MZ-80B - intern Nr. 6

Das hier abgedruckte Unterprogramm (Bild 1) stellt einen INPUT-Ersatz da, bei dem sämtliche Cursorfunktionen, soweit sie sinnvoll sind, erhalten bleiben. Alle Operationen wirken sich aber nur auf ein Fenster aus. Dadurch eignet sich dieses Unterprogramm hervorragend zum Ausfüllen von Masken, da innerhalb des Fensters Korrekturen und ähnliches frei möglich sind, ohne daß der Rahmen zerstört werden kann. Auch leere Eingaben, Anführungszeichen und Kommata werden akzeptiert. Falls nötig, kann eine Vorgabe spezifiziert werden, die dann im Fenster angezeigt wird und geändert oder mit CR akzeptiert werden kann. Ein Klingelzeichen (Ton) läßt sich an einer vorgebbaren Position erzeugen, um das nahende Fensterende anzukündigen.

Aber auch für typische GET-Abfragen, bei denen nur ein Zeichen erwartet wird (z. B. J/N), eignet sich die Eingabe-Routine gut. Der Benutzer erkennt durch das Cursor-Blinken, das eine Eingabe erwartet wird, und nachdem er eine Taste gedrückt hat, gibt das Unterprogramm an das Aufrufprogramm zurück, da das Ende des Fensters (Länge 1) erreicht ist.

Die Eingabe-Routine weist unzulässige Tasten mit einem Piepton zurück. Dazu gehören Cursor auf, Cursor ab, TAB und die programmierbaren Funktionstasten. Shift-Lock, Graph und Revers sind nicht arrettierbar. Die BREAK-Taste wird ohne Piepton abgelehnt, sonst würde das Programm abgebrochen werden. Home stellt den simulierten Cursor an den Anfang des Fensters, Clear löscht das Fenster, Cursor rechts und links funktionieren wie üblich, nur kann das Fenster nicht verlassen werden. Del löscht ein Zeichen und zieht den Rest nach links, am Fensterrand wird ein Leerzeichen ergänzt. Inst fügt ein Zeichen ein, der Rest wird nach rechts verschoben, das Zeichen am Fensterrand verschwindet. 00 erzeugt eine Doppelnull, falls der Platz reicht. Mit CR oder ENT wird die Eingabe beendet und dem Aufrufprogramm zur Verfügung gestellt. Dies geschieht auch, wenn das äußerste rechte Zeichen des Fenster eingegeben wird.

Zusätzlich könnten beim Einsatz der Eingaberoutine in anderen Programmen auch noch Abfragen eingebaut werden, die z. B. nur Zahlen oder bestimmte Buchstaben akzeptieren. Dies sollte durch die reichliche Kommentierung keine Schwierigkeiten bereiten. Die Eingabe-Routine sollte bei längeren Programmen, wie übrigens alle Unterprogramme, an den Anfang des Programms gesetzt werden, da dort die Geschwindigkeit der Programmausführung wesentlich größer ist. Dies liegt daran, daß der Interpreter bei jedem GOTO, GOSUB usw. die angegebenen Zeilennummern im Programm vom Anfang aus sucht, er findet sie also umso langsamer, je weiter sie am Ende liegen. Bei der Eingabe-Routine macht sich dies durch langsameres Blinken des simulierten Cursors bemerkbar. Die Übergabe-Parameter und ein kleines Beispiel für die Anwendung der Eingabe-Routine können Sie Bild 1 entnehmen. Im Beispiel wird keine Vorgabe spezifiziert, die Eingabe beginnt bei Cursorposition 8,6 und ist 20 Zeichen lang. Beim 17. Zeichen wird ein Piepton erzeugt.

