;Definition von Systemadressen

SYSTEM: EQUITE : 00005H

;Adresse für Aufruf von ;Systemfunktionen



:Warmstart

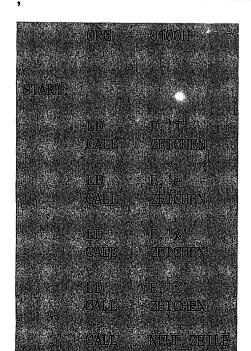
; Ausgabe auf CON: (=Konsole)

;Definition von ASCII-Konstanten



;Cursor an Zeilenanfang ;(Carriage Return) ;Cursor in neue Zeile ;(Line Feed)

;Hauptprogramm ZEICHENAUSGABE



;Das Programm beginnt bei ;0100H

;Es werden die Zeichen ;T, e, x und t geschrieben

;Der Cursor wird in eine

.

;neue Zeile gesetzt ;Die Warmstartfunktion ;wird aufgerufen ;Unterprogramm ZEICHEN - schreibt das Zeichen im E-Register auf den Bildschirm :Der Funktionskode wird ins ;C-Register geladen ;und das Betriebssystem ;aufgerufen ;Rücksprung \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* ;Unterprogramm NEUE ZEILE - fängt auf dem Bildschirm eine neue Zeile an ;Der Cursor wird an den ;Anfang der Zeile... ;... und dann eine Zeile ;tiefer gesetzt ;Rücksprung :Programmende

> ;Der Programmzähler wird ;auf den Programmanfang

;gesetzt

# TITLE TEXTAUSGABE - Programm 2

```
******************
;Dieses Programm gibt ASCII-Zeichen und im Speicher abgelegte
;Strings auf dem Bildschirm aus
und kehrt dann ins Betriebssystem zurück
***************
;Definition von Systemadressen
**************
SYSTEM EQU
           00005H
                      ;Adresse für Aufruf von
                      :Systemfunktionen
*****************
;Definition von Systemfunktionen
******************
WSTARTF EQU
           0
                      ;Warmstart
CONAUSF EQU
           2
                      ; Ausgabe auf CON: (=Konsole)
                      ; Ausgabe von Strings auf CON:
***********************
;Definition von ASCII-Konstanten
*******************
CR
     EQU
           ODH
                      ;Cursor an Zeilenanfang
                      (Carriage Return)
LF
     EQU
           OAH
                      ;Cursor in neue Zeile
                      ;(Line Feed)
                      ;Markierung für Stringende
;Hauptprogramm TEXTAUSGABE
ORG
           100H
                      ;Das Programm beginnt bei
                      ;0100H
START:
          E, 'T'
     LD
                      ;Es werden die Zeichen
     CALL
           ZEICHEN
                      ;T, e. x und t geschrieben
          E, 'e'
     LD,
     CALL
           ZEICHEN
     LD
          E, 'x'
     CALL
           ZEICHEN
```

```
LD
             E, 't'
       CALL
              ZE ICHEN
      LD
             DE, STRING1
                            ;Der unten definierte String
       CALL
             STRING
                            ;wird ausgegeben
       CALL
             NEUE ZEILE
                            ;Der Cursor wird in eine
                            ;neue Zeile gesetzt
             C,WSTARTF
      LD
                           ;Die Warmstartfunktion
       CALL
             SYSTEM
                            ;wird aufgerufen
                            ;eine Leerzeile
                            :Stringende
;Unterprogramm ZEICHEN
 - schreibt das Zeichen im E-Register auf den Bildschirm
******************
ZEICHEN:
      LD
             C, CONAUSF
                           ;Der Funktionskode wird ins
                           ;C-Register geladen
      CALL
             SYSTEM
                           ;und das Betriebssystem
                           ;aufgerufen
      RET
                            ;Rücksprung
 *******************
;Unterprogramm NEUE ZEILE
 - fängt auf dem Bildschirm eine neue Zeile an
******************
NEUE ZEILE:
      LD
             E,CR
                           ;Der Cursor wird an den
      CALL
             ZEICHEN
                           ;Anfang der Zeile...
      LD
             E,LF
                           ;... und dann eine Zeile
      CALL
             ZEICHEN
                           ;tiefer gesetzt
      RET
                           ;Rücksprung
 ********************
;Unterprogramm STRING
 - schreibt den String auf den der Pointer DE zeigt
```

```
STRING:
```

:Der Funktionskode wird ins LD C, STRAUSF ;C-Register geladen ;und das Betriebssystem CALL SYSTEM ;aufgerufen ; Rücksprung RET . \* ;Programmende \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* ;Der Programmzähler wird **END START** ;auf den Programmanfang

;gesetzt

# TITLE TEXTSPEICHER - Programm 3

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;Dieses Programm gibt den Inhalt des TLC90 Textspeichers :auf dem Bildschirm aus und kehrt dann ins Betriebssystem zurück \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* ;Definition von Systemadressen \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* SYSTEM EQU 00005H ;Adresse für Aufruf von ;Systemfunktionen \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* ;Definition von Systemfunktionen \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* WSTÁRTF EQU 0 :Warmstart CONAUSF EQU ; Ausgabe auf CON: (=Konsole) STRAUSF EQU 9 ; Ausgabe von Strings auf CON: TXTBEGF EQU ;Startadresse des Text-247 ;speichers abfragen

;Defini	tion von	ASCII-Konstante	**************************************				
CR	EQU	ODH	;Cursor an Zeilenanfang				
LF	EQU	ОАН	;(Carriage Return) ;Cursor in neue Zeile ;(Line Feed)				
EOF	EQU	1AH	;(End Of File)				
STOP	EQU	'\$'	;Markierung für Stringende				
; ************************************							
	ORG	100Н	;Das Programm beginnt bei ;0100H				
START:							
	LD CALL	DE, MELDUNG STRING	;Anfangsmeldung				
	CALL	NEUE_ZEILE	;Der Cursor wird in eine ;neue Zeile gesetzt				
	LD CALL	C,TXTBEGF SYSTEM	;Der Anfang des Textspeichers ;wird ins HL-Register geholt				
SCHLEIFE:							
	LD INC	A,(HL) HL	;Ein Zeichen ins A-Register ;und Pointer HL erhöhen				
	CP JR	EOF Z,ENDE	;Ist es END OF FILE ? ; Ja: nach ENDE springen				
	LD CALL	E,A ZEICHEN	; Nein: ins E-Register laden, ; ausgeben und				
	JR	SCHLEIFE	; nächstes Zeichen holen				
ENDE:							
	CALL	NEUE_ZEILE					
	LD CALL	C,WSTARTF SYSTEM	;Die Warmstartfunktion ;wird aufgerufen				

```
MELDUNG:
      DB
            CR, LF, 'Inhalt des Textspeichers: ', STOP
*****************
;Unterprogramm ZEICHEN
 - schreibt das Zeichen im E-Register auf den Bildschirm
ZEICHEN:
             C, CONAUSF
                          :Der Funktionskode wird ins
      LD
                          ;C-Register geladen
                          und das Betriebssystem
      CALL
             SYSTEM
                          ;aufgerufen
      RET
                          ;Rücksprung
******************
;Unterprogramm NEUE ZEILE
 - fängt auf dem Bildschirm eine neue Zeile an
 *******************
NEUE ZEILE:
      LD
             E,CR
                          :Der Cursor wird an den
      CALL
             ZEICHEN
                          :Anfang der Zeile...
      LD
             E,LF
                          ;... und dann eine Zeile
             ZEICHEN
      CALL
                          ;tiefer gesetzt
      RET
                          ;Rücksprung
;Unterprogramm STRING
 - schreibt den String auf den der Pointer DE zeigt
          *************
STRING:
      LD
             C, STRAUSF
                          ;Der Funktionskode wird ins
                          ;C-Register geladen
                          ;und das Betriebssystem
      CALL
             SYSTEM
                          ;aufgerufen
      RET
                          ;Rücksprung
******************
;Programmende
 *******************
                          ;Der Programmzähler wird
      END
             START
                          ;auf den Programmanfang
                          ;gesetzt
```

