```
                                Pagesize        105
                                Pagewidth       120
                                cseg
0000 EB3F       0041            Jmps    Start
0002 436F70797269               DB      'Copyright by Ihno Krumreich'
     676874206279
     2049686E6F20
     4B72756D7265
     696368
001D 4F726967696E               DB      'Original in Cobol von Ken Richardsen'
     616C20696E20
     436F626F6C20
     766F6E204B65
     6E2052696368
     61726473656E
0041 B191               Start:  Mov     CL,91H          ; Erhoehung der Prioritaet von
0043 B2C7                       Mov     DL,0C7H         ; C8 ( Standart ) auf C6
0045 CDE0                       Int     224             ;
0047 B12D                       Mov     CL,2DH          ;
0049 B2FF                       Mov     DL,0FFH         ;
004B CDE0                       Int     224             ;
004D FC                         Cld                     ; bleibt fuer den Rest des
                                                        ; Programmes geloescht
004E E81005     0561            Call    Get_Time        ;
0051 03062A0C                   Add     AX,Dummy        ;
0055 03062C0C                   Add     AX,Dummy+2      ;
0059 A30401                     Mov     Old_Random,AX   ;
005C 1E                         Push    DS              ;
005D 07                         Pop     ES              ;
005E BA4101                     Mov     DX,Offset Wellcome  ; Frage nach spielanleitung
                                                        ; ausgeben
0061 E8D104     0535            Call    WriteStr        ;
0064 E8D304     053A            Call    GetChar         ; Antwort erfragen
0067 24DF                       And     AL,0DFH         ; auf Grossbuchstaben setzen
0069 3C4A                       Cmp     AL,'J'          ;
006B 7404       0071            Je      Anleitung       ; Anleitung ausgeben
006D 3C59                       Cmp     AL,'Y'          ;
006F 7503       0074            Jne     Spiel           ; keine Anleitung ==> spielen
0071 E84404     04B8 Anleitung: Call    Inst            ; Anleitung ausgeben
0074 803E010100      Spiel:     Cmp     Byte Ptr RaumNummer,0
0079 7444       00BF            Jz      No_Show         ;
007B A10601                     Mov     AX,Player       ;
007E 3C0A                       Cmp     AL,10           ;
0080 B00A                       Mov     AL,10           ;
0082 7202       0086            Jb      Unten           ;
0084 32C0                       Xor     AL,AL           ;
0086 32E4               Unten:  Xor     AH,AH           ;
0088 E80304     048E            Call    Pos_Curser      ;
008B A11401                     Mov     AX,PunkteG      ;
008E 03061201                   Add     AX,Punkte_R     ;
0092 A31401                     Mov     Punkte_G,AX     ;
0095 BF1C02                     Mov     DI,Offset ASPunkteG ;
0098 B104                       Mov     CL,4            ;
009A C606000101                 Mov     Byte Ptr Blank,True
009F E82404     04C6            Call    Int_to_Ascii    ;
00A2 A11201                     Mov     AX,Punkte_R     ;
00A5 BFF801                     Mov     DI,Offset ASPunkteR ;
00A8 B104                       Mov     CL,4            ;
00AA C606000101                 Mov     Byte Ptr Blank,True
00AF E81404     04C6            Call    Int_to_Ascii        ;
00B2 BA6C01                     Mov     DX,Offset RaumRahmen;
00B5 E87D04     0535            Call    Write_Str       ;
00B8 E87F04     053A Again3:    Call    GetChar         ; Auf Zeichen warten
00BB 3C0D                       Cmp     AL,CR           ;
00BD 75F9       00B8            Jnz     Again3          ;
00BF BF6502             No_Show:Mov     DI,Offset Screen; Hintergrund Bildschirm loeschen
00C2 B82020                     Mov     AX,2020H        ;
00C5 B93007                     Mov     CX,ScreenBytes  ; nur * 12 da Wortweise
00C8 D1E9                       Shr     CX,1            ; Div 2
00CA F3AB                 Rep   StosW                   ;
00CC BA2F01                     Mov     DX,Offset ClrScr; Bildschirm loeschen
00CF E86304     0535            Call    WriteStr        ;
00D2 FE060101                   Inc     RaumNummer      ; RaumNummer fuer AnZeige erhoehen
00D6 A00101                     Mov     AL,RaumNummer   ;
00D9 98                         CBW                     ; Auf Wort erweitern
00DA BFB201                     Mov     DI,Offset ASRaumNr  ; Offset fuer Anzeige
00DD B102                       Mov     CL,2            ; Anzahl Stellen
00DF C606000101                 Mov     Byte Ptr Blank,True ;
00E4 E8DF03     04C6            Call    Int_to_Ascii    ;
00E7 A10C01                     Mov     AX,Droids       ; Anzahl Droids
00EA 03060A01                   Add     AX,DroidInc     ; Plus Anzahl fuer naechsten Raum
00EE A30C01                     Mov     Droids,AX       ; Anzahl fuer naechsten Raum merken
00F1 A30E01                     Mov     Droid_Count,AX  ; Fuer laufenden Raum
00F4 BFD401                     Mov     DI,Offset ASDroids  ;
00F7 B104                       Mov     CL,4            ;
00F9 C606000101                 Mov     Byte Ptr Blank,True ;
00FE E8C503     04C6            Call    Int_to_Ascii    ;
0101 8B0E0C01                   Mov     CX,Droids       ;
0105 E83503     043D            Call    Gen_Droids      ; CX Androiden generieren
0108 E85103     045C            Call    Gen_Human       ; Spieler generieren
010B C606020100                 Mov     Byte Ptr Screw_D,False  ;
```

```
0110 C606030100                      Mov     Byte Ptr Last_Stand,False ;
0115 C70612010000                    Mov     Word Ptr PunkteR,0      ; Punkte in dieser Runde
011B F606030101          Get_Zug:    Test    Byte Ptr Last_Stand,True;
0120 7522        0144                Jnz     Move_Droids       ;
0122 A10601                          Mov     AX,Player         ;
0125 E86603      048E                Call    Pos_Curser        ;
0128 E80F04      053A                Call    Get_Char          ;
012B 3C39                            Cmp     AL,'9'            ; Zahl ?
