```
PPPPPPPPPPPP        AAAAAAAAA        SSSSSSSSSSS  RRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
PPPPPPPPPPPP        AAAAAAAAA        SSSSSSSSSSS  RRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
PPPPPPPPPPPP        AAAAAAAAA        SSSSSSSSSSS  RRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
PPP          PPP   AAA         AAA  SSS          RRR         RRR        TTT        LLL
PPP          PPP   AAA         AAA  SSS          RRR         RRR        TTT        LLL
PPP          PPP   AAA         AAA  SSS          RRR         RRR        TTT        LLL
PPP          PPP   AAA         AAA  SSS          RRR         RRR        TTT        LLL
PPP          PPP   AAA         AAA  SSS          RRR         RRR        TTT        LLL
PPP          PPP   AAA         AAA  SSS          RRR         RRR        TTT        LLL
PPPPPPPPPPPP       AAA         AAA     SSSSSSSSS     RRRRRRRRRRR        TTT        LLL
PPPPPPPPPPPP       AAA         AAA     SSSSSSSSS     RRRRRRRRRRR        TTT        LLL
PPPPPPPPPPPP       AAA         AAA     SSSSSSSSS     RRRRRRRRRRR        TTT        LLL
PPP                AAAAAAAAAAAAAAA           SSS   RRR   RRR          TTT        LLL
PPP                AAAAAAAAAAAAAAA           SSS   RRR   RRR          TTT        LLL
PPP                AAAAAAAAAAAAAAA           SSS   RRR   RRR          TTT        LLL
PPP                AAA         AAA           SSS   RRR      RRR       TTT        LLL
PPP                AAA         AAA           SSS   RRR      RRR       TTT        LLL
PPP                AAA         AAA           SSS   RRR      RRR       TTT        LLL
PPP                AAA         AAA  SSSSSSSSSSS   RRR         RRR     TTT        LLLLLLLLLLLLLLL
PPP                AAA         AAA  SSSSSSSSSSS   RRR         RRR     TTT        LLLLLLLLLLLLLLL
PPP                AAA         AAA  SSSSSSSSSSS   RRR         RRR     TTT        LLLLLLLLLLLLLLL
PPP                AAA         AAA  SSSSSSSSSSS   RRR         RRR     TTT        LLLLLLLLLLLLLLL
```

```
PPPPPPPP      AAAAAA      SSSSSSSS  RRRRRRRR    EEEEEEEEEE    AAAAAA    EEEEEEEEEE  NN      NN  UU      UU
PPPPPPPP      AAAAAA      SSSSSSSS  RRRRRRRR    EEEEEEEEEE    AAAAAA    EEEEEEEEEE  NN      NN  UU      UU
PP      PP   AA      AA  SS        RR      RR  EE            AA      AA  EE          NN      NN  UU      UU
PP      PP   AA      AA  SS        RR      RR  EE            AA      AA  EE          NN      NN  UU      UU
PP      PP   AA      AA  SS        RR      RR  EE            AA      AA  EE          NNNN    NN  UU      UU
PP      PP   AA      AA  SS        RR      RR  EE            AA      AA  EE          NNNN    NN  UU      UU
PPPPPPPP     AA      AA    SSSSSS  RRRRRRRR    EEEEEEEE      AA      AA  EEEEEEEE    NN  NN  NN  UU      UU
PPPPPPPP     AA      AA    SSSSSS  RRRRRRRR    EEEEEEEE      AA      AA  EEEEEEEE    NN  NN  NN  UU      UU
PP           AAAAAAAAAA          SS  RR  RR      EE          AAAAAAAAAA  EE          NN    NNNN  UU      UU
PP           AAAAAAAAAA          SS  RR  RR      EE          AAAAAAAAAA  EE          NN    NNNN  UU      UU
PP           AA      AA          SS  RR    RR    EE          AA      AA  EE          NN      NN  UU      UU
PP           AA      AA          SS  RR    RR    EE          AA      AA  EE          NN      NN  UU      UU
PP           AA      AA  SSSSSSSS    RR      RR  EEEEEEEEEE  AA      AA  EEEEEEEEEE  NN      NN  UUUUUUUUUU
PP           AA      AA  SSSSSSSS    RR      RR  EEEEEEEEEE  AA      AA  EEEEEEEEEE  NN      NN  UUUUUUUUUU

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

