Übersichtliches Listing

Diese kleine Erweiterung spaltet beim Listen eine Programmzeile in mehrere Zeilen auf. Dadurch wird die Analyse auch komplizierter Programmteile zum Kinderspiel.

Die Idee zu dieser Erweiterung kam durch den Einzeiler-Wettbewerb im 64'er-Magazin, bei dem die abgedruckten Programme durch ihre gedrängte Darstellung schlecht lesbar waren. Damit ist nun Schluß. Wenn in einer Programmzeile ein Doppelpunkt vorkommt, so wird der darauf folgende Befehl in die nächste Zeile geschrieben.
Beispiel:
Aus
10 A=53280:POKE A,I:POKE A+1,I+1:I=PEEK(53248+18):GOTO 10
wird
10 A= 53280
POKE A,I
POKE A+1,I+1
I=PEEK (53248+18)
GOTO 10
Die Erweiterung liegt in dem Bereich von 49152 bis 49239, nimmt also keinen Basic-Speicherplatz in Anspruch. Sie läßt sich aber beliebig verschieben, damit sie auch bei Basic-Erweiterungen funktioniert, die den Bereich ab 49152 benutzen (man muß nur den Wert der Variablen ADRESSE in Zeile 0 ändern). Die 87 Byte lange Erweiterung läßt sich mit POKE 2,0 einschalten und mit POKE 2,1 wieder ausschalten.

So arbeitet die Routine

Das Maschinenprogramm liegt als Basic-Lader (Listing 1) und als Source-Code (Listing 2) vor. Zuerst wird der LIST-Vektor auf die neue Routine gePOKEt, dann erfolgt ein Rücksprung zum Basic. In der neuen LIST-Routine wird abgefragt, ob es sich bei dem zu listenden Zeichen um einen Befehl handelt. Ist dies der Fall, so wird zu der Routine gesprungen, die den Befehl ausgibt.

Anschließend erfolgt eine Überprüfung auf einen Doppelpunkt. Handelt es sich nicht um dieses Zeichen, so wird normal weitergelistet. Andernfalls findet eine Überprüfung der Speicherstelle 2 statt. Ist sie nicht 0, so wird der Doppelpunkt ausgegeben und mit dem normalen Listing weitergemacht. Steht in der Speicherstelle 2 eine 0, so wird ein CHR$(13) ausgegeben.

Im darauffolgenden Teil werden die untereinanderstehenden Teilprogrammzeilen linksbündig gemacht und ausgegeben. Danach wird wieder zum »normalen« Listing gesprungen.

Die Routine »Übersichtliches Listing« ist in erster Linie natürlich für das Arbeiten mit einem Drucker gedacht, denn eine Bildschirmausgabe mit einem Basic-Befehl pro Zeile würde den Vorteil der erhöhten Übersicht wohl bald zunichte machen, jedenfalls bei längeren Programmen. Der Einsatzschwerpunkt dieser Routine liegt damit bei der Analyse fremder, unübersichtlicher Programme mittels eines Druckers.

(Frank Barcikowski/ev)

```
0 ADRESSE = 49152:REM BELIEBIG                          <087>
1 REM ********************                              <113>
2 REM *     WRITTEN BY     *                            <207>
3 REM *FRANK BARCIKOWSKI*                               <009>
4 REM *  MORAENENWEG 33 *                               <250>
5 REM *3320 SALZGITTER-1*                               <077>
6 REM ********************                              <118>
10 FOR I=ADRESSE TO ADRESSE+87                          <213>
20 READ A:S=S+A:POKE I,A                                <215>
30 NEXT                                                 <040>
31 IF S<>10407 THEN PRINT"DATA ERROR!":END              <062>
35 REM DEM JEWEILIGEN BERREICH ANPASSEN                 <103>
36 HI=INT((ADRESSE+11)/256)                             <157>
37 LO=ADRESSE+11-INT((ADRESSE+11)/256)*256              <179>
38 POKE ADRESSE+6,HI                                    <107>
39 POKE ADRESSE+1,LO                                    <069>
40 SYS ADRESSE:POKE 2,0:END                             <006>
50 REM ------------------------                        <100>
60 REM EINSCHALTEN DER ERWEITERUNG :                    <008>
70 REM POKE 2,0                                         <089>
80 REM ------------------------                        <130>
90 REM AUSSCHALTEN DER ERWEITERUNG :                    <104>
99 REM POKE 2,<>0                                       <007>
100 REM------------------------                         <150>
101 DATA 169,11,141,6,3,169,48,141,7,3,96,
    16,3,76,28,167,201,58,240,3,76,243                  <093>
102 DATA 166,166,2,240,3,76,243,166,169,13
    ,32,210,255,152,72,160,2,177,95                     <003>
103 DATA 170,200,177,95,133,98,134,99,162,
    144,56,32,73,188,32,223,189,162                     <209>
104 DATA 0,189,0,1,240,3,232,208,248,169,3
    2,32,210,255,202,208,250,104,168                    <038>
105 DATA 169,32,76,243,166,48,141,7,7,96                <180>

0 64'er
```

