```
241 IFSTTHEN259
242 IFC=254THENMT=119
243 IFC=226THENMT=50
244 IFC=213THENMT=12
245 IFC=242THENMT=12
246 IFC=198THEN248
247 RETURN
248 PRINT#15,"M-R"CHR$(234)CHR$(16):GET#
15,ZB$:ZB=ASC(ZB$+CHR$(0))
249 IFZB=0THENMT=12
250 IFZB=1THEN252
251 IFSTTHEN259
252 PRINT#15,"M-R"CHR$(172)CHR$(16):GET#
15,ZC$:ZC=ASC(ZC$+CHR$(0))
253 IFZC=1THENMT=12
254 IFZC=2THENMT=12
255 RETURN
256 CLOSE15:OPEN15,OD,15
257 PRINT#15,"M-W"CHR$(MT)CHR$(0)CHR$(2)
CHR$(ND+32)CHR$(ND+64)
258 RETURN
259 PRINT"■■ FEHLER AM GERAET !":GOSUB30
2:RETURN
260 REM ******************************
261 REM *    C H E C K   D I S K    *
262 REM ******************************
263 DIMT(100):DIMS(100)
264 PRINT"■■■   CHECK DISK"
265 GOSUB59:OPEN15,GN%,15:N%=RND(TI)*255
266 A$="":FORI=1TO255:A$=A$+CHR$(255AND(
I+N%)):NEXT
267 UR%=1:GOSUB299:IFENTHENCLOSE15:GOTO4
6
268 OPEN2,GN%,2,"#"
269 PRINT:PRINT#2,A$;
270 T=1:S=0
271 PRINT#15,"B-A:0"T;S
272 INPUT#15,EN,EM$,ET,ES
273 IFEN=0THEN276
274 IFET=0THEN284
275 PRINT#15,"B-A:0"ET;ES:T=ET:S=ES
276 PRINT#15,"U2:2,0"T;S
277 NB=NB+1:PRINT"GETESTETE BLOECKE"NB
278 PRINT"■ SPUR      ■■■■"T;" SEKTOR
■■■■"S"□□□"
279 INPUT#15,EN,EM$,ES,ET
280 IFEN=0THEN271
281 T(J)=T:S(J)=S:J=J+1
282 PRINT"■■■■■FEHLERHAFTER BLOCK:",T;S"
□□□□□□"
283 GOTO271
284 GOSUB45
285 GOSUB299:IFENTHENCLOSE15:CLOSE2:GOTO
46
286 CLOSE2:CLOSE15
287 IFJ=0THENPRINT"■■■■■■■KEINE FEHLERHAF
TEN BLOECKE !":RETURN
288 OPEN2,GN%,2,"#"
289 PRINT"■■FEHLERBLOECKE","SPUR","SEKTO
R"
290 FORI=0TOJ-1
291 PRINT#15,"B-A:0"T(I);S(I)
292 PRINT,,T(I),S(I)
293 NEXT
294 PRINT"■"J"FEHLERBLOECKE WURDEN FESTG
ESTELLT":CLOSE15:CLOSE2:GOSUB302:RETURN
295 REM ******************************
296 REM * FEHLERKANAL UEBERPRUEFEN  *
297 REM ******************************
298 CLOSE15:OPEN15,GN%,15
299 INPUT#15,EN,EN$,NE,EE:IFUR%THENRETUR
N
300 CLOSE15
301 RETURN
302 REM ******************************
303 REM *    TASTENDRUCK ABWARTEN    *
304 REM ******************************
305 PRINT"■WEITER-->TASTE !":POKE198,0:W
AIT198,1:POKE198,0:RETURN
306 REM ******************************
307 REM *     DIRECTORY LESEN        *
308 REM ******************************
309 PRINT"■■■■■■   DIRECTORY LESEN":PRIN
T"■■■■■■MOMENT BITTE !"
310 DIML$(140),K$(140),F$(140)
311 OPEN1,GN%,0,"$0":OPEN15,GN%,15:UR%=1
:GOSUB299:IFENTHEN46
312 GET#1,A$,B$:ZE=-1
313 ZE=ZE+1:GET#1,A$,B$
314 GET#1,A$,B$:C=0
315 IFA$<>""THENC=ASC(A$)
316 IFB$<>""THENC=C+ASC(B$)*256
317 L$(ZE)=RIGHT$("    "+MID$(STR$(C),2),
3)
318 GET#1,B$:IFST<>0THENFB$=RIGHT$("    "
+L$,3):GOTO327
319 IFB$<>CHR$(34)THEN 318
320 F$(ZE)=""
321 GET#1,B$:IFB$<>CHR$(34)THENF$(ZE)=F$
(ZE)+B$:GOTO321
322 GET#1,B$:IFB$=CHR$(32)THEN322
323 C$=""
324 C$=C$+B$:GET#1,B$:IFB$<>""THEN324
325 K$(ZE)=LEFT$(C$,3):IFZE=0THENID$=LEF
T$(C$,5)
326 IFST=0THEN313
327 CLOSE1:CLOSE15:RETURN
READY.
```