```
;     1          0001  0 MODULE PAS$READ_ENUMERATED ( %TITLE 'Read an enumerated value'
;     2          0002  0                 IDENT = '1-002'                 ! File: PASREAENU.B32 Edit: SBL1002
;     3          0003  0                 ) =
;     4          0004  1 BEGIN
;     5          0005  1 !
;     6          0006  1 !*****************************************************************************
;     7          0007  1 !                                                                            *
;     8          0008  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
;     9          0009  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                    *
;    10          0010  1 !*  ALL RIGHTS RESERVED.                                                     *
;    11          0011  1 !*                                                                           *
;    12          0012  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
;    13          0013  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
;    14          0014  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER  *
;    15          0015  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY  *
;    16          0016  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY  *
;    17          0017  1 !*  TRANSFERRED.                                                             *
;    18          0018  1 !*                                                                           *
;    19          0019  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
;    20          0020  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT  *
;    21          0021  1 !*  CORPORATION.                                                             *
;    22          0022  1 !*                                                                           *
;    23          0023  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS  *
;    24          0024  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
;    25          0025  1 !*                                                                           *
;    26          0026  1 !*                                                                           *
;    27          0027  1 !*****************************************************************************
;    28          0028  1 !
;    29          0029  1
;    30          0030  1 !++
;    31          0031  1 ! FACILITY:     Pascal Language Support
;    32          0032  1 !
;    33          0033  1 ! ABSTRACT:
;    34          0034  1 !
;    35          0035  1 !       This module contains procedures which read an enumerated value
;    36          0036  1 !       from a textfile or a string.
;    37          0037  1 !
;    38          0038  1 ! ENVIRONMENT:  User mode - AST reentrant
;    39          0039  1 !
;    40          0040  1 ! AUTHOR: Steven B. Lionel, CREATION DATE: 1-April-1981
;    41          0041  1 !
;    42          0042  1 ! MODIFIED BY:
;    43          0043  1 !
;    44          0044  1 ! 1-001 - Original.  SBL 1-April-1981
;    45          0045  1 ! 1-002 - Use PAS$$END_READ.  SBL 26-May-1982
;    46          0046  1 !--
;    47          0047  1
```

```
;    49          0048  1 %SBTTL 'Declarations'
;    50          0049  1 !
;    51          0050  1 ! PROLOGUE DEFINITIONS:
;    52          0051  1 !
;    53          0052  1
;    54          0053  1 REQUIRE 'RTLIN:PASPROLOG';                    ! Externals, linkages, PSECTs, structures
;    55          0117  1
;    56          0118  1 !
;    57          0119  1 ! TABLE OF CONTENTS:
;    58          0120  1 !
;    59          0121  1
;    60          0122  1 FORWARD ROUTINE
;    61          0123  1     PAS$READ_ENUMERATED,                          ! Read from textfile
;    62          0124  1     PAS$READ$_ENUMERATED;                         ! Read from string
;    63          0125  1
;    64          0126  1 !
;    65          0127  1 ! MACROS:
;    66          0128  1 !
;    67          0129  1 !       NONE
;    68          0130  1 !
;    69          0131  1 ! EQUATED SYMBOLS:
;    70          0132  1 !
;    71          0133  1 !       NONE
;    72          0134  1 !
;    73          0135  1 ! FIELDS:
;    74          0136  1 !
;    75          0137  1 !       NONE
;    76          0138  1 !
;    77          0139  1 ! OWN STORAGE:
;    78          0140  1 !
;    79          0141  1 !       NONE
;    80          0142  1 !
```

```
;    82          0143  1 %SBTTL 'PAS$READ_ENUMERATED - Read an enumerated value from textfile'
;    83          0144  1 GLOBAL ROUTINE PAS$READ_ENUMERATED (
;    84          0145  1         PFV: REF $PAS$PFV_FILE_VARIABLE,                ! File variable
;    85          0146  1         PETD: REF VECTOR [, LONG],                      ! Enumerated type descriptor
;    86          0147  1         ERROR                                           ! Error unwind address
;    87          0148  1     ) =
;    88          0149  1
;    89          0150  1 !++
;    90          0151  1 ! FUNCTIONAL DESCRIPTION:
;    91          0152  1 !
;    92          0153  1 !       This function reads an enumerated value from the specified textfile
;    93          0154  1 !       and returns it as the function value.
;    94          0155  1 !
;    95          0156  1 ! CALLING SEQUENCE:
;    96          0157  1 !
;    97          0158  1 !       Enumerated.wlu.v = PAS$READ_ENUMERATED (PFV.mr.r , PETD.r.r [, ERROR.ja.r])
;    98          0159  1 !
;    99          0160  1 ! FORMAL PARAMETERS:
;   100          0161  1 !
;   101          0162  1 !       PFV             - The Pascal File Variable (PFV) passed by reference.
;   102          0163  1 !                         The structure of the PFV is defined in PASPFV.REQ.
;   103          0164  1 !
;   104          0165  1 !       PETD            - Pascal Enumerated Type Descriptor, passed by reference.
;   105          0166  1 !                         The structure of a PETD is as follows:
;   106          0167  1 !
;   107          0168  1 !                           +-------------------------------+
;   108          0169  1 !                           |   offset of ASCIC type name   |   <-- PETD
;   109          0170  1 !                           +-------------------------------+
;   110          0171  1 !                           | count of possible values (n)  |
;   111          0172  1 !                           +-------------------------------+
;   112          0173  1 !                           | offset of ASCIC value name 0  |
;   113          0174  1 !                           +-------------------------------+
;   114          0175  1 !                           | offset of ASCIC value name 1  |
;   115          0176  1 !                           +- - - - - - - - - - - - - - - -+
;   116          0177  1 !                                          ...
;   117          0178  1 !                           +- - - - - - - - - - - - - - - -+
;   118          0179  1 !                           | offset of ASCIC value name n-1|
;   119          0180  1 !                           +-------------------------------+
;   120          0181  1 !
;   121          0182  1 !                           The offsets are relative to the address
;   122          0183  1 !                           of the descriptor (PETD).  The names are
;   123          0184  1 !                           counted strings with 1-byte counts. It
;   124          0185  1 !                           is assumed that the compiler has upcased
;   125          0186  1 !                           all of the strings.
;   126          0187  1 !
;   127          0188  1 !       ERROR           - Optional.  If specified, the address to unwind to
;   128          0189  1 !                         in case of an error.
;   129          0190  1 !
;   130          0191  1 ! IMPLICIT INPUTS:
;   131          0192  1 !
;   132          0193  1 !       NONE
;   133          0194  1 !
;   134          0195  1 ! IMPLICIT OUTPUTS:
;   135          0196  1 !
;   136          0197  1 !       NONE
;   137          0198  1 !
;   138          0199  1 ! ROUTINE VALUE:
```

