errors generated

2 .bxOM
 .bxOM - Indexing a Boolean Vector
 Type:  Monadic System Function
 Form:  indexes _ .bxOM boolean-vector
 Right Argument Domain:
               Type:   Near-Boolean
               Shape:  Vector domain
               Depth:  0 or 1 (simple)
 Result domain:
               Type:   Integer
               Rank:   1 (vector)
               Shape:  +_/near-Boolean
               Depth:  1 (simple vector)
 Implicit Arguments:  None

 Returns the index of every occurrence of a 1 in a
 Boolean vector. Note that .bxOM is .bxIO dependent.

3 Errors
 RANK ERROR (NOT VECTOR DOMAIN)
 The argument is not a singleton and its rank is greater
 than 1.

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 The argument is not a simple, homogeneous array.

 DOMAIN ERROR (INCORRECT TYPE)
 The argument contains characters and is nonempty.

 DOMAIN ERROR
 The argument is not Boolean and is nonempty.

2 .bxPACK
 .bxPACK - Packing and Unpacking Data
 Type:   Ambivalent System Function
 Forms:  packed-data _ .bxPACK variable-names
         success/fail _ data-packets .bxPACK variable-names
 Monadic Argument Domain:
                Type:   Character
                Shape:  Matrix domain
                Depth:  0 or 1 (simple)
 Monadic Result Domain:
                Type:   Integer
                Rank:   1 (vector)
                Shape:  Vector
                Depth:  1 (simple vector)
 Dyadic Argument Domain:
     Left
                Type:   Near-integer
                Shape:  Vector domain
                Depth:  1 (simple)
     Right
                Type:   Character
                Shape:  Matrix domain
                Depth:  0 or 1 (simple)
 Dyadic Result Domain:
                Type:   Boolean
                Rank:   1 (vector)
                Shape:  Vector
                Depth:  1 (simple vector)
 Implicit Arguments:  None

 .bxPACK packs and unpacks data of different types into
 a single variable known as a packet. .bxPACK differs from
 .bxCOQ and .bxCIQ since it allows you to pack and unpack
 variables of different data types with only one invocation
 of the .bxPACK function.

 Monadic .bxPACK packs data. For a single variable, .bxPACK creates a
 .bxCOQ packet with a header, and does not perform any data type conversion
 before creating the packet. For more than one variable, .bxPACK creates
 individual .bxCOQ packets for each variable and combines them in a single
 logical record.

 Dyadic .bxPACK unpacks data. Name a packed object in the left argument,
 and supply variable names for the individual .bxCOQ packets in the
 right argument. The variable names must have one row for each individual
 packet in the left argument.

3 Errors
 RANK ERROR (MUST BE VECTOR)
 A must be in the vector domain.

 RANK ERROR (NOT MATRIX DOMAIN)
 B must be in the matrix domain.

 LENGTH ERROR (ITEM COUNT MISMATCH)
 The number of variable names specified in B must be equal
 to the number of packets contained in A.

 DOMAIN ERROR (ILLEGAL NAME CLASS)
 The right argument must be a variable.

 DOMAIN ERROR (ENCLOSED VALUE NOT ALLOWED)
 One of the names in the right argument has an
 enclosed array as its value.

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 The arguments must be simple, homogeneous arrays.

 DOMAIN ERROR (INCORRECT TYPE)
 An argument is non-empty and is either numeric, when it
 should be character, or character when it should be
 numeric.

 DOMAIN ERROR (NOT AN INTEGER)
 A must be a near-integer.

 LIMIT ERROR (INTEGER TOO BIG)
 A is greater than the largest allowable integer.

 DOMAIN ERROR (INVALID CIQ HEADER)

 DOMAIN ERROR (INVALID LENGTH IN PACK HEADER)
 The expression A[1] must equal .roA.

 DOMAIN ERROR (INVALID RANK IN PACK HEADER)
 The expression A[3] must equal 1. (1 means the packed
 data is a vector.)

 DOMAIN ERROR (INVALID RHO VECTOR IN PACK HEADER)
 The value of .roA must equal 4 + A[4].

 DOMAIN ERROR (INVALID TYPE IN PACK HEADER)
 The value of A[2] must equal 1 (1 means the type is integer).

2 .bxPP
 .bxPP - Print Precision
 Type:   System Variable
 Forms:  .bxPP _ digits-of-precision
         integer-scalar _ .bxPP
 Value Domain:
               Type:     Near-integer
               Shape:    Singleton
               Depth:    0 or 1 (simple)
               Value:    1 to 16
               Default:  10
 Result Domain:
               Type:   Integer
               Rank:   0 (scalar)
               Shape:  .io0
               Depth:  0 (simple scalar)

 Controls precision of numeric noninteger output.

3 Errors
 RANK ERROR (NOT VECTOR DOMAIN)
The value is not a singleton and its rank is greater than
 1. For example: .bxPP_2 2.ro4 is incorrect.

 LENGTH ERROR (NOT SINGLETON)
 The value is not a single item. For example:
 .bxPP.io3 is incorrect.

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 The value is not a simple, homogeneous array.

 DOMAIN ERROR (INCORRECT TYPE)
```

```
  The value is non-empty and character. For example:
  .bxPP_'A' is incorrect.

  DOMAIN ERROR (NOT AN INTEGER)
  The value is not near-integer. For example: .bxPP_2.5
  is incorrect.

  LIMIT ERROR (INTEGER TOO LARGE)
  The value is greater than the largest allowable integer.
  For example: .bxPP_.fl2*33 is incorrect.

  DOMAIN ERROR (PARAMETER OUT OF RANGE)
  An attempt was made to use an unavailable value as the value.
  For example: .bxPP_20 or .bxPP_0 is incorrect.

 2 .bxPW
  .bxPW - Print Width
  Type:   System Variable
  Forms:  .bxPW _ print-positions
          integer-scalar _ .bxPW
  Value Domain:
                 Type:     Near-integer
                 Shape:    Singleton
                 Depth:    0 or 1 (simple)
                 Value:    35 to 2044
                 Default:  Determined when APL is invoked.
  Result Domain:
                 Type:   Integer
                 Rank:   0 (scalar)
                 Shape:  .io0
                 Depth:  0 (simple scalar)

  Sets maximum number of characters in output line.

  The default uses the current VMS setting for SET TERMINAL/WIDTH=n.

 3 Errors
  RANK ERROR (NOT VECTOR DOMAIN)
  The value is not a singleton and its rank is greater than
  1. For example: .bxPW_2 2.ro4 is incorrect.

  LENGTH ERROR (NOT SINGLETON)
  The value is not a single item. For example:
  .bxPW_.io3 is incorrect.

  DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
  The value is not a simple, homogeneous array.

  DOMAIN ERROR (INCORRECT TYPE)
  The value is non-empty and character. For example:
  .bxPW_'A' is incorrect.

  DOMAIN ERROR (NOT AN INTEGER)
  The value is not near-integer. For example: .bxPW_2.5

  LIMIT ERROR (INTEGER TOO LARGE)
  The value is greater than the largest allowable integer.
  For example: .bxPW_.fl2*33 is incorrect.

  DOMAIN ERROR (PARAMETER OUT OF RANGE)
  An attempt was made to use an unavailable value as the value.
  For example: .bxPW_2099 is incorrect.

 2 .bxQCO
  .bxQCO - Copying Objects from a Workspace
  Type:  Monadic System Function (quiet)
  Form:  .io0 _ .bxQCO wsname /PASSWORD:pw /CHECK object-names
  Argument Domain:
                 Type:   Character
                 Shape:  Vector domain
                 Depth:  0 or 1 (simple)
  Result Domain:
                 Type:   Character
                 Rank:   1 (vector)
                 Shape:  Vector
                 Depth:  1 (simple vector)
  Implicit Arguments:  None

  Quietly copies all global objects in a workspace.

  The optional /CHECK qualifier means that APL should examine
  the workspace for possible corruption (damage to the
  internal structure of the workspace). Type )HELP /CHECK  for
  more information.

 3 Errors
  RANK ERROR (NOT VECTOR DOMAIN)
  The argument is not a singleton and its rank is greater
  than 1.

  LENGTH ERROR (ILLEGAL EMPTY ARGUMENT)

  DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
  The argument is not a simple, homogeneous array.

  DOMAIN ERROR (INCORRECT TYPE)
  The argument is non-empty and character.

  DOMAIN ERROR (FILE SPECIFICATION IS MISSING)
  The argument is blank.

 2 .bxQLD
  .bxQLD - Loading Workspaces
  Type:  Monadic System Function (quiet)
  Form:  .bxQLD wsname /PASSWORD:pw /CHECK
  Argument Domain:
                 Type:   Character
                 Shape:  Vector domain
                 Depth:  0 or 1 (simple)
  Result Domain: None
  Implicit Arguments:  None

  Quietly loads a workspace.

  The optional /CHECK qualifier means that APL should examine
  the workspace for possible corruption (damage to the
  internal structure of the workspace). Type )HELP /CHECK  for
  more information.

 3 Errors
  RANK ERROR (NOT VECTOR DOMAIN)
  The argument is not a singleton and its rank is greater
  than 1.

  LENGTH ERROR (ILLEGAL EMPTY ARGUMENT)

  DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
  The argument is not a simple, homogeneous array.

  DOMAIN ERROR (INCORRECT TYPE)
  The argument is non-empty and character.

  DOMAIN ERROR (FILE SPECIFICATION IS MISSING)
  The argument is blank.
```

```
2 .bxQPC
 .bxQPC - Copying Objects with Protection
 Type:  Monadic System Function (quiet)
 Form:  .io0 _ .bxQPC wsname /PASSWORD:pw /CHECK object-names
 Argument Domain:
               Type:   Character
               Shape:  Vector domain
               Depth:  0 or 1 (simple)
 Result Domain:
               Type:   Character
               Rank:   1 (vector)
               Shape:  Vector
               Depth:  1 (simple vector)
 Implicit Arguments:  None

 Quietly copies all global objects in a workspace with protection.

 The optional /CHECK qualifier means that APL should examine
 the workspace for possible corruption (damage to the
 internal structure of the workspace). Type )HELP /CHECK  for
 more information.

3 Errors
 RANK ERROR (NOT VECTOR DOMAIN)
 The argument is not a singleton and its rank is greater
 than 1.

 LENGTH ERROR (ILLEGAL EMPTY ARGUMENT)

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 The argument is not a simple, homogeneous array.

 DOMAIN ERROR (INCORRECT TYPE)
 The argument is non-empty and character.

 DOMAIN ERROR (FILE SPECIFICATION IS MISSING)
 The argument is blank.

2 .bxR
 .bxR - Watched Variable Value
 Type:   System Variable
 Forms:  .bxR _ any
         old-value _ .bxR
 Value Domain:
               Type:     Any
               Shape:    Any
               Depth:    Any
               Default:  ''
 Result Domain:
               Type:   Any
               Rank:   Any
               Shape:  Any
               Depth:  Any

 A variable that is used implicitly by .bxWATCH. .bxR contains the
 previous value of a watched variable that has changed. .bxR is set
 implicitly by the system when a variable changes, but can also be
 set by the user.

3 Errors
 No errors generated

2 .bxRELEASE
 .bxRELEASE - Unlocking Shared Records
 Type:  Monadic System Function (quiet)
 Form:  .io0 _ .bxRELEASE chans
 Argument Domain:
                Type:   Near-integer
                Shape:  Vector domain
                Depth:  0 or 1 (simple)
                Value:  .ng999 through 999 (but not 0)
 Result Domain:
                Type:   Numeric
                Rank:   1 (vector)
                Shape:  0 (empty)
                Depth:  1 (simple vector)
 Implicit Arguments:  None

 Releases all locked records in files on one
 or more channels.

3 Errors
 RANK ERROR (NOT VECTOR DOMAIN)
 The argument is not a singleton and its rank is greater
 than 1.

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 The value specified for chans must be a simple array.

 DOMAIN ERROR (NOT AN INTEGER)
 The channel number is not an integer.

 DOMAIN ERROR (INVALID CHANNEL NUMBER)
 The value specified for chans is outside the argument
 domain.

 LIMIT ERROR (INTEGER TOO LARGE)
 The argument is outside the range of valid integer values.

2 .bxREP
 .bxREP - Replication
 Type:  Dyadic System Function
 Form:  A .bxREP B   A .bxREP[K] B
 Argument Domain:
      Left
                Type:   Near-integer
                Shape:  Vector domain
                Depth:  0 or 1 (simple)
      Right
                Type:   Any
                Shape:  Any
                Depth:  Any
 Result Domain:
                Type:   Same as right argument
                Rank:   1.ce.ro.roB
                Shape:  (K-1)^.roB),(+/.abA),K.da.roB
                        (for .bxIO 1)
                Depth:  1.ce.mtB
 Implicit Arguments:  None

 .bxREP builds arrays by specifying the items to be
 deleted, preserved, or duplicated from an existing array,
 and by indicating where fill items are to be added in the
 new array. When items are preserved or deleted, this is
 known as compression (the left argument is Boolean). When
 items are duplicated, deleted, or filled, this is known as
 replication (the left argument is integer).

 .bxREP works the same as the compress and replicate
 derived functions.  See Compress-Replicate function (Type
 )HELP FUNCTION-NAMES COMPRESS-REPLICATE for more information.)

3 Errors
```

```
 AXIS RANK ERROR (NOT VECTOR DOMAIN)
 K is not a singleton and its rank is greater than 1.

 AXIS LENGTH ERROR (NOT SINGLETON)
 K is not a singleton.

 AXIS DOMAIN ERROR (SEMICOLON LIST NOT ALLOWED)
 There is a semicolon inside of the brackets that surround
 K.

 AXIS DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 K is not a simple array.

 AXIS DOMAIN ERROR (INCORRECT TYPE)
 K is not numeric.

 AXIS DOMAIN ERROR (NOT AN INTEGER)
 K is not a near-integer.

 AXIS DOMAIN ERROR (AXIS LESS THAN INDEX ORIGIN)
 K is less than .bxIO.

 AXIS DOMAIN ERROR (RIGHT ARGUMENT HAS WRONG RANK)
 K is greater than the rank of B.

 RANK ERROR (NOT VECTOR DOMAIN)
 A is not a singleton and its rank is greater than 1.

 LENGTH ERROR
 B is not a singleton and its length along the Kth axis
 is not equal to the number of nonnegative integers in A.

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 A is not a simple, homogeneous array.

 DOMAIN ERROR (INCORRECT TYPE)
 A is not empty and not numeric.

 DOMAIN ERROR (NOT AN INTEGER)
 A is not a near-integer.

 LIMIT ERROR (INTEGER TOO LARGE)
 A or K is greater than the largest allowable integer.

2 .bxRESET
 .bxRESET - Resetting the State Indicator
 Type:  Niladic System Function (no result)
 Form:  .bxRESET
 Result Domain:  None

 Clears the state indicator.

3 Errors
 No errors generated

2 .bxREWIND
 .bxREWIND - Returning Next-record Pointer to Start of File
 Type:   Ambivalent System Function (quiet)
 Forms:  .io0 _ .bxREWIND chans
         .io0 _ key-of-reference .bxREWIND chans
 Monadic Argument Domain:
            Type:   Near-integer
            Shape:  Vector domain
            Depth:  0 or 1 (simple)
            Value:  .ng999 through 999 (but not 0)
 Dyadic Argument Domain:
     Left
```

```
            Type:   Near-integer
            Shape:  Singleton
            Depth:  0 or 1 (simple)
            Value:  0 to 255 inclusive
    Right
            Type:   Near-integer
            Shape:  Singleton
            Depth:  0 or 1 (simple)
            Value:  .ng999 to 999 (not 0)
Result Domain:
            Type:   Numeric
            Rank:   1 (vector)
            Shape:  0 (empty)
            Depth:  1 (simple vector)
Implicit Arguments:  None
```

.bxREWIND repositions the next record pointer to the
first record of a file without closing the file.

If an /AS or /IS file is opened for read operations when
you invoke .bxREWIND, it will remain open for read operations
afterwards. However, if the file is opened for write operations
initially, it will be open for both read and write operations
afterwards.

In the monadic form, you specify a vector of channel numbers in the
right argument. This rewinds each of the files associated with the
specified channel numbers. If any of the files have a keyed organization,
APL performs the rewind on the primary key of reference.

Use the dyadic form for keyed files when you want to rewind
on a key of reference other than the primary key. The right argument
specifies the channel number associated with the keyed file. The left
argument specifies the key of reference: A 0 indicates the primary key,
a 1 indicates the secondary key, and so on. You can only specify one
file at a time when you invoke dyadic .bxREWIND.

3 Errors
RANK ERROR (NOT VECTOR DOMAIN)
The argument is not a singleton and its rank is greater
than 1.

DOMAIN ERROR (INCORRECT TYPE)
The argument is in the character domain; it should be
numeric.

DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
The value specified for chans must be a simple array.

DOMAIN ERROR (INTEGER TOO LARGE)
A channel number is greater than the largest allowable
integer.

DOMAIN ERROR (NOT AN INTEGER)
The argument is not near-integer.

DOMAIN ERROR (INVALID CHANNEL)
The argument is not between .ng999 and 999 or is 0.

DOMAIN ERROR (CHANNEL NOT ASSIGNED)
A value in the argument does not refer to an assigned channel.

DOMAIN ERROR (FILE IS ASSIGNED WRITE ONLY)
The argument specifies a file that cannot be rewound
because it was assigned with the /WRITEONLY qualifier.

IO ERROR (INVALID KEY OF REFERENCE FOR $GET/$FIND)

```
 The key of reference in the left argument does not exist
 in the structure of the keyed file.

 For the dyadic form:

 RANK ERROR (NOT A SINGLETON)
 The value for the left argument (the key of reference
 number) is not a singleton.

 DOMAIN ERROR (INCORRECT TYPE)
 An argument is in the character domain; it should be
 numeric.

 DOMAIN ERROR (NOT AN INTEGER)
 An argument is not near-integer.

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 The valued specified for chans and key-of-reference
 must be simple arrays.

 LIMIT ERROR (INTEGER TOO LARGE)
 The argument is greater than the largest allowable integer.

 DOMAIN ERROR (PARAMETER OUT OF RANGE)
 The value for the left argument (the key of reference number)
 is not between 0 and 255 (inclusive).

 DOMAIN ERROR (INVALID CHANNEL)
 The right argument is not between .ng999 and 999 or is 0.

 DOMAIN ERROR (CHANNEL NOT ASSIGNED)
 The value in the right argument does not refer to an
 assigned channel.

 DOMAIN ERROR (CHANNEL NOT ASSIGNED TO A KEYED FILE)
 The file associated with the channel number is not a /KY file.

 DOMAIN ERROR (FILE IS ASSIGNED WRITE ONLY)
 The file associated with the channel number cannot be
 rewound because it was assigned with the /WRITEONLY
 qualifier.

 IO ERROR (INVALID KEY OF REFERENCE FOR $GET/$FIND)
 The key of reference in the left argument does not exist
 in the structure of the keyed file.

2 .bxRL
 .bxRL - Random Link
 Type:   System Variable
 Forms:  .bxRL _ random-seed
         integer-scalar _ .bxRL
 Value Domain:
                Type:     Near-integer
                Shape:    Singleton
                Depth:    0 or 1 (simple)
                Value:    .ng2*30 through .ng1+2*30
                Default:  695197565
 Result Domain:
                Type:   Integer
                Rank:   0 (scalar)
                Shape:  .io0 (scalar)
                Depth:  0 (simple scalar)

 Forms link in chain of random numbers used in roll
 and deal functions. .bxRL can be set by the user, and
 is also set implicitly by the system when roll and deal
 are executed.
```

```
3 Errors
 RANK ERROR (NOT VECTOR DOMAIN)
 The value is not a singleton and its rank is greater than
 1. For example: .bxRL_2 2.ro4 is incorrect.