TITLE TEXTSPEICHER - Verbessertes Programm 3 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* ;Dieses Programm gibt den Inhalt des Textspeichers auf dem Bildschirm aus und kehrt dann ins Betriebssystem zurück ; Definition von Systemadressen :Adresse für Aufruf von SYSTEM EQU 00005H ;Systemfunktionen ;Definition von Systemfunktionen \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* WSTARTF EQU 0 ;Warmstart CONAUSF EQU ; Ausgabe auf CON: (=Konsole) STRAUSF EQU ; Ausgabe von Strings auf CON: TXTBEGF EQU 247 ;Startadresse des Text-;speichers abfragen \* ; Definition von ASCII-Konstanten \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* CR **EQU** ODH ;Cursor an Zeilenanfang ;(Carriage Return) LF EQU 0AH ;Cursor in neue Zeile ;(Line Feed) **EOF EQU** 1AH ; Markierung für Textende ;(End Of File) **STOP** 181 **EOU** ;Markierung für Stringende ; Hauptprogramm TEXTAUSGABE \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;Das Programm beginnt bei

;0100H

ORG

100H

## START:

CALL STRING

DB CR, LF, 'Inhalt des Textspeichers: ', STOP

CALL NEUE ZEILE ; Der Cursor wird in eine

;neue Zeile gesetzt

LD C,TXTBEGF ;Der Anfang des Textspeichers CALL SYSTEM ;wird ins HL-Register geholt

#### SCHLE IFE:

LD A,(HL) ;Ein Zeichen ins A-Register..
INC HL ;..und Pointer HL erhöhen

CP EOF ; Ist es END OF FILE ?
JR Z,ENDE ; Ja: nach ENDE springen

LD E,A ; Nein: ins E-Register laden, CALL ZEICHEN ; ausgeben und...

JR SCHLEIFE ; ... nächstes Zeichen holen

#### ENDE:

CALL NEUE ZEILE

LD C, WSTARTF ;Die Warmstartfunktion CALL SYSTEM ;wird aufgerufen

### ZEICHEN:

LD C, CONAUSF ; Der Funktionscode wird ins

;C-Register geladen,

PUSH HL ;HL gerettet

CALL SYSTEM ;und das Betriebssystem

;aufgerufen

POP HL ;wiederherstellen

RET ;Rücksprung

```
·*************************
:Unterprogramm NEUE ZEILE
 - fängt auf dem Bildschirm eine neue Zeile an
• *******************
NEUE ZEILE:
       LD
             E,CR
                           ;Der Cursor wird an den
       CALL
              ZEICHEN
                           ;Anfang der Zeile...
       LD
             E.LF
                           :.. und dann eine Zeile
       CALL
             ZEICHEN
                           ;tiefer gesetzt
       RET
                           :Rücksprung
**************************
;Unterprogramm STRING
 - schreibt den String, der nach dem aufrufenden Programm
 ********************
STRING:
      POP
             DE
                           ;Rückkehradresse ins DE-
                           :Registerpaar holen und
       PUSH
             DE
                           ;wieder auf den Stack zurück-
                           ;bringen - DE zeigt jetzt auf
                           ;den Stringanfang
      LD
             C,STRAUSF
                           ;Der Funktionscode wird ins
                           ;C-Register geladen
      CALL
             SYSTEM
                           und das Betriebssystem
                           ;aufgerufen
      POP
             HL
                           ;Rückkehradresse nochmal vom
                           ;Stack holen - jetzt zeigt HL
                           auf den Stringanfang
      LD
             BC, OFFFFH
                           ; Zähler BC auf Maximalwert,
      LD
             A,STOP
                           :Stringendezeichen in Reg A
      CPIR
                           ;und Stringende suchend
                           ;... HL zeigt nun auf die
                           ; Adresse nach dem Stringende
      JP
             (HL)
                           ; diese Adresse anspringen
 ************************
; Programmende
              ******************
      END
             START
                           ;Der Programmzähler wird
                           ;auf den Programmanfang
                           ;gesetzt
```

SYSTEM EQU

STATISTIK - Programm 4 TITLE

;Dieses Programm zählt die Häufigkeit eines bestimmten Buch-;stabens im Textspeicher und kehrt dann ins Betriebssystem zurück \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* ;Definition von Systemadressen

00000H ;Adresse für Warmstart WSTART EQU

> 00005H ;Adresse für Aufruf von

;Systemfunktionen

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* ; Definition von Systemfunktionen \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

WSTARTF EQU 0 ;Warmstart

CONEINF EQU ; Eingabe von CON: (=Tastatur)

CONAUSF EQU ; Ausgabe auf CON: (=Konsole)

STRAUSF EQU ; Ausgabe von Strings auf CON:

PRBYTEF EQU 240 ;Ausgabe eines "BYTE" als ;zweistellige Sedezimalzahl.