```
;  139          0200  1 !
;  140          0201  1 !       The value of the enumerated value read.
;  141          0202  1 !
;  142          0203  1 ! SIDE EFFECTS:
;  143          0204  1 !
;  144          0205  1 !       If the file is the standard file INPUT or OUTPUT, it is implicitly opened.
;  145          0206  1 !
;  146          0207  1 ! SIGNALLED ERRORS:
;  147          0208  1 !
;  148          0209  1 !       INVSYNENU - invalid syntax for enumerated value
;  149          0210  1 !       AMBVALENU - "string" is an ambiguous value for enumerated type "type"
;  150          0211  1 !       NOTVALTYP - "string" is not a value of type "type"
;  151          0212  1 !
;  152          0213  1 !--
;  153          0214  1
;  154          0215  2     BEGIN
;  155          0216  2
;  156          0217  2     LOCAL
;  157          0218  2         RESULT,                         ! Result value
;  158          0219  2         STRING: REF VECTOR [, BYTE],    ! Field in line read
;  159          0220  2         STRING_LENGTH,                  ! Length of STRING
;  160          0221  2         MATCH_COUNT,                    ! Count of values matched
;  161          0222  2         FCB: REF $PAS$FCB_CONTROL_BLOCK,! File Control block
;  162          0223  2         PFV_ADDR: VOLATILE,             ! Enable argument
;  163          0224  2         UNWIND_ACT: VOLATILE,           ! Enable argument
;  164          0225  2         ERROR_ADDR: VOLATILE;           ! Enable argument
;  165          0226  2
;  166          0227  2     BUILTIN
;  167          0228  2         ACTUALCOUNT;                    ! Count of arguments
;  168          0229  2
;  169          0230  2     BIND
;  170          0231  2         TYPE_ENUMERATED = UPLIT BYTE (%CHARCOUNT('-enumerated-'),'-enumerated-');
;  171          0232  2
;  172          0233  2     ENABLE
;  173          0234  2         PAS$$IO_HANDLER (PFV_ADDR, UNWIND_ACT, ERROR_ADDR);     ! Enable error handler
;  174          0235  2
;  175          0236  2     !+
;  176          0237  2     ! Get ERROR parameter, if present.
;  177          0238  2     !-
;  178          0239  2
;  179          0240  2     IF ACTUALCOUNT () GEQU 3
;  180          0241  2     THEN
;  181          0242  2         ERROR_ADDR = .ERROR;            ! Set unwind address
;  182          0243  2
;  183          0244  2     PFV_ADDR = PFV [PFV$R_PFV];         ! Set PFV address
;  184          0245  2
;  185          0246  2     !+
;  186          0247  2     ! Validate PFV and get PFV.
;  187          0248  2     !-
;  188          0249  2
;  189          0250  2     PAS$$VALIDATE_PFV (PFV [PFV$R_PFV]; FCB);
;  190          0251  2
;  191          0252  2     !+
;  192          0253  2     ! Set unwind action to unlock file.
;  193          0254  2     !-
;  194          0255  2
;  195          0256  2     UNWIND_ACT = PAS$K_UNWIND_UNLOCK;
```

```
;  196          0257  2
;  197          0258  2      !+
;  198          0259  2      ! Do common initialization.
;  199          0260  2      !-
;  200          0261  2
;  201          0262  2      PAS$$INIT_READ (PFV [PFV$R_PFV], FCB [FCB$R_FCB]; FCB);
;  202          0263  2
;  203          0264  2      RESULT = 0;          ! Indicate no value found yet.
;  204          0265  2
;  205          0266  2      !+
;  206          0267  2      ! Call utility routine to find a string that looks like an enumerated value.
;  207          0268  2      ! If we can't find one, signal an error.
;  208          0269  2      !-
;  209          0270  2
;  210          0271  2      IF NOT PAS$$GET_ENUMERATED (PFV [PFV$R_PFV], FCB [FCB$R_FCB];
;  211          0272  2          STRING, STRING_LENGTH, FCB)
;  212          0273  2      THEN
;  213        P 0274  2          $PAS$IO_ERROR (PAS$_INVSYNENU,3,.STRING_LENGTH,
;  214          0275  2              STRING [0], .FCB [FCB$L_RECORD_NUMBER]);    ! Invalid syntax
;  215          0276  2
;  216          0277  2      !+
;  217          0278  2      ! Maximum length of user string that we look at is 31 characters.
;  218          0279  2      !-
;  219          0280  2
;  220          0281  2      IF .STRING_LENGTH GTRU 31
;  221          0282  2      THEN
;  222          0283  2          STRING_LENGTH = 31;
;  223          0284  2      STRING = .STRING - 1;       ! Offset so that same index can be used for
;  224          0285  2                                  ! type-value and user string.
;  225          0286  2
;  226          0287  2      !+
;  227          0288  2      ! Scan through the PETD and look for matches.  Upcase user's input and
;  228          0289  2      ! check for ambiguous values.
;  229          0290  2      !-
;  230          0291  2
;  231          0292  2      MATCH_COUNT = 0;
;  232          0293  2      INCR IDX FROM 1 TO .PETD [1] DO
;  233          0294  3          BEGIN
;  234          0295  3          LOCAL
;  235          0296  3              TYPE_VAL: REF VECTOR [, BYTE],      ! Type value
;  236          0297  3              VAL_LEN;                            ! Length of value string
;  237          0298  3          TYPE_VAL = .PETD + .PETD [.IDX+1];      ! Get next value string
;  238          0299  3          VAL_LEN = .TYPE_VAL [0];                ! Get length.
;  239          0300  3
;  240          0301  3          !+
;  241          0302  3          ! If user string length is longer than this value, skip this
;  242          0303  3          ! value.  Otherwise, see if they match.
;  243          0304  3          !-
;  244          0305  3
;  245          0306  3          IF .STRING_LENGTH LEQU .VAL_LEN
;  246          0307  3          THEN
;  247          0308  4              BEGIN
;  248          0309  5              IF (INCR POS FROM 1 TO .STRING_LENGTH DO
;  249          0310  6                  BEGIN
;  250          0311  6                  IF .TYPE_VAL [.POS] NEQ
;  251          0312  7                      (
;  252          0313  7                      IF (.STRING [.POS] GEQ %C'a') AND
```