 LENGTH ERROR (NOT SINGLETON)
 The value is not a single item. For example:
 .bxRL.io3 is incorrect.

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 The argument is not a simple, homogeneous array.

 DOMAIN ERROR (INCORRECT TYPE)
 The value is non-empty and character. For example:
 .bxRL_'A' is incorrect.

 DOMAIN ERROR (NOT AN INTEGER)
 The value is not near-integer. For example: .bxRL_0.1+2*33
 is incorrect.

 LIMIT ERROR (INTEGER TOO LARGE)
 The value is greater than the largest allowable integer.
 For example: .bxRL_.fl2*33 is incorrect.

2 .bxSF
 .bxSF - Quad Input Prompt
 Type:   System Variable
 Forms:  .bxSF _ prompt
         char-vector _ .bxSF
 Value Domain:
                Type:     Character
                Shape:    Vector domain
                Depth:    0 or 1 (simple)
                Value:    prompt length<255 keystrokes
                Default:  '.bx: <cr><lf><6-spaces>'
 Result Domain:
                Type:   Character
                Rank:   1 (vector)
                Shape:  Vector
                Depth:  1 (simple vector)

 Contains the prompt for quad input (also known
 as evaluated input).

3 Errors
 RANK ERROR (NOT VECTOR DOMAIN)
 The value is not a singleton and its rank is greater than
 1. For example: .bxSF_2 2.ro4

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 The argument is not a simple, homogeneous array.

 DOMAIN ERROR (INCORRECT TYPE)
 The value is non-empty and numeric. For example: .bxSF_10
 is incorrect.

 LIMIT ERROR (ARGUMENT STRING IS TOO LONG)
 The length of the argument cannot be greater than 255
 keystrokes.

2 .bxSIGNAL
 .bxSIGNAL - Signaling Errors
 Type:   Ambivalent System Function (no result)
 Forms:  .bxSIGNAL error-number
         message-text .bxSIGNAL error-number
 Monadic Argument Domain
```

```
                Type:   Near-integer
                Shape:  Singleton
                Depth:  0 or 1 (simple)
                Value:  Any APL error number (except 75 or 500 to 999)
 Dyadic Argument Domain:
        Left
                Type:   Character
                Shape:  Vector domain
                Depth:  0 or 1 (simple)
        Right
                Type:   Near-integer
                Shape:  Singleton
                Depth:  0 or 1 (simple)
                Value:  Any APL error number (except 75 or 500 to 999)
 Result Domain:  None
 Implicit Arguments:  None

 Passes an error up the stack one level to the caller of the
 operation in error. User-defined errors are numbered 500 to 999.

3 Errors
 DOMAIN ERROR (CANNOT SIGNAL EOF)
 An attempt was made to use 75 as the right argument to
 .bxSIGNAL.

 RANK ERROR (NOT VECTOR DOMAIN)
 An argument is not a singleton and its rank is greater
 than 1.

 LENGTH ERROR (NOT SINGLETON)
 B must be a single item.

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 The arguments must be simple, homogeneous arrays.

 DOMAIN ERROR (INCORRECT TYPE)
 An argument is non-empty and numeric.

 DOMAIN ERROR (NOT AN INTEGER)
 B must be a near-integer.

 LIMIT ERROR (INTEGER TOO LARGE)
 B is greater than the largest allowable integer.

 DOMAIN ERROR (PARAMETER OUT OF RANGE)
 An attempt was made to use an unavailable value as the
 right argument.

2 .bxSINK
 .bxSINK - Discard Output
 Type:   System Variable
 Forms:  .bxSINK _ any-value
         .io0 _ .bxSINK
 Value Domain:
                Type:   Any
                Shape:  Any
                Depth:  Any
 Result Domain:
                Type:   Numeric
                Rank:   1 (vector)
                Shape:  0 (empty)
                Depth:  1 (simple vector)

 Discards unwanted output.  Always .io0.

3 Errors
 No errors generated
```

```
2 .bxSS
 .bxSS - String search
 Type:  Dyadic System Function
 Form:  Boolean _ target-string .bxSS search-string
 Argument Domain:
        Left
                Type:   Character
                Shape:  Vector domain
                Depth:  0 or 1 (simple)
        Right
                Type:   Character
                Shape:  Vector domain
                Depth:  0 or 1 (simple)
 Result Domain:
                Type:   Boolean
                Rank:   1 (vector)
                Shape:  .ro,search-string
                Depth:  1 (simple vector)
 Implicit Arguments:  None

 Searches the right argument for every occurrence of
 a character string specified in the left argument.

 The result is a Boolean vector with a length as long as
 the search-string. A 1 is in each position that corresponds
 to where the target-string starts in the search-string.

3 Errors
 RANK ERROR (NOT VECTOR DOMAIN)
 An argument is not a singleton and its rank is greater
 than 1.

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 The arguments must be simple, homogeneous arrays.

 DOMAIN ERROR (INCORRECT TYPE)
 An argument that is nonempty must be of type character.

2 .bxSTOP
 .bxSTOP - Suspending Operation Execution
 Type:   Ambivalent System Function (monadic form is for query)
 Forms:  line-numbers _ .bxSTOP operation-names
         success/fail _ line-numbers .bxSTOP operation-name
 Monadic Argument Domain:
                Type:   Character
                Shape:  Vector domain or 1-row matrix
                Depth:  0 or 1 (simple)
 Dyadic Argument Domain:
     Left
                Type:   Near-integer
                Shape:  Vector domain
                Depth:  0 or 1 (simple)
     Right
                Type:   Character
                Shape:  Matrix domain
                Depth:  0 or 1 (simple)
 Result Domain:
                Type:   Boolean (dyadic) or integer (monadic)
                Rank:   1 (vector)
                Shape:  Vector
                Depth:  1 (simple vector)
 Implicit Arguments:  None

 Sets or clears stop bits associated with
 operation lines. When the line-number argument
 is .io0, all stop bits are cleared.
```

```
3 Errors
 For the dyadic form:

 RANK ERROR (NOT MATRIX DOMAIN)
 The right argument is not in the matrix domain.

 RANK ERROR (NOT VECTOR DOMAIN)
 The left argument is not a singleton and its rank is
 greater than 1.

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 The arguments must be simple, homogeneous arrays.

 DOMAIN ERROR (INCORRECT TYPE)
 The right argument must be of type character and the left
 argument must be near-integer.

 DOMAIN ERROR (NOT AN INTEGER)
 The left argument is not near-integer.

 LIMIT ERROR (INTEGER TOO LARGE)
 The left argument is greater than the largest allowable
 integer.

 For the monadic form:

 RANK ERROR (NOT MATRIX DOMAIN)
 The argument is not in the matrix domain.

 LENGTH ERROR
 There is more than one row specified in the argument.

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 The argument is not a simple, homogeneous array.

 DOMAIN ERROR (INCORRECT TYPE)
 The argument is not of type character.

2 .bxTERSE
 .bxTERSE - Terse Error Messages
 Type:   System Variable
 Forms:  .bxTERSE _ terse-verbose
         integer-scalar _ .bxTERSE
 Value Domain:
               Type:     Near-integer
               Shape:    Singleton
               Depth:    0 or 1 (simple)
               Value:    0 or 1
               Default:  0
 Result Domain:
               Type:   Integer
               Rank:   0 (scalar)
               Shape:  .io0 (scalar)
               Depth:  0 (simple scalar)

 Suppresses display of secondary error messages.

 .bxTERSE _ 0   means secondary error messages are printed. (default)
 .bxTERSE _ 1   means secondary error messages are not printed.

3 Errors
 RANK ERROR (NOT VECTOR DOMAIN)
 The value is not a singleton and its rank is greater than
 1. For example: .bxTERSE_2 2.ro4 is incorrect.

 LENGTH ERROR (NOT SINGLETON)
 The value is not a single item. For example:
 .bxTERSE_.io3 is incorrect.

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 The value is not a simple, homogeneous array.

 DOMAIN ERROR (INCORRECT TYPE)
 The value is non-empty and character. For example:
 .bxTERSE_'A' is incorrect.

 DOMAIN ERROR (NOT AN INTEGER)
 The value is not near-integer. For example: .bxTERSE_2.5
 is incorrect.

 LIMIT ERROR (INTEGER TOO LARGE)
 The value is greater than the largest allowable integer.
 For example: .bxTERSE_.fl2*33 is incorrect.

 DOMAIN ERROR (SYSTEM VARIABLE VALUE MAY ONLY BE 0 OR 1)

2 .bxTIMELIMIT
 .bxTIMELIMIT - User Response Time Limit
 Type:   System Variable
 Forms:  .bxTIMELIMIT _ seconds
         integer-scalar _ .bxTIMELIMIT
 Value Domain:
               Type:     Near-integer
               Shape:    Singleton
               Depth:    0 or 1 (simple)
               Value:    .ng1 to 255
               Default:  0 (unlimited response time)
 Result Domain:
               Type:   Integer
               Rank:   0 (scalar)
               Shape:  .io0 (scalar)
               Depth:  0 (simple scalar)

 Limit on time to respond to quote quad and
 del quad input requests.

3 Errors
 RANK ERROR (NOT VECTOR DOMAIN)
 The value is not a singleton and its rank is greater than
 1. For example: .bxTIMELIMIT_2 2.ro4 is incorrect.

 LENGTH ERROR (NOT SINGLETON)
 The value is not a single item. For example:
 .bxTIMELIMIT_.io3 is incorrect.

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 The value is not a simple, homogeneous array.

 DOMAIN ERROR (INCORRECT TYPE)
 The value is non-empty and character. For example:
 .bxTIMELIMIT_'A'

 DOMAIN ERROR (NOT AN INTEGER)
 The value is not near-integer. For example:
 .bxTIMELIMIT_2.5

 LIMIT ERROR (INTEGER TOO LARGE)
 The value is greater than the largest allowable integer.
 For example: .bxTIMELIMIT_.fl2*33 is incorrect.

 DOMAIN ERROR (PARAMETER OUT OF RANGE)
 An attempt was made to use an unavailable value as the
 value. For example: .bxTIMELIMIT_300 is incorrect.
 DOMAIN ERROR (TIMEOUT READ UNSUPPORTED FOR CURRENT VALUE
 OF QUAD TT) An attempt was made to set .bxTIMELIMIT while
 the current value of .bxTT indicates a VT220 or VT240
 terminal.

2 .bxTIMEOUT
 .bxTIMEOUT - Time Limit Report
 Type:   System Variable
 Forms:  .bxTIMEOUT _ 0-or-1
         integer-scalar _ .bxTIMEOUT
 Value Domain:
               Type:     Near-integer
               Shape:    Singleton
               Depth:    0 or 1 (simple)
               Value:    0 or 1
               Default:  0
 Result Domain:
               Type:   Integer
               Rank:   0 (scalar)
               Shape:  .io0 (scalar)
               Depth:  0 (simple scalar)

 Equals 1 if time runs out during quote quad
 or del quad input request; otherwise, equals 0.

 .bxTIMEOUT is set implicitly by the system when a
 timeout occurs, but can also be set by the user.

3 Errors
 RANK ERROR (NOT VECTOR DOMAIN)
 The value is not a singleton and its rank is greater than
 1. For example: .bxTIMEOUT_2 2.ro4 is incorrect.

 LENGTH ERROR (NOT SINGLETON)
 The value is not a single item. For example:
 .bxTIMEOUT_.io3 is incorrect.

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 The value is not a simple, homogeneous array.

 DOMAIN ERROR (INCORRECT TYPE)
 The value is non-empty and character. For example:
 .bxTIMEOUT_'A' is incorrect.

 DOMAIN ERROR (NOT AN INTEGER)
 The value is not near-integer. For example: .bxTIMEOUT_2.5
 is incorrect.

 LIMIT ERROR (INTEGER TOO LARGE)
 The value is greater than the largest allowable integer.
 For example: .bxTIMEOUT_.fl2*33 is incorrect.

 DOMAIN ERROR (SYSTEM VARIABLE VALUE MAY ONLY BE 0 OR 1)

2 .bxTLE
 .bxTLE - Terminal Line Edit Characteristics
 Type:   System Variable
 Forms:  .bxTLE _ 0-or-1
         current-value _ .bxTLE
 Value Domain:
               Type:     Near-integer
               Shape:    Singleton
               Depth:    0 or 1 (simple)
               Value:    0 or 1
               Default:  Determined when APL is invoked
 Result Domain:
```

```
               Type:   Integer
               Rank:   0 (scalar)
               Shape:  .io0 (scalar)
               Depth:  0 (simple scalar)

 Controls the terminal line editing attribute.

    Value          Equivalent DCL Command

      0            $ SET TERMINAL/NOLINE_EDITING
      1            $ SET TERMINAL/LINE_EDITING


 APL determines the default value for .bxTLE depending on your
 terminal designator. For LA, VT102, GIGI, KEY, BIT, HDS201, and
 HDS221 (terminals that form overstruck characters with the BACKSPACE
 key), the default is 0. For VT220, VT240, VS, VT320, VT330, VT340 and
 DECTERM (terminals that form overstruck characters with the COMPOSE
 key or CTRL/D), the default is 1. In all other cases (TTY for example),
 the default is the same as the current VAX/VMS setting when APL is
 invoked.

3 Errors
 RANK ERROR (NOT VECTOR DOMAIN)
 The value is not a singleton and its rank is greater than
 1. For example: .bxTLE_2 2.ro4 is incorrect.

 LENGTH ERROR (NOT SINGLETON)
 The value is not a single item. For example:
 .bxTLE_.io3 is incorrect.

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 The value is not a simple, homogeneous array.

 DOMAIN ERROR (INCORRECT TYPE)
 The value is non-empty and character. For example:
 .bxTLE_'A'

 DOMAIN ERROR (NOT AN INTEGER)
 The value is not near-integer. For example: .bxTLE_2.5
 is incorrect.

 LIMIT ERROR (INTEGER TOO LARGE)
 The value is greater than the largest allowable integer.
 For example: .bxTLE_.fl2*33 is incorrect.

 DOMAIN ERROR (SYSTEM VARIABLE VALUE MAY ONLY BE 0 OR 1)

2 .bxTRACE
 .bxTRACE - Monitoring Operation Execution
 Type:   Ambivalent System Function
 Forms:  line-numbers _ .bxTRACE operation-name
         success/fail _ line-numbers .bxTRACE operation-name
 Monadic Argument Domain:
               Type:   Character
               Shape:  Vector domain or 1-row matrix
               Depth:  0 or 1 (simple)
 Dyadic Argument Domain:
        Left
               Type:   Near-integer
               Shape:  Vector domain
               Depth:  0 or 1 (simple)
        Right
               Type:   Character
               Shape:  Matrix domain
               Depth:  0 or 1 (simple)
 Result Domain:
```

```
               Type:   Integer (dyadic) or Boolean (monadic)
               Rank:   1 (vector)
               Shape:  Vector
               Depth:  1 (simple vector)
Implicit Arguments:  None

Sets or clears trace bits associated with
operation lines. When the line-number argument
is .io0, all trace bits are cleared.
```

3 Errors

For the dyadic form:

RANK ERROR (NOT MATRIX DOMAIN)
The right argument is not in the matrix domain.

RANK ERROR (NOT VECTOR DOMAIN)
The left argument is not a singleton and its rank is
greater than 1.

DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
The arguments must be simple, homogeneous arrays.

DOMAIN ERROR (INCORRECT TYPE)
The right argument must be of type character and the left
argument must be near-integer.

DOMAIN ERROR (NOT AN INTEGER)
The left argument is not near-integer.

LIMIT ERROR (INTEGER TOO LARGE)
The left argument is greater than the largest allowable
integer.

For the monadic form:

RANK ERROR (NOT MATRIX DOMAIN)
The argument is not in the matrix domain.

LENGTH ERROR
There is more than one row specified in the argument.

DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
The argument is not a simple, homogeneous array.

DOMAIN ERROR (INCORRECT TYPE)
The argument is not of type character.

2 .bxTRAP

```
.bxTRAP - Trap Expression
Type:   System Variable
Forms:  .bxTRAP _ apl-expression
        current-value _ .bxTRAP
Value Domain:
               Type:     Character
               Shape:    Vector domain
               Depth:    0 or 1 (simple)
               Default:  ''
Result Domain:
               Type:   Character
               Rank:   1 (vector)
               Shape:  Vector
               Depth:  1 (simple vector)

Causes expression to be executed when error occurs
in user-defined operation.
```

```
3 Errors
 RANK ERROR (NOT VECTOR DOMAIN)
 The value is not a singleton and its rank is greater than
 1. For example: .bxTRAP_2 2.ro'A' is incorrect.

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 The value is not a simple, homogeneous array.

 DOMAIN ERROR (INCORRECT TYPE)
 The value is non-empty and numeric. For example:
 .bxTRAP_10 is incorrect.

 DOMAIN ERROR (UNSUCCESSFUL TRAP IN LOCKED FUNCTION)
 The execution of a locked operation's .bxTRAP expression
 did not transfer control to a new statement.

2 .bxTS
 .bxTS - Time Stamp
 Type:  Niladic System Function
 Form:  current-time/date _ .bxTS
 Result Domain:
               Type:   Integer
               Rank:   1 (vector)
               Shape:  7
               Depth:  1 (simple vector)

 Returns current date and time in base 10 format
 as a 7-element vector: year, month, day, hour, minute,
 second, millisecond.

3 Errors
 No errors generated

2 .bxTT
 .bxTT - Terminal Type
 Type:   System Variable
 Forms:  .bxTT _ terminal-type
         integer-scalar _ .bxTT
 Value Domain:
               Type:     Near-integer
               Shape:    Singleton
               Depth:    0 or 1 (simple)
               Value:    0 through 19
               Default:  Determined when APL is invoked.
 Result Domain:
               Type:   Integer
               Rank:   0 (scalar)
               Shape:  .io0 (scalar)
               Depth:  0 (simple scalar)

 Sets or returns terminal type for current APL session.

 The default is determined in the APL initialization stream.
 Type )HELP APL-COMMAND-LINE TERMSPEC for a list of supported terminals.

 APL stores this information as described below.

 Value                      Meaning

   1          Composite
   2          TTY type terminal
   3          VK100 (GIGI) terminal (key paired)
   4          DIGITAL LA type terminal (key paired)
   5          APL/ASCII key-paired terminal
   6          APL/ASCII bit-paired terminal
   7          VS (composite)
   8          DIGITAL VT102 terminal (key paired)
   9          VT220 (composite)
   10         VT240 (composite)
   11         Tektronix 4013 terminal (key paired)
   12         Tektronix 4015 terminal (key paired)
   13         HDSAVT and HDSGVT  (key paired)
   14         HDS201 and HDS201G (key paired)
   15         HDS221 and HDS221G (key paired)
   16         VT320 (composite)
   17         VT330 (composite)
   18         VT340 (composite)
   19         DECterm (composite)

3 Errors
 RANK ERROR (NOT VECTOR DOMAIN)
 The value is not a singleton and its rank is greater than
 1. For example: .bxTT_2 2.ro4 is incorrect.

 LENGTH ERROR (NOT SINGLETON)
 The value is not a single item. For example:
 .bxTT_.io3 is incorrect.

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 The value is not a simple, homogeneous array.

 DOMAIN ERROR (INCORRECT TYPE)
 The value is non-empty and character. For example:
 .bxTT_'A'

 DOMAIN ERROR (NOT AN INTEGER)
 The value is not near-integer. For example: .bxTT_2.5
 is incorrect.

