Erstellen der Datei HELLO.BAS

Listing 13 abtippen

```
SAVE"HELLO."
CALL 0
```

Listing 14 abtippen

```
SAVE"HELLO.LDR"
RUN
CALL 0
LOAD"HELLO."
OPENOUT"$"
MEMORY &1C6
CLEAR
LOAD"HELLO.BIN", &1C8
POKE &AE66-29*(PEEK(&39) = &39), &98
POKE &AE67-29*(PEEK(&39) = &39), &06
SAVE"HELLO.BAS"
```

Listing 1

```
«1960» 100 'MAUSTEST.BAS
           '
           'Testprogramm fuer die Maus.
           '
«0742» 110 MODE 2:IF HIMEM=&3FFF GOTO 150
«0459» 120 SYMBOL AFTER 256:MEMORY &3FFF
«06EB» 130 LOAD"maustest.bin",&4000
«0195» 140 CALL &4000
«0256» 150 DEFINT a-z
«090B» 160 xadr=&409E:yadr=&40A1
«0C5B» 170 PRINT "ϟ┌┤102   Maeusetest (Abbruch
       bei '*')01"
«0179» 180 PRINT
«0E64» 190 PRINT "Die Maus ist ";:CALL 16387:PR
       INT "angeschlossen."
«0E4A» 200 DEF FN pf(x)=2*x+512*(x>127)+256
«029E» 210 CALL &BB48
«0586» 220 FOR i=72 TO 78
«04DE» 230 KEY DEF i,1,255
«02C1» 240 NEXT i
«0268» 250 GOSUB 390
«0326» 260 a$=INKEY$
«0358» 270 WHILE a$<>"*"
«0223» 280 PRINT a$;
«0344» 290 a$=INKEY$
«0D21» 300 IF JOY(0) AND 16 THEN PLOT x,y:DRAW
       x+4,y+4
«09E6» 310 x1=512-FN pf(PEEK(xadr))
«08DD» 320 y1=FN pf(PEEK(yadr))
«0A31» 330 IF x1=x AND y1=y GOTO 370
«01C3» 340 GOSUB 390
«06D0» 350 x=x1:y=y1
«01D7» 360 GOSUB 390
«014E» 370 WEND
«011B» 380 END
«039C» 390 MOVE x,y
«01B8» 400 DRAWR 20,0
«03D4» 410 MOVER -10,-10
«01CC» 420 DRAWR 0,20
«017E» 430 RETURN
```

Listing 2

```
«09CB» 1   :'MC-Generator: MAUSTEST.ldr
«004B» 2   :
«0906» 3   :'erzeugt     : MAUSTEST.bin
«004D» 4   :
«0C3D» 5   :'Copyright   : Guenter Radestock
«004F» 6   :
«103A» 100 DATA C3,1C,40,3E,4E,CD,1E,BB,C0,21,1
       5,40,7E,23,B7,C8,CD,5A,BB,18,F7,6E,&0A06
«0F98» 101 DATA 69,63,68,74,20,00,11,38,40,01,F
       F,80,21,30,40,CD,EF,BC,21,2E,40,C3,&082C
«0FB5» 102 DATA E3,BC,00,00,00,00,00,80,38,40,F
       F,00,01,0E,F4,F3,ED,49,06,F6,ED,78,&0923
«1036» 103 DATA E6,30,4F,F6,C0,ED,79,ED,49,04,C
       5,0E,92,ED,49,E6,30,F6,49,05,ED,79,&0C16
«104C» 104 DATA 06,F4,ED,78,2F,5F,C1,3E,82,ED,7
       9,05,ED,49,FB,7B,E6,03,21,9C,40,4E,&0AB9
«1007» 105 DATA 77,87,87,B1,CD,7E,40,7B,E6,0C,4
       F,7E,71,0F,0F,B1,23,C6,B0,4F,06,40,&0969
«0F9A» 106 DATA 0A,B7,28,11,FE,02,20,04,7E,87,1
       8,01,77,23,86,EA,97,40,77,23,C9,23,&07A3
«0F0B» 107 DATA 23,C9,00,FF,B8,00,FF,F0,00,00,0
       0,00,00,00,00,00,00,00,00,00,00,00,&0492
«0BC2» 108 DATA 00,FF,01,02,01,00,02,FF,FF,02,0
       0,01,02,01,FF,00,&0408
«01FD» 109 DATA EOF
«0075» 110 :
«029D» 111 MEMORY &3FFF
«0C89» 112 zeile= 100:schritt= 1:adr=&4000
```

```
«06C0» 113 PRINT"Zeile:"zeile ;
«0266» 114  READ b$
«0746» 115   IF      b$      ="EOF" GOTO 127
«06FE» 116   IF MID$(b$,1,1)="&"   GOTO 122
«0700» 117    b    = VAL("&"+b$)
«04A8» 118    POKE adr,b
«0A48» 119    sum = sum + PEEK(adr)
«06ED» 120    adr = adr + 1
«01F4» 121  GOTO 114
«07B4» 122 checksum=VAL(b$)
«0EDC» 123 IF sum=checksum THEN v=6 ELSE v=174
«0527» 124 PRINT CHR$(1)CHR$(v)
«0E44» 125 sum=0:zeile=zeile+schritt
«01D7» 126 GOTO 113
«0918» 127 FOR a=&40C0 TO &437F:POKE a,0:NEXT
«07FB» 128 SAVE"MAUSTEST.bin",b,&4000,&380
«011F» 129 END
```

Listing 3

```
«2B8F» 100 'MOUSE.BAS
           '
           'Lader fuer MOUSE.RSX und PFEIL.POI.
           'PFEIL.POI kann mit PED.BAS editiert
           'werden.
           '
«0152» 110 MODE 1
«3291» 120 IF JOY(0) AND 64 THEN:ELSE PRINT "Fa
       lls Sie eine Maus besitzen, schliessenSi
       e diese bitte jetzt an!":PRINT:PRINT"Wei
       ter: Maus/Enter":WHILE INKEY$<>CHR$(13)
       AND (JOY(0) AND 64)=0:WEND
«0238» 130 SYMBOL AFTER 256
«0F0F» 140 IF HIMEM<&A5FF+65536 THEN PRINT"out
       of memory":STOP
«09F4» 150 MEMORY &8FFF:LOAD"mouse.rsx",&9000:C
       ALL &9000
«0630» 160 LOAD "pfeil.poi",&A000
«0419» 170 |PTSELECT
«0198» 180 MODE 1
«0A8A» 190 |PTON:WHILE INKEY$="":WEND:|PTOFF:NE
       W
«0190» 200 '
«0CB8» 210 'Befehle fuer den Pointer:
           '
«0902» 220 '"|PTON                  Zeiger ein
«0937» 230 '"|PTOFF                 Zeiger aus
«21DD» 240 '"|GETPTPOS,@x%,@y% liefert die Zei-
           '                        gerposition in
       'den Variablen x% und y% zurueck.
«12AC» 250 '"|PTSELECT              Neues Muster
           '                        auswaehlen.
«198C» 260 'Das neue Muster muss zuvor an die
       'Adresse &A000 geladen worden sein.
```