Listing 1. »Disketten-Organisation« (Schluß)

Programmiertes LISTing: LISTX-Y

Bei Hilfsprogrammen, die viele Benutzeranleitungen enthalten, gibt man diese Anleitungen normalerweise über PRINT-Anweisungen auf dem Bildschirm aus. An sich würde es bei geschickter Formulierung jedoch reichen, die sowieso vorhandenen REM-Erläuterungen als Anleitungen für den Benutzer mitzuverwenden.

Nur, wie bringt man diese auf den Bildschirm? Mit LIST wird das Basic-Programm jedesmal zum Direktmodus hin verlassen und, was noch schwerer wiegt, die Anfangs- und Endzeilennummern können nur als direkte Zahlen, nicht über Variablen, angegeben werden. Unser Programmvorschlag (Listing 1) simuliert den im VC 20-Basic nicht vorhandenen Befehl LISTX-Y (X=Variable, die die Anfangszeilennummer des zu listenden Programms übergibt, Y = Endzeilennummer).

Man könnte sich ein solches Hilfsprogramm durch Beschreiben des Tastaturpuffers konstruieren. Dann würden sich aber die in den Puffer geschriebenen LIST-Anweisungen auf dem Bildschirm störend bemerkbar machen. Andererseits kommt aus Geschwindigkeitsgründen nur ein Maschinenprogramm in Frage. Will man ein solches Maschinenprogramm per DATA-Zeilen einlesbar gestalten, wäre der Aufwand recht hoch. Außerdem gäbe es Schwierigkeiten mit dem nur immer auf den Datenanfang zurückstellbaren DATA-Zeiger (kein RESTORE X vorhanden). Einlesen per POKE wäre noch aufwendiger. Umgekehrt muß aber ein solches Maschinenprogramm zur Simulation von LIST Y notwendigerweise viele Teile enthalten, die bereits im Betriebssystem vorkommen. Wir lösen das Problem, indem wir mit zwei einfachen FOR-NEXT-Schleifen geeignete Teile des Betriebssystems in den Kassettenpuffer kopieren und diese Kopien dann durch sechs POKE-Anweisungen so abändern, daß sie unseren Ansprüchen genügen. Listing 1 zeigt das entsprechende Unterprogramm, das beim ersten Aufruf mit »GOSUB 63000« die Maschinenroutine erzeugt. Alle weiteren Aufrufe können mit »GOSUB 6350« erfolgen, wodurch, einiges an Zeit gespart wird. In der Variablen X wird die Anfangszeile, in Y die Endzeile übergeben. Listing 3 zeigt ein Demo-Programm. In Listing 2 gebe wir ein Anwendungsbeispiel an: In einem längeren Programm mögen alle REM-Erläuterungen in den Zeilen 100*a bis 100*a+4 untergebracht sein, also in den ersten fünf Zeilen ab jeder vollen Hunderternummer. Das Beispielprogramm nach Listing 2 wird per GOSUB63100 angesprungen und listet per Cursor-Down-Taste alle oben genannten REM-Zeilen (und nur diese, eventuelle weitere REM-Zeilen werden nicht berücksichtigt). Und zwar geschieht dies in Endlosform, das heißt nach Durchgang durch die letzte zu listende REM-Erläuterung erscheint wieder die erste und so fort. Durch Drücken der Return-Taste gelangt man wieder ins Hauptprogramm.

(Fred Behringer/ev)

```
63000 REM LIST X-Y
63001 REM ========
63002 REM
63010 FORI=0TO45:POKE828+I,PEEK(50707+I)
:NEXT
63020 POKE845,252:POKE856,251
63030 FORI=0TO120:POKE874+I,PEEK(50889+I
):NEXT
63040 POKE896,254:POKE900,253:POKE935,19
:POKE949,96
63050 POKE252,X/256:POKE251,X-256*PEEK(2
52)
63060 POKE254,Y/256:POKE253,Y-256*PEEK(2
54)
63070 SYS828:RETURN
READY.
```

Listing 1. Programm zur Simulation von LISTX-Y. Ansprung per GOSUB63000. Einlesezeit 2,6 Sekunden. Jeder weitere Ansprung per GOSUB63050 mit »sofortiger« Abarbeitung.