 LIMIT ERROR (INTEGER TOO LARGE)
 The value is greater than the largest allowable integer.
 For example: .bxTT_.fl2*33 is incorrect.

 DOMAIN ERROR (NEGATIVE INTEGER NOT ALLOWED)
 For example: .bxTT_.ng1

 DOMAIN ERROR (PARAMETER OUT OF RANGE)
 An attempt was made to use an unavailable value as the
 value. For example: .bxTT_16 is incorrect.

 DOMAIN ERROR (FONT FILE COULD NOT BE OPENED)
 There was an attempt to enter VT220, VT240, VT320, VT330,
 or VT340 mode when the APL font file was not accessible.

 DOMAIN ERROR (ERROR ACTIVATING IMAGE)
 There was an attempt to enter VT220, VT240, VT320, VT330,
 or VT340 mode when SYS$SYSTEM:APLSHR.EXE was not accessible.

2 .bxUL
 .bxUL - User Load
 Type:  Niladic System Function
 Form:  pid _ .bxUL
 Result Domain:
               Type:   Integer
               Rank:   0 (scalar)
               Shape:  .io0 (scalar)
               Depth:  0 (simple scalar)

 Returns process identification number in decimal.
 To convert to hex, use the following expression:
 '0123456789ABCDEF'[.bxIO+(8.ro16).en.bxUL]

3 Errors
 No errors generated
```

```
2 .bxVERSION
 .bxVERSION - Interpreter and Workspace Version
 Type:  Niladic System Function
 Form:  version-info _ .bxVERSION
 Result Domain:
               Type:   Character
               Rank:   1 (vector)
               Shape:  Vector
               Depth:  1 (simple vector)

 Returns interpreter and workspace versions.

 Type )HELP GLOSSARY VERSION-NUMBER  for more information.

3 Errors
 No errors generated

2 .bxVI
 .bxVI - Validating Input
 Type:  Monadic System Function
 Form:  valid/invalid-number _ .bxVI _ character-vector
 Argument Domain:
               Type:   Character
               Shape:  Vector domain
               Depth:  1 (simple)
 Result Domain:
               Type:   Boolean
               Rank:   1 (vector)
               Shape:  Vector
               Depth:  1 (simple vector)
 Implicit Arguments:  .bxNG (determines minus sign placement)

 Returns logical vector giving position of valid
 numbers in .bxFI of argument.

3 Errors
 RANK ERROR (NOT VECTOR DOMAIN)
 The argument is not a singleton and its rank is greater
 than 1.

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 The value is not a simple, homogeneous array.

 DOMAIN ERROR (INCORRECT TYPE)
 The argument is non-empty and character.

2 .bxVPC
 .bxVPC - Vector Process Control
 Type:  System Variable(session)
 Forms: .bxVPC _ session variable
        integer-scalar _ .bxVPC
 Value Domain:
               Type:   Non-negative near-integer
               Shape:  Singleton
               Depth:  0 or 1 (simple)
               Default:Determined when APL is invoked.
 Result Domain:
               Type:   Integer
               Rank:   0
               Shape:  .io0 (scalar)
               Depth:  0 (simple scalar)

 .bxVPC determines the threshold at which the vector processor is used.  A
 value of 0 indicates that the vector processor will never be used; a value
 of 1 indicates that the vector processor will always be used.
```

```
3 Errors
 RANK ERROR (NOT SINGLETON)

 DOMAIN ERROR (NEGATIVE NUMBER NO ALLOWED)

 DOMAIN ERROR (NOT AN INTEGER)

 DOMAIN ERROR (VECTOR PROCESSOR NOT AVAILABLE)

2 .bxVR
 .bxVR - Visual Representation
 Type:  Monadic System Function
 Form:  .bxVR _ value-or-object-name
 Argument Domain:
               Type:   Any
               Shape:  Any
               Depth:  Any
 Result Domain:
               Type:   Character
               Rank:   Any
               Shape:  Any
               Depth:  Any
 Implicit Arguments:  .bxDC (controls display of enclosed arrays)
                      .bxPP (controls print precision)

 Returns a visual representation of a value or
 user-defined operation whose name is the argument
 specified.

3 Errors
 RANK ERROR (NOT VECTOR DOMAIN)
 The argument is of type character, but is not in the
 vector domain.

 DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 The argument is not a simple, homogeneous array.

 DOMAIN ERROR
 The argument is type character and its value does not
 represent the name of an APL object.

 DOMAIN ERROR (OPERATION LOCKED)
 The argument is a locked operation.

2 .bxWA
 .bxWA - Workspace Available
 Type:  Niladic System Function
 Form:  available-space _ .bxWA
 Result Domain:
               Type:   Integer
               Rank:   0 (scalar)
               Shape:  .io0 (scalar)
               Depth:  0 (simple scalar)

 Returns maximum amount in bytes by which the active
 workspace can be increased.

3 Errors
 No errors generated

2 .bxWAIT
 .bxWAIT - Limiting Time on Read Functions
 Type:   Ambivalent System Function (dyadic form is quiet)
 Forms:  current-timelimit _ .bxWAIT chans
         .io0 _ timelimit .bxWAIT chan
 Monadic Argument Domain:
               Type:   Near-integer
               Shape:  Vector domain
               Depth:  0 or 1 (simple)
               Value:  .ng999 through 999 (but not 0)
Dyadic Argument Domain:
       Left
               Type:   Near-integer
               Shape:  Singleton
               Depth:  0 or 1 (simple)
               Value:  .ng1 through 255 (seconds)
       Right
               Type:   Near-integer
               Shape:  Singleton
               Depth:  0 or 1 (simple)
               Value:  .ng999 through 999 (but not 0)
Result Domain:
               Type:   Integer
               Rank:   1 (vector)
               Shape:  Vector
               Depth:  1 (simple vector)
Implicit Arguments:  None
```

The dyadic form specifies the amount of time you want APL to wait when it tries to read a shared record that is locked by another user.

The left argument determines the timelimit:

Value	Meaning
.ng1	Don't wait, return immediately
0	Wait indefinitely (this is the default)
n	Wait for n seconds

The right argument is the channel number of the file you want to read.

The monadic form queries for the current timelimits associated with individual channel numbers. The argument specifies the channel numbers you wish to query. For each channel number in the argument, .bxWAIT returns a value between .ng1 and 255 as used by the dyadic form:

3 Errors

For the monadic form:

LENGTH ERROR (NOT VECTOR DOMAIN)
The argument is not in the vector domain.

DOMAIN ERROR (INCORRECT TYPE)
The argument is not empty and nonnumeric.

DOMAIN ERROR (NOT AN INTEGER)
The argument is not a near-integer.

DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
The argument must be a simple array.

DOMAIN ERROR (INVALID CHANNEL NUMBER)
The right argument is not between .ng999 and 999 or is 0.

LIMIT ERROR (INTEGER TOO LARGE)
The argument is outside the range of valid integer values.

FILE NOT FOUND (FILE NOT FOUND)
The file assigned to the channel does not exist and .bxWAIT opened the file for input.

For the dyadic form:

LENGTH ERROR (NOT SINGLETON)
One of the arguments is not a singleton.

DOMAIN ERROR (INCORRECT TYPE)
One of the arguments is not empty and nonnumeric.

DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
The arguments must be simple arrays.

DOMAIN ERROR (NOT AN INTEGER)
One of the arguments is not a near-integer.

DOMAIN ERROR (PARAMETER OUT OF RANGE)
The left argument is less than .ng1 or greater than 255.

DOMAIN ERROR (INVALID CHANNEL NUMBER)
The right argument is not between .ng999 and 999 or is 0.

DOMAIN ERROR (CHANNEL NOT ASSIGNED)
The right argument does not refer to an assigned channel.

LIMIT ERROR (INTEGER TOO LARGE)
One of the arguments is outside the range of valid integer
values.

FILE NOT FOUND (FILE NOT FOUND)
The file assigned to the channel does not exist and
.bxWAIT opened the file for input.

IO ERROR (TIMEOUT PERIOD EXPIRED)
A record does not become unlocked within the timelimit.

2 .bxWATCH

```
.bxWATCH - Monitoring Variable Changes
Type:   Ambivalent System Function
Forms:  current-mode _ .bxWATCH variable-names
        success/fail _ mode-number .bxWATCH variable-names
Monadic Argument Domain:
               Type:   Character
               Shape:  Vector domain or 1-row matrix
               Depth:  0 or 1 (simple)
Dyadic Argument Domain:
       Left
               Type:   Near-integer
               Shape:  Vector domain
               Depth:  0 or 1 (simple)
       Right
               Type:   Character
               Shape:  Matrix domain
               Depth:  0 or 1 (simple)
Result Domain:
               Type:   Integer
               Rank:   1 or 2
               Shape:  Vector or matrix
               Depth:  1 (simple)
Implicit Arguments:  None
```

The dyadic form enables watchpoints on one or more
variables. The monadic form queries for the current mode
that is set for the variables specified in the argument.

A watchpoint looks for changes in the values of variables.
When a change occurs, .bxWATCH either displays or signals
information on the before and after values of the variables
it is watching. In display mode, .bxWATCH sends information to

the current output and continues execution of the operation. In signal mode, .bxWATCH signals an error that is trappable with .bxTRAP.

Implicit in the use of .bxWATCH are the .bxL and .bxR system variables. Each time a change occurs, .bxL contains the name of a changed object; .bxR contains the previous value of the changed object.

Dyadic .bxWATCH enables watchpoints on one or more variables. The right argument specifies the variables you want to watch. The left argument determines the watch mode. The possible watch modes are:

0	Object not being watched
2	Signal if modified
3	Display if modified
4	Signal if referenced
5	Display if referenced
6	Signal if modified or referenced
7	Display if modified or referenced

3 Errors

For the monadic form:

RANK ERROR (NOT MATRIX DOMAIN)
The argument does not belong to the matrix domain.

LENGTH ERROR
The argument contains more than one row.

DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
The argument is not a simple, homogeneous array.

DOMAIN ERROR (INCORRECT TYPE)
The argument is not of type character.

For the dyadic form:

RANK ERROR (NOT MATRIX DOMAIN)
The right argument is not in the matrix domain.

DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
An argument is not a simple, homogeneous array.

DOMAIN ERROR (INCORRECT TYPE)
The right argument is non-empty and numeric.

RANK ERROR (NOT VECTOR DOMAIN)
The left argument is not a singleton and its rank is greater than 1.

LENGTH ERROR
The left argument is not length 1.

DOMAIN ERROR (INCORRECT TYPE)
The left argument is non-empty and character.

DOMAIN ERROR (NOT AN INTEGER)
The left argument is not a near-integer.

LIMIT ERROR (INTEGER TOO LARGE)
The left argument is greater than the largest allowable integer.

DOMAIN ERROR (INVALID WATCH MODE)
The left argument is outside the range of possible modes.

```
2 .bxXQ
 .bxXQ - Executing Expressions
 Type:  Monadic System Function (sometimes quiet)
 Form:  result _ .bxXQ apl-expression
 Argument Domain:
                Type:   Any
                Shape:  Any
                Depth:  Any
 Result Domain:
                Type:   Any
                Rank:   Any
                Shape:  Any
                Depth:  Any
 Implicit Arguments:  None

 Executes character strings with error handling.

 .bxXQ executes apl-expression as if that expression were
 entered in immediate mode or included in a user-defined
 operation.

 If APL encounters an error while evaluating the .bxXQ function's
 argument, it does not return an error;  instead, it stops
 evaluating the string, and returns an empty array whose shape
 is 0 E, where E is a number indicating the error that was
 encountered. The complete text of the error message is placed
 in .bxERROR.

3 Errors
 RANK ERROR (NOT VECTOR DOMAIN)
 The argument is character and its rank is greater than 1.

1 Relational-Functions

 The dyadic <, .le, =, .ne, >, and .ge functions
 are commonly called relational functions.  The domain of
 relational functions is not restricted; they can take both
 numeric and character arguments. However, only the equal and
 not equal functions can have mismatched arguments, that is,
 one numeric and one character argument simultaneously.
 The range of relational functions is restricted to the
 Boolean values 0 and 1.  A relational function returns the
 result 1 if true and 0 if false.

 When <, .le, .ge, or > have character arguments, the
 order of characters in .bxAV is used as a collating sequence.
 When the relational functions have  numeric arguments,
 the comparisons between the arguments are affected by the
 value of .bxCT.


1 Specification-Function

 Form:  A_B     A[K]_B
 Argument Domain:
        Left
                Type:  Variable name or undefined name
                Shape: Any
        Right
                Type:  Any
                Shape: Conforms to left argument
                Depth: Any
 Result Domain:
                Type:  Same as right argument
                Rank:  .ro.roB
                Shape: .roB
```

```
               Depth: .mtB
Implicit Arguments:  None

The specification function stores values in identifiers.
The left argument (A) must be a variable name or
undefined.  When the function is executed, the value of the
right argument (B) becomes associated with the name A.

The specification function is a quiet function; it does not
return a value if it is the leftmost function in a statement.

Additionally, specification can be used for strand and selective
assignment statements. For more information either of the
following:
           )HELP SPECIFICATION-FUNCTION SELECTIVE-ASSIGNMENT
           )HELP SPECIFICATION-FUNCTION STRAND-ASSIGNMENT

2 Errors
  Specification, not subscripted (form A _ B)

 SYNTAX ERROR (MISSING LEFT ARGUMENT TO ASSIGNMENT
 There is no left argument to the specification function (_).

 NOT A VALID SYSTEM IDENTIFIER
 A is an unknown quad name.

 DOMAIN ERROR (ILLEGAL LEFT ARGUMENT TO ASSIGNMENT)
 A is not a variable or an undefined name.

 DOMAIN ERROR (NOT A SYSTEM VARIABLE)
 A is a quad name but not a system variable.

 VALUE ERROR (NO VALUE TO ASSIGN)
 There is no right argument to the specification function (_).

  Subscripted specification (form A[K]_B)

 DOMAIN ERROR (INVALID OBJECT IN INDEXED ASSIGNMENT)
 The left argument to indexed assignment must be the name
 of a variable.

 DOMAIN ERROR (NOT SYSTEM VARIABLE)
 A is a quad name but not a system variable.

 VALUE ERROR (SUBSCRIPTED NAME IS UNDEFINED)
 A is not a defined name.

 VALUE ERROR (NO VALUE TO ASSIGN)
 There is no right argument to the specification function (_).

 INDEX RANK ERROR (CANNOT INDEX A SCALAR)
 A is a scalar.

 INDEX RANK ERROR
 The number of axes of A does not equal one more than
 the number of semicolons in K.

 INDEX LENGTH ERROR
 The shape of B does not equal the shape of K (does not
 include length 1 axes).

 INDEX LENGTH ERROR (INDEX OUT OF RANGE)
 A value of K is out of range of the corresponding axis length.

 INDEX DOMAIN ERROR (INDEX LESS THAN INDEX ORIGIN)
 A value of K is less than the index origin.
```

INDEX DOMAIN ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
K must be a simple homogeneous array.

INDEX DOMAIN ERROR (INCORRECT TYPE)
K is not numeric, or A and B are not empty and their types
do not match.

INDEX DOMAIN ERROR (NOT AN INTEGER)
K is not a near-integer.

LIMIT ERROR (INTEGER TOO LARGE)
K is greater than the largest allowable integer.

2 Selective-Assignment

```
Form: (fA)_B  (CfA)_B
Argument Domain:
    Left
       Type:  A is a variable name
              f is an eligible function
              C is any valid left argument to f
       Shape: Any
    Right
       Type:  Any
       Shape: Conforms to left argument
       Depth: Any
Result Domain:
       Type:  Same as right argument
       Rank:  .ro.roB
       Shape: .roB
       Depth: .mtB
Implicit Arguments:  None
```

Selective assignment allows you to assign values to specified
items of an array.

The left argument contains an expression that selects items
from an array. The length of the right argument either equals
the number of items selected or is a singleton, in which case APL
performs singleton extension.

The following list describes the primitive functions you can
use in the left argument expression to select items from an array.
The symbol I refers to any expression that is a valid argument to
the function in the form. Note that indexed assignment is the only
form that does not require parentheses.

Assignment Form	Function Name
A[K] _ B	Indexed assignment
(,A) _ B	Ravel
(,[K]A) _ B	Ravel with axis
(.rvA) _ B	Reverse
(.crA) _ B	
(.rv[K]A) _ B	Reverse with axis
(.cr[K]A) _ B	
(.trA) _ B	Transpose
(I.daA) _ B	Drop
(I.da[K]A) _ B	Drop with axis
(I^A) _ B	Take
(I^[K]A) _ B	Take with axis
(I.rvA) _ B	Rotate
(I.crA) _ B	
(I.rv[K]A) _ B	Rotate with axis
(I.cr[K]A) _ B	
(I.trA) _ B	Transpose
(I.roA) _ B	Reshape

```
   (I\A) _ B              Expand
   (I.cbA) _ B
   (I\[K]A) _ B           Expand with axis
   (I.cbA) _ B
   (I/A) _ B              Replicate
   (I.csA) _ B
   (I/[K]A) _ B           Replicate with axis
   (I.csA) _ B
```

3 Examples

```
       .bx_GUT_.io5        "Create GUT
 1 2 3 4 5
       (3^GUT)_48 49 50    "Assign to first 3 items of GUT
 48 49 50 4 5
       (3^GUT)_48          "APL does singleton extension
 48 48 48 4 5
```

You can use more than one of the eligible functions in the left argument expression. For example:

```
       .bx_BOP_3 3.ro.io9
 1 2 3
 4 5 6
 7 8 9
        (2^1 1.trBOP)_0 0     "Change first 2 items on diagonal
 0 2 3
 4 0 6
 7 8 9
```

You can use other primitive functions in the portion of the left argument expression that evaluates the argument of one of the eligible functions. For example:

```
       BOP
 1 2 3
 4 5 6
 7 8 9
       BE_1
       EP_2
       ((BE+EP)^1 1.trBOP)_0 0
 0 2 3
 4 0 6
 7 8 0
```

3 Errors

VALUE ERROR (NO VALUE TO ASSIGN)
There is no right argument.

NOT A VALID SYSTEM IDENTIFIER
A is a quad name that is not supported by this APL implementation.

DOMAIN ERROR (CANNOT MODIFY SELECTIVE ASSIGNMENT TARGET)
The variable being assigned to cannot be modified by
the expression forming the left argument to the selective
assignment.

DOMAIN ERROR (INVALID FUNCTION IN SELECTIVE ASSIGNMENT)
The principal function or functions in the left argument is
ineligible for use with selective assignment.

DOMAIN ERROR (INVALID OBJECT IN SELECTIVE ASSIGNMENT)
The first object inside the parentheses of selective
assignment must be a variable name.

DOMAIN ERROR (NOT A SYSTEM VARIABLE)
A is a quad name but not a system variable.

INDEX LENGTH ERROR
B is not a singleton and its shape does not conform to the
shape of the selected items of A.

INDEX RANK ERROR
B is not a singleton and its rank does not conform to the
rank of the selected items of A.

2 Strand-Assignment