Listing 4

```
«08DB» 1   :'MC-Generator: MOUSE.ldr
«004B» 2   :
«083A» 3   :'erzeugt     : MOUSE.rsx
«004D» 4   :
«0C3D» 5   :'Copyright   : Guenter Radestock
«004F» 6   :
«1012» 100 DATA 01,0F,90,21,1D,90,CD,D1,BC,CD,2
       6,95,C3,5F,90,21,90,C3,D3,94,C3,EA,&0B8A
«0F81» 101 DATA 94,C3,3B,90,C3,5F,90,00,00,00,0
       0,50,54,4F,CE,50,54,4F,46,C6,47,45,&0820
«0F92» 102 DATA 54,50,54,50,4F,D3,50,54,53,45,4
       C,45,43,D4,00,ED,5B,0A,95,21,C7,00,&081D
«101F» 103 DATA B7,ED,52,29,EB,DD,6E,00,DD,66,0
       1,73,23,72,2A,08,95,29,EB,DD,6E,02,&09C9
«0FA9» 104 DATA DD,66,03,73,23,72,C9,21,00,A0,C
       3,C4,94,29,01,7B,90,09,7E,23,66,6F,&08A7
«0FC3» 105 DATA 7B,E6,03,47,AB,B2,0F,0F,85,6F,D
       0,24,C9,00,C0,00,C8,00,D0,00,D8,00,&0907
«0FA2» 106 DATA E0,00,E8,00,F0,00,F8,50,C0,50,C
       8,50,D0,50,D8,50,E0,50,E8,50,F0,50,&0C18
«1032» 107 DATA F8,A0,C0,A0,C8,A0,D0,A0,D8,A0,E
       0,A0,E8,A0,F0,A0,F8,F0,C0,F0,C8,F0,&1130
«101B» 108 DATA D0,F0,D8,F0,E0,F0,E8,F0,F0,F0,F
       8,40,C1,40,C9,40,D1,40,D9,40,E1,40,&0F9D
«100B» 109 DATA E9,40,F1,40,F9,90,C1,90,C9,90,D
       1,90,D9,90,E1,90,E9,90,F1,90,F9,E0,&0F9B
«1019» 110 DATA C1,E0,C9,E0,D1,E0,D9,E0,E1,E0,E
       9,E0,F1,E0,F9,30,C2,30,CA,30,D2,30,&1026
«101B» 111 DATA DA,30,E2,30,EA,30,F2,30,FA,80,C
       2,80,CA,80,D2,80,DA,80,E2,80,EA,80,&0DD6
«1074» 112 DATA F2,80,FA,D0,C2,D0,CA,D0,D2,D0,D
       A,D0,E2,D0,EA,D0,F2,D0,FA,20,C3,20,&10DF
«0FFC» 113 DATA CB,20,D3,20,DB,20,E3,20,EB,20,F
       3,20,FB,70,C3,70,CB,70,D3,70,DB,70,&0C61
«1062» 114 DATA E3,70,EB,70,F3,70,FB,C0,C3,C0,C
       B,C0,D3,C0,DB,C0,E3,C0,EB,C0,F3,C0,&1109
«1000» 115 DATA FB,10,C4,10,CC,10,D4,10,DC,10,E
       4,10,EC,10,F4,10,FC,60,C4,60,CC,60,&0B2B
«1061» 116 DATA D4,60,DC,60,E4,60,EC,60,F4,60,F
       C,B0,C4,B0,CC,B0,D4,B0,DC,B0,E4,B0,&0F94
«1018» 117 DATA EC,B0,F4,B0,FC,00,C5,00,CD,00,D
       5,00,DD,00,E5,00,ED,00,F5,00,FD,50,&0B94
«1051» 118 DATA C5,50,CD,50,D5,50,DD,50,E5,50,E
       D,50,F5,50,FD,A0,C5,A0,CD,A0,D5,A0,&0E1F
«10CD» 119 DATA DD,A0,E5,A0,ED,A0,F5,A0,FD,F0,C
       5,F0,CD,F0,D5,F0,DD,F0,E5,F0,ED,F0,&12C7
«1046» 120 DATA F5,F0,FD,40,C6,40,CE,40,D6,40,D
       E,40,E6,40,EE,40,F6,40,FE,90,C6,90,&0DD8
«1084» 121 DATA CE,90,D6,90,DE,90,E6,90,EE,90,F
       6,90,FE,E0,C6,E0,CE,E0,D6,E0,DE,E0,&1152
«1047» 122 DATA E6,E0,EE,E0,F6,E0,FE,30,C7,30,C
       F,30,D7,30,DF,30,E7,30,EF,30,F7,30,&0E01
«1069» 123 DATA FF,80,C7,80,CF,80,D7,80,DF,80,E
       7,80,EF,80,F7,80,FF,7C,C6,08,67,D0,&0E98
«104A» 124 DATA D6,40,67,7D,C6,50,6F,D0,24,C9,7
       C,D6,08,67,D6,C0,D8,C6,C8,67,7D,D6,&0CB3
«101E» 125 DATA 50,6F,D0,25,C9,AF,C5,47,7E,0F,4
       F,E6,77,B0,12,79,06,88,A0,28,04,A8,&09AE
«0FDD» 126 DATA 20,01,A8,C1,13,23,10,E8,C9,0E,0
       C,06,04,CD,2B,92,0D,20,F8,C9,D5,3E,&0830
«0FDA» 127 DATA 0C,01,03,00,ED,B0,EB,36,00,23,E
       B,3D,20,F3,E1,06,03,C5,CD,45,92,C1,&0940
«0FC8» 128 DATA 10,F9,C9,11,14,93,E5,CD,50,92,E
       1,01,24,00,09,C3,50,92,2A,0A,95,ED,&0988
«1007» 129 DATA 5B,08,95,3E,C8,95,FE,0C,38,02,3
       E,0C,32,13,93,21,3F,01,AF,4F,ED,52,&0797
«0FCD» 130 DATA B4,3E,04,20,0A,47,7D,0C,D6,04,3
       8,02,10,F9,79,32,11,93,2A,0A,95,CD,&06F2
«0F94» 131 DATA 65,90,22,0F,93,EB,68,26,00,29,2
       9,29,29,44,4D,29,09,01,14,93,09,DD,&0627
«1009» 132 DATA 21,94,94,ED,4B,13,93,3A,11,93,4
       7,D5,E5,C5,1A,DD,77,00,DD,23,01,C0,&09FA
«0FF5» 133 DATA 00,09,B6,AE,ED,42,AE,12,23,13,C
       1,10,EA,E1,11,04,00,19,EB,E1,CD,0B,&0900
«1053» 134 DATA 92,EB,0D,20,D6,C9,ED,5B,0F,93,2
       1,94,94,ED,4B,12,93,C5,D5,ED,4B,11,&0B3C
«0AE0» 135 DATA 93,ED,B0,EB,E1,CD,0B,92,EB,C1,1
       0,EF,C9,&08DA
«0218» 136 DATA EOF
«10B4» 137 DATA 3A,0C,95,B7,CA,6B,92,E5,CD,EA,9
       4,E1,CD,6B,92,3A,0C,95,EE,FF,C8,32,&0CF6
«106F» 138 DATA 0C,95,CD,7A,92,3E,FE,32,11,95,2
       1,0D,95,C3,DA,BC,3A,0C,95,EE,FF,C0,&0B32
«1023» 139 DATA 32,0C,95,21,0D,95,CD,DD,BC,C3,F
       2,92,ED,53,08,95,22,0A,95,AF,32,11,&09D3
«0FB0» 140 DATA 95,C9,00,00,00,00,00,00,00,00,0
       0,00,80,16,95,FF,DD,E5,CD,F2,92,CD,&0868
«10A1» 141 DATA 7A,92,3E,FE,32,11,95,DD,E1,C9,0
       1,48,FF,79,CD,27,BB,79,CD,2D,BB,79,&0BBE
«1048» 142 DATA CD,33,BB,0C,3E,4C,B9,20,EE,21,4
       9,95,11,01,00,42,4B,CD,E9,BC,C3,D8,&09C3
«0FC9» 143 DATA 95,00,00,00,00,00,00,00,00,00,8
       1,55,95,CD,9E,95,7C,B5,C8,EB,7B,CD,&082C
«0FAD» 144 DATA 97,95,2A,0A,95,09,CB,7C,28,03,2
       1,00,00,01,C8,00,B7,ED,42,09,38,03,&0684
«0FCE» 145 DATA 21,C7,00,7A,EB,CD,97,95,2A,08,9
       5,09,CB,7C,28,03,21,00,00,01,40,01,&06EB
«101B» 146 DATA B7,ED,42,09,38,03,21,3F,01,EB,C
       3,FC,94,4F,06,00,B7,F0,05,C9,CD,24,&0984
«0FEB» 147 DATA BB,4F,E6,40,21,D0,96,AE,7E,B7,2
       0,0C,21,D6,95,5E,36,00,23,56,36,00,&0895
«0FF5» 148 DATA 18,0C,21,B1,96,56,36,00,23,23,2
       3,5E,36,00,EB,C9,CD,D1,96,CD,D8,95,&0937
«1023» 149 DATA 18,D0,82,E0,92,C9,92,E0,82,C9,0
       0,00,CD,24,BB,E6,40,32,D0,96,01,FF,&0BCC
```

```
«0FEC» 150 DATA 81,20,14,11,08,96,21,E4,96,CD,E
     F,BC,21,DE,96,11,01,00,42,4B,C3,E9,&0957
«1089» 151 DATA BC,11,4B,96,21,E0,96,CD,EF,BC,2
     1,DE,96,C3,E3,BC,CD,24,BB,21,4A,96,&0C61
«0F59» 152 DATA E6,0F,20,02,77,C9,16,04,34,28,1
     1,16,01,35,20,03,34,18,08,35,20,03,&03F9
«1079» 153 DATA 36,FB,C9,34,34,34,35,2A,D6,95,1
     F,4F,7D,DC,D2,95,CB,19,DC,CE,95,6F,&0B20
«1088» 154 DATA 7C,CB,19,DC,D2,95,CB,19,DC,CE,9
     5,67,22,D6,95,C9,00,01,0E,F4,F3,ED,&0C66
«106A» 155 DATA 49,06,F6,ED,78,E6,30,4F,F6,C0,E
     D,79,ED,49,04,C5,0E,92,ED,49,E6,30,&0C16
«1095» 156 DATA F6,49,05,ED,79,06,F4,ED,78,2F,5
     F,C1,3E,82,ED,79,05,ED,49,FB,7B,E6,&0C15
«1047» 157 DATA 03,21,AF,96,4E,77,87,87,B1,CD,9
     1,96,7B,E6,0C,4F,7E,71,0F,0F,B1,23,&097E
«0FF7» 158 DATA C6,C0,4F,06,96,0A,B7,28,11,FE,0
     2,20,04,7E,87,18,01,77,23,86,EA,AA,&0861
«0EF6» 159 DATA 96,77,23,C9,23,23,C9,00,00,00,0
     0,00,00,00,00,00,00,00,00,00,00,00,&0308
«0FA1» 160 DATA 00,00,00,01,FF,02,FF,00,02,01,0
     1,02,00,FF,02,FF,01,00,00,21,DE,96,&059D
«090A» 161 DATA 3A,D0,96,B7,CA,EC,BC,C3,E6,BC,&
     072E
«0232» 162 DATA EOF
«00AA» 163 :
«0322» 164 MEMORY &8FFF
«098B» 165 FOR a%=&9000 TO &96FF:POKE a%,0:NEXT
«100D» 166 zeile= 100:schritt= 1:adr=&9000:REST
     ORE 100:GOSUB 170
«1120» 167 zeile= 137:schritt= 1:adr=&94C4:REST
     ORE 137:GOSUB 170
«072B» 168 SAVE"MOUSE.rsx",b,&9000,&700
«0147» 169 END
«06F9» 170 PRINT"Zeile:"zeile ;
«029F» 171  READ b$
«07B8» 172   IF      b$     ="EOF" GOTO 184
«0770» 173   IF MID$(b$,1,1)="&"   GOTO 179
«0739» 174    b   = VAL("&"+b$)
«04E1» 175    POKE adr,b
«0A81» 176    sum = sum + PEEK(adr)
«0726» 177    adr = adr + 1
«0266» 178  GOTO 171
«07ED» 179 checksum=VAL(b$)
«0F15» 180 IF sum=checksum THEN v=6 ELSE v=174
«0560» 181 PRINT CHR$(1)CHR$(v)
«0E7D» 182 sum=0:zeile=zeile+schritt
«0249» 183 GOTO 170
«0287» 184 PRINT b$
«0188» 185 RETURN
```