PRWORDF EQU 241 ;Ausgabe eines 'WORD!' als ; vierstellige Sedezimalzahl

TXTBEGF EQU 247 ;Startadresse des Text-;speichers abfragen

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

; Definition von ASCII-Konstanten 

CR	EQU	ODH	;Cursor an Zeilenanfang ;(Carriage Return)
LF	EQU	ОАН	;(Curriage Recurr); ;Cursor in neue Zeile ;(Line Feed)
EOF	EQU	1AH	;(End Of File)
STOP	EQU	<b>'\$</b> '	; Markierung für Stringende

```
12
```

```
***************
;Hauptprogramm STATISTIK
                 **********
      ORG
             100H
                           ;Das Programm beginnt bei
                           :0100H
START:
      CALL
             STRING
       DB
             CR, LF, 'Zeichenstatistik: '
              'Bitte Taste drücken - ,STOP
       DB
       CALL
             EINGABE
                           ;ein Zeichen von der
                           :Tastatur holen, Zeichen
                           ;ist im Register A
       CP
             'C'-40H
                           :Ist es 'C?
       JP
             Z, WSTART
                           ; Ja: Rückkehr ins Betriebs-
                               system
                           ; Nein: weiter .
       CALL
             STATISTIK
                           ;auswerten
       JR
             START
                           ;Schleife - kann durch
                           ; Eingabe von 'C abgebrochen
                           ;werden
***********************
:Unterprogramm EINGABE
 - holt ein Zeichen von der Tastatur ins Register A und
   schreibt es auf den Bildschirm
 ***********************
EINGABE:
      LD
             C, CONEINF
                           ;Der Funktionscode wird ins
                           ;C-Register geladen
      CALL
             SYSTEM
                           ;und das Betriebssystem
                           ;aufgerufen
      RET
                           ;Rücksprung
*******************
;Unterprogramm STATISTIK
 - Der Textspeicher wird nach dem Zeichen im A-Register
   durchsucht und die Häufigkeit gezählt.
STATISTIK:
```

;Das zu suchende Zeichen ;wird ins Stack gerettet,

**PUSH** 

AF

	CALL	NEUE_ZEILE	;eine neue Zeile begonnen
	LD	C,TXTBEGF	;und der Anfang des Texts- ;peichers ins HL-Register-
	CALL	SYSTEM	;paar geholt
	POP LD LD	AF C,A DE,O	;Das Zeichen wird ins ;A- und C-Register geholt ;und der Zähler (Register- ;paar DE) auf Null gesetzt
SCHLEIF	Е:		
	LD INC	A,(HL) HL	;Ein Zeichen ins A-Register ;und Pointer HL erhöhen
	CP JR	EOF Z,ERGEBNIS	;Ist es END OF FILE ? ; Ja: nach ERGEBNIS springen ; Nein: weiter
	CP JR	C NZ,SCHLEIFE	; Ist es das gesuchte Zeichen' ; Nein: nächstes Zeichen
	INC JR	DE SCHLEIFE	; Ja: Zähler erhöhen und ; nächstes Zeichen
ERGEBNI	S:		
	PUSH PUSH	DE BC	;Zählerstand und ;gesuchtes Zeichen auf Stack
	CALL DB	STRING 'Zeichen ',STOP	
•	POP PUSH	DE DE	;gesuchtes Zeichen von Stack ;nach Register E, wieder ;zurück auf Stack
	CALL	ZEICHEN	;und ausgeben
	CALL DB	STRING ' (=ASCII ',STOR	
·	POP	DE .	;gesuchtes Zeichen von Stack ;wieder ins Register E und
	LD CALL	C,PRBYTEF SYSTEM	;ASCII-Code als Sedezimal- ;zahl ausgeben
	CALL DB	STRING 'H) kommt ',STO	P
	POP LD CALL	DE C,PRWORDF SYSTEM	;Anzahl von Stack ins ;Registerpaar DE und als ;Sedezimalzahl ausgeben

```
14
```

```
CALL
            STRING
      DB
            'H mal vor ,CR,LF,STOP
      RET
                         ;fertig: Rücksprung
;Unterprogramm ZEICHEN
- schreibt das Zeichen im E-Register auf den Bildschirm
*****************
ZEICHEN:
      LD
            C, CONAUSF
                         ;Der Funktionscode wird ins
                         ;C-Register geladen,
      PUSH
                         ;HL gerettet
            HL
            SYSTEM
                        ;und das Betriebssystem
      CALL
                        ;aufgerufen
      POP
                         ;wiederherstellen
            HL
      RET
                         ;Rücksprung
*****************
;Unterprogramm NEUE ZEILE
; - fängt auf dem Bildschirm eine neue Zeile an
*******************
NEUE ZEILE:
      LD
            E, CR
                         ;Der Cursor wird an den
      CALL
            ZEICHEN
                         ;Anfang der Zeile...
      LD
            E, LF
                        ;... und dann eine Zeile
                        ;tiefer gesetzt
      CALL
            ZEICHEN
      RET
                         ;Rücksprung
;Unterprogramm STRING
 - schreibt den String, der nach dem aufrufenden Programm
          ***************
STRING:
      POP
            DE
                        ;Rückkehradresse ins DE-
                        ;Registerpaar holen und
      PUSH
            DE
                        ;wieder auf den Stack zurück-
                        ;bringen - DE zeigt jetzt auf
                        ;den Stringanfang
```

LD

CALL

POP

LD

LD

JP

:Programmende

**CPIR** 

C,STRAUSF

BC, OFFFFH

A,STOP

(HL)

SYSTEM

HL

ı
3

END START ;Der Programmzähler wird ;auf den Programmanfang ;gesetzt

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;Der Funktionscode wird ins

;Rückkehradresse nochmal vom ;Stack holen - jetzt zeigt HL

;Zähler BC auf Maximalwert, ;Stringendezeichen in Reg. A

;C-Register geladen ;und das Betriebssystem

;auf den Stringanfang

;und Stringende suchend

;... HL zeigt nun auf die ;Adresse nach dem Stringende

;diese Adresse anspringen

;aufgerufen

```
; Hauptprogramm AUTOMATISCHE STATISTIK
     holt nacheinander alle ASCII-Zeichen von A bis z in das A-Register
     und zählt ihre Häufigkeit.
ORG
           100H
                       ;das Programm beginnt bei 0100H
START:
     LD
           A, 'A'
                       ;erstes Zeichen
     PUSH
LOOP:
           AF
                       ; Zeichen im Stack verwahren
     CALL
           STATISTIK
                       ;auswerten
     POP
           AF
                       ; Zeichen zurückholen
     INC
           Α
                       ;nächstes Zeichen holen
     CP
           · ß · +1
                       ;ist es noch druckbar?
     JR
           NZ, LOOP
                       ; ja: Auswerten
     JP
           WSTART
                       ;nein: Ende, Warmstart
 *******************************
;Unterprogramm STATISTIK
     Der Textspeicher wird nach dem Zeichen im A-Register durchsucht und
     die Häufigkeit gezählt.
***********************
```

usw.