012D 7737        0166                Ja      No_Number         ; Nein ==>
012F 2C31                            Sub     AL,'1'            ; in Integer wandeln
0131 7C55        0188                Jl      Ignore            ; falsches Zeichen Beep und Ignorieren
0133 98                              Cbw                       ;
0134 D1E0                            Shl     AX,1              ;
0136 8BD8                            Mov     BX,AX             ;
0138 8B9F1C01                        Mov     BX,Word Ptr Zug_Tafel[BX]    ;
013C 83FB00                          Cmp     BX,0              ;
013F 7403        0144                Jz      Move_Droids       ;
0141 E83802      037C                Call    Zug_Spieler       ;
0144 E84D01      0294 Move_Droids:   Call    Zug_Droids        ;
0147 F6062E0101                      Test    Byte Ptr Dead,True  ;
014C 7554        01A2                Jnz     Show_Top_Ten      ;
014E 833E0E0100                      Cmp     Word Ptr Droid_Count,0
0153 75C6        011B                Jnz     Get_Zug           ;
0155 E91CFF      0074                Jmp     Spiel             ;
0158 E85D03      0488 Show_Inst:     Call    Inst              ; Anleitung ausgeben
015B BA6102            Redraw:       Mov     DX,Offset WScreen;
015E E8D403      0535                Call    Write_Str         ;
0161 EBB8        011B                Jmps    Get_Zug           ;
0163 E9AE03      0514 J_Ende:        Jmp     Ende              ;
0166 3C12              No_Number:    Cmp     AL,18             ;
0168 74F1        015B                Jz      Redraw            ;
016A 24DF                            And     AL,0DFH           ; Klein auf Grossbuchstaben
016C 3C59                            Cmp     AL,'Y'            ;
016E 74F3        0163                Jz      J_Ende            ;
0170 3C45                            Cmp     AL,'E'            ;
0172 742E        01A2                Jz      Show_Top_Ten      ;
0174 3C48                            Cmp     AL,'H'            ;
0176 74E0        0158                Jz      Show_Inst         ;
0178 3C52                            Cmp     AL,'R'            ;
017A 74DF        015B                Jz      Redraw            ;
017C 3C4C                            Cmp     AL,'L'            ;
017E 740E        018E                Jz      Move_L            ;
0180 3C54                            Cmp     AL,'T'            ;
0182 7412        0196                Jz      Call_Teleport     ;
0184 3C53                            Cmp     AL,'S'            ;
0186 7414        019C                Jz      Call_Screw        ;
0188 E89B03      0526 Ignore:        Call    Beep              ;
018B E98DFF      011B                Jmp     Get_Zug           ;
018E C606030101          Move_L:     Mov     Byte Ptr Last_STand,True
0193 E9AEFF      0144                Jmp     Move_Droids       ;
0196 E80C00      01A5 Call_Teleport: Call    Teleport          ;
0199 E9A8FF      0144                Jmp     Move_Droids       ;
019C E8A200      0241 Call_Screw:    Call    Screw             ;
019F E9A2FF      0144                Jmp     Move_Droids       ;
01A2 E96F03      0514 Show_Top_Ten:  Jmp     Ende              ;
01A5 E84D03      04F5 Teleport:      Call    Calc_Random       ;
01A8 E8CD02      0478                Call    Calc_S_Pos        ;
01AB 803F20                          Cmp     Byte Ptr [BX],' ';
01AE 75F5        01A5                Jnz     Teleport          ;
01B0 C60758                          Mov     Byte Ptr [BX],'X'; Neue Position
01B3 8BD0                            Mov     DX,AX             ;
01B5 A10601                          Mov     AX,Player         ;
01B8 A30801                          Mov     Old_Pla,AX        ;
01BB 89160601                        Mov     Player,DX         ; Aktuelle Position speichern
01BF E8B602      0478                Call    Calc_S_Pos        ;
01C2 C60720                          Mov     Byte Ptr [BX],' ';
01C5 C6061A0101                      Mov     Byte Ptr T_First,True
01CA 52                              Push    DX                ;
01CB 50                              Push    AX                ;
01CC E81300      01E2                Call    T_Next_Step       ;
01CF 58                              Pop     AX                ;
01D0 E8BB02      048E                Call    Pos_Curser        ;
01D3 B220                            Mov     DL,' '            ;
01D5 E85003      0528                Call    Write_Char        ;
01D8 5A                              Pop     DX                ;
01D9 C6061A0100                      Mov     Byte Ptr T_First,False
01DE E80100      01E2                Call    T_Next_Step       ;
01E1 C3                              Ret                       ;
01E2 52                T_Next_Step:  Push    DX                ;
01E3 C70616010101                    Mov     Word Ptr T_X_Dir,101H
01E9 2AD0                            Sub     DL,AL             ;
01EB 7707        01F4                Ja      T_Pos_Y           ;
01ED F6DA                            Neg     DL                ;
01EF C6061701FF                      Mov     Byte Ptr T_Y_Dir,-1
01F4 8ADA                T_Pos_Y:    Mov     BL,DL             ;
01F6 2AF4                            Sub     DH,AH             ;
01F8 7707        0201                Ja      T_Pos_X           ;
01FA F6DE                            Neg     DH                ;
01FC C6061601FF                      Mov     Byte Ptr T_X_Dir,-1
0201 8AFE                T_Pos_X:    Mov     BH,DH             ;
0203 8ACE                            Mov     CL,DH             ;
```

```
0205 32ED                          Xor     CH,CH           ;
0207 3ACA                          Cmp     CL,DL           ;
0209 7702          020D            Ja      T_NoY_Count     ;
020B 8ACA                          Mov     CL,DL           ;
                   T_NoY_Count:                            ;
020D 80FF00        T_Next:         Cmp     BH,0            ;
0210 7E06          0218            Jle     T_No_X_Move     ;