Listing 5

```
                    MOUSE.MAX
ARNOR MAXAM Assembler Listing fuer MOUSE.RSX


nolist:write "mouse.rsx"

addfly  equ &bcda
delfly  equ &bcdd
addtik  equ &bce9
deltik  equ &bcec
getjoy  equ &bb24


        org &9000

;platz fuer rohshape

sbitmap equ &a000

;pointergeschichte als rsx einhaengen

        ld bc,rsxtable
        ld hl,rsxchn
        call &bcd1          ;kl log ext
        call minit
        jp newshape


rsxtable
        defw nametable
        jp pointeron
        jp pointeroff
        jp askpointer
        jp newshape
rsxchn  defs 4
nametable
        defb "PTO","N"+&80
        defb "PTOF","F"+&80
        defb "GETPTPO","S"+&80
        defb "PTSELEC","T"+&80
        defb 0


;askpointer wird von basic mit
;der uebergabe zweier intpointer
;aufgerufen (getptpos,@x,@y

askpointer
        ld de,(ptycor)
        ld hl,199
        or a
        sbc hl,de
        add hl,hl
        ex de,hl
        ld l,(ix+0)
        ld h,(ix+1)         ;pointer auf y
        ld (hl),e
        inc hl
        ld (hl),d
        ld hl,(ptxcor)
        add hl,hl
        ex de,hl

        ld l,(ix+2)
        ld h,(ix+3)
        ld (hl),e
        inc hl
        ld (hl),d
        ret

newshape
        ld hl,sbitmap
        jp selectpointer

;holen einer bildschirmadresse
;uebernimmt (de,hl) - bildposition
;gibt zurueck hl=adresse,
;             b=wievielter pixel

scrpos  add hl,hl
        ld bc,scrtab
        add hl,bc
        ld a,(hl)
        inc hl
        ld h,(hl)
        ld l,a
        ld a,e
        and 3
        ld b,a
        xor e
        or d
        rrca
        rrca
        add a,l
        ld l,a
        ret nc
        inc h
        ret

scrtab  defw &C000,&C800,&D000,&D800,&E000,&E800,&F000,&F800,&C050,&C850,&D050
        defw &D850,&E050,&E850,&F050,&F850,&C0A0,&C8A0,&D0A0,&D8A0,&E0A0,&E8A0
        defw &F0A0,&F8A0,&C0F0,&C8F0,&D0F0,&D8F0,&E0F0,&E8F0,&F0F0,&F8F0,&C140
        defw &C940,&D140,&D940,&E140,&E940,&F140,&F940,&C190,&C990,&D190,&D990
        defw &E190,&E990,&F190,&F990,&C1E0,&C9E0,&D1E0,&D9E0,&E1E0,&E9E0,&F1E0
        defw &F9E0,&C230,&CA30,&D230,&DA30,&E230,&EA30,&F230,&FA30,&C280,&CA80
        defw &D280,&DA80,&E280,&EA80,&F280,&FA80,&C2D0,&CAD0,&D2D0,&DAD0,&E2D0
        defw &EAD0,&F2D0,&FAD0,&C320,&CB20,&D320,&DB20,&E320,&EB20,&F320,&FB20
        defw &C370,&CB70,&D370,&DB70,&E370,&EB70,&F370,&FB70,&C3C0,&CBC0,&D3C0
        defw &DBC0,&E3C0,&EBC0,&F3C0,&FBC0,&C410,&CC10,&D410,&DC10,&E410,&EC10
        defw &F410,&FC10,&C460,&CC60,&D460,&DC60,&E460,&EC60,&F460,&FC60,&C4B0
        defw &CCB0,&D4B0,&DCB0,&E4B0,&ECB0,&F4B0,&FCB0,&C500,&CD00,&D500,&DD00
        defw &E500,&ED00,&F500,&FD00,&C550,&CD50,&D550,&DD50,&E550,&ED50,&F550
        defw &FD50,&C5A0,&CDA0,&D5A0,&DDA0,&E5A0,&EDA0,&F5A0,&FDA0,&C5F0,&CDF0
        defw &D5F0,&DDF0,&E5F0,&EDF0,&F5F0,&FDF0,&C640,&CE40,&D640,&DE40,&E640
        defw &EE40,&F640,&FE40,&C690,&CE90,&D690,&DE90,&E690,&EE90,&F690,&FE90
        defw &C6E0,&CEE0,&D6E0,&DEE0,&E6E0,&EEE0,&F6E0,&FEE0,&C730,&CF30,&D730
        defw &DF30,&E730,&EF30,&F730,&FF30,&C780,&CF80,&D780,&DF80,&E780,&EF80
        defw &F780,&FF80


;naechste / vorige zeile anspringen
;hl=bildadr, akku verfaelscht

nextlin ld a,h

        add a,8
        ld h,a
        ret nc
        sub &40
        ld h,a
        ld a,l
        add a,80
        ld l,a
        ret nc
        inc h
        ret

prevlin ld a,h
        sub 8
        ld h,a
        sub &c0
        ret c
        add a,&c8
        ld h,a
        ld a,l
        sub 80
        ld l,a
        ret nc
        dec h
        ret


;routinen fuer mousepointer

clines  equ 12              ;pointergroesse
cbytes  equ 3


;zeile mit b bytes von (hl) nach (de)
;kopieren u. einen pkt versetzen

rcopy   xor a               ;links 0
nxprot  push bc
        ld b,a              ;neuer punkt
        ld a,(hl)
        rrca
        ld c,a              ;neuer+alter
        and &77             ;alter weg
        or b                ;+neuer
        ld (de),a           ;kopieren
        ld a,c              ;alt ausmask
        ld b,&88            ;maske alter
        and b               ;nur alter
        jr z,endrot         ;nicht gesetzt
        xor b               ;bits tauschen
        jr nz,endrot        ;farbe 1 or 2
        xor b               ;kein tausch
endrot  pop bc
        inc de
        inc hl
        djnz nxprot
        ret


;bitmap von (hl) nach (de) versetzt
;kopieren
```

```
bmrcopy ld c,clines
nxlbmr  ld b,cbytes+1      ;4 byte/zeile
        call rcopy
        dec c
        jr nz,nxlbmr
        ret


;vier versetzte bmaps von (hl) nach
;(de) einrichten

mrotmap push de            ;anfang erste
        ld a,clines
nxmrtm  ld bc,cbytes       ;alte zeilen
        ldir
        ex de,hl
        ld (hl),0          ;neue eins mehr
        inc hl
        ex de,hl
        dec a
        jr nz,nxmrtm
        pop hl             ;neuer start
        ld b,3             ;noch 3 versetzte
nxmrtm2 push bc
        call bmrcopy
        pop bc
        djnz nxmrtm2
        ret


;sprite (hl) in puffer expandieren
expspr  ld de,sprbuf
        push hl
        call mrotmap       ;bitmap sprite
        pop hl
        ld bc,clines*cbytes
        add hl,bc
        jp mrotmap         ;hintergrund


;sprite an position (de,hl) zeichnen
;und hintergrund speichern

cdraw   ld hl,(ptycor)     ;zuerst groesse
        ld de,(ptxcor)
        ld a,200
        sub l
        cp clines
        jr c,pssz1
        ld a,clines
pssz1   ld (sysize),a
        ld hl,319
        xor a
        ld c,a
        sbc hl,de
        or h
        ld a,cbytes+1
        jr nz,pssz3
        ld b,a             ;max breite

        ld a,1
pssz2   inc c
        sub 4
        jr c,pssz4
        djnz pssz2
pssz4   ld a,c
pssz3   ld (sxsize),a
        ld hl,(ptycor)
        call scrpos        ;adresse holen
        ld (ctadr),hl
        ex de,hl           ;scadr nach de
        ld l,b
        ld h,0
        add hl,hl
        add hl,hl
        add hl,hl
        add hl,hl          ;mal 16
        ld b,h
        ld c,l
        add hl,hl
        add hl,bc          ;mal 48
        ld bc,sprbuf
        add hl,bc          ;adr sprbody
        ld ix,sprback
        ld bc,(sysize)     ;ld c
dsnlin  ld a,(sxsize)
        ld b,a
        push de            ;scrnadr
        push hl            ;bitmapadr
dsnbyte push bc
        ld a,(de)          ;scrnbyte
        ld (ix+0),a        ;wegspeichern
        inc ix
        ld bc,4*12*4
        add hl,bc          ;adr sprframe
        or (hl)
        xor (hl)
        sbc hl,bc
        xor (hl)
        ld (de),a
        inc hl
        inc de
        pop bc
        djnz dsnbyte
        pop hl             ;bitmapadr
        ld de,cbytes+1
        add hl,de
        ex de,hl
        pop hl             ;scnbyte zanf
        call nextlin
        ex de,hl
        dec c
        jr nz,dsnlin
        ret


;pointer vom schirm loeschen

cremove ld de,(ctadr)
        ld hl,sprback
        ld bc,(sysize-1)   ;ld b

nlrem   push bc
        push de
        ld bc,(sxsize)
        ldir
        ex de,hl
        pop hl
        call nextlin
        ex de,hl
        pop bc
        djnz nlrem
        ret


;systemvariable
```

```
ctadr   defs 2             ;scadr pointer
sxsize  defs 1             ;groesse
        defb 0             ;hibyte
sysize  defs 1


;puffer fuer sprite und hintergrund

sprbuf  defs 12*4*4 *2
sprback defs 12*4


;routinen zur auswahl und bewegung -----
;des pointers -------------------------
;-------------------------------------


;einsprung hl-->adresse pointertabelle
;aussprung af,bc,hl,de,ix zerstoert

selectpointer
        ld a,(ptflag)
        or a
        jp z,expspr
        push hl
        call pointeroff
        pop hl
        call expspr
        ;jetzt pointeron


;einsprung -
;aussprung af,bc,de,hl,ix zerstoert

pointeron
        ld a,(ptflag)
        xor &ff
        ret z              ;bereits ein
        ld (ptflag),a
        call cdraw
        ld a,-2
        ld (evzahl),a      ;deaktiviert
        ld hl,evblock
        jp addfly          ;ereignis anfuegen


;einsprung -
;aussprung af,bc,de,hl,ix zerstoert

pointeroff
        ld a,(ptflag)
        xor &ff
        ret nz             ;ist aus
        ld (ptflag),a
        ld hl,evblock
        call delfly
        jp cremove


;einsprung de,hl bildschirmkoordinaten
;aussprung af zerstoert

.setpointer
        ld (ptxcor),de
        ld (ptycor),hl
        xor a
        ld (evzahl),a      ;event zulassen
        ret


;systemvariable

ptxcor  defs 2             ;position
ptycor  defs 2
ptflag  defb 0             ;ein/aus


;eventblock

evblock defs 2             ;frame kette
        defs 2             ;event kette
evzahl  defs 1             ;zahl
        defb &80           ;klasse
        defw evrout        ;routine
evrom   defb &ff           ;nur ram


;eventroutine

evrout  push ix
        call cremove
        call cdraw
        ld a,-2
        ld (evzahl),a      ;deaktivieren
        pop ix
        ret


;-------------------------------------
;routinen zur joystickabfrage
;-------------------------------------


;joystickbelegung loeschen

minit   ld bc,&ff48
ncljoy  ld a,c:call &bb27 ;normal
        ld a,c:call &bb2d ;shift

        ld a,c:call &bb33 ;control
        inc c
        ld a,76
        cp c
        jr nz,ncljoy
        ld hl,tikblk
        ld de,1
        ld b,d
        ld c,e
        call addtik
        jp mvinit


;tickerblock

tikblk  defs 2             ;tick kette
        defs 2             ;tick zahl
        defs 2             ;auflade zahl
        defs 2             ;event kette
        defb 0             ;-zahl
        defb &81           ;-klasse
        defw newmpos       ;-routine

; Position aktualisieren

newmpos call getmove
        ld a,h
        or l
        ret z
```

```
        ex de,hl
        ld a,e
        call ldbc
        ld hl,(ptycor)
        add hl,bc
        bit 7,h
        jr z,nwmp2
        ld hl,0
nwmp2   ld bc,200
        or a
        sbc hl,bc
        add hl,bc
        jr c,nwmp3
        ld hl,199
nwmp3   ld a,d
        ex de,hl        ;X nach DE
        call ldbc
        ld hl,(ptxcor)
        add hl,bc
        bit 7,h
        jr z,nwmp4
        ld hl,0
nwmp4   ld bc,320
        or a
        sbc hl,bc
        add hl,bc
        jr c,nwmp5
        ld hl,319
nwmp5   ex de,hl
        jp setpointer


.ldbc   ld c,a
        ld b,0
        or a
        ret p
        dec b
        ret


; Bewegung abfragen (Maus, Joystick, Tastaturroutine kommt hierher)
;
; H     Bewegung in X-Richtung (-128..+127)
; L     Bewegung in Y-Richtung (-128..+127)
; A     Zustand der Knoepfe
;       Bit-0   Knopf Nr 1
;       Bit-1   Knopf Nr 2
; Cflag gesetzt, bei Eingabe von Joystick/Maus, geloescht bei Tastatur

getmove call &bb24      ;Zustand von Joystick-0 nach A
        ld c,a
        and &40
        ld hl,mjflag
        xor (hl)
        jr nz,mjchange
        ld a,(hl)
        or a
        jr nz,getmla
;
        ld hl,evtoff    ;vom Event bemerkte Verschiebung
        ld e,(hl)       ;in Y-Richtung holen
        ld (hl),0       ;und zuruecksetzen
        inc hl
        ld d,(hl)       ;in X-Richtung holen
        ld (hl),0       ;und zuruecksetzen
        jr getmlb
;
getmla  ld hl,xmove     ;das gleiche bei der Maus
        ld d,(hl)
        ld (hl),0
        inc hl
        inc hl
        inc hl
        ld e,(hl)
        ld (hl),0
;
getmlb  ex de,hl        ;Bewegung nach HL
        ret


; Eingabegeraet wechseln

.mjchange
        call mvexit
        call mvinit
        jr getmove


; ohne Ueberlauf addieren/subtrahieren

dadd    add a,d
        ret po
        sub d
        ret


dsub    sub d
        ret po
        add a,d
        ret


evtoff  defs 1          ;Verschiebung in Y-Richtung
        defs 1          ;Verschiebung in X-Richtung


; Bewegungsroutine initialisieren

mvinit  call &bb24
        and &40         ;Spare-Bit am Joystick
        ld (mjflag),a
        ld bc,&81ff     ;&81=NEAR ADDRESS
        jr nz,mvinit2   ;Maus ist eingesteckt
        ld de,movevt
        ld hl,tikbk+6
        call &bcef      ;KL init event - Tikblock initialisieren
        ld hl,tikbk
        ld de,1
        ld b,d
        ld c,e
        jp &bce9        ;KL add ticker - Abfrageroutine einhaengen
;
mvinit2 ld de,mausint
        ld hl,tikbk+3
        call &bcef      ;KL init event - Tikblock initialisieren
        ld hl,tikbk
        jp &bce3        ;KL add fast Ticker


; ************************** Joystick-Abfrage Interrupt-Routine

movevt  call &bb24      ;Joystick abfragen
        ld hl,kprflag
        and &f          ;Knoepfe ausblenden
        jr nz,mve2      ;irgendwas ist gedrueckt
        ld (hl),a       ;sonst flag loeschen
        ret


mve2    ld d,4
        inc (hl)        ;Flag=FF ?
        jr z,mve5
        ld d,1
        dec (hl)        ;Flag=00 ?
        jr nz,mve3
        inc (hl)
        jr mve6
mve3    dec (hl)        ;Flag=01 ?
        jr nz,mve4
        ld (hl),-5
        ret
mve4    inc (hl)
        inc (hl)        ;sonst inkrementieren
mve6    inc (hl)
mve5    dec (hl)        ;wieder auf &ff bringen
        ld hl,(evtoff)
        rra             ;bit 0 testen
        ld c,a
        ld a,1

        call c,dsub     ;nach oben
        rr c            ;Bit 1
        call c,dadd     ;nach unten
        ld l,a
        ld a,h
        rr c            ;Bit 2
        call c,dsub     ;nach links
        rr c            ;Bit 3
        call c,dadd     ;nach rechts
        ld h,a
        ld (evtoff),hl  ;neues offset
        ret

kprflag defs 1          ;Zustand des Joysticks beim letzten Aufruf
        ;0= war nicht gedrueckt --> 1, Einzelschritt
        ;1= wurde gerade gedrueckt --> -5, keine Aktion
        ;-5..-2= Einzelschrittmodus --> +1, Einzelschritt
        ;-1= Schnellmodus --> -1, grosser Schritt


; ****************************** Mausabfrage Interrupt-Routine
;
; Joystick-0 nach A lesen
;
mausint ld bc,&f40e     ;Joystickabfrage wie
        di              ;Tastaturabfrage in
        out (c),c       ;der Interrupt-Routine
        ld b,&f6
        in a,(c)
        and &30
        ld c,a
        or &c0
        out (c),a
        out (c),c
        inc b
        push bc
        ld c,&92
        out (c),c
        and &30
        or &49
        dec b
        out (c),a
        ld b,&f4
        in a,(c)
        cpl
        ld e,a
        pop bc
        ld a,&82
        out (c),a
        dec b
        out (c),c
        ei
        ld a,e
;
; Auswertung der Maus-Bits
;
; Bit 0-1       auf / ab Raedchen
; Bit 2-3       links / rechts Raedchen
;
        and 3           ;links/rechts Bits isolieren

        ld hl,oldxbit
        ld c,(hl)
        ld (hl),a       ;alte X-Bits aktualisieren
        add a,a
        add a,a
        or c            ;alte 0-1, neue 2-3
        call getmdir
        ld a,e
        and 12          ;auf/ab Bits isolieren
        ld c,a
        ld a,(hl)
        ld (hl),c       ;alte Y-Bits aktualisieren
        rrca
        rrca
        or c            ;alte 0-1, neue 2-3
getmdir inc hl          ;alte Bewegungsrichtung
        add a,mdirtab mod 256
        ld c,a
        ld b,mdirtab/256
        ld a,(bc)
        or a
        jr z,gmdir4     ;keine Bewegung
        cp 2            ;Flag fuer alte Bewegrichtung beibehalten
        jr nz,gmdir2
        ld a,(hl)
        add a,a
        jr gmdir2b
gmdir2  ld (hl),a       ;neue Bewegungsrichtung eintragen
gmdir2b inc hl          ;summierte Bewegung
        add a,(hl)
        jp pe,gmdir3
        ld (hl),a       ;zurueckschreiben
gmdir3  inc hl          ;naechstes old?bit
        ret
gmdir4  inc hl
        inc hl
        ret


oldxbit defs 1          ;Mausstatus bei letzter Abfrage
oldxdir defs 1          ;Bewegungsrichtung bei letzter Abfrage
xmove   defs 1          ;summierte Bewegung

oldybit defs 1          ;Werte fuer X-Achse (Y siehe oben)
oldydir defs 1
ymove   defs 1


mdrest  equ $ mod 16    ;zu mdirtab mod 256 muss 15 addiert werden koennen
        defs 16-mdrest

mdirtab defb 0,1,-1,2   ;Tabelle der Bewegungsrichtungen
        defb -1,0,2,1
        defb 1,2,0,-1
        defb 2,-1,1,0
```

```
mjflag  defs 1          ;Flag =0 fuer Joystick, &40 fuer Maus


; Bewegungsroutine abmelden

mvexit  ld hl,tikbk
        ld a,(mjflag)
        or a
        jp z,&bcec      ;KL del ticker
        jp &bce6        ;KL del fast ticker

tikbk   defs 16


list:end
```