AF

**PUSH** 

```
CONWAY's Life-Game
Programm 1
;Definition von Systemadressen
:Adresse für Warmstart
WSTART
    EQU
          00000H
          00005H
                     :Adresse für Aufruf von
    EOU
SYSTEM
                     :Systemfunktionen
*******************
Definition von Systemfunktionen
****************
WSTARTF EQU
                     ;Warmstart
DIRCONF EQU
                     ;direkte Konsol Ein-/Ausgabe
TXTBEGF EQU
          247
                     ;Startadresse des Text-
                     ;speichers abfragen
;Definition von ASCII-Konstanten
****************
CONTROL EQU
          1FH
                     ;Erzeugung von Control-Codes
                     ;mit ... AND CONTROL
CR
     EQU
          ODH
                     ;Cursor an Zeilenanfang
                     ;(Carriage Return)
LF
     EQU
          OAH
                     ;Cursor in neue Zeile
                     ;(Line Feed)
EOF
     EQU
          1AH
                     ;Markierung für Textende
                     ;(End Of File)
ESC
     EQU
          1BH
                     ;Escape
                     ;(für Cursor-Positionierung)
;sonstige Konstanten
24
ZEILEN EQU
                     ;Anzahl Zeilen
SPALTEN EQU
          80
                     ;Anzahl Spalten
GROESSE EQU
          ZEILEN*SPALTEN
                     :Spielfeldgröße
;Hauptprogramm LIFE
*********************
     ORG
          100H
                     ;Das Programm beginnt bei
```

;0100H

```
START:
       CALL
               HOLE FELD
                               ;Feld aus Textspeicher holen
                               ;und auf Schirm schreiben
       CALL
               SCHREIBE FELD
LIFE:
       CALL
               DIRIN
                               ;Tastatureingabe?
       OR
                               ;kein Zeichen (also A=0)?
                               ; ja: warten
       JR
               Z, LIFE
                               ;nein, Zeichen da:
       CP
               'C' AND CONTROL : Control-C ?
       JP
               Z, WSTART
                               ; ja: Warmstart
       JR
               LIFE
                               ;Schleife
**************
:Unterprogramm HOLE FELD
 - Holt vom Textspeicher eine Aufstellung in das interne
    Spielfeld
 HOLE FELD:
               C, TXTBEGF
                               :Anfang des Textspeichers
       LD
               SYSTEM
                               ;ins HL-Registerpaar
       CALL
       LD
               DE, FELD
                               ; .. Spielfeld in DE
                               ; und Feldgroesse in BC
       LD
               B, SPALTEN
       LD
               C, ZEILEN
HOLE WEITER:
               A,(HL)
       LD
                               ;ein Zeichen holen
       INC
               HL
                               ;Textzeiger weiter
       CP
               LF
                               ; Zeichen = Line Feed?
       JR
               Z, HOLE WEITER
                               ; ja: ignorieren
       CP
                               :Zeichen = Textende?
       JR
               Z, LOESCHE REST
                               ; ja: Feld löschen
       CP
               CR
                               ; Zeichen = neue Zeile?
       JR
               Z,NEUE_ZEILE
                               ; ja: in neue Zeile
       CP
                               ; Zeichen = Zwischenraum
       JR
               Z,LOESCHE EL
                               ; ja: Feldelement loeschen
       LD
               A, 1
                               ; nein: Feldelement:=1
SETZE EL:
               (DE), A
                               ;Feldelement setzen
       LD
       LD
               A,B
                               ; Zeile schon voll,
       OR
               A
                               ;d.h. Spaltenzähler auf Null?
       JR
               Z, HOLE WEITER
                               ; ja: weiter (bis neue Zeile)
       DEC
               В
                               ; nein: Spaltenzähler zurück
                                       Feldzeiger weiter
       INC
               DE
       JR
               HOLE WEITER
                                      und weitermachen
LOESCHE EL:
       LD
               A, 0
               SETZE EL
       JR
```

```
LOESCHE REST:
                               ;Textzeiger zurück auf EOF,
       DEC
               HL
                               :dann NEUE ZEILE
                               :dadurch so oft NEUE ZEILE bis
                               ;Feld voll
                               :Rest der Zeile wird gelöscht
NEUE ZEILE:
                               ;Spaltenzähler in A
       LD
               A, B
                               ;Spaltenzähler setzen
               B, SPALTEN
       LD
LOESCHE ZEILE:
                               ;(alter) Spaltenzähler Null?
       OR
               Z, ZEILE VOLL
                               ; ja: Zeile schon voll
        JR
       EX
               DE,HL
       LD
               (HL),0
                               ; nein: Feldelement löschen
        EX
               DE, HL
        INC
               DE
                                Feldzeiger inkrementieren,
                               : Zähler dekrementieren
       DEC
               LOESCHE ZEILE
                               ; und weiter loeschen
        JR
ZEILE VOLL:
        DEC
               C
                                ; Zeilenzähler zurück, Null?
        RET
                Z
                                ; ja: alle Zeilen voll, fertig
                                ; nein: weitermachen
        JR
               HOLE WEITER
******************
;Unterprogramm SCHREIBE FELD
 - Gibt das interne Spielfeld auf dem Bildschirm aus
 *******************
SCHREIBE FELD:
        LD
               HL,0
        CALL
                CURSOR
                                ;Cursor nach links oben
        LD
                                ;Adresse Spielfeld in HL
               HL, FELD
        LD
                BC, GROESSE-1
                                ;Groesse Spielfeld in BC
SCHREIBE EL:
                E, '*'
        LD
                                ;schreibe * falls gesetzt
        LD
                A,(HL)
                                ;hole Feldelement
        OR
                                ;Element gesetzt (nicht 0)?
                Α
        JR
                NZ, NICHT LEER
                                ; ja: weiter, * lassen
                E,''
        LD
                                ; nein: Leerschritt statt *
NICHT LEER:
        PUSH
                BC
                                ;Zeichen ausgeben
        PUSH
                HL
        CALL
                DIROUT
        POP
                HL
        POP
                BC
        INC
                HL
                                ;Feldzeiger erhöhen
        DEC
                BC
                                ;Zähler dekrementieren
        LD
                                ;und prüfen, ob Null?
                A,B
        OR
        JR
                NZ, SCHREIBE EL
                               ; nein: weiter
        RET
                                ; ja: fertig
```

```
*************
;Unterprogramm CURSOR
 - setzt den Cursor an Zeile L, Spalte H
**************
CURSOR:
      PUSH
                         ; Zeile/Spalte auf Stack
            HL
                         ;ESC ausgeben
            E,ESC
      LD
      CALL
            DIROUT
            E, '='
      LD
                         ;= ausgeben
      CALL
            DIROUT
      POP
                         ;Zeile/Spalte von Stack
            HL
      PUSH
            HL
                         ;holen und wieder zurück
      LD
            A,L
                         ; Zeile ausgeben
      CALL
            CURSOR1
                                  mit CURSOR1
      POP
            HL
                         ; Zeile/Spalte von Stack
      LD
            A,H
                         ;Spalte ausgeben
CURSOR1:
            A,' '
      ADD
                         ;20H addieren
      LD
            E,A
                         ;und dann ausgeben
            DIROUT
      CALL
      RET
;Unterprogramme DIROUT und DIRIN
; - gibt das Zeichen im E-Register mit Systemfunktion 6 aus
   oder holt Zeichen mit Systemfunktion 6 (DIRCONF)
****************
DIRIN:
            E, OFFH
      LD
DIROUT:
      LD
            C, DIRCCNF
      CALL
            SYSTEM
      RET
******************
; Variable im RAM
FELD:
      DS
            GROESSE, 0
                         ;Spielfeld
      END
            START
```