0212 FECF                          Dec     BH              ;
0214 02261601                      Add     AH,T_X_Dir      ;
0218 80FB00        T_No_X_Move:    Cmp     BL,0            ;
021B 7E06          0223            Jle     T_No_Y_Move     ;
021D FECB                          Dec     BL              ;
021F 02061701                      Add     AL,T_Y_Dir      ;
0223 E86802        048E T_No_Y_Move: Call  Pos_Curser      ;
0226 B258                          Mov     DL,'X'          ;
0228 F6061A0101                    Test    Byte Ptr T_First,True
022D 7507          0236            Jnz     T_Write_X       ;
022F 53                            Push    BX              ;
0230 E84502        0478            Call    Calc_S_Pos      ;
0233 8A17                          Mov     DL,Byte Ptr [BX];
0235 5B                            Pop     BX              ;
0236 E8EF02        0528 T_Write_X: Call    Write_Char      ;
0239 E2D2          020D            Loop    T_Next          ;
023B 5A                            Pop     DX              ;
023C 3BC2                          Cmp     AX,DX           ;
023E 75A2          01E2            Jnz     T_Next_Step     ;
0240 C3                            Ret                     ;
0241 A10601        Screw:          Mov     AX,Player       ;
0244 8BD0                          Mov     DX,AX           ;
0246 B90300                        Mov     CX,3            ;
0249 FEC8                          Dec     AL              ;
024B 3C01          S_X_Loop:       Cmp     AL,Nord         ;
024D 7C3B          028A            Jl      S_X_Inc         ;
024F 3C17                          Cmp     AL,Sued         ;
0251 7740          0293            Ja      Screw_Ende      ;
0253 8AE6                          Mov     AH,DH           ;
0255 FECC                          Dec     AH              ;
0257 51                            Push    CX              ;
0258 B103                          Mov     CL,3            ;
025A 80FC01        S_Y_Loop:       Cmp     AH,West         ;
025D 7C26          0285            Jl      S_Y_Inc         ;
025F 80FC50                        Cmp     AH,Ost          ;
0262 7721          0285            Ja      S_Y_Inc         ;
0264 3B060601                      Cmp     AX,Player       ;
0268 741B          0285            Jz      S_Y_Inc         ;
026A E80B02        0478            Call    Calc_S_Pos      ;
026D 803F41                        Cmp     Byte Ptr [BX],'A' ;
0270 7513          0285            Jnz     S_Y_Inc         ;
0272 C60720                        Mov     Byte Ptr [BX],' ' ;
0275 E81602        048E            Call    Pos_Curser      ;
0278 B220                          Mov     DL,' '          ;
027A E8AB02        0528            Call    Write_Char      ;
027D FF061201                      Inc     Punkte_R        ;
0281 FF0E0E01                      Dec     Droid_Count     ;
0285 FEC4          S_Y_Inc:        Inc     AH              ;
0287 E2D1          025A            Loop    S_Y_Loop        ;
0289 59                            Pop     CX              ;
028A FEC0          S_X_Inc:        Inc     AL              ;
028C E2BD          024B            Loop    S_X_Loop        ;
028E C606020101                    Mov     Byte Ptr Screw_D,True;
0293 C3            Screw_Ende:     Ret                     ;
0294 C70610010000  Zug_Droids:     Mov     Word Ptr Droid_Move,0
029A 8B360601                      Mov     SI,Player       ;   SI ist Player
029E 33FF                          Xor     DI,DI           ;   DI ist Distanz
02A0 47            Inc_Dist:       Inc     DI              ;;
02A1 83FF50                        Cmp     DI,Col_Size     ;;  Notbremse
02A4 77ED          0293            Ja      Screw_Ende      ;;  Hauptsache RET
02A6 C6061B0101                    Mov     Byte Ptr Direktion,1;;
02AB 8BC7                          Mov     AX,DI           ;;  Fuer Rechnung
02AD 8BDE                          Mov     BX,SI           ;;  Ausgangsposition
02AF 2AD8                          Sub     BL,AL           ;;
02B1 80FB01                        Cmp     BL,Nord         ;;  Nach norden vergleichen
02B4 7C03          02B9            Jl      Move_Ost        ;;
02B6 E83300        02EC            Call    Move_NS         ;;  Unterp fuer Nord Sued Richtung
02B9 8BDE          Move_Ost:       Mov     BX,SI           ;;
02BB 2AF8                          Sub     BH,AL           ;;
02BD 80FF50                        Cmp     BH,Ost          ;;  Nach norden vergleichen
02C0 7703          02C5            Ja      Move_Sued       ;;
02C2 E86F00        0334            Call    Move_WO         ;; Fuer West Ost Richtung
02C5 C6061B01FF    Move_Sued:      Mov     Byte Ptr Direktion,-1
02CA 8BDE                          Mov     BX,SI           ;;
02CC 02D8                          Add     BL,AL           ;;
02CE 80FB17                        Cmp     BL,Sued         ;;  Nach norden vergleichen
02D1 7703          02D6            Ja      Move_West       ;;
02D3 E81600        02EC            Call    Move_NS         ;;
02D6 8BDE          Move_West:      Mov     BX,SI           ;;
02D8 02F8                          Add     BH,AL           ;;
02DA 80FF01                        Cmp     BH,West         ;;  Nach norden vergleichen
02DD 7C03          02E2            Jl      No_More_Move    ;;
02DF E85200        0334            Call    Move_WO         ;;
02E2 A10E01        No_More_Move:   Mov     AX,Droid_Count  ;;
```

```
02E5 3B061001                       Cmp     AX,Droid_Move   ;;
02E9 75B5       02A0                Jnz     Inc_Dist        ;;
02EB C3                 Ende_Zug_D: Ret                     ;
02EC 57                 Move_NS:    Push    DI              ;;
02ED 56                             Push    SI              ;;
02EE 8AEF                           Mov     CH,BH           ;;  Ausgangs spalte
02F0 2AE8                           Sub     CH,AL           ;;  Untere Grenze
02F2 80FD01                         Cmp     CH,West         ;;  noch im Spielfeld ?