Listing 6

```
«30A0» 100 'PED.BAS
           '
           'Editor fuer Maeuse, Ratten, Zeiger,
           'Spruehdosen, Pointer und was es da
           'noch so alles gibt.
           '
«0B70» 110 '04/01/86 Guenter Radestock
           '
«05A6» 120 INK 0,0:INK 1,24:INK 2,2:INK 3,6:BOR
     DER 0
«05C6» 130 |PTOFF:MODE 1:PEN 1:PAPER 0
«0C6E» 140 KEY DEF 76,0,255,255,255:KEY DEF 77,
     0,255,255,255
«0DAE» 150 PRINT "▌     Gsoft Shape Editor vers
     ion 1       ▌"
«09C2» 160 PRINT "     Shape          Hintergrund"
«12E7» 170 PRINT "┌────────────┐ ┌────────────┐
     "
«04C9» 180 FOR i=1 TO 12
«077B» 190 PRINT "|            | |            |
     "
«0299» 200 NEXT i
«1303» 210 PRINT "└────────────┘ └────────────┘
     "
«01A1» 220 PRINT
«18B5» 230 PRINT "┌────────┐ ┌──────────────┐ ┌
     ─────────┐"
«09E1» 240 PRINT "|        | |              | |
            |"
«0FD1» 250 PRINT "| Farbe  | | Abspeichern  | |
      Zeigen  |"
«08F6» 260 PRINT "|        | |              | |
            |"
«17CC» 270 PRINT "└────────┘ └──────────────┘ └
     ─────────┘"
«0812» 280 LOCATE 31,13:PRINT "┌────────┐"
«04BF» 290 LOCATE 31,14:PRINT "|        |"
«060E» 300 LOCATE 31,15:PRINT "| Laden |"
«04D5» 310 LOCATE 31,16:PRINT "|        |"
«0838» 320 LOCATE 31,17:PRINT "└────────┘"
«066D» 330 WINDOW #1,1,40,25,25:PAPER #1,1:PEN
     #1,0:CLS #1
«069D» 340 PRINT #1," Filename: ▌←";
«021F» 350 DEFINT a-z
«06D8» 360 DIM pt(11,11),bk(11,11):GOSUB 820
«2581» 370 DEF FN ib (x1, y1, x2, y2) = x >= 16
     *(x1-1) AND x < 16*x2 AND y >= 16*(25-y1
     ) AND y < 16*(26-y2)
«0911» 380 color = 1:x = 0:y = 0
«07ED» 390 |PTON:WHILE (JOY (0) AND 16):WEND
«0743» 400 |GETPTPOS, @x, @y
«0A21» 410 a$ = INKEY$:IF a$ <> "" THEN GOSUB 1
     000
«078F» 420 IF (JOY (0) AND 16) = 0 GOTO 400
«19C6» 430 IF FN ib (30,22, 38,20) THEN IF if1
     THEN |PTSELECT:GOTO 390 ELSE GOSUB 640:i
     f1 = -1 ELSE if1 = 0
«0771» 440 IF FN ib (2,16, 13,5) GOTO 500
«06C1» 450 IF FN ib (17,16,28,5) GOTO 560
«1BDB» 460 IF FN ib (2,22, 9,20) THEN color = (
     color+1) MOD 4:LOCATE 3,21:PEN color:|PT
     OFF:PRINT "Farbe":GOTO 390
«083B» 470 IF FN ib (13,22, 26,20) THEN GOSUB 1
     100
«086C» 480 IF FN ib (32,16, 38,14) THEN GOSUB 1
     120
«0264» 490 GOTO 400
«0C75» 500 'punkt im pointerfeld setzen
«0CF8» 510 xx = x\16-1:yy = 20-y\16
«108B» 520 LOCATE xx+2,yy+5:PEN color:|PTOFF:PR
     INT "▌":|PTON
«0A78» 530 pt (xx,yy) = color:GOSUB 630
«164B» 540 WHILE FN ib (xx+2,yy+5, xx+2,yy+5):|
     GETPTPOS, @x, @y:WEND
«01A1» 550 GOTO 400
«1374» 560 xx = x\16-16:yy = 20-y\16:c = bk (xx
     ,yy)
«09DE» 570 LOCATE xx+17,yy+5:PEN 3:|PTOFF
«0765» 580 IF c THEN PRINT " " ELSE PRINT "▌"
«0C42» 590 GOSUB 630:|PTON:bk (xx,yy) = NOT c
«1E8D» 600 |GETPTPOS,@x,@y:IF (JOY (0) AND 16)
     = 0 GOTO 400 ELSE IF FN ib (xx+17,yy+5,
     xx+17,yy+5) GOTO 600
«173A» 610 IF NOT FN ib (17,16,28,5) THEN IF (J
     OY (0) AND 16) = 0 THEN 400 ELSE |GETPTP
     OS,@x,@y:GOTO 610
«0DA3» 620 xx = x\16-16:yy = 20-y\16:GOTO 570
«0F5F» 630 PLOT 264+16*xx,328-16*yy,pt (xx,yy):
     RETURN
«1030» 640 'muster im speicherbereich ab a000 a
     blegen
«0313» 650 |PTOFF
«0575» 660 FOR yy = 0 TO 11
«057D» 670 FOR xx = 0 TO 11
«0F57» 680 PLOT 2*xx+504,286-2*yy,pt (xx,yy)
«1062» 690 PLOT 2*xx+528,286-2*yy,-3*bk (xx,yy)
«0317» 700 NEXT xx
«0323» 710 NEXT yy
«0520» 720 bad! = 49775
«04F2» 730 FOR l=0 TO 11
«04DD» 740 FOR b=0 TO 2
«0CC4» 750 POKE &A000+3*l+b, PEEK(bad!+b)
«0DF9» 760 POKE &A024+3*l+b, PEEK(bad!+b+3)
«01CF» 770 NEXT b
«05C4» 780 bad! = bad!+&800
«0C3F» 790 IF bad! >= 65536 THEN bad! = bad!-16
     304
«043C» 800 NEXT l:|PTON
«00FC» 810 RETURN
«0EA4» 820 'vorhandenes muster in feld einlesen
«048F» 830 bad! = 49775
«052A» 840 FOR yy = 0 TO 11
«0495» 850 FOR i = 0 TO 2
«0CC8» 860 POKE bad!+i,PEEK (&A000+i+3*yy)
«0DFD» 870 POKE bad!+i+3,PEEK (&A024+i+3*yy)
«0244» 880 NEXT i
«055A» 890 FOR xx = 0 TO 11
«114C» 900 pt (xx,yy) = TEST (2*xx+504,286-2*yy
     )
«0EB2» 910 LOCATE xx+2,yy+5:PEN pt (xx,yy):PRIN
     T "▌"
«11A6» 920 bk (xx,yy) = TEST (2*xx+528,286-2*yy
     ) = 3
«10EA» 930 LOCATE xx+17,yy+5:PEN bk (xx,yy)* -3
     :PRINT "▌"
«020E» 940 GOSUB 630
«0312» 950 NEXT xx
«0678» 960 bad! = bad!+&800
«0CF3» 970 IF bad! >= 65536 THEN bad! = bad!-16
     304
«0332» 980 NEXT yy
«01B0» 990 RETURN
«062E» 1000 IF a$ < " " THEN RETURN
«0760» 1010 IF a$ = CHR$ (127) THEN 1060
«07F7» 1020 IF LEN (fi$) >= 16 THEN RETURN
«05CC» 1030 fi$ = fi$+a$
«0946» 1040 |PTOFF:PRINT #1, a$;"▌";"←";:|PTON
«00ED» 1050 RETURN
«05B7» 1060 IF fi$ = "" THEN RETURN
«09F9» 1070 |PTOFF:PRINT #1, " ";"←";"←";"▌";"←
     ";:|PTON
«0A18» 1080 fi$ = LEFT$(fi$, LEN (fi$)-1)
«0115» 1090 RETURN
«0EF3» 1100 IF fi$ <> "" THEN GOSUB 640:SAVE "!
     "+fi$, b, &A000, 12*3*2
«0129» 1110 RETURN
«1361» 1120 IF fi$<>"" THEN LOAD "!"+fi$, &A000
     :|PTOFF:GOSUB 820:|PTON:if1 = -1
«013D» 1130 RETURN
```