F

```
CONWAY's Life-Game
Programm 2
*****************
;Definition von Systemadressen
***************
                        ;Adresse für Warmstart
WSTART
     EQU
           00000H
SYSTEM
     EQU
           00005H
                        ;Adresse für Aufruf von
                        ;Systemfunktionen
***************
;Definition von Systemfunktionen
******************
WSTARTF EQU
           0
                        :Warmstart
DIRCONF EQU
                        ;direkte Konsole Ein-/Ausgabe
TXTBEGF EQU
            247
                        ;Startadresse des Text-
                        ;speichers abfragen
*******************
Definition von ASCII-Konstanten
CONTROL EQU
            1FH
                        ;Erzeugung von Control-Codes
                        ;mit ... AND CONTROL
CR
      EQU
            ODH
                        ;Cursor an Zeilenanfang
                        ;(Carriage Return)
LF
      EQU
            OAH
                        ;Cursor in neue Zeile
                        ;(Line Feed)
      EQU
EOF
            1AH
                        ; Markierung für Textende
                        ; (End Of File)
ESC
      EQU
            1BH
                        ;Escape
                        ;(für Cursor-Positionierung)
********************
;sonstige Konstanten
******************
            24
ZEILEN EQU
                        ;Anzahl Zeilen
SPALTEN EQU
            80
                        ;Anzahl Spalten
GROESSE EQU
            ZEILEN*SPALTEN
                        ;Spielfeldgröße
*****************
;Hauptprogramm LIFE
* ************************
      ORG
            100H
                        ;Das Programm beginnt bei
                        ;0100H
```

START:

LIFE:

CALL

CALL

CALL

OR

JR

CP

CALL

INC

INC DEC

LD

OR

JR RET IX

ΙY

BC

A,B

HOLE FELD

DIRIN

Z, LIFE

SCHREIBE FELD

;Feld aus Textspeicher holen

:und auf Schirm schreiben

;kein Zeichen (also A=0)?

:Tastatureingabe?

:nein. Zeichen da:

;feld entsprechend

;erhöhe Zeiger Ausgangsfeld

erhöhe Zeiger Ergebnisfeld

;dekrementiere Zähler ;und prüfe, ob Null?