Bild 1: Listing der Eingaberoutine

```
100 REM Beispiel
110 REM ========
120 REM
130 CONSOLE 40,N,30,24: CURSOR 0,5
140 PRINT "        ┌──────────────────┐"
150 PRINT " Name: │                  │"
160 PRINT "        └──────────────────┘"
170 VA$="": VL=20: VX=8: VY=6: VT=17: GOSUB 60000
180 NA$=VA$: CURSOR 0,10: PRINT "Die Eingabe war '";NA$;"'."
190 CURSOR 0,15: END
```

```
60000 REM Eingabe-Routine
60010 REM ===============
60020 REM
60030 REM Uebergabe-Parameter
60040 REM VA$    = eventuelle Vorgabe, enthaelt am Ende die Eingabe.
60050 REM VL     = Zulaessige Hoechstlaenge der Eingabe. Eine Vorgabe muss
60060 REM          ebenfalls genau diese Laenge haben, sonst wird sie
60070 REM          durch einen VL-langen Leerzeichenstring ersetzt.
60080 REM VX, VY = Position des ersten Zeichen des Eingabefeldes.
60090 REM VT     = Position des Zeichens, bei dem ein Ton erzeugt wird
60100 REM
60110 IF LEN(VA$)<>VL THEN VA$=SPACE$(VL):   REM Vorgabe ueberpruefen
60120 CURSOR VX,VY: PRINT VA$:               REM und anzeigen
60130 V2=0:                                  REM Hilfsvar. fuer Kursorblinken
60140 V1=0:                                  REM Var. f. aktuelle Position
60150 V1=V1+1
60160 IF V1=VT THEN 60440:                   REM Ton bei VT erzeugen
60170 GET VB$:                               REM Tastaturzeichen einlesen
60180 V2=V2+1
60190 IF VB$<>"" THEN 60240:                 REM Falls keine Taste gedrueckt,
60200 IF V2<16 THEN 60230:                   REM Blinken erzeugen
60210 IF V2>32 THEN V2=0
60220 CURSOR VX+V1-1,VY: PRINT MID$(VA$,V1,1):: GOTO 60170
60230 CURSOR VX+V1-1,VY: PRINT CHR$(31):: GOTO 60170
60240 VB=ASC(VB$): V2=0:                     REM Tastaturzeichen auswerten
60250 IF VB>31 THEN 60400:                   REM Normales Zeichen
60260 IF VB=13 THEN 60670:                   REM <CR> oder <ENT>
60270 IF (VB=1)+(VB=2) THEN 60450:           REM <↓> oder <↑>
60280 IF VB=7 THEN 60700:                    REM <DEL>
60290 IF VB=8 THEN 60750:                    REM <INST>
60300 IF VB=11 THEN 60170:                   REM <BREAK>
60310 IF VB=27 THEN 60450:                   REM <TAB>
60320 IF (VB>15)*(VB<26) THEN 60450:         REM Funktionstasten <F1> - <F10>
60330 IF (VB>8)*(VB<16) THEN 60450:          REM <SFTLOCK>, <GRPH> oder <RVS>
60340 IF (VB=26)*(V1=VL) THEN 60450:         REM <00>
60350 IF VB=26 THEN 60470
60360 IF VB=6 THEN 60530:                    REM <CLR>
60370 IF VB=5 THEN 60560:                    REM <HOME>
60380 IF VB=3 THEN 60590:                    REM <→>
60390 IF VB=4 THEN 60630:                    REM <←>
60400 CURSOR VX+V1-1,VY: PRINT VB$::         REM Normales Zeichen
60410 VA$=LEFT$(VA$,V1-1)+VB$+RIGHT$(VA$,VL-V1)
60420 IF V1=VL THEN RETURN:                  REM RETURN, falls VL erreicht
60430 GOTO 60150:                            REM Wieder von vorn
60440 REM --- Ton bei unzulaessiger Eingabe ---
60450 TEMPO 7: MUSIC "+G1": GOTO 60170
60460 REM --- <00> ---
60470 CURSOR VX+V1-1,VY: PRINT "00":
60480 VA$=LEFT$(VA$,V1-1)+"00"+RIGHT$(VA$,VL-V1-1)
60490 V1=V1+1
60500 IF V1=VL THEN RETURN
60510 GOTO 60150
60520 REM --- <CLR> ---
60530 CURSOR VX,VY:PRINT SPACE$(VL);
60540 VA$=SPACE$(VL):VB$="": GOTO 60140
60550 REM --- <HOME> ---
60560 CURSOR VX+V1-1,VY: PRINT MID$(VA$,V1,1);
60570 GOTO 60140
60580 REM --- <→> ---
60590 IF V1=VL THEN 60450
60600 CURSOR VX+V1-1,VY: PRINT MID$(VA$,V1,1);
60610 GOTO 60150
```

```
60620 REM --- <←> ---
60630 IF V1=1 THEN 60450
60640 CURSOR VX+V1-1,VY: PRINT MID$(VA$,V1,1);
60650 V1=V1-1: GOTO 60170
60660 REM --- <ENT> oder <CR> ---
60670 CURSOR VX+V1-1,VY: PRINT MID$(VA$,V1,1);
60680 RETURN
60690 REM --- <DEL> ---
60700 IF V1=1 THEN 60450
60710 VA$=LEFT$(VA$,V1-2)+RIGHT$(VA$,VL-V1+1)+" "
60720 CURSOR VX,VY: PRINT VA$;
60730 V1=V1-1:GOTO 60170
60740 REM --- <INST> ---
60750 IF V1=VL THEN 60450
60760 VA$=LEFT$(VA$,V1-1)+" "+MID$(VA$,V1,VL-V1)
60770 CURSOR VX,VY: PRINT VA$;
60780 GOTO 60170
```

* * *