Listing 7

```
«08C2» 1   :'MC-Generator: PFEIL.ldr
«004B» 2   :
«080C» 3   :'erzeugt     : PFEIL.poi
«004D» 4   :
«0C3D» 5   :'Copyright   : Guenter Radestock
«004F» 6   :
«0EE0» 100 DATA 00,00,00,70,E0,00,70,C0,00,70,8
       0,00,70,C0,00,60,E0,00,40,70,00,00,&0690
«0F88» 101 DATA 30,80,00,10,C0,00,00,E0,00,00,4
       0,00,00,00,77,EE,00,FF,FF,00,FF,EE,&07F0
«1078» 102 DATA 00,FF,CC,00,FF,EE,00,FF,FF,00,E
       E,FF,88,44,77,CC,00,33,EE,00,11,FF,&0BE3
«0597» 103 DATA 00,00,EE,00,00,44,&0132
«01F8» 104 DATA EOF
«0070» 105 :
«02F8» 106 MEMORY &9FFF
«0CE4» 107 zeile= 100:schritt= 1:adr=&A000
«06BB» 108 PRINT"Zeile:"zeile ;
«0261» 109  READ b$
«073C» 110   IF      b$     ="EOF" GOTO 122
«06F4» 111   IF MID$(b$,1,1)="&"   GOTO 117
«06FB» 112    b   = VAL("&"+b$)
«04A3» 113    POKE adr,b
«0A43» 114    sum = sum + PEEK(adr)
«06E8» 115    adr = adr + 1
«01EA» 116  GOTO 109
«07AF» 117 checksum=VAL(b$)
«0ED7» 118 IF sum=checksum THEN v=6 ELSE v=174
«0522» 119 PRINT CHR$(1)CHR$(v)
«0E3F» 120 sum=0:zeile=zeile+schritt
«01CD» 121 GOTO 108
«0720» 122 SAVE"PFEIL.poi",b,&A000,&48
«0119» 123 END
```

Listing 8

```
«19B9» 100 'DOODLE.BAS
           '
           'Einfaches Zeichnen mit Pointer
           '
«02F5» 110 |PTOFF
«032E» 120 DEFINT x,y:MODE 1
«0B07» 130 x=0:y=0:|GETPTPOS,@x,@y
«0942» 140 x1=x:y1=y:|PTON
«0549» 150 WHILE INKEY$<>CHR$(13)
«057A» 160 FOR i=1 TO 30:NEXT
«071A» 170 |GETPTPOS,@x,@y
«0AF9» 180 IF x=x1 AND y=y1 GOTO 210
«1158» 190 IF JOY (0) AND 16 THEN |PTOFF:DRAW x
       ,y:|PTON ELSE MOVE x,y
«0739» 200 x1=x:y1=y
«0431» 210 WEND:|PTOFF
```

Listing 9

```
'program mausdemo;

|  (c) 1987 Guenter Radestock  |


const
  steine = 15;
  laenge = 4;

var
  feld: array [1..laenge, 1..laenge] of 0..steine;
  x, y: integer;


{$Imouse.inc}
{$Imdemo1.inc}



procedure initialisieren;

var
  x, y, nr: integer;

begin
  nr := 1;
  for y := 1 to laenge do
    for x := 1 to laenge do begin
      feld [x, y] := nr;
      nr := (nr+1) mod (steine+1)
    end
end;


function infeld (x1, y1, x2, y2: integer): boolean;

begin
  infeld := (xpos >= x1) and (xpos <= x2) and
            (ypos >= y1) and (ypos <= y2)
end;



function fwert(x, y: integer): integer;

begin
  if (x >= 1) and (x <= laenge) and (y >= 1) and (y <= laenge) then
    fwert := feld [x, y]
  else
    fwert := -1
end;



procedure swap (x, y, xleer, yleer: integer);

begin
  steinzeigen (x, y, feld [x, y], true);
  steinzeigen (xleer, yleer, feld [x, y], true);
  steinzeigen (x, y, 0, false);
  steinzeigen (xleer, yleer, feld [x, y], false);
  feld [xleer, yleer] := feld [x, y];
  feld [x, y] := 0
end;

procedure probiere (var xleer, yleer: integer; x, y: integer);

begin
  if (fwert (x, y) <> -1) then begin
    steinzeigen (xleer, yleer, feld [x, y], false);
    steinzeigen (x, y, 0, false);
    feld [xleer, yleer] := feld [x, y];
    feld [x, y] := 0;
    xleer := x;
    yleer := y
  end
end;



procedure mischen;

var
  i, xleer, yleer: integer;

begin
  while lbutton do
    ;
  randomize;
  xleer := 1;
  yleer := 1;
  while (feld [xleer, yleer] <> 0) do begin
    xleer := xleer+1;
    if (xleer > laenge) then begin
      xleer := 1;
      yleer := yleer+1
    end
  end;
  i := 1;
  while (i < 250) and not lbutton do begin
    case random (4) of
      0: probiere (xleer, yleer, xleer+1, yleer);
      1: probiere (xleer, yleer, xleer-1, yleer);
      2: probiere (xleer, yleer, xleer, yleer+1);
      3: probiere (xleer, yleer, xleer, yleer-1)
    end;
    i := i+1
  end
end;



begin
  initialisieren;
  write (chr (4), chr (2), ' Mausdemo in Turbo');
  write ('                    [ mischen ] [ Ausgangsstellung ] [ Ende ]');
  move (0, 380);
  draw (639, 380);
  xpos := 320;
  ypos := 100;
  initmouse;
  feldzeigen;
  repeat
    while lbutton do
      ;
    while not lbutton do
      ;
    if infeld (400,0,554,8) then begin
      initialisieren;

      loeschen;
      feldzeigen
    end
    else if infeld (150,43,374,172) then begin
      x := trunc ((xpos-150)/56)+1;
      y := trunc ((ypos-43)/32)+1;
      if (x < 1) then x := 1 else if (x > laenge) then x := laenge;
      if (y < 1) then y := 1 else if (y > laenge) then y := laenge;
      if (fwert (x+1, y) = 0) then
        swap (x, y, x+1, y)
      else if (fwert (x-1, y) = 0) then
        swap (x, y, x-1, y)
      else if (fwert (x, y+1) = 0) then
        swap (x, y, x, y+1)
      else if (fwert (x, y-1) = 0) then
        swap (x, y, x, y-1)
      else
        write (chr (7))
    end
    else if infeld (306,0,388,8) then
      mischen
  until infeld (570,0,626,8);
  delmouse;
  gotoxy (1,24)
end.
```