SCHREIBE ELEMENT; schreibe \* oder Leerzeichen

;auf Schirm

; ja: fertig

NZ, GENERATION 1; nein: weiter

; ja: warten

'C' AND CONTROL ; Control-C ?

```
;Unterprogramm PRUEFE ELEMENT
: - Prüft, ein einzelnes Feldelement in der nächsten
   Generation belegt ist
PRUEFE ELEMENT:
      LD
             A,(IX)
                           :Feldelement laden
                           und 4 mal zu sich selbst
      ADD
             A,A
      ADD
             A,A
                           ;addieren, d.h. mit 16
      ADD
                           :multiplizieren
             A.A
      ADD
                           ;dann Nachbarelemente addieren:
             A,A
      ADD
             A,(IX-81)
                           ;links oben
      ADD
             A, (IX-80)
                           ;oben
      ADD
             A_{1}(IX-79)
                           :rechts oben
      ADD
             A,(IX-1)
                           :links
             A,(IX+1)
      ADD
                           :rechts
      ADD
             A, (IX+79)
                           ;links unten
      ADD
             A,(IX+80)
                           ;unten
      ADD
             A, (IX+81)
                           ;rechts unten
      LD
             HL, REGELN
                           ;Adresse Regeltabelle nach HL
      LD
             D.0
                           ;und Feldelement*16+Anzahl der
                           :Nachbarn addieren
      LD
             E,A
      ADD
             HL, DE
                           :Tabellenwert laden
      LD
             A,(HL)
      OR
                           und verknüpfen für Flags
             Α
      RET
REGELN:
      DB
             0,0,0,1,0,0,0,0 ;Spielregeln, erste Hälfte
      DB
             0,0,0,0,0,0,0;(16 Byte) für bisher
                           ;leeres Feldelement
      DB
             0,0,1,1,0,0,0,0; zweite für bisher belegtes
      DB
             0,0,0,0,0,0,0 ;Feldelement
*******************
;Unterprogramm SETZE ELEMENT
; - Setzt ein einzelnen Feldelement im neuen Feld
******************
SETZE ELEMENT:
      LD
             (IY),A
      RET
```

```
р—,
```

```
;Unterprogramm SCHREIBE ELEMENT
 - Schreibt ein einzelnes Feldelement auf Schirm
SCHREIBE ELEMENT:
       LD
               E, '*'
       OR
               Α
       JR
               NZ, NO SPACE
       LD
NO SPACE:
       PUSH
               IX
       PUSH
               IY
       PUSH
               BC
       CALL
               DIROUT
       POP
               BC
       POP
               ΙY
       POP
               IX
       RET
;Unterprogramm HOLE FELD
 - Holt vom Textspeicher eine Aufstellung in das interne
   Spielfeld
HOLE FELD:
               C, TXTBEGF
       LD
                              ;Anfang des Textspeichers
       CALL
               SYSTEM
                              ;ins HL-Registerpaar
       LD
               DE, (VON FELD)
                              ; .. Spielfeld in DE
       LD
               B, SPALTEN
                              ; und Feldgroesse in BC
       LD
               C, ZEILEN
HOLE WEITER:
       LD
               A,(HL)
                              ;ein Zeichen holen
       INC
               HL
                              ;Textzeiger weiter
       CP
               LF
                              ; Zeichen = Line Feed?
       JR
               Z,HOLE WEITER
                              ; ja: ignorieren
       CP
               EOF
                              ; Zeichen = Textende?
       JR
               Z,LOESCHE REST
                              ; ja: Feld löschen
       CP
               CR
                              ; Zeichen = neue Zeile?
       JR
               Z,NEUE ZEILE
                              ; ja: in neue Zeile
       CP
                              ; Zeichen = Zwischenraum
       JR
               Z,LOESCHE EL
                              ; ja: Feldelement loeschen
       LD
               A, 1
                              ; nein: Feldelement:=1
SETZE EL:
               (DE),A
       LD
                              ;Feldelement setzen
       LD
               A,B
                              ; Zeile schon voll,
       OR
                              ;d.h. Spaltenzähler auf Null?
       JR
               Z,HOLE WEITER
                              ; ja: weiter (bis neue Zeile)
       DEC
               В
                              ; nein: Spaltenzähler zurück
       INC
                                     Feldzeiger weiter
               HOLE WEITER
       JR
                                     und weitermachen
```

```
LOESCHE EL:
        LD
                A,0
        JR
                SETZE EL
LOESCHE REST:
                                ;Textzeiger zurück auf EOF,
                HL
        DEC
                                 ;dann NEUE ZEILE
                                 :dadurch so oft NEUE ZEILE bis
                                 ;Feld voll
                                 ;Rest der Zeile wird gelöscht
NEUE ZEILE:
                                 ;Spaltenzähler in A
        LD
                A,B
                                ;Spaltenzähler setzen
        LD
                B, SPALTEN
LOESCHE ZEILE:
        OR
                Α
                                 ;(alter) Spaltenzähler Null?
        JR
                Z, ZEILE VOLL
                                 ; ja: Zeile schon voll
        EX
                DE, HL
                                 ; nein: Feldelement löschen
        LD
                (HL),0
        EX
                DE, HL
        INC
                DE
                                 ; Feldzeiger inkrementieren,
        DEC
                                 ; Zähler dekrementieren
                Α
                                 ; und weiter loeschen
        JR
                LOESCHE ZEILE
ZEILE VOLL:
        DEC
                C
                                 ; Zeilenzähler zurück, Null?
                Z
        RET
                                 ; ja: alle Zeilen voll, fertig
        JR
                                 ; nein: weitermachen
                HOLE WEITER
*****************
;Unterprogramm SCHREIBE FELD
 - Gibt das interne Spielfeld auf dem Bildschirm aus
SCHREIBE FELD:
        LD
                HL,0
        CALL
                CURSOR
                                 ;Cursor nach links oben
                HL, (VON FELD)
                                 ;Adresse Spielfeld in HL
        LD
                BC, GROESSE-1
                                 Groesse Spielfeld in BC
        LD
SCHREIBE EL:
        LD
                E, '*'
                                 ;schreibe * falls gesetzt
        LD
                A,(HL)
                                 ;hole Feldelement
        OR .
                                 ;Element gesetzt (nicht 0)?
                Α
                NZ, NICHT LEER
        JR
                                 ; ja: weiter, * lassen
                E.' '
        LD
                                 ; nein: Leerschritt statt *
NICHT LEER:
        PUSH
                BC
                                 ; Zeichen ausgeben
        PUSH
                HL
        CALL
                DIROUT
        POP
                HL
        POP
                BC
        INC
                HL
                                 ;Feldzeiger erhöhen
        DEC
                BC
                                 ;Zähler dekrementieren
        LD
                A,B
                                 ;und prüfen, ob Null?
        OR
        JR
                NZ, SCHREIBE EL
                                 ; nein: weiter
        RET
                                 ; ja: fertig
```

```
************
;Unterprogramm CURSOR
 - setzt den Cursor an Zeile L, Spalte H
CURSOR:
       PUSH
                             ; Zeile/Spalte auf Stack
              HL
              E, ESC
                             ;ESC ausgeben
       LD
       CALL
              DIROUT
              E, '='
       LD
                             ;= ausgeben
              DIROUT
       CALL
       POP
              HL
                             ; Zeile/Spalte von Stack
       PUSH
              HL
                             ;holen und wieder zurück
       LD
              A,L
                             ;Zeile ausgeben
       CALL
              CURSOR1
                                        mit CURSOR1
       POP
              HL
                             ;Zeile/Spalte von Stack
                             ;Spalte ausgeben
       LD
              A,H
CURSOR1:
              A,' '
                             ;20H addieren
       ADD
       LD
              E,A
                             ;und dann ausgeben
              DIROUT
       CALL
       RET
******************
;Unterprogramme DIROUT und DIRIN
; - gibt das Zeichen im E-Register mit Systemfunktion 6 aus
   oder holt Zeichen mit Systemfunktion 6 (DIRCONF)
DIRIN:
       LD
              E, OFFH
DIROUT:
       LD
              C, DIRCONF
              SYSTEM
       CALL
       RET
:Variable im RAM
*****************
       DS
              81,0
              GROESSE, 0
FELD1:
       DS
                             ;Spielfeld 1
       DS
              2*81,0
FELD2:
       DS
              GROESSE, 0
                             ;Spielfeld 2
       DS
              81,0
       END
              START
```

```
TITLE
           CONWAY's Life-Game schneller Programm 1
****************
;Definition von Systemadressen
WSTART
     EOU
           00000H
                      :Adresse für Warmstart
     EQU
           00005H
                      ;Adresse für Aufruf von
SYSTEM
                      ;Systemfunktionen
****************
; Definition von Systemfunktionen
******************
WSTARTF EQU
           0
                      :Warmstart
DIRCONF EQU
                      :direkte Konsole Ein-/Ausgabe
           247
                      :Startadresse des Text-
TXTBEGF EQU
                      ;speichers abfragen
;Definition von ASCII-Konstanten
****************
CONTROL EQU
                      ;Erzeugung von Control-Codes
           1FH
                      ;mit ... AND CONTROL
CR
     EQU
           0DH
                      ;Cursor an Zeilenanfang
                      ;(Carriage Return)
LF
     EQU
           0AH
                      ;Cursor in neue Zeile
                      ;(Line Feed)
EOF
     EQU
           1AH
                      ;Markierung für Textende
                      ;(End Of File)
ESC
                       :Escape
     EQU
           1BH
                      ;(für Cursor-Positionierung)
***************
;sonstige Konstanten
*******************
ZEILEN EOU
           24
                       ;Anzahl Zeilen
SPALTEN EOU
           80
                       ;Anzahl Spalten
           ZEILEN*SPALTEN
                      ;Spielfeldgröße
GROESSE EQU
```

```
; Hauptprogramm LIFE-GAME
********************
       ORG
              100H
                            ;Das Programm beginnt bei
                            ;0100H
START:
                            ;Feld aus Textspeicher holen
       CALL
              HOLE FELD
       CALL
              SCHREIBE FELD
                            ;und auf Schirm schreiben
LIFE:
              DIRIN
                            ;Tastatureingabe?
       CALL
                            ;kein Zeichen (also A=0)?
       OR
              Α
       JR
              Z,LIFE
                            ; ja: warten
                            ;nein, Zeichen da:
       CP
              'C' AND CONTROL : Control-C ?
                            : ia: Warmstart
       JP
              Z, WSTART
       CP
                            ; Carriage Return?
              CR
       CALL
              Z, GENERATION
       JR
              LIFE
                            :Schleife
   ********************
;Unterprogramm GENERATION
 - Rechnet eine neue Generation aus
 *****************
VON FELD:
              DW
                     FELD1
NACH FELD:
              DW
                     FELD2
GENERATION:
       LD
              HL,0
                             ;Cursor nach links oben
       CALL
              CURSOR
       LD
              IX, (VON FELD)
                             ;IX := Zeiger Ausgangsfeld
              IY, (NACH FELD)
                             ; IY := Zeiger Ergebnsisfeld
       LD
       LD
              (VON FELD), IY
                             ; Ausgangsfeld und Ergebnis-
       LD
              (NACH FELD), IX
                             ;feld für nächste Generation
                             ;tauschen
       LD
              BC, GROESSE-1
                             ;BC := Anzahl der Feldelemente
                             ;die zu berechnen sind
       EXX
              HL,0
                             ;Spalte/Zeile auf Schirm und
         LD
         LD
              DE,0
                             ;Zähler für ausgelassene
                             :Zeichen
```

