Data Entry Routines

General Information

Author: Paul B. Loux
Language: ACTION!
Compiler/Interpreter: ACTION!
Published: 1986

Description: Universal string input routine, offering full control over keyboard input and screen display. Allows program to limit responses to acceptable parameters.

Calling parameters:

FIELD	The field buffer.	
MIN	Minimum number of characters for valid response, 0-MAX.	
MAX	Maximum number of characters, 1-36.	
TYPEC	Type Code (see below)	
XIT	Exit record if the first character in FIELD is ESC.	
COL	Screen display horizontal position for input echo, 2-37.	
ROW	Screen display vertical position for input echo, 1-22.	
ERRPTR	Pointer variable to pass error code on record aborts (Ctrl-Z) or XIT's (above).	

Note: User entry of ESC will restart field entry, or exit (see above). Entry of Ctrl-Z aborts record. The routine uses the BYTE FUNC Fetch() to obtain the keystokes, allowing timeout control.

(Note: no range check is provided on the numeric response)

Type Codes

code	type	
1	Alphanumeric	
2	Force Upper Case	
3	Signed integer	
4	Signed real (float)	
5	Unsigned integer	
6	Unsigned real	
7	Yes/No check	

EntryS#

```
;**********************************
;*                                *
;*(C)Copyright 1986 by Paul B. Loux *
```

```
;*                                     *
;* These routines are in the public *
;* domain,  and  are not to be sold *
;* for a profit. They may be freely *
;* distributed, provided  that this *
;* header remains in place. Use and *
;* enjoy! PBL, CIS 72337,2073.      *
;*                                     *
;***************************************
;*                                     *
;*  File "ENTRYS.LIB"                *
;*                                     *
;*  Description: Universal string   *
;*   input routine, offering full  *
;*   control over keyboard input    *
;*   and screen display.  Allows    *
;*   program to limit responses  *
;*   to acceptable parameters.    *
;*                                     *
;*  Calling parameters:              *
;*                                     *
;*   FIELD  The field buffer.     *
;*                                     *
;*   MIN      Minimum number of   *
;*              characters for        *
;*              valid response, 0-MAX.*
;*                                     *
;*   MAX      Maximum number of   *
;*              characters, 1-36.     *
;*                                     *
;*   TYPEC  Type Code:               *
;*              1 Alphanumeric        *
;*              2 Force Upper Case   *
;*              3 Signed integer        *
;*              4 Signed real (float) *
;*              5 Unsigned integer   *
;*              6 Unsigned real      *
;*              7 Yes/No check        *
;*              (Note: no range check *
;*              is provided on the   *
;*              numeric response)     *
;*                                     *
;*   XIT      Exit record if the     *
;*              first character in   *
;*              FIELD is ESC.           *
;*                                     *
;*   COL      Screen display          *
;*              horizontal position *
;*              for input echo, 2-37. *
;*                                     *
;*   ROW      Screen display          *
;*              vertical position     *
;*              for input echo, 1-22. *
;*                                     *
;*   ERRPTR  Pointer variable to    *
;*              pass error code on   *
;*              record aborts (Ctrl-Z)*
;*              or XIT's (above).     *
;*                                     *
```

```
;*   Note:  User entry of ESC will*
;*              restart field entry,  *
;*              or exit (see above).  *
;*              Entry of Ctrl-Z aborts*
;*              record. The routine *
;*              uses the BYTE FUNC   *
;*              Fetch() to obtain the *
;*              keystokes, allowing *
;*              timeout control.        *
;*                                           *