02F5 7302       02F9                Jae     In1_West        ;;
02F7 B501                           Mov     CH,West         ;;  Nein, Korregieren
02F9 8ACF               In1_West:   Mov     CL,BH           ;;  Ausgangsspalte
02FB 02C8                           Add     CL,AL           ;;  Obere Grenze
02FD 80F950                         Cmp     CL,Ost          ;;  Im Spielfeld ?
0300 7E02       0304                Jle     In1_Ost         ;;
0302 B150                           Mov     CL,Ost          ;;  Nein, korregieren
0304 8AC3               In1_Ost:    Mov     AL,BL           ;;
0306 8AE1               Move_N_Droids: Mov  AH,CL           ;;;
0308 8AF1                           Mov     DH,CL           ;;; AH AL = Neue Position
030A 53                             Push    BX              ;;;
030B E86A01     0478                Call    Calc_S_Pos      ;;;
030E 803F41                         Cmp     Byte Ptr [BX],'A';;;
0311 5B                             Pop     BX              ;;;
0312 7517       032B                Jnz     No_N_Droid      ;;;
0314 3AE7                           Cmp     AH,BH           ;;;
0316 7408       0320                Je      No_N_x_Korr     ;;;
0318 7704       031E                Ja      Dec_N_X         ;;;
031A FEC4                           Inc     AH              ;;;
031C EB02       0320                Jmps    No_N_X_korr     ;;;
031E FECC               Dec_N_X:    Dec     AH              ;;;
0320 8AD0               No_N_X_korr: Mov    DL,AL           ;;;
0322 02061B01                       Add     AL,Direktion    ;;;
0326 E8BA00     03E3                Call    Z_Move_Droids   ;;; Retten
0329 8AC3                           Mov     AL,BL           ;;;
032B 3ACD               No_N_Droid: Cmp     CL,CH           ;;;
032D E0D7       0306                Loopne  Move_N_Droids   ;;;
032F 5E                             Pop     SI              ;;;
0330 5F                             Pop     DI              ;;
0331 8BC7                           Mov     AX,DI           ;;  Fuer Rechnung
0333 C3                             Ret                     ;;

0334 57                 Move_WO:    Push    DI              ;;
0335 56                             Push    SI              ;;
0336 8AEB                           Mov     CH,BL           ;;  Ausgangs Zeile
0338 2AE8                           Sub     CH,AL           ;;  Untere Grenze
033A 80FD01                         Cmp     CH,Nord         ;;  noch im Spielfeld ?
033D 7302       0341                Jae     In1_Nord        ;;
033F B501                           Mov     CH,Nord         ;;  Nein, Korregieren
0341 8ACB               In1_Nord:   Mov     CL,BL           ;;  Ausgangsspalte
0343 02C8                           Add     CL,AL           ;;  Obere Grenze
0345 80F917                         Cmp     CL,Sued         ;;  Im Spielfeld ?
0348 7E02       034C                Jle     In1_Sued        ;;
034A B117                           Mov     CL,Sued         ;;  Nein, korregieren
034C 8AE7               In1_Sued:   Mov     AH,BH           ;;
034E 8AC1               Move_W_Droids: Mov  AL,CL           ;;;
0350 8AD1                           Mov     DL,CL           ;;; AH AL = Neue Position
0352 53                             Push    BX              ;;;
0353 E82201     0478                Call    Calc_S_Pos      ;;;
0356 803F41                         Cmp     Byte Ptr [BX],'A';;;
0359 5B                             Pop     BX              ;;;
035A 7517       0373                Jnz     No_W_Droid      ;;;
035C 3AC3                           Cmp     AL,BL           ;;;
035E 7408       0368                Je      No_W_Y_Korr     ;;;
0360 7704       0366                Ja      Dec_W_Y         ;;;
0362 FEC0                           Inc     AL              ;;;
0364 EB02       0368                Jmps    No_W_Y_korr     ;;;
0366 FEC8               Dec_W_Y:    Dec     AL              ;;;
0368 8AF4               No_W_Y_korr: Mov    DH,AH           ;;;
036A 02261B01                       Add     AH,Direktion    ;;;
036E E87200     03E3                Call    Z_Move_Droids   ;;; Retten
0371 8AE7                           Mov     AH,BH           ;;;
0373 3ACD               No_W_Droid: Cmp     CL,CH           ;;;
0375 E0D7       034E                Loopne  Move_W_Droids   ;;;
0377 5E                             Pop     SI              ;;;
0378 5F                             Pop     DI              ;;
0379 8BC7                           Mov     AX,DI           ;;  Fuer Rechnung
037B C3                             Ret                     ;;
                        ;   Eingabe in Zug_Spieler   BH = 0,1,-1 BL = 0,1,-1
                        ;                                 = Zugrichtung
037C A10601             Zug_Spieler: Mov    AX,Player       ;
037F 02C3                           Add     AL,BL           ;
0381 02E7                           Add     AH,BH           ;
0383 3C01                           Cmp     AL,Nord         ;
0385 7C58       03DF                Jl      Illegal         ;
0387 3C17                           Cmp     AL,Sued         ;
0389 7754       03DF                Ja      Illegal         ;
038B 80FC01                         Cmp     AH,West         ;
038E 7C4F       03DF                Jl      Illegal         ;
0390 80FC50                         Cmp     AH,Ost          ;
0393 774A       03DF                Ja      Illegal         ;
0395 8B1E0601                       Mov     BX,Player       ;
0399 891E0801                       Mov     Old_Pla,BX      ; Alte Position retten
```

```
039D A30601                         Mov     Player,AX       ;
03A0 50                             Push    AX              ; Retten
03A1 8BC3                           Mov     AX,BX           ;