```
 Form: (A1...An)_B
 Argument Domain:
     Left
        Type:  List of variable or undefined names
        Shape: Any
     Right
        Type:  Any
        Shape: Vector domain
        Depth: Any
 Result Domain:
        Type:  Same as right argument
        Rank:  .ro.roB
        Shape: .roB
        Depth: .mtB
 Implicit Arguments:  None
```

Strand assignment (also known as vector assignment) allows you
to assign a list of values to a list of objects. APL applies
the assignment along successive pairs of items in the left (A)
and right (B) arguments in a manner similar to scalar extension.

The objects in A may be undefined names, variable names, or system
variable names. The result of the strand assignment function is
the right argument.

The length of B must conform to the number of objects in A
or it must be a singleton, in which case APL performs singleton
extension.

You can use strand assignment to allow multiple arguments in
user-defined operations.

Strand assignment is an atomic operation; if any of the
assignments fail, no change occurs to any of the names in the
left argument list. If you have set the display option on the
.bxWATCH system function, the displays occur as the strand
assignment proceeds. But if you have set the signal option,
the signal is held until APL completes the entire strand
assignment and only the last watched name is signaled.

3 Examples

```
       BURR _ 32 .dm TEMP _ 0 .dm COLD _ .ng12   "Create 3 objects
       BURR .dm TEMP .dm COLD                    "Display values
32
0
.ng12
       (BURR TEMP COLD) _ 20 .ng4 .ng15       "Parentheses required
       BURR .dm TEMP .dm COLD                 "Display values
20
.ng4
.ng15
       (BURR TEMP COLD) _ .ng3           "Perform singleton extension
       BURR .dm TEMP .dm COLD            "Display values
.ng3
.ng3
.ng3
```

Use strand assignment to allow multiple arguments in user-defined operations. FRET is a monadic user-defined operation containing three local variables (X, Y, and Z). The header definition of FRET is as follows:

```
        .dlFRET B;X;Y;Z .dl
```

When FRET is called, the argument (B) contains three items. Inside FRET, there is an expression that performs a strand assignment in which each item in B is assigned to a local variable. For example:

```
       BIP_(23 41 'RUE')   "BIP contains 3 items
       FRET BIP            "The call to FRET is still monadic
       (X Y Z)_BIP         "This is expression inside of FRET
```

Note that the length (3) of the left argument to the specification function conforms to the number of items in BIP. If BIP were a singleton, APL would perform singleton extension.

3 Errors

SYNTAX ERROR (MISSING LEFT ARGUMENT TO ASSIGNMENT)
There is no left argument.

NOT A VALID SYSTEM IDENTIFIER
A is a quad name that is not supported by this APL implementation.

DOMAIN ERROR (NOT A SYSTEM VARIABLE)
A is a quad name but not a system variable.

DOMAIN ERROR (INVALID OBJECT IN STRAND ASSIGNMENT)
The left argument to a strand assignment must contain either the names of variables or undefined names.

VALUE ERROR (NO VALUE TO ASSIGN)
There is no right argument.

LENGTH ERROR
B is not a singleton and its length is not equal to the number of names in A.

RANK ERROR (NOT VECTOR DOMAIN)
B is not a singleton and its rank is greater than 1.

1 Statements

Statements consist of one or more expressions executed as a unit. You can include more than one statement on a line if you separate the statements with the diamond character (.dm).

Statements separated by diamonds are executed from left to right. Do not confuse the purpose of the semicolon with that of the diamond character; the semicolon is an output catenator, not a statement separator.

The right-to-left evaluation rule does not explain how APL evaluates expressions in all situations. There is also the concept of binding strength, which refers to how APL groups objects for evaluation. The relative binding strengths for various objects are listed below in descending order:

Object	Binding Strength
Brackets	to what is on the left
Left assignment	to the identifier on the left
Right operand	to dyadic operator
Strand	array to array

```
 Left operand     to the operator
 Left argument    to the function
 Right argument   to the function
 Right assignment to the value on the right

1 Qualifiers

2 /ACCESS
 /ACCESS is a qualifier on the left argument of the .bxMAP function that
 specifies whether a formal parameter is read only or modifiable.
 /ACCESS:IN indicates that the external routine reads the parameter and
 does not modify its value. /ACCESS:INOUT indicates that the routine
 reads the parameter and may modify it. /ACCESS:OUT indicates that the
 routine writes a value to the parameter.

2 /APL
 /APL is a qualifier used on the )INPUT (or )OUTPUT) command which
 specifies the use of the character set specified as your terminal
 designator, unless your terminal designator is TTY.

2 /APPEND
 /APPEND is a qualifier used on the )OUTPUT command which allows you
 to add data to the end of an existing file.

2 /AS
 /AS is a qualifier used on the .bxASS function meaning ASCII
 sequential file organization.

2 /BIT
 /BIT is a qualifier used on the )INPUT (or )OUTPUT) command which
 specifies the use of the bit-paired character set.

2 /BLOCKSIZE
 /BLOCKSIZE is a qualifier that has the following meanings:

   o  For input on nondisk devices, it tells APL the minimum
      size memory buffer to have available.  The default is
      2048 bytes.

   o  For output, it specifies the maximum segment size (in
      bytes) for segmented records.  The default is the
      smaller of 2048 and the /MAXLEN value.  Note that
      you should specify /BLOCKSIZE:512 if you want to be
      able to use DECnet to pass the file to another
      VAX/VMS system.

   o  In all other cases, it is ignored.  In addition, it is
      always ignored for ASCII sequential files (the blocksize is
      always 2048).

2 /BUFFERCOUNT
 /BUFFERCOUNT:n is a qualifier used on the .bxASS function. It specifies
 how many I/O buffers you want allocated to read/write to a file. The
 value for n is an integer from 0-127. The default number of allocated
 buffers is the same as the current system default value.

2 /CHECK
 /CHECK is a qualifier used on the following system commands and
 system functions:

       )LOAD       )PCOPY
       )XLOAD      .bxQLD
       )SAVE       .bxQCO
       )COPY       .bxQPC

 When a workspace is loaded or copied, the /CHECK qualifier means
 that APL should examine the workspace for possible corruption
```

(damage to the internal structure of the workspace). If damage is detected, a message is displayed and APL tries to recover as much information as possible from the workspace and continue the load or the copy. The 'recovered' workspace may be missing APL variables, user-defined operations, and other APL objects that were damaged. The user must determine what named objects have been removed from the workspace.

When /CHECK is specified on)SAVE, APL checks for possible damage before saving the current workspace on disk. If damage is detected, APL signals an error and aborts the)SAVE. If this occurs, use)SAVE without /CHECK to save the damaged workspace; use)LOAD with /CHECK to recover as much as possible from the damaged workspace; determine what APL objects have been lost from the damaged workspace.

2 /COMMAND
/COMMAND is a qualifier used on the)EDIT system command.)EDIT/COMMAND:filename tells VAXTPU to use an initialization file.

2 /COMPOSITE
/COMPOSITE is a qualifier used on the)INPUT and)OUTPUT system commands. For)INPUT, it specifies the character set of a file being read into APL. For)OUTPUT, it specifies the desired character set of an object being written to a file.

2 /DA
/DA is a qualifier used on the .bxASS function meaning direct access file organization.

2 /DEFAULTFILE
/DEFAULTFILE:filespec is a qualifier used on the .bxASS function. It specifies a default to be applied to any missing components of a file specification that is used in the argument to .bxASS.

2 /DISPLAY
/DISPLAY is a qualifier used on the)EDIT system command.)EDIT/DISPLAY tells VAXTPU that you are using a supported ANSI CRT terminal. This is the default. You should only specify /DISPLAY during an interactive session.

2 /DISPOSE
/DISPOSE is a qualifier used with .bxASS and)OUTPUT. /DISPOSE:KEEP, which is the default, means the file is permanent; /DISPOSE:DELETE means the file is temporary, and will be deleted when the file is closed; /DISPOSE:PRINT means send the file to a print queue when the file is closed; /DISPOSE:SUBMIT means send the file to a batch queue when the file is closed; /DISPOSE:PRINTDELETE and /DISPOSE:SUBMITDELETE mean send the file to the appropriate queue with instructions to delete the file when the job is finished.

2 /EFN
/EFN is a qualifier used on the .bxASS function which sets up event flags.

2 /ENTRY
/ENTRY:symbol is a qualifier on the right argument of the .bxMAP function that specifies that the value of symbol is the name of the entry point in the shared image. An entry point is the starting address of executable code.

2 /EXECUTE
/EXECUTE is a qualifier used on the)EDIT system command.)EDIT/EXECUTE:tpucommand specifies a VAXTPU command

```
 string that you want to execute after the editor finishes
 any command or section files.

2 /FNS
 /FNS is a qualifier used on the )ERASE system command. It specifies
 that you want to erase only objects that are operators.

 You can use the /FNS qualifier in conjunction with wildcards to
 limit the name class of the objects being erased.

2 /forg
 /forg is the file organization of the file identified by
 fspec.  Possible file organizations are /AS, /IS, /RF, /KY and
 /DA.  The default is /DA.

 Type )HELP FILE-SYSTEM FILE-ORGANIZATION-QUALIFIERS for more information.

2 /GRPS
 /GRPS is a qualifier used on the )ERASE system command. It specifies
 that you want to erase only objects that are operators.

 You can use the /GRPS qualifier in conjunction with wildcards to
 limit the name class of the objects being erased.

2 /INTO
 /INTO is a qualifier used on the )STEP system command.
 )STEP/INTO tells APL to step into any called operation.

2 /IS
 /IS is a qualifier used on the .bxASS function meaning internal
 sequential file organization.

2 /KEY
 /KEY is a qualifier used on the )INPUT (or )OUTPUT) command which
 specifies the use of the key-paired character set.

2 /KY
 /KY is a qualifier used on the .bxASS function meaning keyed
 file organization.

2 /LC
 /LC is a qualifier used on the )EDIT system command.
 )EDIT/LC means that you want the line numbers of a
 user-defined operation to appear in the VAXTPU editor

2 /LIBRARY
 /LIBRARY:filespec is a qualifier on the )HELP command that specifies
 a file containing a help library other than the default APL HELP
 library. This feature allows you to write your own help libraries
 and reference them through the APL )HELP facility.

2 /LIST
 /LIST is a qualifier used on the )INPUT and )OUTPUT commands.
 )INPUT/LIST lists the nested input files. )OUTPUT/LIST displays
 the diverted output file and SYS$OUTPUT.

2 /LOWERCASE
 /LOWERCASE is a qualifier used on the )DO system command.
 )DO/LOWERCASE means that any ASCII lowercase characters
 contained in the recovered output from the execution of
 the command string (which follows the )DO command) should
 not be converted to uppercase.

2 /MAXLEN
 /MAXLEN is a qualifier that allows you to specify a
 maximum record length (in bytes) for a new file (it is
 ignored for existing files).  The default length is the
 value of the .bxDML system variable.

2 /MBX
 /MBX is a qualifier used on the .bxASS function which associates
 a mailbox with a channel.

2 /MECHANISM
 /MECHANISM {:IMMEDIATE |  :REFERENCE | :DESCRIPTOR} is a qualifier on
 the left argument of .bxMAP that specifies a technique for passing
 formal parameters from APL to the external routine. IMMEDIATE specifies
 that the value of the parameter is the value you want to pass.
 REFERENCE specifies that the value of the parameter is the address
 of the value you want to pass. DESCRIPTOR specifies that the value
 is the address of a descriptor that contains the address and length
 of the data as well as other attributes (if the descriptor requires them).

2 /MODE
 /MODE is a qualifier used on the )EDIT system command.
 )EDIT/MODE determines the input/output mode for the data
 moving between the APL and VAXTPU environments. /MODE:2
 means use .qq; /MODE:3 means use .qd.

2 /NC
 /NC is a qualifier used on the )EDIT system command.
 )EDIT/NC specifies the name class of the object you want
 to edit. /NC:2 means the type of the data is character;
 /NC:3 means the type is a function. /NC:4 means the type
 is an operator.

2 /NFS
 /NFS is a qualifier (nonfile structured) that tells
 APL to read from the device without trying to interpret the
 data; in other words, return the data on the device as a
 string of bits.  This qualifier is useful for reading foreign
 devices.

2 /NG
 /NG is a qualifier used on the )EDIT system command.
 )EDIT/NG determines how VAXTPU displays the high minus
 sign.

   /NG:1 means precede negative numbers with a high minus sign
   /NG:0 means do not use the high minus sign.
   /NG:2 means the ASCII minus sign (-) is used as the negative
         sign. Note that the ASCII minus sign translates to an
         APL plus sign. (/NG:2 is used to handle negative numbers
         in strings being read or written in ASCII.)

2 /NOCOMMAND
 /NOCOMMAND is a qualifier used on the )EDIT system command.
 )EDIT/NOCOMMAND tells VAXTPU not to use an initialization
 file.

2 /NODISPLAY
 /NODISPLAY is a qualifier used on the )EDIT system command.
 )EDIT/NODISPLAY tells VAXTPU that you are not using a
 supported terminal. You should use this qualifier when you run
 VAXTPU procedures in batch mode.

2 /NOKEYPAD
 /NOKEYPAD is a qualifier on the )DO and )PUSH commands that specifies
 you do not want the keypad characteristics of the current process
 to be available to the new subprocess.

2 /NOLOGICALS
 /NOLOGICALS is a qualifier on the )DO and )PUSH commands that specifies
 you do not want the logical name table from the current process to be
```

available to the new subprocess.

2 /NOSECTION
/NOSECTION is a qualifier used on the)EDIT system command.
)EDIT/NOSECTION tells VAXTPU not to use a section file.

2 /NOSYMBOLS
/NOSYMBOLS is a qualifier on the)DO and)PUSH commands that specifies
you do not want the global and local symbol table (defined at the DCL
level) from the current process to be available to the new subprocess.

2 /NOTIFY
/NOTIFY is a qualifier used on the)PUSH system command.
)PUSH/NOTIFY tells APL to broadcast a message to your
current process when the new subprocess completes or aborts.

2 /NOWAIT
/NOWAIT is a qualifier used on the)PUSH system.)PUSH/NOWAIT
allows you to create a detached subprocess; control
returns to either APL or the command level when the
subprocess begins execution.

2 /NOWRITERS
/NOWRITERS is a qualifier used on .bxASS. It allows you to
write to a shareable file, but prevents other users from
doing so.

2 /OPEN
/OPEN is a qualifier used on the .bxASS function. It specifies that
you want APL to open or create a file when the channel is assigned.
Normally, the file is not opened or created until the time of the
first I/O operation.

/OPEN:NEW is the default; it means that APL creates a new file.

/OPEN:OLD means that APL opens an existing file.

2 /OPS
/OPS is a qualifier used on the)ERASE system command. It specifies
that you want to erase only objects that are operators.

You can use the /OPS qualifier in conjunction with wildcards to
limit the name class of the objects being erased.

2 /OVER
/OVER is a qualifier used on the)STEP system command.
)STEP/OVER tells APL to step over any called operations.

2 /PARENT
/PARENT is a qualifier used on the)ATTACH system command.
)ATTACH/PARENT means you want to attach to the first process
established in the current job.

2 /PASSWORD
/PASSWORD is a qualifier used on the system commands)LOAD,)SAVE,
)COPY,)PCOPY,)WSID, and)PASSWORD and on the system functions
.bxQLD, .bxQPC, and .bxQCO.

A /PASSWORD or /PASSWORD: specification that is not followed by
a password is ignored.

2 /PP
/PP is a qualifier used on the)EDIT system command.
)EDIT/PP determines the print precision of non-integer
numeric values sent to VAXTPU. /PP is the equivalent of
.bxPP and accepts the same values 1 to 16.

```
2 /PROCESSNAME
 /PROCESSNAME is a qualifier used on the )PUSH system
 command. )PUSH/PROCESSNAME:name specifies the name for the
 new subprocess that you are creating.

2 /PROTECTION
 /PROTECTION is a qualifier used on .bxASS. It allows you to
 specify the protection to be associated with a new file
 (it is ignored for existing files).

2 /PW
 /PW is a qualifier used on the )EDIT system command.
 )EDIT/PW specifies the maximum number of characters in a
 single line of output to the VAXTPU editor. You can use
 values from 35 to 900 for /PW.

 Type )HELP SYSTEM-COMMANDS )EDIT LINE-WRAPPING-SEMANTICS
 for more information.

2 /READONLY
 /READONLY is a qualifier that allows you to read the
 file, but not write to it.

2 /RECORDTYPE
 /RECORDTYPE:value is a qualifier used on the .bxASS function. It
 specifies the record format used by RMS for each record of the
 file. The default is variable length records. APL ignores this
 qualifier if the file already exists or if the file type is /DA,
 /RF, or /KY. You can use the following keywords as values:

    Keyword             Meaning

     VARIABLE           Variable length
     FIXED              Fixed length
     STREAM             Stream format
     STREAMCR           Stream format delimited with <CR>
     STREAMLF           Stream format delimited with <LF>

2 /REVERT
 /REVERT is a qualifier used on the )INPUT and )OUTPUT commands.
 )INPUT/REVERT cancels all nested input files and makes your
 terminal the source of input. )OUTPUT/REVERT directs terminal
 output to your terminal once again.

2 /RF
 /RF is a qualifier used on the .bxASS function meaning relative
 access file organization.

2 /SECTION
 /SECTION is a qualifier used on the )EDIT system command.
 )EDIT/SECTION:filename tells VAXTPU to use a section file.

2 /SHADOW
 /SHADOW is a qualifier used on the )OUTPUT command which displays
 the output on your terminal along with the file.

2 /SHARE
 /SHARE is a qualifier that indicates that several users may
 access the file at the same time.  All users sharing the
 file must use the /SHARE qualifier when associating
 the file with a channel.