```
;  253          0314  8                        (.STRING [.POS] LEQ %C'z')
;  254          0315  7                    THEN
;  255          0316  8                        .STRING [.POS] - (%C'a' - %C'A')
;  256          0317  7                    ELSE
;  257          0318  7                        .STRING [.POS]
;  258          0319  7                    )
;  259          0320  6                THEN
;  260          0321  6                    EXITLOOP 0; ! Mismatch found
;  261          0322  4                END) NEQ 0
;  262          0323  4            THEN
;  263          0324  5                BEGIN
;  264          0325  5                !+
;  265          0326  5                ! Match found.  If exact match, set RESULT, reset
;  266          0327  5                ! MATCH_COUNT to 1 and exit the loop.  If it is a partial
;  267          0328  5                ! match, set RESULT, increment MATCH_COUNT, and continue.
;  268          0329  5                !-
;  269          0330  5
;  270          0331  5                IF .STRING_LENGTH EQL .VAL_LEN
;  271          0332  5                THEN
;  272          0333  6                    BEGIN
;  273          0334  6                    !+
;  274          0335  6                    ! Exact match
;  275          0336  6                    !-
;  276          0337  6                    RESULT = .IDX;
;  277          0338  6                    MATCH_COUNT = 1;
;  278          0339  6                    EXITLOOP;
;  279          0340  5                    END;
;  280          0341  5                RESULT = .IDX;          ! Set this result
;  281          0342  5                MATCH_COUNT = .MATCH_COUNT + 1; ! Increment count of matches
;  282          0343  4                END;
;  283          0344  3            END;
;  284          0345  3
;  285          0346  2        END;    ! Of INCR loop
;  286          0347  2
;  287          0348  2    !+
;  288          0349  2    ! Check for no matches (MATCH_COUNT LSS 1) or ambiguous match
;  289          0350  2    ! (MATCH_COUNT GTR 1).
;  290          0351  2    !+
;  291          0352  2
;  292          0353  2    IF .MATCH_COUNT NEQ 1       ! Not exactly one match?
;  293          0354  2    THEN
;  294          0355  3        BEGIN
;  295          0356  3        LOCAL
;  296          0357  3            DESCR: BLOCK [8, BYTE];
;  297          0358  3        DESCR [DSC$W_LENGTH] = .STRING_LENGTH;
;  298          0359  3        DESCR [DSC$B_CLASS] = 0;
;  299          0360  3        DESCR [DSC$B_DTYPE] = 0;
;  300          0361  3        DESCR [DSC$A_POINTER] = STRING [1];
;  301          0362  3        IF .MATCH_COUNT EQL 0
;  302          0363  3        THEN
;  303        P 0364  3            $PAS$IO_ERROR (PAS$_NOTVALTYP,3,DESCR,
;  304        P 0365  3                (IF .PETD [0] EQL 0
;  305        P 0366  3                THEN
;  306        P 0367  3                    TYPE_ENUMERATED
;  307        P 0368  3                ELSE
;  308        P 0369  3                    .PETD + .PETD [0]),
;  309          0370  4                .FCB [FCB$L_RECORD_NUMBER])     ! Not a value of type
```

```
;  310          0371  3          ELSE
;  311        P 0372  3              $PAS$IO_ERROR (PAS$_AMBVALENU,3,DESCR,
;  312        P 0373  3                  (IF .PETD [0] EQL 0
;  313        P 0374  3                  THEN
;  314        P 0375  3                      TYPE_ENUMERATED
;  315        P 0376  3                  ELSE
;  316        P 0377  3                      .PETD + .PETD [0]),
;  317          0378  3                  .FCB [FCB$L_RECORD_NUMBER]);    ! Ambiguous value
;  318          0379  2          END;
;  319          0380  2
;  320          0381  2      !+
;  321          0382  2      ! Do end-of-READ processing.
;  322          0383  2      !-
;  323          0384  2
;  324          0385  2      PAS$$END_READ (PFV [PFV$R_PFV], FCB [FCB$R_FCB]);
;  325          0386  2
;  326          0387  2      RETURN .RESULT - 1; ! Compensate for loop starting at 1
;  327          0388  2
;  328          0389  1      END;                                    ! End of routine PAS$READ_ENUMERATED