03A3 E8E800       048E              Call    Pos_Curser      ;
03A6 B220                           Mov     DL,' '          ;
03A8 E87D01       0528              Call    Write_Char      ; Old Human killed
03AB 58                             Pop     AX              ; Player Position
03AC 8BD0                           Mov     DX,AX           ; Neue Pos berechnen
03AE E8C700       0478              Call    Calc_S_Pos      ;
03B1 53                             Push    BX              ; retten
03B2 8BC2                           Mov     AX,DX           ;
03B4 E8D700       048E              Call    Pos_Curser      ;
03B7 5B                             Pop     BX              ;
03B8 803F20                         Cmp     Byte Ptr [BX],' ';
03BB 740E         03CB              Je      Legal_Pos       ;
03BD 803F58                         Cmp     Byte Ptr [BX],'X';
03C0 7409         03CB              Je      Legal_Pos       ;
03C2 C6062E0101                     Mov     Byte Ptr Dead,True  ;
03C7 B22B                           Mov     DL,'+'          ;
03C9 EB10         03DB              Jmps    Show_Player     ;
03CB 53           Legal_Pos:        Push    BX              ; neue Position retten
03CC A10801                         Mov     AX,Old_Pla      ;
03CF E8A600       0478              Call    Calc_S_Pos      ;
03D2 C60720                         Mov     Byte Ptr [BX],' ';
03D5 5B                             Pop     BX              ;
03D6 C60758                         Mov     Byte Ptr [BX],'X';
03D9 B258                           Mov     DL,'X'          ;
03DB E84A01       0528 Show_Player: Call    Write_Char      ;
03DE C3                             Ret                     ;
03DF E84401       0526 Illegal:     Call    Beep            ;
03E2 C3                             Ret                     ;
                    ; AH AL = Neue Position
                    ; DH DL = Alte Position
03E3 51             Z_Move_Droids:  Push    CX              ;
03E4 53                             Push    BX              ;
03E5 50                             Push    AX              ; Retten
03E6 8BC2                           Mov     AX,DX           ;
03E8 E88D00       0478              Call    Calc_S_Pos      ;
03EB C60720                         Mov     Byte Ptr [BX],' '; Alte Position geloescht
03EE E89D00       048E              Call    Pos_Curser      ;
03F1 B220                           Mov     DL,' '          ;
03F3 E83201       0528              Call    Write_Char      ; Auf Bildschirm geloescht
03F6 58                             Pop     AX              ; neue Pos nach AX
03F7 E87E00       0478              Call    Calc_S_Pos      ;
03FA 803F20                         Cmp     Byte Ptr [BX],' ';
03FD B241                           Mov     DL,'A'          ;
03FF 7431         0432              Jz      Z_Move          ; Pos frei ==> moven
0401 803F2A                         Cmp     Byte Ptr [BX],'*'
0404 7421         0427              Jz      Dis_Dust        ;
0406 803F58                         Cmp     Byte Ptr [BX],'X';
0409 7509         0414              Jnz     Cmp_Dead        ;
040B B22B           Dis_Dead:       Mov     DL,'+'          ;
040D C6062E0101                     Mov     Byte Ptr Dead,True;
0412 EB15         0429              Jmps    Dec_Droids      ;
0414 803F2B         Cmp_Dead:       Cmp     Byte Ptr [BX],'+';
0417 74F2         040B              Jz      Dis_Dead        ;
0419 803F41                         Cmp     Byte Ptr [BX],'A';
041C 75F6         0414              Jnz     Cmp_Dead        ;
041E FF0E0E01                       Dec     Droid_Count     ;
0422 830612010A                     Add     Punkte_R,10     ;
0427 B22A           Dis_Dust:       Mov     DL,'*'          ;
0429 FF0E0E01       Dec_Droids:     Dec     Droid_Count     ;
042D 830612010A                     Add     Punkte_R,10     ; fuer 2 androiden
0432 8817           Z_Move:         Mov     Byte Ptr [BX],DL;
0434 E85700       048E              Call    Pos_Curser      ;
0437 E8EE00       0528              Call    Write_Char      ;
043A 5B                             Pop     BX              ;
043B 59                             Pop     CX              ;
043C C3                             Ret                     ;
                    ;
                    ;   Unterprogramm Gen_Droids
                    ;   generiert CX Androiden und Positioniert sie auf dem Bildschirm
                    ;   Eingangs Parameter : CX  = Anzahl der zu generierenden Androiden
                    ;
                    Gen_Droids:                             ;
043D 51             Loop_G_D:       Push    CX              ;
043E E8B400       04F5              Call    Calc_Random     ;
0441 50                             Push    AX              ;
0442 E83300       0478              CAll    Calc_S_Pos      ;
0445 803F20                         Cmp     Byte Ptr [BX],' ' ;
0448 58                             Pop     AX              ;
0449 59                             Pop     CX              ;
044A 75F1         043D              Jnz     Loop_G_D        ; Nochmal berechnen falls
044C 51                             Push    CX              ;
044D C60741                         Mov     Byte Ptr [BX],'A' ;
0450 E83B00       048E              Call    Pos_Curser      ; AX enthaelt aktuelle Position
0453 B241                           Mov     DL,'A'          ;
0455 E8D000       0528              Call    Write_Char      ;
0458 59                             Pop     CX              ;
0459 E2E2         043D              Loop    Loop_G_D        ;
045B C3                             Ret                     ;
```

```
045C E89600       04F5 Gen_Human:      Call    Calc_Random     ;