2 /SIGNAL
 /SIGNAL is a qualifier used on the .bxASS function. It specifies
 that APL signal the end-of-file indicator when you perform a
 read operation on a non-existent record.
```

For /AS and /IS files, the indicator is 68 END OF FILE ENCOUNTERED.
For /DA, /RF, and /KY files, the indicator is 68 END OF FILE
ENCOUNTERED for a sequential read, and 69 RECORD NOT FOUND
for a random read.

If you do not specify /SIGNAL, APL returns an empty numeric
matrix with the shape 0 75 as the end-of-file indicator.

2 /SILENT
/SILENT is a qualifier used on the)STEP system command.
)STEP/SILENT tells APL not to display the operation name
and current line that are at the top of the state
indicator after the execution of the lines of the
operation.

2 /TERMINAL
/TERMINAL is a qualifier used on the)EDIT system command
)EDIT/TERMINAL:termtype specifies the terminal type you
plan to use during the)EDIT session. The values for
termtype and the character sets they represent are as
follows:

Terminal Type	Character Set
TTY	TTY
KEY	KEY
BIT	BIT
VT102	KEY
LA	KEY
VT220	COMPOSITE
VT240	COMPOSITE
HDS201	COMPOSITE
HDS221	COMPOSITE
VS	COMPOSITE
VT320	COMPOSITE
VT330	COMPOSITE
VT340	COMPOSITE
DECTERM	COMPOSITE

2 /TTY
/TTY is a qualifier used on the)INPUT (or)OUTPUT) command which
specifies the use of the TTY character set.

2 /TYPE
/TYPE:vms-data-type is a qualifier on the left argument of .bxMAP that
specifies a data type for formal parameters and the result (if any) of
the external function. On a formal parameter, /TYPE specifies the VAX
data type that the external routine is expecting. On the result, /type
specifies the VAX data type that will be returned.

Data internal to APL has one of the following types:

Character data in APL character set (8-bits per value)
Boolean data, a subset of numeric data (1-bit per value)
Integer data, a subset of numeric data (32-bits, signed, per value)
Floating point, a subset of numeric (64-bits, D-floating, per value)

The value for vms-data-type can be one of the following:

Type	Name	Type	Name
Z	Unspecified	T	8-bit Text
BU	Byte Logical	VT	Varying Text
WU	Word Logical	NU	Numeric String
LU	Longword Logical	NL	Left Sign String
QU	Quadword Logical*	NLO	Left Overpunch String
OU	Octaword Logical*	NR	Right Sign String
B	Byte Integer	NZ	Zoned Sign String
W	Word Integer	P	Packed Decimal*

```
    L     Longword Integer     V     Bit
    Q     Quadword Integer*    VU    Bit Unaligned*
    O     Octaword Integer*    ZI    Instructions*
    F     F-floating           ZEM   Entry Mask*
    D     D-floating           DSC   Descriptor*
    G     G-floating           BPV   Bound Procedure*
    H     H-floating           BLV   Bound Label*
    FC    F complex            ADT   Date/Time*
    DC    D complex            other DEC or user reserved*
    GC    G complex
    HC    H complex
    CIT   COBOL Temp*
```

Note that the asterisk (*) means unsupported data type.

2 /VALUE
/VALUE:symbol is a qualifier on the right argument of the .bxMAP function that specifies that the value of symbol is the name of a global constant in the shared image. A global constant is a 32-bit signed longword value. When you specify /VALUE, the function header in the left argument of .bxMAP must specify a niladic function that returns a value with a return type of L (for example, 'Z/TYP:L _ F').

2 /VARS
/VARS is a qualifier used on the)ERASE system command. It specifies that you want to erase only objects that are operators.

You can use the /VARS qualifier in conjunction with wildcards to limit the name class of the objects being erased.

1 Symbols
Below is a list of symbols used in VAX APL. The TTY mnemonics are provided within the explanations of these symbols. However, they are not preceded by the required period in the explanations.

Type)HELP GLOSSARY TTY for a listing of TTY mnemonics, symbol names, and the constituent characters of overstrikes.

2)
Right Parenthesis TTY mnemonic is   )

2 <
Less than TTY mnemonic is <

There is no monadic form of <

To obtain help on dyadic < type)HELP RELATIONAL-FUNCTIONS

2 =
Equals TTY mnemonic is =

There is no monadic form of =

To obtain help on dyadic = type)HELP RELATIONAL-FUNCTIONS

2 >
Greater than TTY mnemonic is >

There is no monadic form of >

To obtain help on dyadic > type)HELP RELATIONAL-FUNCTIONS

2]
Right Bracket   TTY mnemonic is]

2 &
And   TTY mnemonic is the ampersand

```
 There is no monadic form of &

 To obtain help on dyadic & type )HELP LOGICAL-FUNCTIONS

2 ,
 Comma  TTY mnemonic is   ,

 To obtain help on monadic , type )HELP FUNCTION-NAMES RAVEL

 To obtain help on dyadic , (and .cc) type

    )HELP FUNCTION-NAMES CATENATE-LAMINATE

2 +
 Plus  TTY mnemonic is  +

 To obtain help on monadic + type )HELP ARITHMETIC-FUNCTIONS CONJUGATE

 To obtain help on dyadic + type )HELP ARITHMETIC-FUNCTIONS ADD

2 .al
 Alpha  TTY mnemonic is   al

2 .ap
 Ampersand  TTY mnemonic is   ap

2 .bx
 Quad  TTY mnemonic is  bx

 To obtain more help on .bx type

    )HELP QUAD-NAMES
    )HELP TERMINAL-INPUT-OUTPUT EVALUTED-INPUT
    )HELP TERMINAL-INPUT-OUTPUT QUAD-OUTPUT

2 .cb
 Column Backslash  TTY mnemonic is   cb

 To obtain help on monadic .cb type )HELP OPERATORS BACKSLASH

 To obtain help on dyadic .cb type

    )HELP FUNCTION-NAMES EXPAND
    )HELP FUNCTION-NAMES SCAN

2 .cc
 Column Comma  TTY mnemonic is   cc

 There is no monadic form of .cc

 To obtain help on dyadic .cc type )HELP FUNCTION-NAMES CATENATE-LAMINATE

2 .ce
 Ceiling  TTY mnemonic is   ce

 To obtain help on monadic .ce type )HELP ARITHMETIC-FUNCTIONS CEILING

 To obtain help on dyadic .ce type )HELP ARITHMETIC-FUNCTIONS MAXIMUM

2 .cf
 Circumflex TTY mnemonic is   cf

2 .co
 Contains   TTY mnemonic is   co

 There is no monadic form of .co
 To obtain help on dyadic .co type )HELP FUNCTION-NAMES CONTAINS

2 .cr
 Column Rotate  TTY mnemonic is   cr

 To obtain help on monadic .cr type )HELP FUNCTION-NAMES REVERSE

 To obtain help on dyadic .cr type )HELP FUNCTION-NAMES ROTATE

2 .cs
 Column Slash  TTY mnemonic is   cs

 To obtain help on monadic .cs type )HELP OPERATORS SLASH

 To obtain help on dyadic .cs type

                  )HELP FUNCTION-NAMES COMPRESS-REPLICATE
                  )HELP FUNCTION-NAMES REDUCE

2 .da
 Down Arrow  TTY mnemonic is   da

 There is no monadic form of .da

 To obtain help on dyadic .da type )HELP FUNCTION-NAMES DROP

2 .dd
 Dieresis  TTY mnemonic is   dd

 To obtain help on monadic .dd type )HELP OPERATORS EACH

 There is no dyadic form of .dd

2 .de
 Decode  TTY mnemonic is   de

 There is no monadic form of .de

 To obtain help on dyadic .de type )HELP FUNCTION-NAMES BASE

2 .dl
 Del  TTY mnemonic is   dl

 To obtain help on .dl type )HELP EDITOR

2 .dm
 Diamond  TTY mnemonic is   dm

 To obtain help on .dm type )HELP STATEMENTS

2 .dq
 Domino  TTY mnemonic is   dq

 To obtain help on monadic .dq type )HELP FUNCTION-NAMES MATRIX-INVERSE

 To obtain help on dyadic .dq type )HELP FUNCTION-NAMES MATRIX-DIVIDE

2 .du
 Down U  TTY mnemonic is   du

 There is no monadic form of .du

 To obtain help on dyadic .du type )HELP FUNCTION-NAMES INTERSECTION

2 .en
 Encode  TTY mnemonic is   en
 There is no monadic form of .en

 To obtain help on dyadic .en type )HELP FUNCTION-NAMES REPRESENT

2 .fl
 Floor  TTY mnemonic is   fl

 To obtain help on monadic .fl type )HELP ARITHMETIC-FUNCTIONS FLOOR

 To obtain help on dyadic .fl type )HELP ARITHMETIC-FUNCTIONS MINIMUM

2 .fm
 Thorn  TTY mnemonic is   fm

 To obtain help on monadic .fm type )HELP FUNCTION-NAMES FORMAT-MONADIC

 To obtain help on dyadic .fm type )HELP FUNCTION-NAMES FORMAT-DYADIC

2 .gd
 Grade Down  TTY mnemonic is   gd

 To obtain help on monadic .gd type )HELP FUNCTION-NAMES GRADE-DOWN

 To obtain help on dyadic .gd type )HELP FUNCTION-NAMES DYADIC-GRADE-DOWN

2 .ge
 Greater than or Equal  TTY mnemonic is   ge

 There is no monadic form of .ge

 To obtain help on dyadic .ge type )HELP RELATIONAL-FUNCTIONS

2 .go
 Right Arrow  TTY mnemonic is   go

 To obtain help on .go type )HELP FUNCTION-NAMES BRANCH

2 .gu
 Grade Up TTY mnemonic is   gu

 To obtain help on monadic .gu type )HELP FUNCTION-NAMES GRADE-UP

 To obtain help on dyadic .gu type )HELP FUNCTION-NAMES DYADIC-GRADE-UP

2 .ib
 I-Beam  TTY mnemonic is    ib

2 .io
 Iota  TTY mnemonic is    io

 To obtain help on monadic .io type )HELP FUNCTION-NAMES INDEX-GENERATOR

 To obtain help on dyadic .io type )HELP FUNCTION-NAMES INDEX-OF

2 .iq
 Input Quad  TTY mnemonic is   iq

 There is no dyadic form of .iq

 To obtain more help on monadic .iq type )HELP FILE-SYSTEM INPUT-QUAD

2 .ja-.jz
 Lowercase letters   TTY mnemonic is  ja-jz

 When you enter lowercase letters from a TTY terminal, APL converts
 them to uppercase letters.  APL recognizes lowercase letters only
 if your terminal is not capable of recognizing lowercase letters.

2 .ld
 Delta (Lower Del)  TTY mnemonic is   ld

2 .le
 Less than or Equal  TTY mnemonic is   le

 There is no monadic form of .le

 To obtain help on dyadic .le type )HELP RELATIONAL-FUNCTIONS

2 .lg
 Logarithm  TTY mnemonic is   lg

 To obtain help on monadic .lg type

    )HELP ARITHMETIC-FUNCTIONS NATURAL-LOGARITHM

 To obtain help on dyadic .lg type

    )HELP ARITHMETIC-FUNCTIONS LOGARITHM

2 .lk
 Left Tack  TTY mnemonic is   lk

2 .lo
 Circle (Large O)  TTY  mnemonic is    lo

 To obtain help on monadic .lo type

    )HELP ARITHMETIC-FUNCTIONS PI-TIMES

 To obtain help on dyadic .lo type

    )HELP ARITHMETIC-FUNCTIONS CIRCLE

2 .lu
 Left U  TTY mnemonic is   lu

 There is no dyadic for of .lu

 To obtain help on monadic .lu type )HELP FUNCTION-NAMES ENCLOSE

2 .mt
 Match  TTY mnemonic is   mt

 To obtain help on monadic .mt type )HELP FUNCTION-NAMES DEPTH

 To obtain help on dyadic .mt type )HELP FUNCTION-NAMES MATCH

2 .ne
 Not Equal  TTY mnemonic is   ne

 There is no monadic form of .ne

 To obtain help on dyadic .ne type )HELP RELATIONAL-FUNCTIONS

2 .ng
 High minus (Negation)   TTY mnemonic is   ng

 This symbol is used as part of a numeric constant.

2 .nn
 Nand  TTY mnemonic is    nn

 There is no monadic form of .nn
 To obtain help on dyadic .nn type )HELP LOGICAL-FUNCTIONS

2 .nr
 Nor  TTY mnemonic is   nr

 There is no monadic form of .nr

 To obtain help on dyadic .nr type )HELP LOGICAL-FUNCTIONS

2 .om
 Omega  TTY mnemonic is   om

2 .oq
 Output Quad  TTY mnemonic is   oq

 To obtain help on .oq type )HELP FILE-SYSTEM OUTPUT-QUAD

2 .or
 Or  TTY mnemonic is   or

 There is no monadic form of .or

 To obtain help on dyadic .or type )HELP LOGICAL-FUNCTIONS

2 .pc
 Percent Sign  TTY mnemonic is   pc

2 .pd
 Protected Del  TTY mnemonic is   pd

 To obtain help on .pd type )HELP EDITOR

2 .ps
 Pound Sign  TTY mnemonic is   ps

2 .qd
 Quad Del  TTY mnemonic is    qd

 To obtain help on .qd type

    )HELP TERMINAL-INPUT-OUTPUT DEL-QUAD-INPUT
    )HELP TERMINAL-INPUT-OUTPUT BARE-OUTPUT

2 .qq
 Quote Quad  TTY mnemonic is   qq

 To obtain help on .qq type

    )HELP TERMINAL-INPUT-OUTPUT QUOTE-QUAD-INPUT
    )HELP TERMINAL-INPUT-OUTPUT BARE-OUTPUT

2 .qu
 Double Quote  TTY mnemonic is   qu

2 .rk
 Right Tack  TTY mnemonic is   rk

2 .ro
 Rho  TTY mnemonic is   ro

 To obtain help on monadic .ro type )HELP FUNCTION-NAMES SHAPE

 To obtain help on dyadic .ro type )HELP FUNCTION-NAMES RESHAPE

2 .ru
 Right U  TTY mnemonic is   ru
 To obtain help on monadic .ru type )HELP FUNCTION-NAMES DISCLOSE

 To obtain help on dyadic .ru type )HELP FUNCTION-NAMES PICK

2 .rv
 Reverse   TTY mnemonic is    rv

 To obtain help on monadic .rv type )HELP FUNCTION-NAMES REVERSE

 To obtain help on dyadic .rv type )HELP FUNCTION-NAMES ROTATE

2 .so
 Jot (Small O)  TTY mnemonic is   so

 For more information type )HELP OPERATORS DOT

 For more information type )HELP FUNCTION-NAMES OUTER-PRODUCT

2 .sq
 Squish Quad  TTY mnemonic is   sq

 The squish quad symbol is formed by overstriking [ and ].  Characters
 in .bxAV that have no meaning in APL are printed out as squish quads.
 Squish quads may not be used as input.

2 .ss
 Subset  TTY mnemonic is   ss

 There is no monadic form of .ss

 To obtain help on dyadic .ss type )HELP FUNCTION-NAMES SUBSET

2 .tr
 Transpose  TTY mnemonic is   tr

 To obtain help on monadic .tr type )HELP FUNCTION-NAMES MONADIC-TRANSPOSE

 To obtain help on dyadic .tr type )HELP FUNCTION-NAMES DYADIC-TRANSPOSE

2 .ud
 Underscored Delta  TTY mnemonic is   ud

2 .us
 Underscore  TTY mnemonic is   us

2 .uu
 Up U  TTY mnemonic is   uu

 To obtain help on monadic .uu type )HELP FUNCTION-NAMES UNIQUE

 To obtain help on dyadic .uu type )HELP FUNCTION-NAMES UNION

2 .xq
 Hydrant  TTY mnemonic is   xq

 To obtain help on monadic .xq type )HELP FUNCTION-NAMES EXECUTE

 There is no dyadic form of .xq

2 .za-.zz
 Underscored letters   TTY mnemonic is   za-zz

2 [
 Left Bracket  TTY mnemonic is  [

 To obtain more help on [ type
```

```
   )HELP AXIS
   )HELP INDEXING

2 ;
 Output Catenator  TTY mnemonic is   ;

 To obtain more help type )HELP TERMINAL-INPUT-OUTPUT OUTPUT-CATENATOR

2 #
 Times  TTY mnemonic is pound sign

 To obtain help on monadic # type )HELP ARITHMETIC-FUNCTIONS SIGNUM

 To obtain help on dyadic # type )HELP ARITHMETIC-FUNCTIONS MULTIPLY

2 :
 Colon  TTY mnemonic is  :

 The colon is used to delimit function line labels.

2 \
 Backslash  TTY mnemonic is   \

 To obtain help on monadic \ type )HELP OPERATORS BACKSLASH

 To obtain help on dyadic \ type
   )HELP FUNCTION-NAMES EXPAND
   )HELP FUNCTION-NAMES SCAN

2 '
 Quote  TTY mnemonic is  '

2 |
 Stile (Absolute Value)  TTY mnemonic is   |

 To obtain help on monadic | type )HELP ARITHMETIC-FUNCTIONS MAGNITUDE

 To obtain help on dyadic | type  )HELP ARITHMETIC-FUNCTIONS RESIDUE

2 ~
 Tilde (Not)  TTY mnemonic is   ~ or nt

 To obtain help on monadic .nt type )HELP LOGICAL-FUNCTIONS

 To obtain help on dyadic .nt type )HELP FUNCTION-NAMES WITHOUT

2 ^
 Up arrow  TTY mnemonic is   circumflex

 To obtain help on monadic ^ type )HELP FUNCTION-NAMES FIRST

 To obtain help on dyadic ^ type )HELP FUNCTION-NAMES TAKE

2 _
 Left Arrow  TTY mnemonic is   underscore

 There is no monadic form of _

 To obtain help on dyadic _ type )HELP SPECIFICATION-FUNCTION

2 -
 Minus  TTY mnemonic is   -

 To obtain help on monadic - type )HELP ARITHMETIC-FUNCTIONS NEGATIVE

 To obtain help on dyadic - type )Help ARITHMETIC-FUNCTIONS SUBTRACT
```

```
2 Atsign
 AT sign   TTY mnemonic is at sign

2 Divide
 Divide  TTY mnemonic is percent sign

 To obtain help on monadic % type )HELP ARITHMETIC-FUNCTIONS RECIPROCAL

 To obtain help on dyadic % type )HELP ARITHMETIC-FUNCTIONS DIVIDE

2 Dollar
 Dollar Sign  TTY mnemonic is   $

2 Dot
 Period  TTY mnemonic is  .

 Represents decimal point, inner product, outer product, or the
 beginning of a TTY mnemonic.

 To obtain help on the . operator type
    )HELP OPERATORS DOT
    )HELP FUNCTION-NAMES INNER-PRODUCT
    )HELP FUNCTION-NAMES OUTER-PRODUCT

2 Lamp
 Lamp  TTY mnemonic is  double-quote

 For more information type )HELP COMMENTS

2 Leftparenthesis
 Left Parenthesis  TTY mnemonic is  (

2 Questionmark
 Question mark  TTY mnemonic is   ?

 To obtain help on monadic ? type )HELP ARITHMETIC-FUNCTIONS ROLL

 To obtain help on dyadic ? type )HELP FUNCTION-NAMES DEAL

2 Slash
 Slash  TTY mnemonic is   /

 To obtain help on monadic / type )HELP OPERATORS SLASH

 To obtain help on dyadic / type
      )HELP FUNCTION-NAMES COMPRESS-REPLICATE
      )HELP FUNCTION-NAMES REDUCE

2 Star
 Star  TTY mnemonic is   asterisk

 To obtain help on monadic * type )HELP ARITHMETIC-FUNCTIONS EXPONENTIAL

 To obtain help on dyadic * type )HELP ARITHMETIC-FUNCTIONS POWER

2 Shriek
 Shriek   TTY mnemonic is   !

 To obtain help on monadic ! type )HELP ARITHMETIC-FUNCTIONS FACTORIAL

 To obtain help on dyadic ! type )HELP ARITHMETIC-FUNCTIONS COMBINATIONS

2 {
 Left Brace  TTY mnemonic is   { or lb

2 }
 Right Brace  TTY mnemonic is   } or rb
```

```
2 `
 Accent Grave  TTY mnemonic is   ag

1 System-Commands

2 )ATTACH
 Type: System action system command
 Form: )ATTACH {/PARENT  | process-name}

 Temporarily suspends the APL session, returns control to
 the parent process (/PARENT) or the process specified by
 /process-name.

 To return to APL, you can use the DCL ATTACH command on the
 process name of the APL process. When you return to the
 interrupted APL session, program execution resumes at the
 point after the execution of the )ATTACH command.

3 Errors
 ERROR PROCESSING ATTACH (NONEXISTENT PROCESS)

 INCORRECT PARAMETER (EXTRANEOUS CHARACTERS AFTER COMMAND)
 For )ATTACH, the extraneous characters could be the result
 of specifying both /PARENT and process-name.

 INCORRECT PARAMETER (MISSING ARGUMENT)

 INCORRECT PARAMETER (PARENT QUALIFIER REPEATED)

 INCORRECT PARAMETER (UNRECOGNIZED QUALIFIER KEYWORD)
 A qualifier other than /PARENT was specified.

 ERROR PROCESSING ATTACH (INVALID LOGICAL NAME)
 The value specified for process-name is incorrect.

 ERROR PROCESSING ATTACH (ATTACH REQUEST REFUSED)
 The value specified for process-name is the name of a
 nonexistent process.

2 )CHARGE
 Type: Query system command
 Form: )CHARGE

 )CHARGE displays a record of activity during the current APL
 session. It includes:

     Your terminal identification.
     Current time and date.
     Length of time connected to APL.
     Amount of computer CPU time used inside APL.
     Number of APL statements executed .
     Number of APL operations executed.
     Number of page faults while inside APL.
     Number of buffered IO and number of direct IO while inside APL.