60010	Einlesen von $C613-$C640 (50707-50752). Maschinenprogramm zur Berechnung der Startadresse einer Basic-Zeile. Zeilennummereingabe (der Anfangszeile) in $0014/$0015 (20/21).
63020	Verlegen dieser Parameterübergabestellen nach $00FB/$00FC (251/252), da $0014/$0015 (20/21) beim Ansprung SYS828 (siehe unten) gestört werden
63030	$C6009-$C741 (50889-51009). Programm zum Auflisten der gewünschten Zeilen. In diesem Programm wird die laufende Nummer der gerade gelisteten Zeile mit der beim Ansprung dieser Routine an $0014/$0015 (20/21) übergebenen Nummer der Endzeile verglichen, und die Routine wird verlassen, wenn die Endnummer erreicht ist. Auch hier läßt sich aus dem eben erwähnten Grund die Stelle $0014/$0015 (20/21) nicht als Parameterübergabestelle (von Basic zu Maschinenprogramm) verwenden und wird in 63040 zu $00FD/$00FE (253/254) abgeändert. Überschreiben des JMP-Befehls in $C714 (50964) mit einem RTS-Befehl sorgt dafür, daß nicht zum Basic-Warmstart zurückgekehrt wird, sondern zum Hauptprogramm. Die Sprungadresse $0306/$0307 (774/775), die in $C717 (50967) wirksam wird, hätte geändert werden müssen, da sich der Sprung ja nun auf die der Stelle $C71A (50970) entsprechenden Stelle in der Kopie im Kassettenpuffer bezieht. Es war einfacher, den (hier) überflüssigen JMP-Befehl in $C717 (50967) zu umgehen, indem die Sprungadresse in der der Stelle $C705 (50949) entsprechenden Stelle der Kopie abgeändert wurde.
63050	Die vom Hauptprogramm in X stammende Nummer der Anfangszeile wird in Low Byte/High Byte gespalten und der Stelle 251/252 übergeben (siehe oben)
63060	Entsprechend für die aus Y stammende Nummer der Endzeile und 253/254
63070	Aufruf des Maschinenprogramms und Rückkehr ins Basic-Hauptprogramm. Nach Rückkehr steht in $030D/$030C (781/780) die Nummer (Low Byte/High Byte) der nächstanstehenden Basic-Zeile zur Verfügung.

Beschreibung des Programms nach Listing 1 zur Simulation von LISTX-Y

```
63100 REM AUSDRUCKEN DER ZEILEN 100*A BI
S 100*A+4
63105 REM
63110 GOSUB63000:PRINTCHR$(147);
63120 X=0
63130 GETX$:IFX$=""THEN63130
63140 IFX$=CHR$(13)THEN63200
63150 IFX$<>CHR$(17)THEN63120
63160 Y=X+4:GOSUB63050:PRINTCHR$(145);
63170 X=X+100:IFX>63999THENX=0:GOTO63160
63180 I=PEEK(781)+256*PEEK(780):IFI<Y+10
RI>X+4THENX=INT(I/100)*100
63190 GOTO63130
63200 RETURN
READY.
```

Listing 2. Programm zum Ausdrucken aller Zeilen mit den Nummern 100*a bis 100*a+4, wobei a von 0 bis 633 geht. Nicht vorhandene Zeilen werden ohne Verzögerung übersprungen. Der Ausdruck läuft, solange die Cursor-Down-Taste gedrückt ist, auch über 63304 hinaus (Wiederanfang bei 0).

63110	Einlesen des Simulationsprogramms für LISTX-Y nach Listing 1
63120	Anfangszeile auf Null
63130 63140	Wenn Return-Taste, dann zurück ins Hauptprogramm

63150	Wenn nicht Cursor-Down- oder Return-Taste, dann Anfang wieder 0.
63160	Wenn Cursor-Down-Taste, dann Listen der Zeilen X bis X+4 und Ausgleich für Zeilenvorschub
63170	Sprung um 100 nach vorn. Wenn Vorrat erschöpft, Zeilennummer wieder auf 0 (Endlosdurchlauf).
63180	Schnelles Vortasten zur nächsten vorhandenen Zeile mit Nummer zwischen 110*a und 100*a+4
63190	Weiter in Eingabeschleife

Beschreibung des Programms nach Listing 2 zum Ausdrucken aller Kopf-REM-Zeilen. Das Programm verwendet die Routine für LISTX-Y aus Listing 1.

```
10 REM LIST-DEMO
20 REM ==========
30 REM
40 REM
50 REM
60 REM
70 PRINT"LIST-DEMO"
80 PRINT
90 INPUT"VON";X:INPUT"BIS";Y
100 PRINT
110 GOSUB63000:REM LIST X-Y
120 PRINT:PRINT"** OK ** (TASTE)"
130 GETA$:IFA$=""THEN130
140 PRINT:PRINT
150 GOTO70
160 REM
170 REM
180 REM
READY.
```

Listing 3. LIST-Demo zum Testen der Routine nach Listing 1

Kopieren mit Komfort: Super Copy

Bereits im ersten 64'er wurde Ihnen mit »Disk Copy« ein Kopierprogramm für Disketten vorgestellt. Wir möchten Ihnen heute eine völlig revidierte Fassung vorstellen, die erheblich leistungsfähiger und komfortabler ist.