                                                                     .TITLE  PAS$READ_ENUMERATED Read an enumerated value
                                                                     .IDENT  \1-002\

                                                                     .PSECT  _PAS$CODE,NOWRT,  SHR,  PIC,2

                                                          0C  00000 P.AAA:  .BYTE   12                                    ;
                2D  64  65  74  61  72  65  6D  75  6E  65  2D  00001       .ASCII  \-enumerated-\                          ;

                                                    TYPE_ENUMERATED=    P.AAA
                                                                     .EXTRN  PAS$READ_ENUMERATED
                                                                     .EXTRN  PAS$READV_ENUMERATED
                                                                     .EXTRN  PAS$$IO_HANDLER
                                                                     .EXTRN  PAS$$VALIDATE_PFV
                                                                     .EXTRN  PAS$$INIT_READ, PAS$$GET_ENUMERATED
                                                                     .EXTRN  PAS$$SIGNAL, PAS$K_INVSYNENU
                                                                     .EXTRN  PAS$K_NOTVALTYP
                                                                     .EXTRN  PAS$K_AMBVALENU
                                                                     .EXTRN  PAS$$END_READ

                                                        0FFC 00000       .ENTRY  PAS$READ_ENUMERATED, Save R2,R3,R4,R5,R6,-  ; 0144
                                                                                 R7,R8,R9,R10,R11                            ;
                                                    5E          14   C2 00002       SUBL2   #20, SP                         ;
                                                            08  AE   7C 00005       CLRQ    ERROR_ADDR                      ; 0215
                                                            10  AE   D4 00008       CLRL    PFV_ADDR                        ;
                                                    6D      0110 CF   DE 0000B       MOVAL   21$, (FP)                       ;
                                                    03           6C   91 00010       CMPB    (AP), #3                        ; 0240
                                                                 05   1F 00013       BLSSU   1$                              ;
                                                08  AE       0C  AC   D0 00015       MOVL    ERROR, ERROR_ADDR               ; 0242
                                                    50          04   AC   D0 0001A 1$:   MOVL    PFV, R6                         ; 0244
                                                10  AE           56   D0 0001E       MOVL    R6, PFV_ADDR                    ;
                                                        00000000G 00   16 00022       JSB     PAS$$VALIDATE_PFV               ; 0250
                                                0C  AE           01   D0 00028       MOVL    #1, UNWIND_ACT                  ; 0256
                                                        00000000G 00   16 0002C       JSB     PAS$$INIT_READ                  ; 0262
                                                                 59   D4 00032       CLRL    RESULT                          ; 0264
                                                        00000000G 00   16 00034       JSB     PAS$$GET_ENUMERATED             ; 0271
                                                    10           50   E8 0003A       BLBS    R0, 2$                          ;
```