3 Example

        )CHARGE

 VTA76:  MONDAY 27-SEP-1982 16:14:56.68
 CONNECTED  01:51:47.89  CPU TIME 00:00:02.77
 3 STATEMENTS 2 OPERATIONS
 176 PAGE FAULTS  22 BUFFERED IO  20 DIRECT IO

3 Errors
 INCORRECT PARAMETER (EXTRANEOUS CHARACTERS AFTER COMMAND)
```

```
2 )CLEAR
 Type: Action system command
 Form: )CLEAR

 Replaces active workspace with clear workspace.

3 Errors
 INCORRECT PARAMETER (EXTRANEOUS CHARACTERS AFTER COMMAND)

2 )CONTINUE
 Type: System action system command
 Form: )CONTINUE {HOLD | LOGOUT}
 Default: HOLD

 Saves active workspace and exits APL.

3 Errors
 INCORRECT PARAMETER (UNRECOGNIZED QUALIFIER KEYWORD)

 INCORRECT PARAMETER (EXTRANEOUS CHARACTERS AFTER COMMAND)

2 )COPY
 Type:  Workspace manipulation system command
 Form: )COPY wsname {/PASSWORD:pw} {/CHECK} {list}

 Copies objects from another workspace.

 )COPY retrieves global user-defined operations, global variables,
 and groups from a stored workspace (wsname) and places them
 into your active workspace. If there is a password associated
 with the workspace, you must include it in the command string.

 You can copy all the objects that have names in a workspace or
 a subset of them; list identifies a subset of objects to be copied.
 When you specify a list of objects, you can use the * and %
 wildcards. If you omit the list parameter, all user-defined
 operations, variables, and groups are copied.

 )COPY does not transfer local values for variables and operations,
 nor does it copy the state indicator, channel assignments, or any
 system variable such as the print width, index origin, or print
 precision.

 The )COPY command displays the same message as the )LOAD command.
 The size printed in this message is the size (in disk pages) of
 the active workspace after execution of the )COPY command completes.
 If the list to be copied contains an object that is not in the
 specified workspace, APL returns the message NOT FOUND: followed by
 a list of the objects (separated by tabs) that were not found.

 The optional /CHECK qualifier means that APL should examine
 the workspace for possible corruption (damage to the
 internal structure of the workspace). Type )HELP /CHECK  for
 more information.

3 Examples
     Example 1 copies the entire contents of the workspace named AVER.

            )COPY AVER
      SAVED MONDAY 27-SEP-1982 15:45:03.98 6 BLKS


     Example 2 copies the object B from the workspace named AVER.

            )COPY AVER B
      SAVED MONDAY 27-SEP-1982 15:45:03.98 5 BLKS
     Example 3 cannot find the object C in the workspace named AVER.

            )COPY AVER C
      SAVED MONDAY 27-SEP-1982 15:45:03.98 7 BLKS
      NOT FOUND: C

3 Errors
 INCORRECT PARAMETER

 INCORRECT PARAMETER (ILL FORMED NAME)

 LIMIT ERROR (ARGUMENT STRING IS TOO LONG)

 DAMAGED WORKSPACE HAS BEEN CORRECTED (SOME SYMBOLS MAY
 HAVE BEEN ERASED)

2 )DIGITS
 Type: Query/Change system command
 Form: )DIGITS {n}
 Default in Clear WS: 10

 Displays or changes the number of significant
 digits to be displayed.

3 Examples
            )DIGITS
      10
            1.23456789123456789
      1.234567891
            )DIG 5
      WAS 10
            1.23456789123456789
      1.2346
            )DIG 2
      WAS 5
            1.23456789123456789
      1.2

3 Errors
 INCORRECT PARAMETER (ILL FORMED NUMERIC CONSTANT)

 INCORRECT PARAMETER (PARAMETER OUT OF RANGE)

 INCORRECT PARAMETER (EXTRANEOUS CHARACTERS AFTER COMMAND)

2 )DO
 Type: System action system command
 Form: )DO {/LOWERCASE} command-string
           {/NOKEYPAD}
           {/NOLOGICALS}
           {/NOSYMBOLS}

 Executes a VAX/VMS command;  returns output to APL.

3 Errors
 INCORRECT PARAMETER (NOLOGICALS QUALIFIER REPEATED)

 INCORRECT PARAMETER (NOSYMBOLS QUALIFIER REPEATED)

 INCORRECT PARAMETER (NOKEYPAD QUALIFIER REPEATED)

 INCORRECT PARAMETER (LOWERCASE QUALIFIER REPEATED)

 INCORRECT PARAMETER (MISSING ARGUMENT)
 SUBPROCESS ERROR (COMMAND BUFFER OVERFLOW - SHORTEN
 EXPRESSION OR COMMAND LINE)

2 )DROP
 Type: Workspace manipulation system command
 Form: )DROP file-spec

 Deletes workspaces or files from a directory-structured device.

 )DROP is equivalent to executing the DCL command DELETE/LOG. This
 is true even if you have a symbol definition for DELETE that has
 different qualifiers.

3 Errors
 INCORRECT PARAMETER (MISSING ARGUMENT)

 INCORRECT PARAMETER (LINE TOO LONG TO TRANSLATE)

2 )EDIT
 Type: Action system command
 Form: )EDIT object-name {/COMMAND:filespec |  /NOCOMMAND}
                         {/DISPLAY | /NODISPLAY}
                         {/EXECUTE:tpucommand}
                         {/LC}
                         {/MODE:mode}
                         {/NC:nc}
                         {/NG:ng}
                         {/PP:pp}
                         {/PW:pw}
                         {/SECTION:filespec | /NOSECTION}
                         {/TERMINAL:termtype}

         (Note that all qualifiers follow the name of the
          object you want to edit.)

 The )EDIT system command allows you to edit global APL objects with
 the VAXTPU editor. You can edit user-defined operations and variables.
 You cannot edit enclosed arrays, and you cannot modify an operation that
 is suspended or pendent.

 If you are creating an object, such as a function, with VAXTPU, you
 should specify its name class with the /NC qualifier. The /NC values are:

      Name class    Data type    Shape

           2 (D)    Character    Vector
           3        Function     Not applicable
           4        Operator     Not applicable

 The values 3 and 4 are interchangeable in this case, the actual class
 of the object created depends on the header line.  If you use either
 of these values to create an object, the object-name given in the
 )EDIT command will be inserted in the file as the first line.

3 Line-wrapping-semantics

 When you execute )EDIT, some line wrapping may occur when
 you enter TPU. This could cause unexpected changes in the
 edited object, and may result in an error when you attempt
 to end the editing session.

 When a wrap occurs as you enter the editing session, APL places
 a warning message in TPU's message buffer: LINE WRAP HAS OCCURRED

 The semantics for line wrapping are as follows:

        -  if /PW is not specified,
               APL wraps records with length > 900
        -  if /PW is specified,
               APL wraps records with length > 900 .fl .bxPW
        -  if /PW=n is specified,
               APL wraps records with length > 900 .fl n
```

3 Terminal-information

```
 )EDIT allows VAXTPU to communicate with the VT220, VT240, HDS201,
 HDS221(not the HDSAVT), VS, VT320, VT330, VT340 and DECTERM in APL
 character set. The /TERMINAL  qualifier accepts VT220, VT240, HDS201,
HDS221,
 VS, VT320, VT330, VT340 or DECTERM.

 When in VT220, VT240, VT320, VT330 or VT340 mode, if you leave APL via
 )EDIT and change the terminal type, APL cannot restore
 the terminal setting when you return to APL. The same is true if
 the font file becomes inaccessible while you are in VAXTPU. When the
 terminal setting cannot be restored, APL signals an error and sets
 .bxTT to 2 (TTY).

 The rest of the information in this HELP level refers to when you
 are inside TPU via )EDIT on either the VT2xx, VT3xx or the HDS2xx terminal:

 CTRL/Y aborts your current image, and the memory image is
 not saved. This means you cannot resume with the DCL command
 CONTINUE. The following message appears when you use CTRL/Y:

        %SYSTEM-F-ABORT, abort
        $


 CTRL/T displays an APL-specific status line:

       node::name dd-mmm-yyyy hh:mm:ss.tt CPU=hh:mm:ss.tt
       operation [line .dm statement] M=memory used/allocated

       If there is no operation on the top of the SI stack,
       CTRL/T displays '(APL)' instead of 'operation'.

       Memory is in bytes.


 When searching for a string, certain APL single and overstrike
 characters are matched with normal ASCII characters. For example,
 'y' or 'Y' will also find the lamp character.  This is because
 the default search technique is not case sensitive. You can fix
 this behavior in the following ways:

   If using the EDT emulation section file, then you can enter
   SET SEARCH EXACT to the EDT command processor to enforce
   case sensitivity.

   If using the EVE section file, then refer to the EVE User's
   Manual to determine how to modify case sensitivity.

   If using a custom section file, then modify the TPU-written
   search routine to use the keyword EXACT when calling the
   TPU SEARCH builtin (See VAX TPU Reference Manual).


 Due to a known bug in VAXTPU, some APL characters are displayed
 as reverse questions marks. This behavior may be fixed in the future.
 The affected characters are listed below:

       & % .fl .ep .dl .ld .lo .uu .go .zn .ib .cs
```

Additionally, on the HDS201, the reverse question mark and .SQ appear
as a shaded rectangle. On the HDS221, they appear as an inverted !.

CTRL/R or REFRESH may unmap the APL character set. This means all
APL characters will not be displayed correctly (their internal
representation remains intact). There is a function that remaps
the APL character set and restores the display of your APL text.
To invoke this function, enter APL$MAP_FONT at the TPU command prompt.

Note that the terminal continues to receive in APL mode when the
screen becomes unmapped. This means that when you type APL$MAP_FONT,
the APL '$' appears as '~' and the APL '.us' will appear as capital 'F'.

3 Errors

DOMAIN ERROR (ERROR ACTIVATING IMAGE)
For)EDIT, there is an attempt to enter VT220, VT240,
HDS201, HDS221, VT320, VT330 or VT340 mode when SYS$SYSTEM:APLSHR
is not accessible.

EDIT COMMAND ERROR (ARGUMENT TO xx IS OUT OF RANGE)
For)EDIT, a numeric value that is outside the acceptable
range was specified for a qualifier. xx is the name of the
qualifier.

EDIT COMMAND ERROR (BAD ARGUMENT TO xx)
For)EDIT, an invalid value was specified for a qualifier.
xx is the name of the qualifier.

EDIT COMMAND ERROR (EXECUTE QUALIFIER ARGUMENT IS TOO LONG)
For /EXECUTE, the string specified for tpucommand
is too long.

EDIT COMMAND ERROR (OPERATION LOCKED)
For)EDIT, an attempt was made to edit a locked function.

DEFN ERROR (OPERATION SUSPENDED OR PENDENT)
For)EDIT, an attempt was made to end the VAXTPU session
with an EXIT command when you are not allowed to modify
the function.

EDIT COMMAND ERROR (ILL FORMED NUMERIC CONSTANT)
For)EDIT, there is nonnumeric data (data unacceptable to
.bxVI) inside a numeric array that is returning from
VAXTPU.

EDIT COMMAND ERROR (ILL FORMED NUMERIC MATRIX))
For)EDIT, a record or records in the matrix returning
from VAXTPU have either more or fewer values than the
number of values in the first record.

EDIT COMMAND ERROR (ILLEGAL NAME CLASS)
For /NC, either a value other than 2, 3, or 4 was
specified, or the specified value does not match the
current name class value for objectname.

EDIT COMMAND ERROR (INCORRECT PARAMETER)
For)EDIT, an unknown qualifier was specified.

EDIT COMMAND ERROR (MISSING ARGUMENT)
For)EDIT, an attempt was made to edit a system function
or variable.

EDIT COMMAND ERROR (xx QUALIFIER REPEATED)
For)EDIT, the same qualifier was specified more than once.
xx is the name of the repeated qualifier.

```
 EDIT COMMAND ERROR (EDIT COMMAND UNAVAILABLE DURING
 FUNCTION DEFINITION)

 EDIT COMMAND ERROR (ENCLOSED/HETEROGENEOUS ARRAY NOT ALLOWED)
 An attempt was made to edit an enclosed array.

 EDIT COMMAND ERROR (UNRECOGNIZED QUALIFIER KEYWORD)

 EDIT COMMAND ERROR (VOLUME TOO LARGE)

2 )ERASE
 Type: Action system command
 Form: )ERASE {/FNS} {/VARS} {/GRPS} {/OPS} list

 Erases the named global object from the current workspace.

 You can use the * or % wildcards as an argument. You can use the
 /FNS, /GRPS, /OPS and /VARS qualifiers in conjunction with wildcards to
 limit the name class of the objects being erased.

3 Errors
 INCORRECT PARAMETER (INVALID KEYWORD OR QUALIFIER)

 INCORRECT PARAMETER (ILL FORMED NAME)

 LIMIT ERROR (ARGUMENT STRING IS TOO LONG)

2 )FNS
 Type: Query system command
 Form: )FNS {/WSID:wsname {/PASSWORD:pw}} {wildcard | {start stop}}

 Displays a list of the global names used as user-defined function
 names in a workspace. By default, APL displays the list from the
 currently active workspace. The optional /WSID qualifier  allows you
 to specify a nonactive workspace. If the nonactive workspace was
 saved with a password, you must also specify the /PASSWORD qualifier.

 The optional string parameters identify starting and stopping points
 for the list. When you specify the string parameters, you can use
 the * and % wildcards.

 Note that for all of the system commands that accept wildcards ()FNS,
 )VARS, )GRPS, and )NMS, the wildcard determines the start string. There
 is no wildcard for the stop string.

3 Errors
 INCORRECT PARAMETER (INVALID KEYWORD OR QUALIFIER)

 INCORRECT PARAMETER (FILE SPECIFICATION IS MISSING)

 FILE NOT FOUND (FILE NOT FOUND)

 FILE DOES NOT CONTAIN A WORKSPACE

 INCORRECT PARAMETER (NOT A LETTER)

 INCORRECT PARAMETER (EXTRANEOUS CHARACTERS AFTER COMMAND)

2 )GROUP
 Type: Action system command
 Form: )GROUP group-name {group-member-list}

 Collects named objects into a group.

3 Errors
 INCORRECT PARAMETER (MISSING ARGUMENT)
 INCORRECT PARAMETER (EXTRANEOUS CHARACTERS AFTER COMMAND)

 INCORRECT PARAMETER (ILL FORMED NAME)

 NOT GROUPED, NAME IN USE

2 )GRP
 Type: Query system command
 Form: )GRP {/WSID:wsname {/PASSWORD:pw}} group-name

 Lists members of a group. By default, APL displays the list from the
 currently active workspace. The optional /WSID qualifier  allows you
 to specify a nonactive workspace. If the nonactive workspace was
 saved with a password, you must also specify the /PASSWORD qualifier.

3 Errors
 INCORRECT PARAMETER (MISSING ARGUMENT)

 INCORRECT PARAMETER (EXTRANEOUS CHARACTERS AFTER COMMAND)

 INCORRECT PARAMETER (INVALID KEYWORD OR QUALIFIER)

 INCORRECT PARAMETER (FILE SPECIFICATION IS MISSING)

 FILE NOT FOUND (FILE NOT FOUND)

 FILE DOES NOT CONTAIN A WORKSPACE

 INCORRECT PARAMETER (ILL FORMED NAME)

 INCORRECT PARAMETER (NOT A GROUP)

2 )GRPS
 Type: Query system command

 Form: )GRPS {/WSID:wsname {/PASSWORD:pw}} {start-string} {stop-string}

 Displays an alphabetical list of group names. By default, APL
 displays the list from the currently active workspace. The
 optional /WSID qualifier  allows you to specify a nonactive workspace.
 If the nonactive workspace was saved with a password, you must
 also specify the /PASSWORD qualifier.

 You can use the * or % wildcards as an argument.

3 Errors
 INCORRECT PARAMETER (INVALID KEYWORD OR QUALIFIER)

 INCORRECT PARAMETER (FILE SPECIFICATION IS MISSING)

 FILE NOT FOUND (FILE NOT FOUND)

 FILE DOES NOT CONTAIN A WORKSPACE

 INCORRECT PARAMETER (NOT A LETTER)

 INCORRECT PARAMETER (EXTRANEOUS CHARACTERS AFTER COMMAND)

2 )HELP
 Type: Query system command
 Form: )HELP {/LIBRARY:filespec} {topic}

 )HELP provides controlled access to the APL HELP facility via
 the VAX/VMS HELP librarian.

 The APL HELP library is a file associated with the VAX/VMS
 logical name APL$HELP:. You can define that logical name if you
 want your own HELP library to be the default. If APL$HELP: is
 not defined, VAX APL looks for a file named SYS$HELP:VAXAPL.HLB,
 which is placed on your system during installation.

 For information on creating a HELP file, see the
 LIBRARY command as described in the DCL dictionary.

3 Errors
 ERROR PROCESSING HELP (ERROR PARSING ARGUMENT TO LIBRARY)

 ERROR PROCESSING HELP (INVALID KEY)

 ERROR PROCESSING HELP (TOO MANY HELP KEYS SPECIFIED)

 ERROR PROCESSING HELP (ERROR OPENING AS INPUT)

2 )INPUT
 Type:  Query/Change System Command
 Forms: )INPUT
        )INPUT {filespec /character-set}
        )INPUT {/REVERT}
        )INPUT {/LIST}  (This is the query form)

 )INPUT allows you to change the source of APL input from your
 terminal to other devices. Typically, you would select a file
 to be the new source.

3 Errors
 FILE NOT FOUND (FILE NOT FOUND)

 INCORRECT PARAMETER (EXTRANEOUS CHARACTER AFTER COMMAND)
 Unrecognized input, such as an undefined or repeated
 qualifier, appeared at the end of the command.

 INCORRECT PARAMETER (INVALID CHARACTER SET QUALIFIER)

 INVALID FILE SPECIFICATION (WILD CARDS NOT ALLOWED IN FILE
 SPECIFICATION)

 DEPTH ERROR (TOO MANY DIVERTED INPUTS)

2 )LIB
 Type: Query system command
 Form: )LIB {filespec}

 Displays names of workspaces or files on a
 directory-structured device.

 If you do not specify filespec, APL uses *.APL;*.

3 Errors
 INCORRECT PARAMETER (LINE TOO LONG TO TRANSLATE)

2 )LOAD
 Type: Workspace manipulation system command
 Form: )LOAD wsname {/PASSWORD:pw} {/CHECK}

 Retrieves a workspace from secondary storage.

 The optional /CHECK qualifier means that APL should examine
 the workspace for possible corruption (damage to the
 internal structure of the workspace). Type )HELP /CHECK  for
 more information.

3 Errors
 INCORRECT PARAMETER
 INCORRECT PARAMETER (INVALID KEYWORD OR QUALIFIER)

2 )MAXCORE
 Type: Query/Change system command
 Form: )MAXCORE {n}
 Default in Clear WS: 512P / 1048576P

 Displays or changes setting for maximum workspace size.

3 Errors
 INCORRECT PARAMETER (ILL FORMED NUMERIC CONSTANT)

 INCORRECT PARAMETER (PARAMETER OUT OF RANGE)

 INCORRECT PARAMETER (EXTRANEOUS CHARACTERS AFTER COMMAND)

2 )MINCORE
 Type: Query/Change system command
 Form: )MINCORE {n}
 Default in Clear WS: 40P

 Displays or changes setting for minimum workspace size.