Listing 10

```
|       Mausroutinen fuer CP/M 2.2 und Turbopascal      |
|       auf CPC-464,  (c) Guenter Radestock     |

CONST
  xpos: INTEGER = 0;     |       Koordinaten des Mauspfeils      |
  ypos: INTEGER = 0;
  xshft: INTEGER = 1;    |       Verschiebung (jede Mausbewegung |
  yshft: INTEGER = 0;    |       wird mit 2^Verschiebung multipliziert)  |

VAR
  mpslct: INTEGER;


PROCEDURE mousecall;

|       Maschinenprogramme (siehe MAUS.SOU)     |

BEGIN
  INLINE (
  $21/*+$0287/$E5/$2A/MPSLCT/$11/*+$0004/$19/$E9/$C3/*+$0017/$C3/*+$0069
  /$C3/*+$0005/$C3/*+$0008/$CD/$9B/$BE/*+$00F4/$C9/$CD/$9B/$BE/*+$0115/$C9
  /$CD/$9B/$BE/*+$0003/$C9/$CD/*+$00E4/$01/$48/$FF/$79/$CD/$27/$BB/$79
  /$CD/$2D/$BB/$79/$CD/$33/$BB/$0C/$3E/$4E/$B9/$20/$EE/$06/$FF/$3E/$4C
  /$CD/$27/$BB/$06/$FF/$3E/$4D/$CD/$27/$BB/$01/$FF/$00/$11/*+$003E
  /$21/$02/$9D/$CD/$EF/$BC/$21/$00/$9D/$CD/$E3/$BC/$01/*+$0010/$2A/$01/$00
  /$23/$5E/$71/$23/$56/$70/$ED/$53/*+$000F/$C9/$CD/*+$0005/$C3/$00/$00
  /$CD/$9B/$BE/*+$000D/$11/$00/$00/$2A/$01/$00/$23/$73/$23/$72/$C9
  /$21/$00/$9D/$CD/$E6/$BC/$C3/*+$00A6/$01/$0E/$F4/$F3/$ED/$49/$06/$F6
  /$ED/$78/$E6/$30/$4F/$F6/$C0/$ED/$79/$ED/$49/$04/$C5/$0E/$92/$ED/$49
  /$E6/$30/$F6/$49/$05/$ED/$79/$06/$F4/$ED/$78/$2F/$5F/$C1/$3E/$82/$ED/$79
  /$05/$ED/$49/$FB/$7B/$E6/$03/$21/*+$0034/$4E/$77/$87/$87/$B1/$CD/*+$000B
  /$7B/$E6/$0C/$4F/$7E/$71/$0F/$0F/$B1/$23/$01/*+$0025/$81/$4F/$30/$01/$04
  /$0A/$B7/$28/$11/$FE/$02/$20/$04/$7E/$87/$18/$01/$77/$23/$86/$EA/*+$0003
  /$77/$23/$C9/$23/$23/$C9/$00/$00/$00/$00/$00/$00/$00/$01/$FF/$02
  /$FF/$00/$02/$01/$01/$02/$00/$FF/$02/$FF/$01/$00/$21/*+$0037/$7E/$B7/$C0
  /$36/$01/$2A/YPOS/$ED/$5B/XPOS/$DF/*+$0014/$01/$FF/$80/$11/*+$0023
  /$21/$12/$9D/$CD/$EF/$BC/$21/$10/$9D/$C3/$DA/$BC/*+$00B9/$FF/$21/*+$0010
  /$7E/$B7/$C8/$36/$00/$21/$10/$9D/$CD/$DD/$BC/$C3/*+$00EE/$00/$3A/*+$FFB2
  /$21/*+$FFB2/$B6/$C8/$DD/$E5/$7E/$36/$00/$21/YPOS/$11/$BC/$00
  /$ED/$4B/YSHFT/$CD/*+$0025/$21/*+$FF98/$7E/$36/$00/$21/XPOS/$11/$6F/$02
  /$ED/$4B/XSHFT/$CD/*+$0012/$CD/*+$00BD/$2A/YPOS/$ED/$5B/XPOS/$CD/*+$006C
  /$DD/$E1/$C9/$E5/$0C/$6F/$26/$00/$B7/$F2/*+$0003/$25/$0D/$28/$03/$29
  /$18/$FA/$E3/$C1/$E5/$7E/$23/$66/$6F/$09/$CB/$7C/$28/$05/$21/$00/$00
  /$18/$07/$B7/$ED/$52/$19/$38/$01/$EB/$EB/$E1/$73/$23/$72/$C9/$70/$00
  /$7E/$00/$7F/$C0/$7F/$F8/$7F/$00/$77/$80/$63/$5C0/$41/$E0/$00/$F0/$00/$78
  /$00/$20/$00/$00/$00/$00/$00/$00/$00/$00/$00/$00/$00/$00/$00/$00/$00/$00
  /$00/$00/$00/$00/$00/$00/$00/$00/$00/$00/$00/$00/$00/$00/$00/$00/$00/$00
  /$00/$CD/*+$006F/$22/*+$FFFA/$4F/$06/$0B/$DD/$21/*+$FFBC/$11/*+$FFCF/$C5
  /$E5/$79/$01/$03/$00/$ED/$B0/$EB/$E3/$E5/$DD/$56/$00/$DD/$5E/$01/$B7/$47
  /$3E/$00/$28/$07/$CB/$3A/$CB/$1B/$1F/$10/$F9/$47/$7E/$B2/$77/$23/$7E/$B3
  /$77/$23/$78/$B6/$77/$E1/$CD/*+$0025/$DD/$23/$DD/$23/$D1/$C1/$10/$CA/$C9
  /$21/*+$FF95/$ED/$5B/*+$FFB2/$06/$0B/$C5/$D5/$01/$03/$00/$ED/$B0/$E3
  /$CD/*+$0008/$EB/$E1/$C1/$10/$F0/$C9/$7C/$C6/$08/$67/$D0/$D6/$40/$67/$7D
  /$C6/$50/$6F/$D0/$24/$C9/$7D/$0F/$0F/$0F/$67/$E6/$1F/$6F/$87/$87/$85/$6F
  /$7C/$26/$00/$29/$29/$29/$29/$E6/$B0/$0F/$0F/$C6/$C0/$84/$67/$7B/$E6/$07
  /$CB/$3A/$CB/$1B/$CB/$3A/$CB/$1B/$CB/$3A/$CB/$1B/$19/$C9/$00
  )
END;

PROCEDURE initmouse;    |       Mausroutinen einschalten        |

BEGIN
  mpslct := $00;
  mousecall
END;

PROCEDURE delmouse;     |       Mausroutinen ausschalten        |

BEGIN
  mpslct := $03;
  mousecall
END;

PROCEDURE showmouse;    |       Mauspfeil einschalten   |

BEGIN
  mpslct := $06;
  mousecall
END;

PROCEDURE hidemouse;    |       Mauspfeil ausschalten   |

BEGIN
  mpslct := $09;
  mousecall
END;
```

Listing 11

```
|       Firmware-Interface fuer Turbo Pascal    |
|       Jede Firmware-Routine kann mit USRCALL(adresse) aufgerufen
|       werden. Die Variablen rAF, rBC, rDE, rHL enthalten vor  |
|       und nach dem Firmware-Aufruf den Zustand der Prozessor- |
|       register.       |

|       (c) Guenter Radestock   |

VAR
  rAF, rBC, rDE, rHL, usradr: INTEGER;


PROCEDURE usrcall (address: INTEGER);

BEGIN
  usradr := address;
  INLINE (
  $2A/USRADR/$22/*+$0015/$2A/RAF/$E5/$F1/$ED/$4B/RBC/$ED/$5B/RDE/$2A/RHL
  /$CD/$9B/$BE/$00/$00/$22/RHL/$ED/$53/RDE/$ED/$43/RBC/$F5/$E1/$22/RAF
  )
END;


FUNCTION lbutton: BOOLEAN;      |       linke Maustaste abfragen        |

BEGIN
  usrcall ($bb24);
  lbutton := (rAF AND $1000) > 0
END;


FUNCTION rbutton: BOOLEAN;      |       rechte Maustaste abfragen       |

BEGIN
  usrcall ($bb24);
  rbutton := (rAF AND $2000) > 0
END;
procedure move (x, y: integer);

begin
  rDE := x;
  rHL := y;
  usrcall ($bbc0)
end;


procedure draw (x, y: integer);

begin
  rDE := x;
  rHL := y;
  usrcall ($bbf6)
end;


procedure rahmen (x1, y1, x2, y2: integer);

begin
  move (x1, y1);
  draw (x2, y1);
  draw (x2, y2);
  draw (x1, y2);
  draw (x1, y1)
end;
procedure gcolor (col: integer);

begin
  rAF := col shl 8;
  usrcall ($bbde)
end;


procedure steinzeigen (x, y, nummer: integer; invers: boolean);

begin
  hidemouse;
  if invers then
    write (chr (24));
  gotoxy (14+7*x, 3+4*y);
  write ('      ');
  gotoxy (14+7*x, 4+4*y);
  if (nummer > 0) then
    write (' ', nummer: 2, '  ')
  else begin
    write ('      ');
    gcolor (0)
  end;
  gotoxy (14+7*x, 5+4*y);
  write ('      ');
  if invers then
    write (chr (24));
  rahmen (101+56*x, 370-64*y, 146+56*x, 316-64*y);
  gcolor (1);
  showmouse
end;


procedure feldzeigen;

var
  x, y: integer;

begin
  rahmen (152, 312, 375, 54);
  for x := 1 to laenge do
    for y := 1 to laenge do
      steinzeigen (x, y, feld [x, y], false)
end;


procedure loeschen;

var
  x, y: integer;

begin
  for y := laenge downto 1 do
    for x := laenge downto 1 do
      steinzeigen (x, y, 0, false)
end;
```