045F 50                                Push    AX              ;
0460 E81500       0478                 Call    Calc_S_Pos      ;
0463 803F20                            Cmp     Byte Ptr [BX],' ' ;
0466 58                                Pop     AX              ;
0467 75F3         045C                 Jnz     Gen_Human       ; Nochmal berechnen falls
0469 A30601                            Mov     Player,AX       ;
046C C60758                            Mov     Byte Ptr [BX],'X' ;
046F E81C00       048E                 Call    Pos_Curser      ;
0472 B258                              Mov     DL,'X'          ;
0474 E8B100       0528                 Call    Write_Char      ;
0477 C3                                Ret                     ;
                        ; Berechnen der Position in Screen
                        ; AX enthaelt Koordinaten
                        ; Ergebnis ist in BX Offset zu DS:
0478 50                 Calc_S_Pos:    Push    AX              ; Retten
0479 8BD8                              Mov     BX,AX           ;
047B B450                              Mov     AH,Col_Size     ;
047D FEC8                              Dec     AL              ;
047F F6E4                              Mul     AH              ;
0481 93                                Xchg    AX,BX           ;
0482 8AC4                              Mov     AL,AH           ;
0484 98                                Cbw                     ;
0485 48                                Dec     AX              ;
0486 03D8                              Add     BX,AX           ;
0488 81C36502                          Add     BX,Offset Screen;
048C 58                                Pop     AX              ;
048D C3                                Ret                     ;
                        ; Curser wird auf die in AX angegebene Position gesetzt
                        ; AX BX CX DX werden gesichert
048E 50                 Pos_Curser:    Push    AX              ;
048F 53                                Push    BX              ;
0490 51                                Push    CX              ;
0491 52                                Push    DX              ;
0492 50                                Push    AX              ;
0493 C606000100                        Mov     Byte Ptr Blank,False;
0498 BF9809                            Mov     DI,Offset C_Pos_Z;
049B B102                              Mov     CL,2            ;
049D 98                                Cbw                     ;
049E E82500       04C6                 Call    Int_to_Ascii    ;
04A1 58                                Pop     AX              ;
04A2 8AC4                              Mov     AL,AH           ;
04A4 BF9B09                            Mov     DI,Offset C_Pos_S;
04A7 B102                              Mov     CL,2            ;
04A9 98                                Cbw                     ;
04AA E81900       04C6                 Call    Int_to_Ascii    ;
04AD BA9609                            Mov     DX,Offset C_Pos ;
04B0 E88200       0535                 Call    Write_Str       ;
04B3 5A                                Pop     DX              ;
04B4 59                                Pop     CX              ;
04B5 5B                                Pop     BX              ;
04B6 58                                Pop     AX              ;
04B7 C3                                Ret                     ;
04B8 BA9F09             Inst:          Mov     DX,Offset InStr ; Anleitung ausgeben
04BB E87700       0535                 Call    WriteStr        ;
04BE E87900       053A Again:          Call    GetChar         ; Auf Zeichen warten
04C1 3C0D                              Cmp     AL,CR           ;
04C3 75F9         04BE                 Jnz     again           ;
04C5 C3                                Ret                     ;
                        ;
                        ;  UnterProgramm Int_to_Ascii
                        ;  Umwandeln einer Integer Zahl in einen Ascii String.
                        ;  Eingabe Parameter :
                        ;    AX = Integer Zahl
                        ;    CL = Laenge des Strings ( sollte ausreichend lang sein )
                        ;    ES:DI = Beginn des Strings
                        ;  Ausgabe : String
                        ;               fuehrende Nullen werden Blank entsprechend unterdrueckt
                        ;               Blank = True Unterdreucken
                        ;
                        ;    AX,CX  haben  danach undefinierte Werte
04C6 53                 Int_to_Ascii:  Push    BX              ;  Retten der Register
04C7 52                                Push    DX              ;
04C8 56                                Push    SI              ;
04C9 BB0A00                            Mov     BX,10           ;  durch 10 wird geteilt
04CC 32ED                              Xor     CH,CH           ;  Loop Befehl braucht CX
04CE 8BF1                              Mov     SI,CX           ;  Retten fuer 2. Schleife
04D0 99                 Div_Loop:      CWD                     ;;
04D1 F7F3                              Div     BX              ;;
04D3 52                                Push    DX              ;; Rest der Division merken
04D4 E2FA         04D0                 Loop    Div_Loop        ;; CX mal
04D6 8BCE                              Mov     CX,SI           ;  alter CX wert
04D8 8A1E0001                          Mov     BL,Blank        ;
04DC 58                 Write_Loop:    Pop     AX              ;; werte vom Stapel holen
04DD 3C00                              Cmp     AL,0            ;; = 0 ?
04DF 7509         04EA                 Jne     No_Blank        ;; Nein ==> kein Blank
04E1 F6C301                            Test    BL,True         ;; Blank gesetzt ?