3 Errors
 INCORRECT PARAMETER (ILL FORMED NUMERIC CONSTANT)

 INCORRECT PARAMETER (PARAMETER OUT OF RANGE)

 INCORRECT PARAMETER (EXTRANEOUS CHARACTERS AFTER COMMAND)

2 )MON
 Type: System action system command
 Form: )MON

 Returns you to operating system command level.

3 Errors
 INCORRECT PARAMETER (EXTRANEOUS CHARACTERS AFTER COMMAND)

2 )NMS
 Type: Query system command
 Form: )NMS {/WSID:wsname {/PASSWORD:pw}} {start-string} {stop-string}

 Displays an alphabetical list of all names in the symbol table.
 By default, APL displays the list from the currently active workspace.
 The optional /WSID qualifier  allows you to specify a nonactive workspace.
 If the nonactive workspace was saved with a password, you must also
 specify the /PASSWORD qualifier.

 You can use the * or % wildcards as an argument for the start-string.

3 Errors
 INCORRECT PARAMETER (INVALID KEYWORD OR QUALIFIER)

 INCORRECT PARAMETER (FILE SPECIFICATION IS MISSING)

 FILE NOT FOUND (FILE NOT FOUND)

 FILE DOES NOT CONTAIN A WORKSPACE

 INCORRECT PARAMETER (NOT A LETTER)

 INCORRECT PARAMETER (EXTRANEOUS CHARACTERS AFTER COMMAND)

2 )OFF
 Type: System action system command
 Form: )OFF {HOLD | LOGOUT}
 Default: HOLD

 Ends current APL session.

3 Errors
 INCORRECT PARAMETER (UNRECOGNIZED QUALIFIER KEYWORD)

 INCORRECT PARAMETER (EXTRANEOUS CHARACTERS AFTER COMMAND)

2 )OPS
 Type: Query System Command
 Form: )OPS [[/WSID:wsname[[/PASSWORD:pw]]]]
           [[start-string[[stop-string]]]]
 Default: Displays all user-defined operators
          from active workspace.

 )OPS displays a list of the global names used as user-defined
 operator names in a workspace. The optional /WSID qualifier allows
 you to specify a nonactive workspace. If the nonactive workspace was
 saved with a password, you must also specify the /PASSWORD qualifier.

 The optional string parameters identify starting and stopping points
 for the list. When you specify the string parameters, you can use the
 * and % wildcards.

 The objects are listed in .bxAV order, separated by tabs.

3 Errors
 INCORRECT PARAMETER (INVALID KEYWORD OR QUALIFIER)

 INCORRECT PARAMETER (FILE SPECIFICATION IS MISSING)

 FILE NOT FOUND (FILE NOT FOUND)

 FILE DOES NOT CONTAIN A WORKSPACE

 INCORRECT PARAMETER (NOT A LETTER)

 INCORRECT PARAMETER (EXTRANEOUS CHARACTERS AFTER COMMAND)

2 )ORIGIN
 Type: Query/Change system command
 Form: )ORIGIN {n}
 Default in Clear WS: 1

 Displays or changes index origin.

3 Errors
 INCORRECT PARAMETER (ILL FORMED NUMERIC CONSTANT)

 INCORRECT PARAMETER (SYSTEM VARIABLE VALUE MAY ONLY BE 0 OR 1)

 INCORRECT PARAMETER (EXTRANEOUS CHARACTERS AFTER COMMAND)

2 )OUTPUT
 Type:  Query/Change System Command
 Forms: )OUTPUT
        )OUTPUT filespec {/character-set}
                {/APPEND}
                {/DISPOSE: KEEP | DELETE | PRINT |
                           SUBMIT | PRINTDELETE | SUBMITDELETE}
                {/SHADOW}
        )OUTPUT /REVERT
        )OUTPUT /LIST        (This is the query form)

 )OUTPUT allows you to change the destination of output to a device
 other than your terminal. Typically, you would send the output to
```

a file or to another terminal.

Note that)OUTPUT files cannot be nested.

3 Errors
INCORRECT PARAMETER (REDUNDANT KEYWORD OR QUALIFIER)
A keyword or qualifier was repeated.

INCORRECT PARAMETER (INVALID KEYWORD OR QUALIFIER)

INCORRECT PARAMETER (EXTRANEOUS CHARACTERS AFTER COMMAND)
Unrecognized input, such as an incorrect keyword, appeared
at the end of the command.

IO ERROR (INVALID WILDCARD OPERATION)

2)OWNER
Type: Query system command
Form:)OWNER

Displays information about the creation of the
current workspace.

3 Examples

```
       )CLEAR
       )OWNER
CREATED ON TUESDAY 16-MAY-1983 11:04:25.02 BY USER1 [100,1]
       AT TTA1: WITH V1.1-150
       )SAVE USER1WS
TUESDAY 16-MAY-1983 11:25:04.37 15 BLKS
       )OWNER
CREATED ON TUESDAY 16-MAY-1983 11:25:04.37 BY USER1 [100,1]
       AT TTA1: WITH V1.1-150
       )LOAD USER1WS
SAVED TUESDAY 16-MAY-1983 11:25:04.37 15 BLKS
       )OWNER
CREATED ON TUESDAY 16-MAY-1983 11:25:04.37 BY USER1 [100,1]
       AT TTA1: WITH V1.1-150
```

3 Errors
INCORRECT PARAMETER (EXTRANEOUS CHARACTERS AFTER COMMAND)

2)PASSWORD
Type: Query/Change system command
Form:)PASSWORD {/PASSWORD:pw | pw}
Default in Clear WS: Empty

Displays or changes the workspace password.

3 Errors
INCORRECT PARAMETER (EXTRANEOUS CHARACTERS AFTER COMMAND)

INCORRECT PARAMETER (ILL FORMED NAME)

2)PCOPY
Type: Workspace manipulation system command
Form:)PCOPY wsname {/PASSWORD:pw} {/CHECK} {list}

Copies objects from another workspace while protecting names
already in use.

The optional /CHECK qualifier means that APL should examine
the workspace for possible corruption (damage to the
internal structure of the workspace). Type)HELP /CHECK for
more information.

```
3 Errors
 INCORRECT PARAMETER

 INCORRECT PARAMETER (ILL FORMED NAME)

 LIMIT ERROR (ARGUMENT STRING IS TOO LONG)

 DAMAGED WORKSPACE HAS BEEN CORRECTED (SOME SYMBOLS MAY
 HAVE BEEN ERASED)

2 )PUSH
 Type: System action system command
 Form: )PUSH {/NOWAIT /NOTIFY} {command-string}
             {/PROCESSNAME:process-name}
             {/NOKEYPAD}
             {/NOLOGICALS}
             {/NOSYMBOLS}

 Temporarily suspends the APL session, returning
 control to the operating system.

3 Errors
 INCORRECT PARAMETER (NOLOGICALS QUALIFIER REPEATED)

 INCORRECT PARAMETER (NOSYMBOLS QUALIFIER REPEATED)

 INCORRECT PARAMETER (NOKEYPAD QUALIFIER REPEATED)

 INCORRECT PARAMETER (NOWAIT QUALIFIER REPEATED)

 INCORRECT PARAMETER (NOTIFY QUALIFIER REPEATED)

 INCORRECT PARAMETER (PROCESS NAME QUALIFIER REPEATED)

 INCORRECT PARAMETER (MISSING ARGUMENT)

 INCORRECT PARAMETER (ILLEGAL ASCII CHARACTER)

 SUBPROCESS ERROR (COMMAND BUFFER OVERFLOW - SHORTEN
 EXPRESSION OR COMMAND LINE)

2 )SAVE
 Type: Workspace manipulation system command
 Form: )SAVE {wsname} {/PASSWORD:pw} {/MAXLEN:n} {/CHECK}

 )SAVE creates a copy of the active workspace.

 Note that wsname is optional if the workspace has already been
 given a name (a CLEAR workspace cannot be saved). Otherwise, APL
 assigns the name you specify for wsname as the name of the current
 workspace (regardless of any previous name) and saves a copy of
 this workspace under the new name.

 The optional /CHECK qualifier means that APL should examine
 the workspace for possible corruption (damage to the
 internal structure of the workspace). Type )HELP /CHECK  for
 more information.

3 Errors
 WS NOT SAVED, THIS WS IS CLEAR WS

 INCORRECT PARAMETER (INVALID KEYWORD OR QUALIFIER)

 INCORRECT PARAMETER (EXTRANEOUS CHARACTERS AFTER COMMAND)

2 )SI
 Type: Query system command
```

Form:)SI

)SI displays the state indicator of the active workspace.
The state indicator contains the status of the execution of
user-defined operations, quad input requests, and execute
functions.

For user-defined operations, APL displays the operation name
followed by, within brackets, the line and statement numbers
at which the operation stopped executing. No statement number
is displayed if the statement at which execution stopped is
the first or only statement on the line.

A star following the bracketed line and statement number
indicates that the operation is currently suspended; no
star indicates that the operation is pendent.

Pendent quad input requests are indicated by a .bx character.

Pendent execute functions are indicated by the .bxXQ or .xq
characters.

Locked operations in the state indicator are flagged with a
.pd character, and no line number is displayed.

You can clear individual operations from the state indicator
by using the branch function (.go) to restart or terminate
suspended operations, or you can use the system function
.bxRESET or)SIC to clear the state indicator entirely.

When the state indicator is clear,)SI returns no result.

3 Errors
INCORRECT PARAMETER (EXTRANEOUS CHARACTERS AFTER COMMAND)

2)SIC
Type: Action system command
Form:)SIC

)SIC clears the state indicator.

Once cleared, the state indicator shows no suspended operations
and no pending quad input requests or execute functions.

The)SIC system command behaves in the same manner as the
.bxRESET system function, and they can be used interchangeably.

)SIC does not return a value.

3 Errors
INCORRECT PARAMETER (EXTRANEOUS CHARACTERS AFTER COMMAND)

DEFN ERROR (NAME IN USE)

2)SINL
Type: Query system command
Form:)SINL

)SINL displays the same information as)SI. In addition,
)SINL lists the local symbols of each operation, and displays
the argument expression of any pending execute function.
Local symbols in locked operations (flagged with a .pd
character) are not displayed.

When the state indicator is clear,)SINL returns no result.

3 Errors

```
 INCORRECT PARAMETER (EXTRANEOUS CHARACTERS AFTER COMMAND)

2 )SIS
 Type: Query system command
 Form: )SIS

 Displays workspace state indicator, currently executing
 line, and the argument expression to any pending execute
 functions.

 )SIS does not display the executing line of a locked
 operation.

 When the state indicator is clear, )SIS returns no result.

3 Errors
 INCORRECT PARAMETER (EXTRANEOUS CHARACTERS AFTER COMMAND)

2 )SIV
 )SIV has been phased out. Use )SINL in its place.
 To obtain help on )SINL, type )HELP SYSTEM-COMMANDS )SINL.

2 )STEP
 Type: Action system command
 Form: )STEP {n} {/SILENT} {/INTO | /OVER}

 Executes lines of an operation one at a time.

3 Errors
 OPERATION INVALID IN THIS CONTEXT
 )STEP was used in one of the following instances: as an
 argument to the execute function, when there was no
 suspended operation, or when you were in function
 definition mode or evaluated input mode.

 INCORRECT PARAMETER (PARAMETER OUT OF RANGE)
 For )STEP, the value for n was 0 (or less).

 INCORRECT PARAMETER (EXTRANEOUS CHARACTERS AFTER COMMAND)

 INCORRECT PARAMETER (INVALID KEYWORD OR QUALIFIER)

2 )VARS
 Type: Query system command
 Form: )VARS {/WSID:wsname {/PASSWORD:pw}} {start-string} {stop-string}

 Displays an alphabetic list of global variables. By default, APL
 displays the list from the currently active workspace. The optional
 /WSID qualifier  allows you to specify a nonactive workspace. If the
 nonactive workspace was saved with a password, you must also specify
 the /PASSWORD qualifier.

 You can use the * or % wildcards as an argument for the start-string.

3 Errors
 INCORRECT PARAMETER (INVALID KEYWORD OR QUALIFIER)

 INCORRECT PARAMETER (FILE SPECIFICATION IS MISSING)

 FILE NOT FOUND (FILE NOT FOUND)

 FILE DOES NOT CONTAIN A WORKSPACE

 INCORRECT PARAMETER (NOT A LETTER)

 INCORRECT PARAMETER (EXTRANEOUS CHARACTERS AFTER COMMAND)
```

```
2 )VERSION
 Type: Query system command
 Form: )VERSION

 Displays the APL version numbers for the workspace
 and interpreter.

 Type )HELP GLOSSARY VERSION-NUMBER  for more information.

3 Errors
 INCORRECT PARAMETER (EXTRANEOUS CHARACTERS AFTER COMMAND)

2 )WIDTH
 Type: Query/Change system command
 Form: )WIDTH {n}
 Default in Clear WS: System setting

 Displays or changes the terminal line width.

 When using the VT220, VT240, VT102, HDSAVT, HDS201, HDS221, VS, VT320,
 VT330, VT340 or DECTERM,  APL toggles the screen between 80- and
 132-column mode when you use )WIDTH to set the print width to above or
 below 80. Setting .bxPW does not cause this behavior.

 The VT240, VT320, VT330, and VT340 support uses two font files, one for
 80- and the other for 132-column mode. If you suspend the APL session and
 change the terminal width at the DCL level, the screen will be in the new
 mode and APL will be in the previous mode when you return to APL. Use the
 appropriate value to )WIDTH to correct it.

3 Errors
 INCORRECT PARAMETER (ILL FORMED NUMERIC CONSTANT)

 INCORRECT PARAMETER (PARAMETER OUT OF RANGE)

 INCORRECT PARAMETER (EXTRANEOUS CHARACTERS AFTER COMMAND)

2 )WSID
 Type: Query/Change system command
 Form: )WSID {wsname} {/PASSWORD:pw}
 Default in Clear WS: CLEAR WS with a blank password

 Displays or changes workspace name; optionally changes
 workspace password.

3 Errors
 INCORRECT PARAMETER (EXTRANEOUS CHARACTERS AFTER COMMAND)

2 )XLOAD
 Type: Workspace manipulation system command
 Form: )XLOAD wsname {/PASSWORD:pw} {/CHECK}

 Retrieves a workspace from secondary storage without
 executing .bxLX.

 The optional /CHECK qualifier means that APL should examine
 the workspace for possible corruption (damage to the
 internal structure of the workspace). Type )HELP /CHECK  for
 more information.

3 Errors
 INCORRECT PARAMETER

 INCORRECT PARAMETER (INVALID KEYWORD OR QUALIFIER)


1 Terminal-Input-Output
```

Input and output operations not involving external files --
the default terminal I/O and the use of I/O variables -- is
sometimes called terminal I/O, because the only I/O device
involved is your terminal.

The default terminal I/O is straightforward: you enter input
from your terminal; APL echoes your input beginning in
column 7, and if the statement you entered does not have a
quiet function as the leftmost function, APL prints the
result beginning at column 1 on the next line of your
terminal.

2 Bare-Output

```
Type:  System Variable
Forms: .qq _ apl-expression
       .qd _ apl-expression
       .qq is formed with .bx and '
       .qd is formed with .bx and .dl
```

If the del-quad or quote-quad input symbols appear immediately
to the left of a specification function (_), the result of the
is called bare-output. Bare output works the same way
as quad output, except that bare output does not print any
<CR><LF>s (not even a closing one) that are not entered by the
user. Thus, bare output provides a convenient way to request
input on the same line as an output string.

Note also that the input value is preceded by a number of
spaces equal to the length of the .qq output.
If you do not want the spaces, you can use the .bxARBOUT
function to reset the bare output buffer.

The format of bare output is the APL default terminal
output, except that bare output does not depend upon the
value of the .bxPW system variable (it is .bxPP-dependent,
however). Thus, APL does not insert a <CR><LF> and begin a
new line after it displays .bxPW characters.

2 Quad-Del-Input

```
Type:  System Variable
Form:  untranslated-character-array _ .qd
       .qd is formed with .bx and .dl
Default Prompt:      None
User-defined Prompt: See description of bare output
```

Del quad input is similar to quote quad input, except that
the characters returned remain untranslated; thus, you can
enter special characters like <BS> without having APL
evaluate them.

What is normally a legal APL overstruck character
becomes three characters: first character, <BS>, second
character.

TTY mode APL mnemonics are treated as three characters.

To escape from .qd input without entering a value, type the
abort input signal. Type)HELP TERMINAL-SUPPORT INTERRUPTING-APL
for more information.

2 Diverted-Input

The)INPUT system command allows you to change the source of
APL input from your terminal to other devices.

```
 Type )HELP SYSTEM-COMMANDS )INPUT  for more information.

2 Diverted-Output

 The )OUTPUT command allows you to divert output to a device
 other than your terminal.

 Type )HELP SYSTEM-COMMANDS )OUTPUT  for more information.

2 Evaluated-Input

 Type:  System Variable
 Form:  evaluated-result-of-input _ .bx
 Default Prompt: .bx: <CR><LF><6-spaces>
 User-defined Prompt: Use .bxSF

 The system variable for evaluated input returns the value of
 the expression you enter in response to the prompt specified
 by .bxSF.  (The default prompt is .bx: followed by a <CR><LF>
 and six spaces.)

 When you use .bx, APL prompts you for input from the terminal,
 then returns your input value as the result.  Typically,
 evaluated input is used with the specification function, so
 that the variable which is the left argument of the
 specification function is assigned the value of the
 evaluated input.

 While the system is awaiting your input, you can execute a
 system command, or you can define or edit an operation; the
 input request remains pending until you supply a value.
 However, the input request is canceled if you enter CTRL/Z,
 execute one of the system functions .bxRESET, .bxBREAK, or
 .bxSIGNAL, or execute a system command that changes the
 state of the active workspace, that is, a )LOAD, )CLEAR,
 )OFF, )CONTINUE, or )SIC command.

 To escape from evaluated input without entering a value,
 type the right-arrow (.go) character, or type the abort
 input signal. Type )HELP TERMINAL-SUPPORT INTERRUPTING-APL
 for more information.

2 Quad-Output

 Type:  System Variable
 Form:  .bx _ apl-expression

 If the quad symbol appears immediately to the left of a
 specification function (_), the result of the
 expression to the right of the _ prints on the terminal.
 This is called quad output.

 Note that using quad output has the same effect as merely
 typing the expression to the right of _. Quad output
 may be helpful when an APL statement contains multiple
 specification operations.

 The format of quad output is the APL default terminal output
 and depends upon the values of the system variables .bxPW and
 .bxPP.  The last character printed in quad output is a <CR><LF>.

