

ASSEMBLERPROGRAMM SIDE-SCROLL.ASM:

```
.org 0x7200          // Programm in Grafikspeicher laden
.run 0x7200          // und dort auch starten

.def screen: 0x7000 // Bildschirmspeicheradresse

LD HL, screen:      // HL = Bildschirm Startadresse
LD DE, screen:      // DE = Bildschirm Startadresse + 1
INC DE
LD BC, 0x01ff       // BC = Zähler

loop:               // Schleife 1

LD A, (DE)          // Zeichen aus DE laden
LD (HL), A           // und nach HL schreiben
INC HL              // Adresspointer erhöhen
INC DE
DEC BC              // Zähler - 1
LD A, B              // Zähler = 0 ?
OR C
JR NZ, loop:        // nein, weiter

LD HL, screen:      // HL = Bildschirm Startadresse
LD DE, 0x0020        // DE = Zeilenlänge
ADD HL, DE           // HL + Zeilenlänge - 1 = Ende der 1. Zeile
DEC HL
LD B, 0x10           // B = Zähler für 16 Zeilen
LD A, 0x20           // A = Leerzeichen

loop2:              // Schleife

LD (HL), A           // Leerzeichen schreiben
ADD HL, DE           // nächste Zeile
DEC B               // Zähler - 1
JR NZ, loop2:       // weiter, solange Zähler != 0
RET                 // Ende
```

KODIERUNG IN Z80-MASCHINENODE:

HEX	DEZIMAL	ASSEMBLER	
21 00 70	33, 0, 112	LD HL,0x7000	HL = Bildschirm Startadresse
11 00 70	17, 0, 112	LD DE,0x7000	DE = Bildschirm Startadresse + 1
13	19	INC DE	
01 ff 01	1, 255, 1	LD BC,0x01FF	BC = Zähler
1a	26	LD A,(DE)	Schleife 1; Zeichen aus DE laden
77	119	LD (HL),A	und nach HL schreiben
23	35	INC HL	Adresspointer erhöhen
13	19	INC DE	
0b	11	DEC BC	Zähler - 1
78	120	LD A,B	Zähler = 0 ?
b1	177	OR C	
20 f7	32, 247	JR NZ, -9	nein, weiter
21 00 70	33, 0, 112	LD HL,0x7000	HL = Bildschirm Startadresse
11 20 00	17, 32, 00	LD DE,0x0020	DE = Zeilenlänge
19	25	ADD HL,DE	HL + Zeilenlänge - 1 = Ende der 1. Zeile
2b	43	DEC HL	
06 10	6, 16	LD B,0x10	B = Zähler für 16 Zeilen
3e 20	62, 32	LD A, 0x20	A = Leerzeichen
77	119	LD (HL), A	Schleife 2; Leerzeichen schreiben
19	25	ADD HL,DE	nächste Zeile
05	5	DEC B	Zähler - 1
20 fb	32, 251	JR NZ, -4	weiter, solange Zähler!=0
c9	201	RET	Ende

BASIC - PROGRAMM:

```
10 REM *** SCROLL-LEFT-DEMO ***
20 CLS:COLOR,0
30 PRINT"***** SIDE SCROLLING DEMO *****"
40 PRINT@160,"      <W> UP"
50 PRINT"      <S> DOWN"
50 PRINT"      <X> EXIT"
40 PRINT@320,"  !! TRY TO AVOID STARS !! "
60 PRINT@480,"          LOADING ASSEMBLER ...";
80 GOSUB 900:SOUND 0,5
90 CLS:COLOR,1:P=165:Q=165
100 REM *** GAME LOOP ***
110 PRINT@Q,"  ";
120 X=USR(X)
125 IF PEEK(28672+P+3)<>32 THEN PRINT@P-2,"%=#$!#";:GOTO140
130 PRINT@P,">=- ";
140 POKE 28672+31+32*(RND(16)-1), 42
150 Q=P
170 B$=INKEY$
180 IF P>=32 AND B$="W" THEN P=P-32
190 IF P<480 AND B$="S" THEN P=P+32
200 IF B$<>"X" THEN 100
210 CLS:COLOR,0
220 END
900 REM *** ASM-PROGRAMM-INSTALLIEREN ***
910 A = 28672 + 512
920 A1 = INT(A / 256): A0 = A - 256 * A1
930 POKE 30862, A0:POKE 30863, A1
940 READ B:REM *** BYTE-LESEN
950 IF B > 255 THEN RETURN:REM *** FERTIG?
960 POKE A, B:REM *** BYTE-SCHREIBEN
970 A=A+1
980 GOTO 940
1000 DATA 33, 0, 112, 17, 0, 112, 19, 1
1010 DATA 255, 1, 26, 119, 35, 19, 11, 120
1020 DATA 177, 32, 247, 33, 0, 112, 17, 32
1030 DATA 00, 25, 43, 6, 16, 62, 32, 119
1040 DATA 25, 5, 32, 251, 201
1020 DATA 999
READY
```

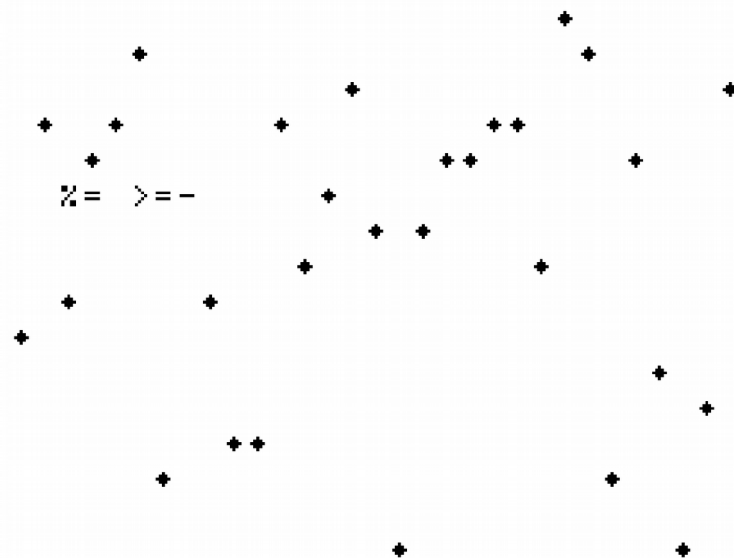
RUN ■

***** SIDE SCROLLING DEMO *****

<W> UP
<S> DOWN
<X> EXIT

!! TRY TO AVOID STARS !!

LOADING ASSEMBLER ...



⌘= >=-

The image shows a side-scrolling demo screen. It features a black background with numerous small white stars scattered across the field. In the lower-left quadrant, there is a small text prompt consisting of a symbol followed by an equals sign and a greater-than sign followed by a minus sign. The stars are distributed in a way that suggests a star field in a game environment.