04E4 7404         04EA                 Jz      No_Blank        ;; Nein 0 anzeigen
04E6 B020                              Mov     AL,20H          ;; Blank anzeigen
04E8 EB04         04EE                 Jmps    Stosb_AL        ;;
04EA 0430               No_Blank:      Add     AL,30H          ;; aus Zahl Ascii Zeichen machen
```

```
04EC B300                       Mov     BL,False        ;; Blank ist jetzt false
04EE AA             Stosb_AL:   StosB                   ;; Zahl speichern
04EF E2EB     04DC  W_Write_Loop: Loop    Write_Loop      ;  naechte Zahl
04F1 5E                         Pop     SI              ;  zurueckladen der gesicherten Register
04F2 5A                         Pop     DX              ;
04F3 5B                         Pop     BX              ;
04F4 C3                         Ret                     ;
                    ; Ende Int_to_Ascii
                    ; berechnen von Zufalls koordinaten
                    ; Zeilenwert  in AL   0 < AL < Line_Size
                    ; Spaltenwert in AH   0 < AH < Col_Size
04F5 E85000   0548  Calc_Random:    Call    Get_Random      ;
04F8 B350                       Mov     BL,Col_Size     ; Screen Format nach BX
04FA 32FF                       Xor     BH,BH           ;
04FC 33D2                       Xor     DX,DX           ;
04FE F7F3                       Div     BX              ; AH = AX Mod BH
0500 42                         Inc     DX              ;
0501 52                         Push    DX              ; Retten
0502 E84300   0548              Call    Get_Random      ;
0505 B317                       Mov     BL,Line_Size    ;
0507 32FF                       Xor     BH,BH           ;
0509 33D2                       Xor     DX,DX           ;
050B F7F3                       Div     BX              ; AH = AX Mod BL
050D 42                         Inc     DX              ;
050E 58                         Pop     AX              ; Geretten Wert zurueck
050F 8AE2                       Mov     AH,DL           ;
0511 86E0                       Xchg    AH,AL           ; AH = Spaltenwert
                                                        ; AL = Zeilenwert
0513 C3                         Ret                     ;
0514 BA3401         Ende:       Mov     DX,Offset High_Light;
0517 E81B00   0535              Call    Write_Str       ;
051A BA2F01                     Mov     DX,Offset CLrScr;
051D E81500   0535              Call    Write_Str       ;
0520 B100                       Mov     CL,0            ; Beenden des Programmes
0522 CDE0                       Int     224             ;
0524 EBEE     0514              Jmps    Ende            ;
0526 B207           Beep:       Mov     DL,7            ;
                    ; DL enthaelt auszugebendes Zeichen
                    ; Alle Regs werden gerettet
0528 50             Write_Char: Push    AX              ;
0529 53                         Push    BX              ;
052A 51                         Push    CX              ;
052B 52                         Push    DX              ;
052C B102                       Mov     CL,2            ; Aufruf C_Write
052E CDE0                       Int     224             ;
0530 5A                         Pop     DX              ;
0531 59                         Pop     CX              ;
0532 5B                         Pop     BX              ;
0533 58                         Pop     AX              ;
0534 C3                         Ret                     ;
                    ; DS:DX Anfang des auszugebenden Strings
                    ; String wird durch  '$' beendet
0535 B109           Write_Str:  Mov     CL,9            ; Aufruf C_WRITESTR
0537 CDE0                       Int     224             ;
0539 C3                         Ret                     ;
053A B106           GetChar:    Mov     CL,6            ; Aufruf C_RawIO
053C B2FD                       Mov     DL,0FDH         ;
053E CDE0                       Int     224             ;
0540 3C03                       Cmp     AL,3            ;
0542 7503     0547              Jnz     ReturnGC        ;
0544 E9CDFF   0514              Jmp     Ende            ;
0547 C3             ReturnGC:   Ret                     ;
0548 E81600   0561  GET_Random: Call    Get_Time        ;
054B F7260401                   Mul     Word Ptr Old_Random ;
054F 33060401                   Xor     AX,Old_Random   ;
0553 7404     0559              Jz      Init_Random     ;
0555 A30401                     Mov     Old_Random,AX   ;
0558 C3                         Ret                     ;
0559 C70604015555   Init_Random: Mov    Old_Random,5555H;
055F EBE7     0548              Jmps    Get_Random      ;
0561 B169           Get_Time:   Mov     CL,T_Get        ;
0563 BA2A0C                     Mov     DX,Offset Dummy ;
0566 CDE0                       Int     224             ;
0568 C3                         Ret                     ;
                            Dseg
  000D              CR            EQU   0DH             ;
  000A              LF            EQU   0AH             ;
  001B              ESC           EQU   27              ;
  0001              West          EQU   1               ; Feldgrenzen
  0050              Ost           EQU   80              ;
  0017              Sued          EQU   23              ;
  0001              Nord          EQU   1               ;
  0001              True          EQU   1               ;
  0000              False         EQU   0               ;
  0069              T_Get         EQU   105             ; System Aufruf Get_Time
                                                        ; dabei ist AL = Sekunden
  0017              Line_Size     EQU   Sued            ; Anzahl Zeilen
  0050              Col_Size      EQU   Ost             ; Anzahl Spalten
  5017              Screen_Size   EQU   Col_Size*256+Line_Size  ; BH = 80 , BL = 24
  0730              ScreenBytes   EQU   Col_Size*Line_Size      ;
                            Org   100H                  ;
```

```
0100 00                Blank           DB      0               ;
0101 00                RaumNummer      DB      0               ; Anfangs nummer
0102 00                Screw_D         DB      0               ;
0103 00                Last_Stand      DB      0               ;
0104                   Old_Random      RW      1               ;
0106 0000              Player          DW      0               ;
0108 0000              Old_Pla         DW      0               ; Low = Zeile , High = Spalte
010A 0500              DroidInc        DW      5               ; Anzahl der Droidenerhoehung pro Raum
010C 0000              Droids          DW      0               ; Aktuelle Anzahl
010E 0000              Droid_count     DW      0               ;
0110 0000              Droid_Move      DW      0               ;
0112 0000              Punkte_R        DW      0               ; Punkte im Raum
0114 0000              Punkte_G        DW      0               ; Punkte ingesammt