;*************************************
;
;   Atari OS Variables for sound
;   control and PROC Sound_reset.
;
MODULE

BYTE AudCtl=$D208, SKCtl=$D20F,
      AudC1 =$D201, AudC2=$D203,
      AudC3 =$D205, AudC4=$D207

PROC Sound_reset()


AudCtl=0 SKCtl=3
AudC1 =0 AudC2=0 AudC3=0 AudC4=0

RETURN
;
;*************************************
;
;   BYTE FUNC FetchD
;     (BYTE ioch,Time_out)
;
;   This function is an enhancement
;   to the library  routine GetD().
;   Basically it is a GetD() with a
;   timeout spec.  The routine uses
;   the Atari System Timer 4, whose
;   counter is located at $21E,$21F
;   and  whose zero-flag is located
;   at $22C. Normally, the function
;   RETURNs the  same ATASCII  code
;   that a  GetD() would.  However,
;   if no key is pressed  prior  to
;   time-out, the function RETURNs
;   an ATASCII value of 255. Should
;   a user actually try to  enter a
;   character-255, the key sequence
;   is ignored.
;
;   Through use of the  OS variable
;   POKMSK, FetchD()  disables  the
;   Break key upon each call.
;
;   The value in the  BYTE Time_Out
;   represents  the  time limit  in
;   seconds. A value of zero  means
;   that there  is  no time  limit.
```

```
;   The maximum value for Time_out,
;   255,  represents 4 minutes  and
;   fifteen seconds.  If  more time
;   is required,  try using a short
;   Time_out but use a loop to make
;   the call to FetchD(), or modify
;   the routine to use a CARD.
;
;   The  BYTE ioch  represents  the
;   IOCB  channel from  which input
;   is to be obtained. The  routine
;   assumes the channel has already
;   been opened for input from K:.
;
;

BYTE FUNC FetchD(BYTE ioch,Time_Out)

BYTE POKMSK=$10      ; IRQ nabl bits
CARD CDTMV3=$21E        ; timer 4 value
BYTE CDTMF3=$22C        ; timer 4 flag
BYTE CH=$2FC             ; key pressed
BYTE IRQEN=$D20E        ; Pokey IRQ

CARD jiffies
BYTE response

POKMSK==&127             ; disable Break
IRQEN==&127

jiffies=Time_Out*60
CH=255
CDTMF3=255
CDTMV3=jiffies

DO
 IF CH#255 THEN
  response=GETD(ioch)
  IF response#255 THEN
    RETURN(response)
  FI
 FI
UNTIL CDTMF3=0
OD

;POKMSK==%128           ; uncomment to
;IRQEN==%128             ; restore BRKKEY

RETURN(255)
;
;**********************************
;
;   This routine places a message
;   at the bottom  of the screen.
;

PROC MSG(BYTE code)

BYTE ARRAY mesag
```

```
CARD ARRAY index(10)
CARD ctr

BYTE ROWCRS=$54,row
CARD COLCRS=$55,col

index(0)="                                                  "
index(1)="Too short-- continue"
index(2)="Too long-- press RETURN or Edit"
index(3)="Record aborted-- press ESC to clear"
index(4)="Numeric input only-- continue"
index(5)="Positive numbers only-- continue"
index(6)="Integers only-- continue"
index(7)="Value out of range-- press ESC"
index(8)="Invalid date-- press ESC"
index(9)="PROGRAM ERROR-- press ESC to bypass"


IF code>9 THEN code=9 FI
mesag=index(code)

IF code>0 THEN Sound_reset()
                    SOUND(0,20,10,10)
                    FOR ctr=1 TO 2000
                      DO
                        ;
                      OD
                    Sound_reset()
FI

row=ROWCRS
col=COLCRS

POSITION(1,23)
PUT(156)         ; delete line
PRINT(mesag)

IF code=9 THEN
  PUT(253)
FI

IF code=3 OR code=7 OR
    code=8 OR code=9 THEN
  DO
    ctr=FETCHD(7,10)         ; 10 second
  UNTIL ctr=27 OR ctr=255 ; timeout
  OD
FI

COLCRS=col+1
ROWCRS=row
PUT(30)

RETURN

;**********************************
;
;  This is the actual Entry routine.
;
```

```
PROC ENTRYS (BYTE ARRAY field
                BYTE min,max,typec,xit,
                col,row
                BYTE POINTER errptr)

BYTE dotflg,code,ctr,chr,intrpt,accept

BYTE ROWCRS=$54
CARD COLCRS=$55

DEFINE FILLCHR="46"     ; "."

BYTE INVFLG=$2B6
BYTE SHFLOK=$2BE
BYTE SHFTMP

IF col+max> 38 OR row>22 OR col<2
  OR max<min OR max<1 THEN
  MSG(9)
  RETURN
FI

SHFTMP=SHFLOK
SHFLOK=0
INVFLG=0
dotflg=0

MSG(0)

COLCRS=col
ROWCRS=row
FOR ctr=1 TO max DO PUT(FILLCHR)