Was soll ein gutes Kopierprogramm leisten? Nun, zunächst einmal soll es kopieren. Dazu muß man auswählen können, was kopiert werden soll. Dieser Vorgang dauert beim »Disk Copy« sehr lange, vor allem, wenn auf der Quelldiskette viele Programme sind. Bei »Super Copy« geht es genauso schnell wie das Einlesen eines Directory. Fehler beim Kopieren sollen möglichst ohne Programmabsturz und völligen Neubeginn behebbar sein. Gerade hier liegt eine Stärke von »Super Copy«. Alle Funktionen, die man sonst noch beim Kopieren braucht (Formatieren, Gültigkeitskontrolle (Vality check) etc.), sollen integriert sein. Dazu gehört auch eine komfortable Löschmöglichkeit, um Disketten »aufzuräumen«. Schließlich soll das Programm möglichst wenig Speicherplatz belegen, damit zum Kopieren genug zur Verfügung steht.

Aus all dem ergibt sich eine Konsequenz: Ein solches Programm läßt sich nur in Maschinensprache schreiben, da Basic einfach zu langsam ist und zuviel Speicherplatz belegt. Trotzdem habe ich einige Einschränkungen gemacht:

Relative Files können nicht kopiert werden. Dies verlangt eine zu aufwendige Verwaltung und kommt auch zu selten vor, um es ins Programm zu integrieren. In einem Programmdurchlauf können höchstens 32 Files kopiert werden. Mehr Filenamen kann das Programm nicht speichern.

Ansonsten aber läßt »Super Copy« kaum noch Wünsche offen, höchstens den nach einem schnelleren Laufwerk. Aber auch da läßt sich wohl noch etwas machen.

Wie arbeitet »Super Copy«?

Nach dem Programmstart meldet sich das Programm mit einem Menü:

1. Directory
2. Kopieren
3. Formatieren
4. Scratchen
5. Validieren
6. Ende

Durch Druck auf eine Ziffer wählen Sie die entsprechende Funktion aus. Übrigens können Sie im Programm immer dann, wenn Sie irgendeine Taste drücken müssen, mit »←« in dieses Menü zurückkehren. Gehen wir nun die einzelnen Funktionen einmal durch.

Zur Funktion »Directory« ist nicht viel zu sagen. Es erscheint das Verzeichnis aller Files auf der Diskette.

Beim »Formatieren« müssen Sie den Diskettennamen und die ID — wie üblich durch ein Komma getrennt — angeben. Eine ID ist nur bei einer neuen Diskette wichtig. Verzichten Sie darauf, werden zwar alle Einträge im Directory gelöscht, aber es entfällt das Neuformatieren der einzelnen Spuren. »Validieren« (Gültigkeitskontrolle) entspricht dem Basic-Befehl OPEN 1,8,15,"V":CLOSE 1.

Entscheiden Sie sich für »Kopieren«, werden Sie aufgefordert, die Quelldiskette einzulegen. Nach Tastendruck erscheinen nun die Namen der Programme. Files, die kopiert werden sollen, kennzeichnen Sie mit der »J«-Taste, die anderen mit »N«. Relative Files können nicht kopiert werden, daher erscheint eine Fehlermeldung, wenn Sie versuchen, solche Files mit »J« zu markieren. Das Programm kann maximal 32 Namen speichern. Wenn Sie mehr als 32 Files kopieren wollen, erscheint die Fehlermeldung »Kopierliste voll«. Sie können nun die bisher markierten Programme kopieren und nach Abschluß einen neuen Programmdurchlauf starten. Haben Sie Ihre Auswahl beendet, gibt das Programm an, wieviele Blöcke insgesamt zu kopieren sind, damit Sie genügend Platz auf der Zieldiskette bereitstellen können. Ein neues Menü erscheint:

1. Directory
2. Formatieren
3. Validieren

*** Space***
für weiter

Sie können nun in aller Ruhe eine Zieldiskette aussuchen, eventuell noch formatieren etc. Sie kommen in jedem Fall in dieses Menü zurück. Sind alle Vorbereitungen abgeschlossen, drücken Sie »Space«, um mit dem Kopieren fortzufahren. Das Programm fordert nun auf, die Quelldiskette einzulegen, und liest die vorher markierten Programme ein. Sollte dabei ein Fehler auftreten, weil Sie zum Beispiel aus Versehen die falsche Diskette eingelegt haben, wird eine entsprechende Meldung ausgegeben und gefragt, ob dieses File übersprungen oder ein neuer Versuch unternommen werden soll. Auch Lese-