Listing 1. »Übersichtliches Listing« als Basic-Lader. Beachten Sie die Eingabehinweise auf Seite 6.

```
2
20:   3000                    .opt p1
25:   3000          list      =    $0306
26:   3000          altlist   =    $a71c
30:   3000                    *=   $3000
40:   3000 a9 0b              lda  #<nlist
50:   3002 8d 06 03           sta  list    ;listvektor verbiegen
60:   3005 a9 30              lda  #>nlist
70:   3007 8d 07 03           sta  list +1
80:   300a 60                 rts          ;wieder zu basic
                    ;--------- neue listroutine -------
90:   300b 10 03    nlist     bpl  nint    ;pruefung auf inter-
100:  300d 4c 1c a7           jmp  altlist ;pretercode (>$80)
110:  3010 c9 3a    nint      cmp  #$3a    ;code fuer
110:  3012
120:  3012 f0 03              beq  doppelp ;code gefunden
130:  3014 4c f3 a6           jmp  $a6f3   ;weiterlisten
                    ;------ doppelpunkt gefunden -----
137:  3017 a6 02    doppelp   ldx  2       ;2 laden
138:  3019 f0 03              beq  ok      ;wenn 0 dann newlist
139:  301b 4c f3 a6           jmp  $a6f3   ;sonst altlist
140:  301e a9 0d    ok        lda  #13     ;return
150:  3020 20 d2 ff           jsr  $ffd2   ;printen
160:  3023 98                 tya          ;y zwischenspeichern
170:  3024 48                 pha          ;fuer spaeter
                    ;
                    ;----- berechnung der zu --------
                    ;----- printenden spaces  --------
210:  3025 a0 02              ldy  #2
220:  3027 b1 5f              lda  ($5f),y ;zeilennr. lo
230:  3029 aa                 tax          ;merken
240:  302a c8                 iny          ;zeiger erhoehen
250:  302b b1 5f              lda  ($5f),y ;zeilennr. hi
260:  302d 85 62              sta  $62
270:  302f 86 63              stx  $63
280:  3031 a2 90              ldx  #$90
290:  3033 38                 sec
300:  3034 20 49 bc           jsr  $bc49   ;zeilennr. in fac
310:  3037 20 df bd           jsr  $bddf   ;fac nach ascii und $0100
                    ;              ;nach $0100
330:  303a a2 00              ldx  #0
340:  303c bd 00 01 loop      lda  $100,x
350:  303f f0 03              beq  endloop ;0 = stringende
360:  3041 e8                 inx          ;zaehler fur space
370:  3042 d0 f8              bne  loop
                    ;----- space x mal printen -------
380:  3044 a9 20    endloop   lda  #32     ;space laden
400:  3046 20 d2 ff prin      jsr  $ffd2   ;printen
410:  3049 ca                 dex
420:  304a d0 fa              bne  prin    ;schon fertig
430:  304c 68                 pla          ;y wiederholen
435:  304d a8                 tay
440:  304e a9 20              lda  #32     ;space in accu
450:  3050 4c f3 a6           jmp  $a6f3   ;accu printen
                    ;und weiterlisten
```

Listing 2. Assembler-Listing zu »Übersichtliches Listing«. Es dient nur zur Dokumentation, braucht also nicht eingegeben zu werden.