                         field(ctr)=32
                    OD
field(0)=max               ; needed?
col==+1
ctr=0

DO
     accept=0
     intrpt=0
     COLCRS=col+ctr
     ROWCRS=row
     PUT(30)
     ctr==+1

 DO
     chr=FETCHD(7,30)    ; 30 second
                                ; timeout

     IF chr=255 THEN      ; timeout
            errptr^=3
            intrpt=1
            EXIT

     ELSEIF chr=155 THEN ; RETURN
            IF ctr<=min THEN
                  MSG(1)
```

```
       ELSE intrpt=1
            EXIT
       FI

ELSEIF chr=126 THEN ; BS
       IF ctr>1 THEN
         ctr==-1
         IF (typec=5 OR typec=6)
           AND field(ctr)=46
           THEN dotflg=0
         FI
         field(ctr)=32
         PUT(30)
         PUT(FILLCHR)
         PUT(30)
         MSG(0)
       FI

ELSEIF chr=26 THEN  ; Ctrl-Z
       FOR ctr=1 TO max
        DO
         field(ctr)=32
        OD
       ctr=0
       MSG(3)
       errptr^=2
       intrpt=1
       EXIT

ELSEIF chr=27 THEN  ; ESC
       IF ctr>1 THEN
        COLCRS=col-1
        ROWCRS=row
        FOR ctr=1 TO max
         DO
           PUT(FILLCHR)
           field(ctr)=32
         OD
        ctr=1
        dotflg=0
        COLCRS=col
        ROWCRS=row
        PUT(30)
       ELSE
        IF xit=1 THEN
         errptr^=1
         intrpt=1
         EXIT
        FI
       FI

ELSEIF ctr>max THEN
       MSG(2)

ELSEIF (chr<32
       OR chr>122
       OR chr=96)    ; Goofy Keys
       THEN SHFLOK=0
            INVFLG=0
```

```
    ELSEIF typec=7 THEN ;yes/no
           IF chr=110 OR  ;'n
               chr=121 THEN;'y
            chr==-32
           FI
           IF chr=89  OR  ;'Y
               chr=78  THEN;'N
            accept=1
           FI

    ELSEIF typec=2 THEN ; FORCE UC
           IF chr>96 AND chr<123 THEN
            chr==-32
           FI
           accept=1

    ELSEIF chr=45 THEN  ; "-"
           code=4
           IF ctr=1 THEN code=5 FI
           IF typec>4 THEN
            MSG(code)
           ELSEIF typec>2 THEN
            IF ctr>1 THEN
             MSG(4)
            ELSE accept=1
            FI
           ELSE accept=1
           FI

    ELSEIF chr=46 THEN  ; "."
           IF typec=4 OR typec=6 THEN
            IF dotflg=0 THEN
             dotflg=1
             accept=1
            ELSE MSG(4)
            FI
           ELSEIF typec=3
            OR typec=5 THEN
            MSG(6)
           ELSE accept=1
           FI

    ELSEIF typec>2 THEN ; digits only
           IF chr<48 OR chr>57 THEN
            MSG(4)
           ELSE accept=1
           FI

    ELSE accept=1

    FI

 UNTIL accept

OD

IF intrpt=1 THEN
   IF chr=155 THEN   ; RTN
```

```
     MSG(0)
     COLCRS=col-1
     ROWCRS=row
     field(0)=ctr-1
     PRINT(field)
     errptr^=0
    FI
    EXIT                    ; ESC,^Z,time
 ELSE PUT(chr)          ; accept chr
        field(ctr)=chr
        MSG(0)
 FI

OD

SHFLOK=SHFTMP

RETURN

MODULE                   ; continue

;
;**********************************
;
; END OF FILE.
;
;**********************************

;**********************************
;
; Example of usage of EntryS().

PROC Test()

BYTE x,y,min,max,typec,xit

BYTE ARRAY  name="xxxxxxxxxxxxxxx",
              id_no="xxxx",
              state="xx",
              price="xxxxxxxx"

BYTE errcde
BYTE POINTER errptr

errcde=0
errptr=@errcde

PUT(125)
POSITION(5,5)
PRINT("Enter Full Name: ")
x=22  y=5  min=0  max=15
typec=1 xit=1

EntryS(name,min,max,typec,
         xit,x,y,errptr)

POSITION(5,7)
PRINT("Enter I.D. Number: ")
x=24  y=7  min=4  max=4
```

```
typec=5 xit=0

EntryS(id_no,min,max,typec,
         xit,x,y,errptr)

POSITION(5,9)
PRINT("Enter State: ")
x=18  y=9  min=2  max=2
typec=2

EntryS(state,min,max,typec,
         xit,x,y,errptr)

POSITION(5,11)
PRINT("Enter Price: ")
x=18  y=11  min=0  max=8
typec=6

EntryS(price,min,max,typec,
         xit,x,y,errptr)

POSITION(5,14)
PUTE()
PRINTE(name)
PRINTE(id_no)
PRINTE(state)
PRINTE(price)

PUTE()
PRINTE("Done...")

RETURN
```