2 Quote-Quad-Input

 Type:  System Variable
 Forms: .qq _ apl-expression
```

The system variable for .qq input treats the value you enter as character data. When APL encounters a .qq symbol, it considers the data between the current cursor position and the next carriage return as a character vector.

Note that there is no prompt for .qq input.

Do not enclose the input in quotation marks; if you do, the quotation marks are taken as part of the vector

Because whatever you type is accepted as part of a character vector, you cannot execute system commands or invoke the function editor while .qq input is pending.

To escape from .qq input without entering a value, type the abort input signal. Type)HELP TERMINAL-SUPPORT INTERRUPTING-APL for more information.

1 Terminal-Support

APL language functions and operators are represented by a variety of special characters. The way these characters are supported depends on the type of terminal you have. On APL terminals, you can enter these special characters directly; on non-APL terminals, you must substitute ASCII mnemonics. (These terminals are known as TTY terminals.)

Type)HELP APL-COMMAND-LINE TERMSPEC for information on specifying your terminal type when you invoke APL.

APL terminals use one of three APL character sets: APL key-paired (also called typewriter-paired), APL bit-paired, or COMPOSITE APL character set. The COMPOSITE APL character set is a superset of the APL key-paired character set.

2 BIT

BIT terminals use the bit-paired character set.

APL assumes that any BIT terminal uses the BACKSPACE key to create overstrikes. When you invoke APL with the bit designator, APL prepares the environment for creating overstruck characters using <BACKSPACE> (regardless of the current terminal setting). The default for line editing is .bxTLE_0 (noline-edit). For example, <shift L><BACKSPACE><shift K> produces .qq (quote-quad).

2 COMPOSITE

COMPOSITE terminals use the COMPOSITE character set.

APL inherits the terminal characteristics for line editing and for receiving broadcast messages. When the terminal is set for broadcast and line-edit, the defaults are: .bxTLE_1 (line-edit) and .bxGAG_0 (display messages).

2 DECTERM

The DECwindows terminal emulator is treated as a key-paired terminal. The keyboard types in both APL and ASCII. For example, unshifted 'r' is APL 'R' and shifted 'R' is APL 'rho'. Note that the terminal itself runs in COMPOSITE APL character set.

To create an APL overstruck character, press CTRL/D followed by the two constituent characters. For example, CTRL/D <shift L><shift K> produces .qq (quote-quad). The constituent characters may be in either order.

The BACKSPACE on the DECterm performs a VMS terminal line-editing function inside APL (it sends the cursor to the beginning of the line). Use the arrow keys to position the cursor on an input line for editing.

When you use the DECTERM designator, the APL environment inherits the terminal characteristics for line editing and for receiving broadcast messages. When the terminal is set for broadcast and line-edit, the defaults are: .bxTLE_1 (line-edit) and .bxGAG_0 (display messages).

2 GIGI

A GIGI terminal runs APL on the DIGITAL VK100 (GIGI) graphics terminal. The APL character set on the GIGI terminal is key-paired and must be loaded manually.

GIGI users must make sure that the G0 active character set contains the ASCII character set, and that the G1 active character set contains the APL character set.

To create overstrikes, use the BACKSPACE key. For example, <shift L><BACKSPACE><shift K> produces .qq (quote-quad).

When you invoke APL with the gigi designator, APL prepares the environment for creating overstruck characters using <BACKSPACE> (regardless of the current terminal setting) and inherits the terminal characteristic for receiving broadcast messages. The default for line editing is .bxTLE_0 (noline-edit). The inherited characteristics for broadcasts is either .bxGAG_2 (trap, translate, and display message) or .bxGAG_1 (refuse message).

2 HDSAVT

The HDSAVT is a key-paired APL terminal manufactured by Human Design Systems.

To create overstrikes, use the BACKSPACE key. For example, <shift L><BACKSPACE><shift K> produces .qq (quote-quad).

When you invoke APL with the HDSAVT designator, APL prepares the environment for creating overstruck characters using <BACKSPACE> (regardless of the current terminal setting) and inherits the terminal characteristic for receiving broadcast messages. The default for line editing is .bxTLE_0 (noline-edit). The inherited characteristics for broadcasts is either .bxGAG_2 (trap, translate, and display message) or .bxGAG_1 (refuse message).

2 HDS201

The HSD201 is a key-paired APL terminal manufactured by Human Design Systems.

To create overstrikes, use the BACKSPACE key. For example, <shift L><BACKSPACE><shift K> produces .qq (quote-quad).

When you invoke APL with the HDS201 designator, APL prepares the environment for creating overstruck characters using <BACKSPACE> (regardless of the current terminal setting) and inherits the terminal characteristic for receiving broadcast messages. The default for line editing is .bxTLE_0 (noline-edit). The inherited characteristics for broadcasts is either .bxGAG_2 (trap, translate, and display message) or .bxGAG_1 (refuse message).

2 HDS221

The HSD221 is a key-paired APL terminal manufactured by Human Design

Systems.

To create overstrikes, use the BACKSPACE key. For example, <shift L><BACKSPACE><shift K> produces .qq (quote-quad).

When you invoke APL with the HDS221 designator, APL prepares the environment for creating overstruck characters using <BACKSPACE> (regardless of the current terminal setting) and inherits the terminal characteristic for receiving broadcast messages. The default for line editing is .bxTLE_0 (noline-edit). The inherited characteristics for broadcasts is either .bxGAG_2 (trap, translate, and display message) or .bxGAG_1 (refuse message).

2 Interrupting-APL

You can interrupt APL execution by entering any of the three forms of the attention signal:

```
    CTRL/C -- The weak attention signal means to suspend execution
              of the current operation after executing the current
              statement, and return control to immediate mode.

    CTRL/C CTRL/C -- The strong attention signal means to suspend
              the current operation as soon as possible, even in
              the middle of the statement, and return control to
              immediate mode.

    CTRL/Y -- The panic exit means to suspend the current operation
              immediately, and give control to the VAX/VMS. After a
              panic exit, you can return to where you left off by
              executing the DCL command CONTINUE.
```

The abort input signal allows you to escape to immediate mode when APL is waiting for input. The abort input signal is particularly useful when APL is executing .bx, .qq, or .qd input, or when APL is in the .dl editor or super-edit mode. In all cases, APL cancels the current input request and returns you to immediate mode.

Different terminal types form the abort input signal differently. Note that <dot> represents the dot (.) symbol.

Signal Form	Terminal Designator
o<BS>u<BS>t	APL, BIT, HDSxx, KEY, LA, TTY, 4013, 4015
O<BS>U<BS>T	TTY
o<BS>u<BS>t	TTY
<dot>ou	TTY
CTRL/D O U	VT2xx, VT3xx, DECTERM
COMPOSE O U	VS

For terminals that for the abort input signal with o<BS>u<BS>t, you must enter the five keystrokes exactly in the order shown, with no embedded spaces or tabs.

For a TTY terminal, note that the dot (.) of <dot>ou (or <dot>OU) must appear in the first column of a line or must be preceded by a space. (See example for more information.)

For terminals that for the abort input signal with CTRL/D O U or COMPOSE O U, it does not matter the order in which you type the O and U.

3 Examples

```
            "First, the <dot>OU is part of a filetype
      )OUTPUT DATA.OUT
            "Now, the <dot>OU is the abort input signal
      )OUTPUT DATA <dot>OUT
   51 INPUT ABORTED
      )OUTPUT DATA <dot>OUT
```

Note that <dot> represents the dot (.) symbol.

2 KEY

KEY terminals use the key-paired character set. When you invoke APL, the KEY designator is synonymous with the APL designator.

APL assumes that any KEY terminal uses the BACKSPACE key to create overstrikes. For example, <shift L><BACKSPACE><shift K> produces .qq (quote-quad).

When you invoke APL with the key designator, APL prepares the environment for creating overstruck characters using <BACKSPACE> (regardless of the current terminal setting) and inherits the terminal characteristic for receiving broadcast messages. The default for line editing is .bxTLE_0 (noline-edit). The inherited characteristics for broadcasts is either .bxGAG_2 (trap, translate, and display message) or .bxGAG_1 (refuse message).

2 LA

LA represents any of the following terminal types:

   LA12, LA34, LA36, LA38, LA100, LA120

Each of these terminals uses the key-paired character set.

The LA terminals shift character sets automatically when you enter and leave APL. You do not have to push a button on the terminal to switch to the APL or ASCII character set. (However, if you want to switch character sets manually, you can specify KEY as you terminal designator when you invoke APL.)

To create overstrikes, use the BACKSPACE key. For example, <shift L><BACKSPACE><shift K> produces .qq (quote-quad).

When you invoke APL with the LA designator, APL prepares the environment for creating overstruck characters using <BACKSPACE> (regardless of the current terminal setting) and inherits the terminal characteristic for receiving broadcast messages. The default for line editing is .bxTLE_0 (noline-edit). The inherited characteristics for broadcasts is either .bxGAG_2 (trap, translate, and display message) or .bxGAG_1 (refuse message).

2 Line-editing

Line editing enables you to correct typographical errors and other errors in lengthy input lines and saves you the trouble of retyping the entire line.

To edit input lines, you can use various control characters as well as keypad keys. The following list describes some of these editing functions:

CTRL/B (up arrow)	Displays the last command line entered. You can display up to twenty previously entered command lines. The down arrow displays lines in the opposite direction.
<BACKSPACE>	Moves the cursor to the beginning of the line.

CTRL/E	Moves the cursor to the end of the line
CTRL/D (left arrow)	Moves the cursor one position to the left.
CTRL/F (right arrow)	Moves the cursor one position to the right.
DELETE	Moves the cursor back one character and erases that character.
CTRL/U	Deletes the current command line from the cursor to the left margin and issues a carriage return.
CTRL/C CTRL/Y	Cancels or interrupts an entire command line.

2 TTY

Terminals that do not have an APL keyboard are known at TTY terminals.

TTY terminals use ASCII mnemonics to represent APL symbols. For example, to represent the APL 'rho' symbol (.ro) on a TTY terminal, you type the mnemonic RO.

Note that you can use any APL terminal as a TTY terminal by specifying TTY as your terminal designator when you invoke APL (or you can set .bxTT _ 2 once inside APL).

When you use the TTY designator, the APL environment inherits the terminal characteristics for line editing and for receiving broadcast messages. When the terminal is set for broadcast and line-edit, the defaults are: .bxTLE_1 (line-edit) and .bxGAG_0 (display messages).

Type)HELP SYMBOLS or)HELP GLOSSARY TTY-Character-Set for more information on the ASCII mnemonics.

2 VS

The VS designator is for the DIGITAL VAXstation, which uses the COMPOSITE character set with a key-paired keyboard. APL maps a loadable APL character set into G1. The system command)EDIT uses VAXTPU, which maps DEC Supplemental into G2 and Special Graphics into G3. G0 remains 7-bit ASCII.

When you leave the APL environment, APL loads DEC Supplemental into G1 and maps it into GR. (APL switches between character sets automatically when you enter and leave APL.)

To create an APL overstruck character on VS terminals, press the COMPOSE key followed by the two constituent characters. For example, COMPOSE <shift L><shift K> produces .qq (quote-quad). The constituent characters may be typed in either order.

When you invoke APL with the VS designator, APL inherits the terminal characteristics for line editing and for receiving broadcast messages. When the terminal is set for broadcast and line_edit, the defaults are: .bxTLE_1 (line_edit) and .bxGAG_0 (display messages). When the terminal is set for nobroadcast and noline_edit, the defaults are: .bxTLE_0 (noline_edit) and .bxGAG_1 (refuse messages).

2 VT102

The VT102 terminal has an optional APL feature that uses the APL character set. If you have this feature, the terminal shifts

character sets automatically when you enter and leave APL.
If you do not have the optional APL feature, then specify
TTY as your terminal designator when you invoke APL.

The keyboard is treated as typewriter-paired (key-paired) APL/ASCII.
For example, unshifted 'r' is APL 'R' and shifted 'R' is APL 'rho'.

To create overstrikes, use the BACKSPACE key. For example,
<shift L><BACKSPACE><shift K> produces .qq (quote-quad).

When you invoke APL with the VT102 designator, APL prepares the
environment for creating overstruck characters using <BACKSPACE>
(regardless of the current terminal setting) and inherits the
terminal characteristic for receiving broadcast messages. The default
for line editing is .bxTLE_0 (noline-edit). The inherited
characteristics for broadcasts is either .bxGAG_2 (trap, translate,
and display message) or .bxGAG_1 (refuse message).

2 VT220

The VT220 is treated as a key-paired terminal. The keyboard types
in both APL and ASCII. For example, unshifted 'r' is APL 'R' and
shifted 'R' is APL 'rho'. Note that the terminal itself runs in
COMPOSITE APL character set.

To create overstrikes, press CTRL/D followed by the two constituent
characters. For example, CTRL/D <shift L><shift K> produces .qq
(quote-quad). The constituent characters may be in either order.

The BACKSPACE on the VT220 performs a VMS terminal line-editing
function inside APL (it sends the cursor to the beginning of the
line). Use the arrow keys to position the cursor on an input line
for editing.

When you use the VT220 designator, the APL environment inherits the
terminal characteristics for line editing and for receiving
broadcast messages. When the terminal is set for broadcast and
line-edit, the defaults are: .bxTLE_1 (line-edit) and .bxGAG_0
(display messages).

3 Font-information

APL character support for the VT220 uses font files provided with
the APL software. The following logical name is used to find
the associated font file:

APL$VT220_FONT	the loadable APL font for the VT220

You can define this logical name to point to your own font file.
Otherwise, APL uses the font file installed with VAX APL.

APL maps the APL character set into G1. TPU maps DEC Supplemental
into G2 and Special Graphics into G3. G0 remains 7-bit ASCII.

2 VT240

The VT240 is treated as a key-paired terminal. The keyboard types
in both APL and ASCII. For example, unshifted 'r' is APL 'R' and
shifted 'R' is APL 'rho'. Note that the terminal itself runs in
COMPOSITE APL character set.

To create an APL overstruck character, press CTRL/D followed by the
two constituent characters. For example, CTRL/D <shift L><shift K>
produces .qq (quote-quad). The constituent characters may be in either
order.

The BACKSPACE on the VT240 performs a VMS terminal line-editing function inside APL (it sends the cursor to the beginning of the line). Use the arrow keys to position the cursor on an input line for editing.

When you use the VT240 designator, the APL environment inherits the terminal characteristics for line editing and for receiving broadcast messages. When the terminal is set for broadcast and line-edit, the defaults are: .bxTLE_1 (line-edit) and .bxGAG_0 (display messages).

3 Font-information

APL character support for the VT240 uses font files provided with the APL software. The following logical names are used to find the associated font files:

```
      APL$VT240_FONT        the loadable APL font for the VT240
                             in 80 column mode
      APL$VT240_FONT_132    the loadable APL font for the VT240
                             in 132 column mode
```

You can define these logical names to point to your own font files. Otherwise, APL uses the font files installed with VAX APL.

APL maps the APL character set into G1. TPU maps DEC Supplemental into G2 and Special Graphics into G3. G0 remains 7-bit ASCII.

2 VT320

The VT320 is treated as a key-paired terminal. The keyboard types in both APL and ASCII. For example, unshifted 'r' is APL 'R' and shifted 'R' is APL 'rho'. Note that the terminal itself runs in COMPOSITE APL character set.

To create an APL overstruck character, press CTRL/D followed by the two constituent characters. For example, CTRL/D <shift L><shift K> produces .qq (quote-quad). The constituent characters may be in either order.

The BACKSPACE on the VT320 performs a VMS terminal line-editing function inside APL (it sends the cursor to the beginning of the line). Use the arrow keys to position the cursor on an input line for editing.

When you use the VT320 designator, the APL environment inherits the terminal characteristics for line editing and for receiving broadcast messages. When the terminal is set for broadcast and line-edit, the defaults are: .bxTLE_1 (line-edit) and .bxGAG_0 (display messages).

3 Font-information

APL character support for the VT320 uses font files provided with the APL software. The following logical names are used to find the associated font files:

```
      APL$VT320_FONT        the loadable APL font for the VT320
                             in 80 column mode
      APL$VT320_FONT_132    the loadable APL font for the VT320
                             in 132 column mode
```

You can define these logical names to point to your own font files. Otherwise, APL uses the font files installed with VAX APL.

APL maps the APL character set into G1. TPU maps DEC Supplemental into G2 and Special Graphics into G3. G0 remains 7-bit ASCII.

```
2 VT330

 The VT330 is treated as a key-paired terminal. The keyboard types
 in both APL and ASCII. For example, unshifted 'r' is APL 'R' and
 shifted 'R' is APL 'rho'. Note that the terminal itself runs in
 COMPOSITE APL character set.

 To create an APL overstruck character, press CTRL/D followed by the
 two constituent characters. For example, CTRL/D <shift L><shift K>
 produces .qq (quote-quad). The constituent characters may be in either
 order.

 The BACKSPACE on the VT330 performs a VMS terminal line-editing
 function inside APL (it sends the cursor to the beginning of the
 line). Use the arrow keys to position the cursor on an input line
 for editing.

 When you use the VT330 designator, the APL environment inherits the
 terminal characteristics for line editing and for receiving
 broadcast messages. When the terminal is set for broadcast and
 line-edit, the defaults are: .bxTLE_1 (line-edit) and .bxGAG_0
 (display messages).

3 Font-information

 APL character support for the VT330 uses font files provided with
 the APL software. The following logical names are used to find
 the associated font files:

       APL$VT330_FONT        the loadable APL font for the VT330
                              in 80 column mode
       APL$VT330_FONT_132    the loadable APL font for the VT330
                              in 132 column mode

 You can define these logical names to point to your own font files.
 Otherwise, APL uses the font files installed with VAX APL.

 APL maps the APL character set into G1.  TPU maps DEC Supplemental
 into G2 and Special Graphics into G3. G0 remains 7-bit ASCII.

2 VT340

 The VT340 is treated as a key-paired terminal. The keyboard types
 in both APL and ASCII. For example, unshifted 'r' is APL 'R' and
 shifted 'R' is APL 'rho'. Note that the terminal itself runs in
 COMPOSITE APL character set.

 To create an APL overstruck character, press CTRL/D followed by the
 two constituent characters. For example, CTRL/D <shift L><shift K>
 produces .qq (quote-quad). The constituent characters may be in either
 order.

 The BACKSPACE on the VT340 performs a VMS terminal line-editing
 function inside APL (it sends the cursor to the beginning of the
 line). Use the arrow keys to position the cursor on an input line
 for editing.

 When you use the VT340 designator, the APL environment inherits the
 terminal characteristics for line editing and for receiving
 broadcast messages. When the terminal is set for broadcast and
 line-edit, the defaults are: .bxTLE_1 (line-edit) and .bxGAG_0
 (display messages).

3 Font-information

 APL character support for the VT340 uses font files provided with
```

the APL software. The following logical names are used to find the associated font files:

```
      APL$VT340_FONT        the loadable APL font for the VT340
                             in 80 column mode
      APL$VT340_FONT_132    the loadable APL font for the VT340
                             in 132 column mode
```

You can define these logical names to point to your own font files. Otherwise, APL uses the font files installed with VAX APL.

APL maps the APL character set into G1. TPU maps DEC Supplemental into G2 and Special Graphics into G3. G0 remains 7-bit ASCII.

2 4013

The 4013 refers to the Tektronix 4013 terminal, which uses the key-paired character set.

The 4013 shifts character sets automatically when you enter and leave APL. You do not have to push a button on the terminal to switch to the APL or ASCII character set. (However, if you want to switch character sets manually, you can specify KEY as you terminal designator when you invoke APL.)

When you invoke APL with the 4013 designator, APL prepares the environment for creating overstruck characters using <BACKSPACE> (regardless of the current terminal setting) and inherits the terminal characteristic for receiving broadcast messages. The default for line editing is .bxTLE_0 (noline-edit). The inherited characteristics for broadcasts is either .bxGAG_2 (trap, translate, and display message) or .bxGAG_1 (refuse message).

2 4015

The 4015 refers to the Tektronix 4015 terminal, which uses the key-paired character set.

The 4015 shifts character sets automatically when you enter and leave APL. You do not have to push a button on the terminal to switch to the APL or ASCII character set. (However, if you want to switch character sets manually, you can specify KEY as your terminal designator when you invoke APL.)

When you invoke APL with the 4015 designator, APL prepares the environment for creating overstruck characters using <BACKSPACE> (regardless of the current terminal setting) and inherits the terminal characteristic for receiving broadcast messages. The default for line editing is .bxTLE_0 (noline-edit). The inherited characteristics for broadcasts is either .bxGAG_2 (trap, translate, and display message) or .bxGAG_1 (refuse message).