;im 2. Registersatz

EXX

```
GENERATION 1:
       CALL
              PRUEFE ELEMENT
                             :prüfe, ob Feldelement in
                             :nächster Gen. besetzt ist
       CALL
                             ;setze Feldelement in Ergebnis-
              SETZE ELEMENT
                             ;feld entsprechend
              SCHREIBE ELEMENT; schreibe * oder Leerzeichen
       CALL
                             :auf Schirm
       LD
                             ;lösche altes Ausgangsfeld
              (IX),0
       INC
              IX
                             ;erhöhe Zeiger Ausgangsfeld
       INC
              ΙY
                             erhöhe Zeiger Ergebnisfeld
       DEC
              BC
                             ;dekrementiere Zähler
       LD
              A,B
                             ;und prüfe, ob Null?
       OR
              C
       JR
              NZ, GENERATION 1; nein: weiter
       RET
                             ; ja: fertig
;Unterprogramm PRUEFE ELEMENT
; - Prüft, ein einzelnes Feldelement in der nächsten
   Generation belegt ist
*****************
PRUEFE ELEMENT:
              E,(IX)
       LD
                             ;Feldelement laden
       LD
              D,0
       LD
              HL, REGELN
                             ;Adresse Regeltabelle nach HL
       ADD
              HL, DE
       LD
              A,(HL)
                             ;Tabellenwert laden
       RET
REGELN:
       DB
              0,0,0,1,0,0,0,0 ;Spielregeln, erste Hälfte
       DB
              0,0,0,0,0,0,0,0;(16 Byte) für bisher
                             :leeres Feldelement
       DB
              0,0,1,1,0,0,0,0; zweite für bisher belegtes
       DB
              0,0,0,0,0,0,0; Feldelement
 ************************
;Unterprogramm SETZE ELEMENT
 - Setzt ein einzelnen Feldelement im neuen Feld
• ***********************************
SETZE ELEMENT:
       OR
              Α
       RET
              Z
                             ;A=0: schon fertig!
              A,10H
       LD
       ADD
              A,(IY)
                             ; belegtes Feld markieren
              (IY),A
       LD
       INC
              (IY-81)
       INC
              (1Y-80)
              (IY-79)
       INC
                             ;und Nachbarfelder erhöhen
       INC
              (IY-1)
```

```
INC
               (IY+1)
        INC
               (IY+79)
        INC
               (IY+80)
        INC
               (IY+81)
       RET
;Unterprogramm SCHREIBE ELEMENT
; - Schreibt ein einzelnes Feldelement auf Schirm
********************
; die Befehle, die den zweiten Registersatz benutzen, sind
;zur besseren Übersicht eingerückt!
SCHREIBE ELEMENT:
       XOR
               (IX)
                               ; Vergleich mit
       AND
               10H
                                     Ausgangsfeld
       CALL
               Z,NO CHANGE
                               ;gleich: nichts ändern
       CALL
               NZ, CHANGE
                               ;ungleich: neu schreiben
       EXX
                               ;neue Spalte/Zeile
         INC
               Η
                               ;in HL' berechnen
         LD
               A,H
         CP
               SPALTEN
                               ; Zeile zu Ende?
         JR
               NZ, SCHREIBE RET; nein: Zeile bleibt
         LD
               H, 0
                               ; ja:
                                      Spalte=0 und
         INC
               L
                                      Zeile erhöhen
SCHREIBE RET:
       EXX
                               ;zurück in 1. Registersatz
       RET
NO CHANGE:
       EXX
         INC
               DE
                              ; Zähler in DE' erhöhen
       EXX
       RET
CHANGE:
       EX X
         LD
               A,D
                              ; Zähler für nicht geschr.
         OR
               Α
                              ;Elemente, D -> Z-Flag
         LD
               A,E
                                         E \rightarrow A
         LD
               DE, O
                              ;und Zähler=0
       EXX
       JR
               NZ, NEU CURSOR
                              ;D>0: Cursor neu setzen
       DEC
                              ; D=0: E=1?
       JR
               NZ, NICHT EINS
       ĹD
               E,9
                              ; ja: mit 09H Cursor 1 rechts
       CALL
               NO SPACE
       JR
               SCHREIBE
```

```
NICHT EINS:
                              ;E=0?
       INC
               Α
NEU CURSOR:
       EXX
                               ;nein: Cursor setzen
         CALL
               NZ, CURSOR
       EXX
SCHREIBE:
       LD
               A, (IY)
       AND
               10H
               E, '*'
       LD
               NZ,NO_SPACE
       JR
               E, ' '
       LD
NO SPACE:
       PUSH
               BC
               DIROUT
       CALL
       POP
               BC
       RET
***************
:Unterprogramm HOLE FELD
 - Holt vom Textspeicher eine Aufstellung in das interne
    Spielfeld
HOLE_FELD:
       LD
               C, TXTBEGF
                               ;Anfang des Textspeichers
       CALL
               SYSTEM
                               ;ins HL-Registerpaar
       LD
               IY, (VON FELD)
                               ; .. Spielfeld in IY
       LD
               B, SPALTEN
                               ; und Feldgroesse in BC
       LD
               C, ZEILEN
HOLE WEITER:
       LD
               A,(HL)
                               ;ein Zeichen holen
       INC
               HL
                               ;Textzeiger weiter
       CP
               LF
                               ; Zeichen = Line Feed?
               Z,HOLE WEITER
       JR
                               ; ja: ignorieren
       CP
               EOF
                               ; Zeichen = Textende?
       JR
               Z,LOESCHE REST
                               ; ja: Feld löschen
       CP
               CR
                               ; Zeichen = neue Zeile?
       JR
               Z, NEUE ZEILE
                               ; ja: in neue Zeile
       CP
                               ;Zeichen = Zwischenraum
       CALL
               NZ, SETZE ELEMENT; nein: Feldelement setzen
       LD
               A,B
                               ; Zeile schon voll,
       OR
               Α
                               ;d.h. Spaltenzähler auf Null?
       JR
               Z, HOLE WEITER
                               ; ja: weiter (bis neue Zeile)
       DEC
               В
                               ; nein: Spaltenzähler zurück
       INC
               ΙY
                                      Feldzeiger weiter
       JR
               HOLE WEITER
                                      und weitermachen
```

```
LOESCHE REST:
        DEC
               HL
                               ;Textzeiger zurück auf EOF,
                               :dann NEUE ZEILE
                               ;dadurch so oft NEUE ZEILE bis
                               ;Feld voll
NEUE ZEILE:
                               ;Rest der Zeile wird gelöscht
               A,B
       LD
                               ;Spaltenzähler in A
       LD
               B, SPALTEN
                               ;Spaltenzähler setzen
LOESCHE ZEILE:
       OR
               Α
                               ;(alter) Spaltenzähler Null?
               Z, ZEILE VOLL
        JR
                               ; ja: Zeile schon voll
        INC
               ΙY
                               ; nein: Feldzeiger inkrementieren,
       DEC
                               ; Zähler dekrementieren
               Α
       JR
               LOESCHE ZEILE
                               ; und weiter
ZEILE VOLL:
               C
       DEC
                               ;Zeilenzähler zurück, Null?
       RET
               Z
                               ; ja: alle Zeilen voll, fertig
        JR
               HOLE WEITER
                               ; nein: weitermachen
*******************
:Unterprogramm SCHREIBE FELD
; - Gibt das interne Spielfeld auf dem Bildschirm aus
************************
SCHREIBE FELD:
       LD
               HL,0
       CALL
               CURSOR
                               :Cursor nach links oben
       LD
               HL, (VON FELD)
                               ;Adresse Spielfeld in HL
               BC, GROESSE-1
       LD
                               Groesse Spielfeld in BC
SCHREIBE EL:
       LD
               E. '*'
                               ;schreibe * falls gesetzt
               A,(HL)
       LD
                               ;hole Feldelement
       AND
               10H
                               ;Element gesetzt?
        JR
               NZ, NICHT LEER
                               ; ja: weiter, * lassen
               E,''
       LD
                               ; nein: Leerschritt statt *
NICHT LEER:
       PUSH
               BC
                               ; Zeichen ausgeben
       PUSH
               HL
       CALL
               DI ROUT
       POP
               HL
       POP
               BC
        INC
               HL
                               ;Feldzeiger erhöhen
       DEC
               BC
                               ; Zähler dekrementieren
       LD
               A,B
                               ;und prüfen, ob Null?
       OR
               C
       JR
               NZ, SCHREIBE EL
                               ; nein: weiter
       RET
                               ; ja: fertig
```