EntryYN#

```
;************************************
;*                                  *
;*(C)Copyright 1986 by Paul B. Loux *
;*                                  *
;* These routines are in the public *
;* domain,  and  are not to be sold *
;* for a profit. They may be freely *
;* distributed, provided  that this *
;* header remains in place. Use and *
;* enjoy! PBL, CIS 72337,2073.      *
;*                                  *
;************************************
;
;     File ENTRYYN.ACT
;
;     BYTE FUNC EntryYN()
;
;     Returns  either  one or zero
;     (true/false), representing a
;     one-character (Y or N)  user
;     response to yes/no questions
;     (uses EntryS(),the universal
```

```
;     string entry utility, to get
;     the keystroke).
;
;     The routine supports typical
;     EntryS() features, including
;     screen  coordinates supplied
;     by the calling routine; pass
;     through  of error codes, for
;     ESC and Ctrl-Z handling; and
;     timeouts. This  routine also
;     allows a default value for a
;     null-response (if desired).
;
;  Parameters:
;
;     col=screen echo horiz column
;     row=screen echo vert column
;
;     default= 0 for null="N"
;                 1 for null="Y"
;                 2 to disallow null
;
;     err_ptr= pointer to pass err
;                 to calling routine.
;
;***********************************
;
    INCLUDE "ENTRYS.ACT"
;
;***********************************
;
;
BYTE FUNC Ask_YN(BYTE col,row,default
                     BYTE POINTER err_ptr)

DEFINE max  ="1",
         typec="7",
         xit  ="0"

BYTE response,min
BYTE ARRAY field="x"

IF default=2 THEN
 min=1
ELSE
 min=0
FI

DO

ENTRYS(field,min,max,typec,xit,
         col,row,err_ptr)

IF err_ptr^#0 THEN RETURN(0) FI

IF field(0)=0 THEN       ; null entry
  IF default=0 THEN
     PUT('N)
  ELSE
```

```
     PUT('Y)
  FI
  RETURN(default)
FI

response=field(1)

IF response=89 THEN     ; 'Y
 RETURN(1)
ELSEIF response=78 THEN  ; 'N
 RETURN(0)
FI

OD

RETURN(0)
;
;
;**********************************
;
; Example of usage:

PROC Test8()

BYTE answer
BYTE x,y,default
BYTE errcde
BYTE POINTER err_ptr

errcde=0
err_ptr=@errcde

x=33  y=5
default=0

PUT(125)
POSITION(1,5)
PRINT("Do you own a computer (Y/[N]): ")
answer=Ask_YN(x,y,default,err_ptr)
POSITION(1,7)

IF answer THEN
  x=26  y=7
  default=1
  PRINT("Is it an Atari ([Y]/N): ")
  answer=Ask_YN(x,y,default,err_ptr)
  POSITION(1,9)

  IF answer THEN
     x=17 y=9
     default=2
     PRINT("Is it an 8-bit?      (Y/N)")
     answer=Ask_YN(x,y,default,err_ptr)
     POSITION(1,11)

     IF answer THEN
        PRINT("Congratulations.")
     ELSE
        x=9  y=11
```

```
        default=0
        PRINT("520 ST? (Y/[N])")
        answer=Ask_YN(x,y,default,err_ptr)
        POSITION(1,13)

        IF answer THEN
          PRINT("Congratulations.")
        ELSE
          PRINTE("Must be a 1040 ST then.")
        FI
     FI
  ELSE
     PRINTE("Too bad.")
  FI

ElSE
  x=32  y=7
  default=1

  POSITION(1,7)
  PRINTE("Too bad.")
FI

RETURN
```