Listing 12

```
;;      Mauseinbindung fuer Turbopascal
;;      Quelltext fuer INLASS Inline-Assembler
;;        aus "CHIP Special Turbopascal"
;;      1988, Guenter Radestock


; externe Variable (in Turbo definiert)

MPSLCT  ext             ;Nummer des Maschinen Unterprogramms
xpos    ext             ;Koordinaten des Mauspfeils
ypos    ext
xshft   ext             ;Verschiebmaske fuer Pfeilbewegung
yshft   ext


; Programm MPSLCT ausfuehren
```

```
amscall equ 0be9bh      ;schaltet CP/M interrupt normal

        ld hl,endproc   ;am Programmende nach Endproc springen
        push hl
        ld hl,(MPSLCT)  ;Adresse in der Sprungtabelle errechnen
        ld de,jmptab
        add hl,de
        jp (hl)

jmptab  jp minit        ;00     Initialisierung
        jp delint       ;03     Interrupts loeschen
        jp pointer      ;06     Pfeil einschalten
        jp noptr        ;09     Pfeil ausschalten


pointer call amscall
        defw pfon
        ret

noptr   call amscall
        defw pfoff
        ret

; Mausroutinen initialisieren

minit   call amscall    ;Initialisierung benoetigt AMSdos Vektoren
        defw minitA
        ret


minita  call pfon       ;hier geht es weiter

; Joystick in der Tastaturmatrix so umbelegen, dass
; er keine Tastaturcodes mehr erzeugt

lbutton equ 255
rbutton equ 255
        ld bc,0ff48h    ;Joystick-Tastenbelegung loeschen
clrjoy  ld a,c
        call 0bb27h     ;Taste alein
        ld a,c
        call 0bb2dh     ;Taste mit Shift
        ld a,c
        call 0bb33h     ;Taste mit Control
        inc c
        ld a,78
        cp c
        jr nz,clrjoy
        ld b,lbutton
        ld a,76         ;linken Knopf
        call 0bb27h     ;umbelegen
        ld b,rbutton
        ld a,77         ;rechten Knopf
        call 0bb27h     ;umbelegen


; Interruptroutine fuer Mausabfrage installieren

        ld bc,80ffh     ;80h= far address
        ld de,mausint
        ld hl,tikblk+2
        call 0bcefh     ;KL init event - Tikblock initialisieren
        ld hl,tikblk
        call 0bce3h     ;KL add fast Ticker


; Interrupt bei Warmstart (Runtime/IO Error)
; automatisch entfernen

        ld bc,delint0   ;Absolute Adressen laedt INLASS nur 16BitWeise
        ld hl,(1)       ;Warmstart-Vektor holen
        inc hl          ;erstes Byte ist IMMER ein jp
        ld e,(hl)       ;alten Warmstart nach DE
        ld (hl),c
        inc hl
        ld d,(hl)
        ld (hl),b
        ld (owarm+1),de

        ret


delint0 call delint     ;bei Warmstart zuerst Int entfernen &
                        ;Warmstartvektor korrigieren
        jp 0            ;dann Warmstart ausfuehren

; Interruptroutine entfernen & Warmstartvektor korrigieren

delint  call amscall
        defw deltik     ;Ticker entfernen
owarm   ld de,0         ;(0 wird oben geaendert)
        ld hl,(1)       ;Warmstart-Vektor holen
        inc hl          ;erstes Byte ist IMMER ein jp
        ld (hl),e       ;alte Sprungadresse einsetzen
        inc hl
        ld (hl),d

        ret

; Interrupts aus AMSdos ausklinken

deltik  ld hl,tikblk
        call 0bce6h     ;KL del fast Ticker
        jp pfoff


; ******************************** Mausabfrage Interrupt-Routine

tikblk  equ 9d00h       ;Ticker-Block muss im Kern-Ram liegen

;
; Joystick-0 nach A lesen
;
mausint ld bc,0f40eh    ;Joystickabfrage wie
        di              ;Tastaturabfrage in
        out (c),c       ;der Interrupt-Routine
        ld b,0f6h
        in a,(c)
        and 30h
        ld c,a
        or 0c0h
        out (c),a
        out (c),c
        inc b
        push bc
        ld c,92h
        out (c),c
        and 30h
        or 49h
        dec b
        out (c),a
        ld b,0f4h
        in a,(c)
        cpl
        ld e,a
        pop bc
        ld a,82h
        out (c),a
        dec b
        out (c),c
        ei
        ld a,e
;
; Auswertung der Maus-Bits
;
; Bit 0-1       auf / ab Raedchen
; Bit 2-3       links / rechts Raedchen
;
        and 3           ;links/rechts Bits isolieren
        ld hl,oldxbit
        ld c,(hl)
        ld (hl),a       ;alte X-Bits aktualisieren
        add a,a
        add a,a
        or c            ;alte 0-1, neue 2-3
        call getmdir
        ld a,e
        and 12          ;auf/ab Bits isolieren
        ld c,a
        ld a,(hl)
        ld (hl),c       ;alte Y-Bits aktualisieren
        rrca
        or c            ;alte 0-1, neue 2-3
getmdir inc hl          ;alte Bewegungsrichtung
        ld bc,mdirtab
        add a,c
        ld c,a
        jr nc,gmdir1a
        inc b
gmdir1a ld a,(bc)
        or a
        jr z,gmdir4     ;keine Bewegung
        cp 2            ;Flag fuer alte Bewegrichtung beibehalten
        jr nz,gmdir2
        ld a,(hl)
        add a,a
        jr gmdir2b
gmdir2  ld (hl),a       ;neue Bewegungsrichtung eintragen
gmdir2b inc hl          ;summierte Bewegung
        add a,(hl)
        jp pe,gmdir3
        ld (hl),a       ;zurueckschreiben
gmdir3  inc hl          ;naechstes old?bit
        ret
gmdir4  inc hl
        inc hl
        ret


oldxbit defs 1          ;Mausstatus bei letzter Abfrage
oldxdir defs 1          ;Bewegungsrichtung bei letzter Abfrage
xmove   defs 1          ;summierte Bewegung

oldybit defs 1          ;Werte fuer X-Achse (Y siehe oben)
oldydir defs 1

ymove   defs 1


mdirtab defb 0,1,-1,2   ;Tabelle der Bewegungsrichtungen
        defb -1,0,2,1
        defb 1,2,0,-1
        defb 2,-1,1,0


; ********************** Ende der Fast-Ticker Interrupt Routine

; Pfeil einschalten

pfon    ld hl,pfflag
        ld a,(hl)
        or a
        ret nz          ;Pfeil ist bereits eingeschaltet
        ld (hl),1
        ld hl,(ypos)
        ld de,(xpos)
        rst 18h         ;Far Call, um RAM einzuschalten
        defw pfon2      ;Pfeil zeichnen
        ld bc,80ffh     ;80h= far address
        ld de,pfint
        ld hl,tikblk+16+2
        call 0bcefh     ;KL init event - Tikblock initialisieren
        ld hl,tikblk+16
        jp 0bcdah       ;KL add frame fly

pfon2   defw pdraw
        defb 255


; Pfeil ausschalten

pfoff   ld hl,pfflag
        ld a,(hl)
        or a
        ret z           ;Pfeil ist garnicht eingeschaltet
        ld (hl),0
        ld hl,tikblk+16
        call 0bcddh     ;KL del frame fly
        jp restore      ;Pfeil loeschen


; Flag, ob Pfeil eingeschaltet

pfflag  defb 0


; Interrupt-Routine zum Neuzeichnen des Pfeils
```

```
pfint   ld a,(xmove)
        ld hl,ymove
        or (hl)
        ret z           ;keine Bewegung.
        push ix
        ld a,(hl)       ;Verschiebung nach A
        ld (hl),0
        ld hl,ypos      ;Adresse der Variablen nach HL
        ld de,ymax      ;MAX nach DE
        ld bc,(yshft)   ;Verschiebung nach C
        call pfint2     ;Variable korrigieren
        ld hl,xmove     ;das gleiche mit der X-Koordinate
        ld a,(hl)
        ld (hl),0
        ld hl,xpos
        ld de,xmax
        ld bc,(xshft)
        call pfint2
        call restore    ;alten Pfeil loeschen
        ld hl,(ypos)
        ld de,(xpos)
        call pdraw      ;neuen Pfeil zeichnen
        pop ix
        ret
        ;
pfint2  push hl
        inc c           ;Verschiebung+1
        ld l,a          ;A mit richtigem Vorzeichen nach HL
        ld h,0
        or a
        jp p,pfint3
        dec h           ;(bei negativem A  B = -1)
pfint3  dec c
        jr z,pfint3a
        add hl,hl       ;sooft wie "verschiebung" verdoppeln
        jr pfint3
pfint3a ex (sp),hl      ;Variablenadr holen
        pop bc          ;Offset nach BC
        push hl         ;Adresse der Variablen retten
        ld a,(hl)       ;Variable nach HL holen
        inc hl
        ld h,(hl)
        ld l,a
        add hl,bc       ;Verschiebung addieren
        bit 7,h
        jr z,pfint4     ;Ergebnis ist positiv
        ld hl,0
        jr pfint5
pfint4  or a
        sbc hl,de       ;HL mit MAX vergleichen
        add hl,de
        jr c,pfint5     ;HL < MAX
        ex de,hl        ;sonst HL = MAX.
pfint5  ex de,hl        ;neue Koordinate nach DE
        pop hl          ;Adresse nach HL
        ld (hl),e       ;und zurueckschreiben
        inc hl
        ld (hl),d
        ret


; Bitmuster des Maus Pfeiles

pfeil   defb 1110000b , 0
        defb 1111110b , 0
        defb 1111111b , 11000000b
        defb 1111111b , 11111000b
        defb 1111111b , 0
        defb 1110111b , 10000000b
        defb 1100011b , 11000000b
        defb 1000001b , 11100000b
        defb 0        , 11110000b
        defb 0        ,  1111000b
        defb 0        ,   100000b

hoehe   equ 11          ;Ausmasse der Pfeil-Bitmap
breite  equ 2

ymax    equ 199-hoehe
xmax    equ 623         ;639-(breite*8)

; Platz fuer den Bildschirmhintergrund

breil   equ breite+1    ;bei verschiedenen Assemblern so

hinter  defs breil*hoehe

lstpos  defs 2          ;Bildschirmadresse des Hintergrundes

; Pfeil zeichnen & Hintergrund speichern
;
;       L       Y-Koordinate
;       DE      X-Koordinate

pdraw   call dotpos     ;Bildschirmadresse holen
        ld (lstpos),hl  ;Adresse zum Restaurieren sichern
        ld c,a          ;Verschiebung nach C
        ld b,hoehe      ;Anzahl Zeilen nach B
        ld ix,pfeil
        ld de,hinter
pdraw2  push bc
        push hl
        ld a,c          ;Verschiebung+1
        ld bc,breite+1
        ldir            ;Hintergrund sichern
        ex de,hl
        ex (sp),hl      ;Hintergrundadr retten
        push hl         ;Bildschirmadresse
        ld d,(ix+0)     ;Pfeilbitmuster nach DE
        ld e,(ix+1)
        or a
        ld b,a
        ld a,0
        jr z,pdraw4
pdraw3  srl d           ;DEA verschieben
        rr e
        rra
        djnz pdraw3
pdraw4  ld b,a          ;DEA in den Bildschirm odern
        ld a,(hl)
        or d
        ld (hl),a
        inc hl
        ld a,(hl)
        or e
        ld (hl),a
        inc hl
        ld a,b
        or (hl)
        ld (hl),a
        pop hl
        call nextlin    ;naechste Bildschirmzeile
        inc ix          ;naechste Bitmapzeile
        inc ix
        pop de
        pop bc
        djnz pdraw2
        ret

; Hintergrund restaurieren

restore ld hl,hinter    ;Adresse Hintergrund
        ld de,(lstpos)  ;Bildschirmadresse
        ld b,hoehe      ;Anzahl Zeilen
rstore  push bc
        push de
        ld bc,breite+1  ;Anzahl Bytes pro Zeile
        ldir            ;in den Bildschirm transferieren
        ex (sp),hl      ;Bildschirmadresse wiederholen
        call nextlin
        ex de,hl        ;neue Bildschirmadresse nach de
        pop hl
        pop bc
        djnz rstore
        ret

; Adresse der naechsten Bildschirmzeile

nextlin ld a,h          ;HEX 800 addieren
        add a,8
        ld h,a
        ret nc          ;bei Ueberlauf (Bildshcirm bis FFFF)
        sub 64          ;HEX 4000 subtrahieren
        ld h,a
        ld a,l
        add a,80        ;und 80 addieren
        ld l,a
        ret nc
        inc h
        ret

; Bildschirmadresse und Anzahl Verschiebungen ausrechnen
;
; nimmt         DE      x-Koordinate
;               L       y-Koordinate
;
; liefert       HL      Bildschirmadresse
;               A       Anzahl Verschiebebits

dotpos  ld a,l          ;Zeile
        rrca            ;durch acht
        rrca
        rrca
        ld h,a
        and 1+2+4+8+16  ;geschobene bits ausblenden
        ld l,a
        add a,a         ;*4
        add a,a
        add a,l         ;*5
        ld l,a          ;nach HL
        ld a,h
        ld h,0
        add hl,hl
        add hl,hl
        add hl,hl
        add hl,hl       ;insgesamt *80
        and 128+64+32   ;obere 3 Bits (urspruenglich untere)
        rrca            ;untere Bits *8
        rrca
        add a,192       ;=0c0h, Hibyte Bildschirmanfang
        add a,h         ;zum Zeilenoffset
        ld h,a
        ld a,e
        and 7           ;Spalte untere 3 Bits
        srl d           ;Zeile 3 Bits nach links schieben
        rr e
        srl d
        rr e
        srl d
        rr e
        add hl,de       ;Spalte/8 addieren
        ret


endproc nop             ;lokale Variable wieder ordnungsgemaess entfernen

        end
```

Listing 13

```
«0879» 10 REM (c) Guenter Radestock
«05B5» 20 REM Maus-Version
«01B0» 30 CALL &1C8
«044E» 40 RUN "????????.???"
```