```
                                C8    A7  DD 0003D          PUSHL   -56(FCB)                                  ; 0275
                                      54  DD 00040          PUSHL   STRING
                                      55  DD 00042          PUSHL   STRING_LENGTH
                                      03  DD 00044          PUSHL   #3
                          7E          00G 8F  9A 00046       MOVZBL  #PAS$K_INVSYNENU, -(SP)
                                        00BC  31 0004A       BRW     18$
                          1F              55  D1 0004D 2$:   CMPL    STRING_LENGTH, #31                      ; 0281
                                          03  1B 00050       BLEQU   3$
                          55              1F  D0 00052       MOVL    #31, STRING_LENGTH                      ; 0283
                                          54  D7 00055 3$:    DECL    STRING                                  ; 0284
                                          5B  D4 00057       CLRL    MATCH_COUNT                             ; 0292
                          52          08  AC  D0 00059       MOVL    PETD, R2                                ; 0293
                                          51  D4 0005D       CLRL    IDX                                     ; 0306
                                          54  11 0005F       BRB     11$
                53        52          04 A241 C1 00061 4$:   ADDL3   4(R2)[IDX], R2, TYPE_VAL               ; 0298
                          5A              63  9A 00067       MOVZBL  (TYPE_VAL), VAL_LEN                     ; 0299
                          5A              55  D1 0006A       CMPL    STRING_LENGTH, VAL_LEN                  ; 0306
                                          46  1A 0006D       BGTRU   11$
                                          50  D4 0006F       CLRL    POS                                     ; 0313
                                          27  11 00071       BRB     8$
                      61  8F              6044 91 00073 5$:   CMPB    (POS)[STRING], #97
                                          10  1F 00078       BLSSU   6$
                      7A  8F              6044 91 0007A       CMPB    (POS)[STRING], #122                     ; 0314
                                          09  1A 0007F       BGTRU   6$
                          58              6044 9A 00081       MOVZBL  (POS)[STRING], R8                       ; 0316
                          58                20 C2 00085       SUBL2   #32, R8
                                            04 11 00088       BRB     7$
                          58              6044 9A 0008A 6$:   MOVZBL  (POS)[STRING], R8                       ; 0318
            58      6043  08                00 ED 0008E 7$:   CMPZV   #0, #8, (POS)[TYPE_VAL], R8             ; 0312
                                            04 13 00094       BEQL    8$
                                          50  D4 00096       CLRL    R0                                      ; 0321
                                          07  11 00098       BRB     9$
                D5        50              55  F3 0009A 8$:   AOBLEQ  STRING_LENGTH, POS, 5$                  ; 0309
                          50              01  CE 0009E       MNEGL   #1, R0
                                          12  13 000A1 9$:   BEQL    11$                                     ; 0322
                          5A              55  D1 000A3       CMPL    STRING_LENGTH, VAL_LEN                  ; 0331
                                          08  12 000A6       BNEQ    10$
                          59              51  D0 000A8       MOVL    IDX, RESULT                             ; 0337
                          5B              01  D0 000AB       MOVL    #1, MATCH_COUNT                         ; 0338
                                          0A  11 000AE       BRB     12$                                     ; 0333
                          59              51  D0 000B0 10$:   MOVL    IDX, RESULT                             ; 0341
                                          5B  D6 000B3       INCL    MATCH_COUNT                             ; 0342
                A7        51      04      A2  F3 000B5 11$:   AOBLEQ  4(R2), IDX, 4$                         ; 0293
                          01              5B  D1 000BA 12$:   CMPL    MATCH_COUNT, #1                         ; 0353
                                          53  13 000BD       BEQL    19$
                          6E              55  3C 000BF       MOVZWL  STRING_LENGTH, DESCR                    ; 0358
                  04      AE      01      A4  9E 000C2       MOVAB   1(R4), DESCR+4                          ; 0361
                                          5B  D5 000C7       TSTL    MATCH_COUNT                             ; 0362
                                          20  12 000C9       BNEQ    15$
                                C8        A7  DD 000CB       PUSHL   -56(FCB)                                ; 0370
                                          62  D5 000CE       TSTL    (R2)
                                          09  12 000D0       BNEQ    13$
                          50      FF1D    CF  9E 000D2       MOVAB   TYPE_ENUMERATED, R0
                                          50  DD 000D7       PUSHL   R0
                                          05  11 000D9       BRB     14$
                          52              62  C0 000DB 13$:   ADDL2   (R2), R2
                                          52  DD 000DE       PUSHL   R2
```

```
                                 08   AE  9F 000E0 14$:    PUSHAB  DESCR                                    ;
                                 03   DD 000E3            PUSHL   #3                                       ;
                           7E    00G  8F  9A 000E5        MOVZBL  #PAS$K_NOTVALTYP, -(SP)                  ;
                                      1E  11 000E9        BRB     18$                                      ;
                                 C8   A7  DD 000EB 15$:    PUSHL   -56(FCB)                               ; 0378
                                      62  D5 000EE        TSTL    (R2)                                     ;
                                      09  12 000F0        BNEQ    16$                                      ;
                           50    FEFD CF  9E 000F2        MOVAB   TYPE_ENUMERATED, R0                      ;
                                      50  DD 000F7        PUSHL   R0                                       ;
                                      05  11 000F9        BRB     17$                                      ;
                           52         62  C0 000FB 16$:    ADDL2   (R2), R2                               ;
                                      52  DD 000FE        PUSHL   R2                                       ;
                                 08   AE  9F 00100 17$:    PUSHAB  DESCR                                  ;
                                 03   DD 00103            PUSHL   #3                                       ;
                           7E    00G  8F  9A 00105        MOVZBL  #PAS$K_AMBVALENU, -(SP)                  ;
                  00000000G  00         05  FB 00109 18$:    CALLS   #5, PAS$$SIGNAL                        ;
                                      0A  11 00110        BRB     20$                                      ;
                      00000000G   00    16  00112 19$:    JSB     PAS$$END_READ                            ; 0385
                           50         79  9E 00118        MOVAB   -(R9), R0                                ; 0387
                                      04  00118          RET                                              ;
                                 50   D4  0011C 20$:    CLRL    R0                                       ; 0389
                                      04  0011E          RET                                              ;
                                    0000  0011F 21$:    .WORD   Save nothing                             ; 0215
                           50    08   AC  D0 00121        MOVL    8(AP), R0                                ;
                           50    04   A0  D0 00125        MOVL    4(R0), R0                                ;
                                 F4   A0  9F 00129        PUSHAB  ERROR_ADDR                               ;
                                 F8   A0  9F 0012C        PUSHAB  UNWIND_ACT                               ;
                                 FC   A0  9F 0012F        PUSHAB  PFV_ADDR                                 ;
                                      03  DD 00132        PUSHL   #3                                       ;
                                      5E  DD 00134        PUSHL   SP                                       ;
                           7E    04   AC  7D 00136        MOVQ    4(AP), -(SP)                             ;
                  00000000G  00         03  FB 0013A        CALLS   #3, PAS$$IO_HANDLER                    ;
                                      04  00141          RET                                              ;