0116 00                T_X_Dir         DB      0               ;
0117 00                T_Y_Dir         DB      0               ;
0118 00                T_X_Count       DB      0               ;
0119 00                T_Y_Count       DB      0               ;
011A 00                T_First         DB      0               ;
011B 00                Direktion       DB      0               ; Even Adress
011C 01FF              Zug_Tafel       DB       1,-1           ; 1
011E 0100                              DB       1, 0           ; 2
0120 0101                              DB       1, 1           ; 3
0122 00FF                              DB       0,-1           ; 4
0124 0000                              DB       0, 0           ; 5
0126 0001                              DB       0, 1           ; 6
0128 FFFF                              DB      -1,-1           ; 7
012A FF00                              DB      -1, 0           ; 8
012C FF01                              DB      -1, 1           ; 9
012E 00                Dead            DB      0               ;
012F 1B5B324A24        ClrScr          DB      ESC,'[2J$'      ;
0134 1B5B306D24        High_Light      DB      ESC,'[0m$'      ;
0139 1B237124          Uhraus          DB      ESC,'#q$'       ;
013D 1B237024          Uhran           DB      ESC,'#p$'       ;
0141 1B5B324A1B5B      Wellcome        DB      ESC,'[2J',ESC,'[7m Spielanleitung erwuenscht ? [n]  $'
     376D20537069
     656C616E6C65
     6974756E6720
     65727775656E
     73636874203F
     205B6E5D2020
     24
                       ;               DB      ESC,'[5m','ABCDEFabcdef012345'
                       ;               DB      ESC,'[01m!"@%123456789abcdefghiABCDEFGH'
016C 202020202020      RaumRahmen      DB      '                                                ',CR,LF
     202020202020
     202020202020
     202020202020
     202020202020
     202020202020
     2020200D0A
0195 202052617756D                     DB      '  Raum Nummer :              '
     204E756D6D65
     72203A202020
     202020202020
     2020202020
01B2 000020202000D     ASRaumNr        DB      0,0,'   ',CR,LF
     0A
01B9 2020416E6472                      DB      '  Androiden im Raum :      '
     6F6964656E20
     696D20526175
     6D203A202020
     202020
01D4 000000002020      ASDroids        DB      0,0,0,0,'   ',CR,LF
     200D0A
01DD 202050756E6B                      DB      '  Punkte in diesem Raum :  '
     746520696E20
     646965736560
     205261756D20
     3A2020
01F8 000000002020      ASPunkteR       DB      0,0,0,0,'   ',CR,LF
     200D0A
0201 202050756E6B                      DB      '  Punkte ingesammt :       '
     746520696E67
     6573616D6D74
     203A20202020
     202020
021C 000000002020      ASPunkteG       DB      0,0,0,0,'   ',CR,LF
     200D0A
0225 202020202020                      DB      '                                            ',CR,LF
     202020202020
     202020202020
     202020202020
     202020202020
     202020202020
     2020200D0A
024E 205265747572                      DB      ' Return druecken  $'
     6E2064727565
     636B656E2020
     24
0261 1B5B324A          WScreen         DB      ESC,'[2J'
0265                   Screen          RB      Line_Size*ColSize  ; Sued+1*(Ost+1)
```

```
0995 24                                DB      '$'
0996 1B5B              C_Pos           DB      ESC,'['
0998 30303B            C_Pos_Z         DB      '00;'
099B 30304824          C_Pos_S         DB      '00H$'
099F 1B5B324A0D0A      Instr           DB      ESC,'[2J',CR,LF
09A5 536965207369                      DB      'Sie sind auf einer Raumstation gelandet, die '
     6E6420617566
     2065696E6572
     205261756D73
     746174696F6E
     2067656C616E
     6465742C2064
     696520
09D2 766F6E20416E      ON              DB      'von Androiden bewohnt wird.',CR,LF,CR,LF
     64726F696465
     6E206265776F
     686E74207769
     72642E0D0A0D
     0A
09F1 4A6564657220                      DB      'Jeder Raum der Station ist 80 * 23 gross.',CR,LF
     5261756D2064
     657220537461
     74696F6E2069
     737420383020
     2A2032332067
     726F73732E0D
     0A
0A1C 4A6564657220                      DB      'Jeder Raum enthaelt mehr Androiden.',CR,LF,CR,LF
     5261756D2065
     6E746861656C
     74206D656872
     20416E64726F
     6964656E2E0D
     0A0D0A
0A43 446965736520                      DB      'Diese Androiden haben nur eine Auftrag : Jeden '
     416E64726F69
     64656E206861
     62656E206E75
     722065696E65
     204175667472
     6167203A204A
     6564656E20
0A72 45696E647269                      DB      'Eindringling zu zerstoeren.',CR,LF
     6E676C696E67
     207A75207A65
     7273746F6572
     656E2E0D0A
0A8F 536965206861                      DB      'Sie haben folgende Kommandos :',CR,LF
     62656E20666F
     6C67656E6465
     204B6F6D6D61
     6E646F73203A
     0D0A
0AAF 20312D392020                      DB      ' 1-9     Einen Schritt in die ensprechende Richtung',CR,LF
     20202045696E
     656E20536368
     726974742069
     6E2064696520
     656E73707265
     6368656E6465
     205269636874
     756E670D0A
0AE4 205420202020                      DB      ' T       Teleportieren',CR,LF
     20202054656C
     65706F727469
     6572656E0D0A
0AFC 205220202020                      DB      ' R       Bildschirm neu zeichnen',CR,LF
     20202042696C
     647363686972
     6D206E657520
     7A656963686E
     656E0D0A
0B1E 205320202020                      DB      ' S       Sonic Zapper',CR,LF
     202020536F6E
     6963205A6170
     7065720D0A
0B35 204C20202020                      DB      ' L       laufen lassen ( nur fuer diesen Raum )',CR,LF
     2020206C6175
     66656E206C61
     7373656E2028
     206E75722066
     756572206469
     6573656E2052
     61756D20290D
     0A
0B66 204374726C20                      DB      ' Ctrl R  Bildschirm neu zeichnen',CR,LF
     52202042696C
     647363686972
     6D206E657520
     7A656963686E
     656E0D0A
```

```
0B88 203130202020                      DB      ' 10      Punkte fuer jeden Androiden der explodiert',CR,LF
     20202050756E
     6B7465206675
     6572206A6564
     656E20416E64
     726F6964656E
     206465722065
     78706C6F6469
     6572740D0A
0BBD 202031202020                      DB      '  1      Punkt fuer jeden Androiden der mit dem Sonic Zapper',CR,LF
     20202050756E
     6B7420667565
     72206A656465
     6E20416E6472
     6F6964656E20
     646572206D69
     742064656D20
     536F6E696320
     5A6170706572
     0D0A
0BFB 202020202020                      DB      '         vernichtet wird.',CR,LF
     202020766572
     6E6963687465
     742077697264
     2E0D0A
0C16 202052657475                      DB      '  Return druecken  $'
     726E20647275
     65636B656E20
     2024
0C2A 00000000          Dummy           DW      0,0             ; Dummy Variable fuer TOD Aufruf
```

END OF ASSEMBLY. NUMBER OF ERRORS: 0. USE FACTOR: 5%