EntryI#

```
;************************************
;*                                  *
;*(C)Copyright 1986 by Paul B. Loux *
;*                                  *
;* These routines are in the public *
;* domain,  and  are not to be sold *
;* for a profit. They may be freely *
;* distributed, provided  that this *
;* header remains in place. Use and *
;* enjoy! PBL, CIS 72337,2073.      *
;*                                  *
;************************************
;
;  CARD FUNC EntryI()
;
;  Universal integer-entry routine,
;  requires PROC EntryS(), the
;  universal string entry routine.
;  Includes range check, a null-
;  entry ok flag, and uses the
;  the same XIT flag as ENTRYS.
;
;  This routine takes input from
;  K: in string form (through
;  EntryS) and checks for legal
;  value (<=65535) and other useful
;  features before converting to
;  an actual INT value.
;
;  Use of EntryS allows the same
;  user interface (ESC and ^-Z
;  handling, timeouts, etc.)
```

```
;
;  Parameters are self-explanatory;
;  minval and maxval are the range
;  limits for acceptable response
;  (limted to +/-32767 of course);
;  the XIT and nullok flags are 1
;  for yes and 0 for no.
;
;**********************************
;
 INCLUDE "ENTRYS.ACT"
;
;**********************************

INT FUNC EntryI(BYTE col,row
                    INT minval,maxval
                    BYTE nullok,
                    xeq,xit
                    BYTE POINTER err_ptr)

BYTE ARRAY u_limit(0)="32767",
             l_limit(0)="-32767",
             field(0)="......"
BYTE fldlen=field

BYTE accept,min,max,typec
INT chk,tmp
INT value,tmpval

CARD temp,minchk,maxchk,offset

min=0
IF nullok=0 THEN
 IF minval<0 THEN
  temp=-minval
  min==+1
 ELSE
  temp=minval
 FI
 IF temp>0 THEN min==+1 FI
 IF temp>10 THEN min==+1 FI
 IF temp>100 THEN min==+1 FI
 IF temp>1000 THEN min==+1 FI
 IF temp>10000 THEN min==+1 FI
FI

max=1
IF maxval<0 THEN
 temp=-maxval
 max==+1
ELSE
 temp=maxval
FI
IF temp>0 THEN max==+1 FI
IF temp>10 THEN max==+1 FI
IF temp>100 THEN max==+1 FI
IF temp>1000 THEN max==+1 FI
IF temp>10000 THEN max==+1 FI
```

```
IF max<min THEN
 tmp=max
 max=min
 min=tmp
FI

typec=3                 ; signed int
accept=0
chk=0

DO
 ENTRYS(field,min,max,typec,xit,
          col,row,err_ptr)

 IF err_ptr^#0 THEN RETURN(0) FI
;calling routine does error handling

 IF fldlen=0 THEN
  field(1)='0
  field(0)=1
 FI

 IF fldlen=6 THEN
  chk=SCOMPARE(field,l_limit)
 ELSEIF fldlen=5 THEN
  IF field(1)#45 THEN  ;'-
    chk=SCOMPARE(field,u_limit)
  FI
 FI

 IF chk>0 THEN
  MSG(7)
 ELSE
  value=VALI(field)
  IF minval<0 THEN
     offset=-minval
     minchk=0
     maxval==+offset
     maxchk=maxval
     tmpval=value
     tmpval==+offset
     IF tmpval<0 THEN
      tmpval=maxval+1
     FI
     temp=tmpval
  ELSE
     temp=value
     maxchk=maxval
     minchk=minval
  FI
  IF temp<minchk or temp>maxchk
    THEN MSG(7)
  ELSE accept=1
  FI
 FI

UNTIL accept
OD
```

```
RETURN(value)

;***********************************
;
; Example of use of EntryC()

PROC Test4()

BYTE x,y,nullflg
INT min,max,value
BYTE errcde
BYTE POINTER err_ptr

errcde=0
err_ptr=@errcde

min=-20000
max=-1000

nullflg=0
x=19  y=7

PUT(125)
POSITION(5,5)
PUTE()
PRINTE("Enter a number between ")
PRINTI(min)
PRINT(" and ")
PRINTI(max)
PRINT(": ")

value=EntryI(x,y,min,max,nullflg,
                 0,0,err_ptr)

POSITION(5,17)
PUTE()
PRINTIE(value)
PRINTE("Done...")

RETURN
```