; Routine Size:  322 bytes,    Routine Base:  _PAS$CODE + 000D

;  329          0390  1
;  330          0391  1 !<BLF/PAGE>
```

```
;  332     0392  1 %SBTT' 'PAS$READV_ENUMERATED - Read an enumerated value from string'
;  333     0393  1 GLOBAL ROUTINE PAS$READV_ENUMERATED (
;  334     0394  1         STRING: REF BLOCK [, BYTE],                     ! String descriptor
;  335     0395  1         PETD: REF VECTOR [, LONG],                      ! Enumerated type descriptor
;  336     0396  1         ERROR                                           ! Error unwind address
;  337     0397  1     ) =
;  338     0398  1
;  339     0399  1 !++
;  340     0400  1 ! FUNCTIONAL DESCRIPTION:
;  341     0401  1 !
;  342     0402  1 !       This function reads an enumerated value from the specified string
;  343     0403  1 !       and returns it as the function value.
;  344     0404  1 !
;  345     0405  1 ! CALLING SEQUENCE:
;  346     0406  1 !
;  347     0407  1 !       Enumerated.wl.v = PAS$READV_ENUMERATED (STRING.mt.ds , PETD.rr.r [, ERROR.ja.r])
;  348     0408  1 !
;  349     0409  1 ! FORMAL PARAMETERS:
;  350     0410  1 !
;  351     0411  1 !       STRING          - The string to read from, passed as a class S
;  352     0412  1 !                         (assumed) descriptor.  The length and pointer
;  353     0413  1 !                         are updated to reflect the unread string.
;  354     0414  1 !
;  355     0415  1 !       PETD            - Pascal Enumerated Type Descriptor, passed by reference.
;  356     0416  1 !                         The structure of a PETD is as follows:
;  357     0417  1 !
;  358     0418  1 !                           +-------------------------------+
;  359     0419  1 !                           |   offset of ASCIC type name   |   <-- PETD
;  360     0420  1 !                           +-------------------------------+
;  361     0421  1 !                           | count of possible values (n)  |
;  362     0422  1 !                           +-------------------------------+
;  363     0423  1 !                           | offset of ASCIC value name 0  |
;  364     0424  1 !                           +-------------------------------+
;  365     0425  1 !                           | offset of ASCIC value name 1  |
;  366     0426  1 !                           +- - - - - - - - - - - - - - - -+
;  367     0427  1 !                                        ...
;  368     0428  1 !                           +- - - - - - - - - - - - - - - -+
;  369     0429  1 !                           | offset of ASCIC value name n-1|
;  370     0430  1 !                           +-------------------------------+
;  371     0431  1 !
;  372     0432  1 !                           The offsets are relative to the address
;  373     0433  1 !                           of the descriptor (PETD).  The names are
;  374     0434  1 !                           counted strings with 1-byte counts. It
;  375     0435  1 !                           is assumed that the compiler has upcased
;  376     0436  1 !                           all of the strings.
;  377     0437  1 !
;  378     0438  1 !       ERROR           - Optional.  If specified, the address to unwind to
;  379     0439  1 !                         in case of an error.
;  380     0440  1 !
;  381     0441  1 ! IMPLICIT INPUTS:
;  382     0442  1 !
;  383     0443  1 !       NONE
;  384     0444  1 !
;  385     0445  1 ! IMPLICIT OUTPUTS:
;  386     0446  1 !
;  387     0447  1 !       NONE
;  388     0448  1 !
```

```
;  389          0449  1 ! ROUTINE VALUE:
;  390          0450  1 !
;  391          0451  1 !       The value of the enumerated value read.
;  392          0452  1 !
;  393          0453  1 ! SIDE EFFECTS:
;  394          0454  1 !
;  395          0455  1 !       NONE
;  396          0456  1 !
;  397          0457  1 ! SIGNALLED ERRORS:
;  398          0458  1 !
;  399          0459  1 !       NONE
;  400          0460  1 !
;  401          0461  1 !--
;  402          0462  1
;  403          0463  2     BEGIN
;  404          0464  2
;  405          0465  2     LOCAL
;  406          0466  2         PFV: $PAS$PFV_FILE_VARIABLE,    ! Pascal File Variable
;  407          0467  2         RESULT,                         ! Result value
;  408          0468  2         ARG_LIST: VECTOR [3, LONG],     ! Argument list
;  409          0469  2         PFV_ADDR: VOLATILE,             ! Enable argument
;  410          0470  2         UNWIND_ACT: VOLATILE,           ! Enable argument
;  411          0471  2         ERROR_ADDR: VOLATILE;           ! Enable argument
;  412          0472  2
;  413          0473  2     BUILTIN
;  414          0474  2         ACTUALCOUNT;                    ! Count of arguments
;  415          0475  2
;  416          0476  2     ENABLE
;  417          0477  2         PAS$$IO_HANDLER (PFV_ADDR, UNWIND_ACT, ERROR_ADDR);     ! Enable error handler
;  418          0478  2
;  419          0479  2     !+
;  420          0480  2     ! Get ERROR parameter, if present.
;  421          0481  2     !-
;  422          0482  2
;  423          0483  2     IF ACTUALCOUNT () GEQU 3
;  424          0484  2     THEN
;  425          0485  2         ERROR_ADDR = .ERROR;            ! Set unwind address
;  426          0486  2
;  427          0487  2     PFV_ADDR = PFV [PFV$R_PFV];         ! Set PFV address
;  428          0488  2
;  429          0489  2     !+
;  430          0490  2     ! Set up ARG_LIST.
;  431          0491  2     !-
;  432          0492  2
;  433          0493  2     ARG_LIST [0] = 2;                   ! Two arguments
;  434          0494  2     ARG_LIST [1] = PFV [PFV$R_PFV];     ! PFV address
;  435          0495  2     ARG_LIST [2] = PETD [0];            ! Enumerated type descriptor
;  436          0496  2
;  437          0497  2     !+
;  438          0498  2     ! Call PAS$$DO_READV to do the work, giving it the address of
;  439          0499  2     ! PAS$READ_ENUMERATED to call.
;  440          0500  2     !-
;  441          0501  2
;  442          0502  2     PAS$$DO_READV (PFV [PFV$R_PFV], .STRING, ARG_LIST, PAS$READ_ENUMERATED;
;  443          0503  2         RESULT);
;  444          0504  2
;  445          0505  2     RETURN .RESULT;
```