Listing 14

```
«08C6» 1   :'MC-Generator: HELLO.ldr
«004B» 2   :
«0801» 3   :'erzeugt      : HELLO.bin
«004D» 4   :
«0C3D» 5   :'Copyright    : Guenter Radestock
«004F» 6   :
«0FC9» 100 DATA CD,D2,01,21,30,02,E5,C3,26,03,C
       D,24,BB,E6,40,CA,36,02,3A,39,00,FE,&0909
«0F43» 101 DATA 39,21,FE,1C,28,06,FE,41,21,86,1
       E,C0,22,13,03,21,61,02,11,00,40,01,&0574
«108B» 102 DATA C5,00,ED,B0,01,48,FF,79,CD,27,B
       B,79,CD,2D,BB,79,CD,33,BB,0C,3E,4E,&0ACC
«109E» 103 DATA B9,20,EE,06,E0,3E,4C,CD,27,BB,0
       6,FC,3E,4D,CD,27,BB,CD,24,BB,01,FF,&0ACE
«0FAA» 104 DATA 81,11,10,40,21,02,40,CD,EF,BC,2
       1,00,40,C3,E3,BC,21,00,40,C3,E6,BC,&0946
«1053» 105 DATA 3E,48,06,F0,CD,27,BB,3E,49,06,F
       1,CD,27,BB,3E,4A,06,F2,CD,27,BB,3E,&09C5
«1061» 106 DATA 4B,06,F3,CD,27,BB,3E,4C,06,E0,C
       D,27,BB,3E,4D,06,FC,CD,27,BB,C9,00,&0A17
«0EE0» 107 DATA 00,00,00,00,00,00,00,00,00,00,0
       0,00,00,00,01,0E,F4,F3,ED,49,06,&0332
«1051» 108 DATA F6,ED,78,E6,30,4F,F6,C0,ED,79,E
       D,49,04,C5,0E,92,ED,49,E6,30,F6,49,&0D06
«1062» 109 DATA 05,ED,79,06,F4,ED,78,2F,5F,C1,3
       E,82,ED,79,05,ED,49,FB,7B,E6,03,21,&0AFA
«1022» 110 DATA 79,40,4E,77,87,87,B1,CD,5B,40,7
       B,E6,0C,4F,7E,71,0F,0F,B1,CD,5B,40,&0987
«0F6C» 111 DATA 18,35,23,C6,80,4F,06,40,0A,B7,2
       8,11,FE,02,20,04,7E,87,18,01,77,23,&0621
«0F61» 112 DATA 86,EA,74,40,77,23,C9,23,23,C9,0
       0,00,00,00,00,00,00,00,01,FF,02,FF,&0697
«0F65» 113 DATA 00,02,01,01,02,00,FF,02,FF,01,0
       0,21,7B,40,CD,B4,40,38,0B,01,01,01,&04EA
«0F4E» 114 DATA 20,03,01,00,02,CD,B1,40,21,7E,4
       0,CD,B4,40,D8,01,00,01,20,02,06,04,&058A
«1055» 115 DATA C3,00,00,7E,B7,FA,BF,40,D6,06,D
       8,77,AF,C9,C6,06,77,F6,01,C9,CD,48,&0BAC
«0FCF» 116 DATA BB,CD,A9,03,04,02,1C,00,1A,1A,1
       C,01,00,00,1D,0E,0E,FF,11,00,80,CD,&053D
«103D» 117 DATA 9B,BC,AF,32,92,06,3A,92,06,CD,B
       7,03,FE,E0,20,02,3E,0D,FE,0D,CA,2B,&0974
«106A» 118 DATA 04,FE,11,CA,63,06,CD,22,05,FE,4
       4,CA,A5,05,FE,52,CA,EF,05,FE,56,CA,&0B1C
«101B» 119 DATA 89,05,FE,50,CA,8F,05,FE,FC,CA,2
       6,03,21,92,06,D6,F0,38,0A,20,08,7E,&098E
«0FE8» 120 DATA D6,04,38,C0,77,18,BD,3D,20,0A,7
       E,C6,04,FE,50,30,B3,77,18,B0,3D,20,&089A
«0FE9» 121 DATA 07,34,35,28,A9,35,18,A6,3D,20,A
       3,7E,FE,4F,30,9E,34,18,9B,E3,7E,23,&0838
«106D» 122 DATA FE,FF,28,05,CD,5A,BB,18,F5,E3,C
       9,2E,03,2C,D6,04,30,FB,C6,04,67,87,&0ADF
«1029» 123 DATA 87,84,87,87,3C,67,22,E4,03,CD,7
       5,BB,06,11,CD,E6,03,CD,06,BB,F5,2A,&0A3C
«1065» 124 DATA E4,03,CD,75,BB,06,11,CD,E6,03,F
       1,C9,00,00,CD,8A,BB,3E,09,CD,5A,BB,&0AA6
«1018» 125 DATA 10,F6,C9,2A,E4,03,CD,75,BB,21,1
       9,04,06,0C,CD,60,BB,77,23,3E,09,CD,&08C3
«0FD6» 126 DATA 5A,BB,10,F4,21,21,04,7E,FE,2E,2
       8,04,E1,C3,44,03,CD,A9,03,04,01,00,&079E
«0F7F» 127 DATA 00,00,00,00,00,00,00,00,00,00,0
       0,0D,0A,0A,0A,FF,C9,CD,F1,03,23,E5,&04BC
«1027» 128 DATA 11,BA,04,CD,B0,04,E1,CA,C3,04,E
       5,11,BD,04,CD,B0,04,E1,CA,C3,04,11,&0A7D
«0FF9» 129 DATA C0,04,CD,B0,04,CA,57,06,CD,A9,0
       3,41,6E,7A,65,69,67,65,6E,0D,0A,44,&0871
«102E» 130 DATA 72,75,63,6B,65,6E,FF,21,04,01,E
       5,CD,75,BB,06,08,CD,E6,03,CD,06,BB,&09E1
«1079» 131 DATA E1,F5,E5,CD,75,BB,06,08,CD,E6,0
       3,E1,F1,FE,FC,CA,26,03,FE,E0,28,12,&0D53
«107C» 132 DATA FE,0D,28,0E,FE,F0,38,D6,FE,F2,3
       0,D2,3E,09,95,6F,18,CC,3E,09,95,6F,&0AA9
«1054» 133 DATA F5,CD,75,BB,3E,12,CD,5A,BB,F1,F
       E,05,CA,34,05,C3,64,05,06,03,1A,BE,&0A28
«0F77» 134 DATA C0,23,13,10,F9,C9,42,41,53,42,4
       9,4E,43,4F,4D,21,19,04,11,B0,01,01,&0657
«0F9B» 135 DATA 0C,00,ED,B0,C9,21,00,00,22,93,0
       6,CD,7D,BC,21,19,04,06,0C,11,00,80,&0635
«1087» 136 DATA CD,77,BC,D8,E1,CD,7A,BC,D4,7D,B
       C,CD,2B,05,C3,26,03,21,93,06,7E,B7,&0BA1
«102E» 137 DATA 20,22,CD,80,BC,D0,FE,1A,C8,E6,7
       F,21,94,06,34,FE,0A,20,02,36,00,FE,&09AD
«1008» 138 DATA 09,37,C0,AF,7E,36,00,3D,ED,44,F
       6,08,2B,77,3D,E6,07,77,3E,20,37,C9,&0870
«10D5» 139 DATA FE,61,D8,FE,7B,D0,D6,20,C9,CD,8
       1,BB,CD,06,BB,C3,84,BB,CD,CF,04,CD,&0E45
«101C» 140 DATA A9,03,04,02,FF,CD,F1,04,D2,E5,0
       4,FE,20,30,08,FE,0D,28,04,FE,0A,20,&08E3
«10C4» 141 DATA 03,CD,5A,BB,CD,09,BB,30,E6,CD,2
       B,05,FE,FC,20,DF,CD,0C,BB,C3,E5,04,&0BC2
«1099» 142 DATA CD,A9,03,0A,0A,FF,CD,CF,04,CD,F
       1,04,D2,E5,04,4F,79,CD,2B,BD,38,F3,&0B51
«109F» 143 DATA CD,09,BB,30,F5,FE,FC,20,F1,CD,0
       C,BB,C3,E5,04,CD,F1,03,C3,34,05,CD,&0C8B
«1032» 144 DATA F1,03,CD,A9,03,70,72,69,6E,74,6
       9,6E,67,20,2E,2E,2E,FF,C3,64,05,CD,&097A
«103F» 145 DATA F1,03,CD,A9,03,44,65,6C,65,74,6
       5,3F,20,FF,CD,2B,05,CD,22,05,FE,4E,&095B
«1053» 146 DATA CA,26,03,FE,FC,CA,26,03,FE,59,C
       2,B4,05,CD,A9,03,59,65,73,2E,0D,0A,&09A1
«1044» 147 DATA FF,21,EC,05,CD,D4,BC,DD,21,E5,0
       5,3E,01,CD,1B,00,C3,26,03,E9,05,51,&09A8
«0FD8» 148 DATA 06,0C,19,04,45,52,C1,CD,F1,03,C
       D,A9,03,45,6E,74,65,72,20,6E,65,77,&0829
«1032» 149 DATA 20,6E,61,6D,65,3A,20,FF,21,95,0
       6,06,00,CD,2B,05,FE,FC,CA,26,03,FE,&08C4
«1049» 150 DATA 0D,28,24,FE,20,38,F0,FE,7F,28,0
       E,77,78,FE,0C,28,E6,04,7E,23,CD,5A,&0925
«0FF4» 151 DATA BB,18,DE,04,05,28,DA,05,2B,CD,A
       9,03,08,10,FF,18,D0,78,32,51,06,21,&0786
«0FF7» 152 DATA 54,06,CD,D4,BC,DD,21,E5,05,3E,0
       2,CD,1B,00,C3,26,03,00,95,06,52,45,&07E5
«1011» 153 DATA CE,21,60,06,CD,D4,BC,C3,16,BD,4
       3,50,CD,21,77,BC,11,8F,06,01,03,00,&08A6
«1044» 154 DATA ED,B0,3E,C3,32,77,BC,21,82,06,2
       2,78,BC,CD,A9,03,1F,01,19,0A,FF,C9,&0986
«0F6A» 155 DATA 21,8F,06,11,77,BC,01,03,00,ED,B
       0,AF,C9,00,00,00,00,00,00,00,00,00,&0513
«022C» 156 DATA EOF
«00A4» 157 :
«02CC» 158 MEMORY &3FFF
«0CB8» 159 zeile= 100:schritt= 1:adr=&4000
«06EF» 160 PRINT"Zeile:"zeile ;
«0295» 161  READ b$
«07A4» 162   IF      b$     ="EOF" GOTO 174
«075C» 163   IF MID$(b$,1,1)="&"   GOTO 169
«072F» 164    b    = VAL("&"+b$)
«04D7» 165    POKE adr,b
«0A77» 166    sum = sum + PEEK(adr)
«071C» 167    adr = adr + 1
«0252» 168  GOTO 161
«07E3» 169 checksum=VAL(b$)
«0F0B» 170 IF sum=checksum THEN v=6 ELSE v=174
«0556» 171 PRINT CHR$(1)CHR$(v)
«0E73» 172 sum=0:zeile=zeile+schritt
«0235» 173 GOTO 160
«0775» 174 SAVE"HELLO.bin",b,&4000,&4D0
«014D» 175 END
```