EntryC#

```
;***********************************
;*                                 *
;*(C)Copyright 1986 by Paul B. Loux *
;*                                 *
;* These routines are in the public *
;* domain,  and  are not to be sold *
;* for a profit. They may be freely *
;* distributed, provided  that this *
;* header remains in place. Use and *
;* enjoy! PBL, CIS 72337,2073.      *
;*                                 *
;***********************************
;
;  CARD FUNC EntryC()
;
```

```
;  Universal card-entry routine,
;  requires PROC EntryS(), the
;  universal string entry routine.
;  Includes range check, execute
;  on first digit flag, a null-
;  entry ok flag, and uses the
;  the same XIT flag as ENTRYS.
;
;  This routine takes input from
;  K: in string form (through
;  EntryS) and checks for legal
;  value (<=65535) and other useful
;  features before converting to
;  an actual CARD value.
;
;  Includes range check, execute
;  on single digit flag, a null-
;  entry ok flag, and uses the
;  the same xit flag as EntryS.
;  Use of EntryS allows the same
;  user interface (ESC and ^-Z
;  handling, timeouts, etc.)
;
;  Parameters are self-explanatory;
;  minval and maxval are the range
;  limits for acceptable response
;  (limted to 0-65535 of course);
;  the XEQ, XIT and nullok flags
;  are 1 or yes and 0 for no.
;
;**********************************
;
 INCLUDE "ENTRYS.ACT"
;
;**********************************

CARD FUNC EntryC(BYTE col,row
                      CARD minval,maxval
                      BYTE nullok,
                      xeq,xit
                      BYTE POINTER err_ptr)

BYTE ARRAY limit(0)="65535",
              field(0)="....."
BYTE fldlen=field

BYTE accept,min,max,typec
CARD value
INT chk

min=1
IF minval>10 THEN min==+1 FI
IF minval>100 THEN min==+1 FI
IF minval>1000 THEN min==+1 FI
IF minval>10000 THEN min==+1 FI
IF nullok THEN
 min=0
FI
```

```
IF maxval=0 THEN
 maxval=65535
 max=5
ELSE
 max=1
 IF maxval>10 THEN max==+1 FI
 IF maxval>100 THEN max==+1 FI
 IF maxval>1000 THEN max==+1 FI
 IF maxval>10000 THEN max==+1 FI
FI

typec=5                 ; pos int
accept=0
chk=0

DO
 ENTRYS(field,min,max,typec,xit,
          col,row,err_ptr)

 IF err_ptr^#0 THEN RETURN(0) FI
;Calling routine does error handling

 IF fldlen=0 THEN
  field(1)='0
  field(0)=1
 FI

 IF fldlen=5 THEN
  chk=SCOMPARE(field,limit)
 FI
 IF chk>0 THEN
  chk=0
  MSG(7)
 ELSE
  value=VALC(field)
  IF value<minval OR value>maxval
    THEN MSG(7)
  ELSE accept=1
  FI
 FI

UNTIL accept
OD

RETURN(value)

;*********************************
;
; Example of use of EntryC()

PROC Test3()

BYTE x,y,nullflg
CARD min,max,value
BYTE errcde
BYTE POINTER err_ptr

errcde=0
err_ptr=@errcde
```

```
min=1000
max=2000

nullflg=0
x=17  y=7

PUT(125)
POSITION(5,5)
PUTE()
PRINTE("Enter a number between ")
PRINTC(min)
PRINT(" and ")
PRINTC(max)
PRINT(": ")

value=EntryC(x,y,min,max,nullflg,
                 0,0,err_ptr)

POSITION(5,10)
PUTE()
PRINTCE(value)
PRINTE("Done...")

RETURN
```