```
;  446          0506  2
;  447          0507  1     END;                                ! End of routine PAS$READV_ENUMERATED


                                                        .EXTRN  PAS$$DO_READV

                                        005C 00000      .ENTRY  PAS$READV_ENUMERATED, Save R2,R3,R4,R6      ; 0393
                                    5E          24  C2  00002   SUBL2   #36, SP                            ;
                                                7E  D4  00005   CLRL    ERROR_ADDR                          ; 0463
                                          04    AE  7C  00007   CLRQ    UNWIND_ACT                          ;
                                    6D    0035  CF  DE  0000A   MOVAL   2$, (FP)                            ;
                                    03          6C  91  0000F   CMPB    (AP), #3                            ; 0483
                                                04  1F  00012   BLSSU   1$                                  ;
                                    6E    0C    AC  D0  00014   MOVL    ERROR, ERROR_ADDR                   ; 0485
                              08    AE    18    AE  9E  00018 1$: MOVAB   PFV, PFV_ADDR                      ; 0487
                              0C    AE          02  D0  0001D   MOVL    #2, ARG_LIST                        ; 0493
                              10    AE    18    AE  9E  00021   MOVAB   PFV, ARG_LIST+4                     ; 0494
                              14    AE    08    AC  D0  00026   MOVL    PETD, ARG_LIST+8                    ; 0495
                                    54  FE8F    CF  9E  0002B   MOVAB   PAS$READ_ENUMERATED, R4             ; 0502
                                    53    0C    AE  9E  00030   MOVAB   ARG_LIST, R3                        ;
                                    56    18    AE  9E  00034   MOVAB   PFV, R6                             ;
                                    52    04    AC  D0  00038   MOVL    STRING, R2                          ;
                                00000000G 00      16  0003C   JSB     PAS$$DO_READV                       ;
                                                04  00042   RET                                         ; 0507
                                              0000  00043 2$:   .WORD   Save nothing                       ; 0463
                                    50    08    AC  D0  00045   MOVL    8(AP), R0                           ;
                                    50    04    A0  D0  00049   MOVL    4(R0), R0                           ;
                                          D8    A0  9F  0004D   PUSHAB  ERROR_ADDR                          ;
                                          DC    A0  9F  00050   PUSHAB  UNWIND_ACT                         ;
                                          E0    A0  9F  00053   PUSHAB  PFV_ADDR                           ;
                                                03  DD  00056   PUSHL   #3                                 ;
                                                5E  DD  00058   PUSHL   SP                                 ;
                                    7E    04    AC  7D  0005A   MOVQ    4(AP), -(SP)                        ;
                          00000000G 00          03  FB  0005E   CALLS   #3, PAS$$IO_HANDLER                 ;
                                                04  00065   RET                                         ;

; Routine Size:  102 bytes,    Routine Base:  _PAS$CODE + 014F

;  448          0508  1
;  449          0509  1 !<BLF/PAGE>
```

```
;  451          0510  1 END                                   ! End of module PAS$READ_ENUMERATED
;  452          0511  1
;  453          0512  0 ELUDOM
```

```
;                                   PSECT SUMMARY

;       Name                    Bytes                           Attributes

;  _PAS$CODE                        437  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
```

```
;                                 Library Statistics

;                                        -------- Symbols --------      Pages       Processing
;       File                             Total    Loaded   Percent      Mapped      Time

;  _$255$DUA28:[SYSLIB]STARLET.L32;1       9776         4         0         581        00:01.0
;  _$255$DUA28:[PASRTL.OBJ]PASLIB.L32;1     427        99        23          33        00:00.4
```

```
;                                  COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/NOTRACE/LIS=LIS$:PASREAENU/OBJ=OBJ$:PASREAENU MSRC$:PASREAENU/UPDATE=(ENH$:PASREAENU
;       )

; Size:          424 code + 13 data bytes
; Run Time:          00:10.6
; Elapsed Time:      00:35.7
; Lines/CPU Min:      2892
; Lexemes/CPU-Min: 15163
; Memory Used:  134 pages
; Compilation Complete
```

PASREABOO
LIS

PASREAST1
LIS

PASREAREH
LIS

PASRESETK
LIS

PASREAREG
LIS

PASRAB
LIS

PASRESET2
LIS

PASREADLN
LIS

PASREAREF
LIS

PASREAVAR
LIS

PASREARED
LIS

PASREACHA
LIS

PASREADUT
LIS

PASREAENU
LIS

PASREAUNS
LIS

PASREAINT
LIS

PASREASTR
LIS