EntryB#

```
;************************************
;*                                  *
;*(C)Copyright 1986 by Paul B. Loux *
;*                                  *
;* These routines are in the public *
;* domain,  and  are not to be sold *
;* for a profit. They may be freely *
;* distributed, provided  that this *
;* header remains in place. Use and *
;* enjoy! PBL, CIS 72337,2073.      *
;*                                  *
;************************************
;
;  FILE BYTE FUNC EntryB()
;
;  Universal byte-entry routine,
;  requires PROC EntryS, the
;  universal string entry routine.
;  This routine takes input from
;  K: in string form (through
;  EntryS) and checks for legal
;  value (<=255) and other useful
;  features before converting to
;  an actual BYTE value.
;
;  Includes range check, execute
;  on single digit flag, a null-
;  entry ok flag, and uses the
;  the same xit flag as EntryS.
```

```
;  Use of EntryS allows the same
;  user interface (ESC and ^-Z
;  handling, timeouts, etc.)
;
;  Parameters are self-explanatory;
;  minval and maxval are the range
;  limits for acceptable response
;  (limted to 0-255 of course); the
;  XEQ, XIT and nullok flags are
;  1 or yes and 0 for no.
;
;*********************************
;
 INCLUDE "ENTRYS.ACT"
;
;*********************************

BYTE FUNC EntryB(BYTE col,row,minval,
                     maxval,nullok,xeq,xit
                     BYTE POINTER err_ptr)

BYTE ARRAY limit(0)="255",
              field(0)="..."
BYTE fldlen=field

BYTE accept,min,max,
      typec,value

INT chk

min=0
IF minval>0 THEN min==+1 FI
IF minval>10 THEN min==+1 FI
IF minval>100 THEN min==+1 FI
IF nullok THEN
 min=0
FI

IF maxval=0 THEN
 maxval=255
 max=3
ELSE
 max=0
IF maxval>0 THEN max==+1 FI
IF maxval>10 THEN max==+1 FI
IF maxval>100 THEN max==+1 FI
FI

typec=5                 ; pos int
accept=0
chk=0

DO
 ENTRYS(field,min,max,typec,xit,
          col,row,err_ptr)

 IF err_ptr^#0 THEN RETURN(0) FI
;(calling routine does error handling)
```

```
 IF fldlen=0 THEN
  field(1)='0
  field(0)=1
 FI

 IF fldlen=3 THEN            ; overflow
  chk=SCOMPARE(field,limit)
 FI
 IF chk>0 THEN
  chk=0
  MSG(7)
 ELSE
  value=VALB(field)
  IF value<minval OR value>maxval
    THEN MSG(7)
  ELSE accept=1
  FI
 FI

UNTIL accept
OD

RETURN(value)

;*********************************
;
; Example of use of EntryB()

PROC Test2()

BYTE x,y,min,max,nullflg,value
BYTE errcde
BYTE POINTER err_ptr

errcde=0
err_ptr=@errcde

min=100
max=200

nullflg=0
x=15  y=7

PUT(125)
POSITION(5,5)
PUTE()
PRINTE("Enter a number between ")
PRINTB(min)
PRINT(" and ")
PRINTB(max)
PRINT(": ")

value=EntryB(X,Y,min,max,nullflg,
                 0,0,err_ptr)

POSITION(5,10)
PUTE()
PRINTBE(value)
PRINTE("Done...")
```

RETURN