F

```
:Unterprogramm CURSOR
; - setzt den Cursor an Zeile L, Spalte H
***************
CURSOR:
      PUSH
            HL
                          ; Zeile/Spalte auf Stack
      LD
             E,ESC
                          ;ESC ausgeben
      CALL
             DI ROUT
             E, '='
      LD
                          ;= ausgeben
             DIROUT
      CALL
      POP
             HL
                          ;Zeile/Spalte von Stack
      LD
                          ; Zeile ausgeben
             A,L
      CALL
             CURSOR1
                                    mit CURSOR1
      LD
             A,H
                          ;Spalte ausgeben
CURSOR1:
      PUSH
             HL
                          ; Z/S zurück auf Stack
             A,' '
      ADD
                          ;20H addieren
      LD
             E,A
                          ;und dann ausgeben
      CALL
             DIROUT
      POP
             HL
                          ; Zeile/Spalte von Stack
      RET
                          ;und zurück
;Unterprogramme DIROUT und DIRIN
; - gibt das Zeichen im E-Register mit Systemfunktion 6 aus
   oder holt Zeichen mit Systemfunktion 6 (DIRCONF)
******************
DIRIN:
      LD
             E, OFFH
DIROUT:
      LD
             C, DIRCONF
      CALL
             SYSTEM
      RET
*******************
;Variable im RAM
*******************
      DS
             81,0
FELD1:
      DS
             GROESSE, 0
                          ;Spielfeld 1
      DS
             2*81,0
FELD2:
      DS
             GROESSE, 0
                          ;Spielfeld 2
      DS
             81,0
      END
             START
```

>asm

CRC=9705

```
TITLE
          Fakultaetenberechnung - Programm 2
*****************
;Definition von Systemadressen
SYSTEM
                      ;Adresse für Aufruf von
     EQU
           00005H
                      ;Systemfunktionen
;Definition von Systemfunktionen
******************
CONAUSF EQU
           2
                      ;Konsol-Ausgabe
STRAUSF EQU
                      ;String-Ausgabe
STREINF EQU
           10
                      ;String-Eingabe
;Definition von ASCII-Konstanten
*********************
CR
     EQU
           ODH
                      ;Cursor an Zeilenanfang
                      ;(Carriage Return)
LF
     EQU
           OAH
                      ;Cursor in neue Zeile
                      ;(Line Feed)
; Definition von TRUE/FALSE (WAHR/FALSCH)
**********************
FALSE
     EQU
TRUE
     EQU
          NOT FALSE
```

```
;Sonstige Konstanten
10
MAX LAENGE
             EQU
                           ;Länge des Eingabepuffers
                    57
                           ;größte mögliche Eingabe
MAX FAKT
             EQU
                           ;bei
DEZ STELL
             EQU
                    78
                           ;maximal 78 Dezimalstellen
                           ;(57! = 4*10 \text{ hoch } 76) \text{ bzw.}
BIN STELL
             EQU
                    256
                           ;256-Bit Binärzahl
                           ;(2 \text{ hoch } 256 = 1*10 \text{ hoch } 77)
BIN BYTE
             EQU
                    BIN STELL/8
                           ;Anzahl Bytes d. Binärzahlen
                           ;prüfen, ob Speicher für
CHECK
             EQU
                    TRUE
                           ;Datenbereich frei
***************
; Datenbereich
***************
      DSEG
                           :Datembereich
      ORG
             1000H
                           ;Beginn bei 1000H
PUFFER:
                           ; Puffer für Eingabezeile
                           ;Wert der maximalen Länge
      DB
             MAX LAENGE
      DS
                           ;Platz f. tatsächliche Länge
      DS
             MAX LAENGE
                           ;Platz für Eingabe
DEZ REG:
                           ;Speicher für Dezimalzahl
      DS
             DEZ STELL/2
BIN REG1:
             BIN BYTE
                           ;Speicher für Binärzahlen
      DS
BIN REG2:
      DS
             BIN BYTE
      ΙF
             CHECK
                           ;erzeugt beim assemblieren
      DB
                           ; I-Fehler falls in ge-
      ENDIF
                           ;schütztem Bereich
; Hauptprogramm FAKULTAET
********************
      CSEG
                           ; Programmbereich
      ORG
             100H
                           ;Das Programm beginnt bei
                           ;0100H
```

```
START:
                             ;Eingabezeile von der
       CALL
              EINGABE
                             :Tastatur holen
                             ;Eingabezeile auswerten und
              HOLE ZAHL
       CALL
                             ; Neueingabe bei Fehler
              C, START
       JR.
                             ;Fakultät berechnen
       CALL
              FAKT
                             ;in Dezimalzahl wandeln
       CALL
              WANDLE DEZ
       CALL
              AUSGABE
                             :ausgeben
              START
                             ;und zurück zur Eingabe
       JR
 *****************
;Unterprogramm EINGABE
       holt eine Eingabezeile von der Tastatur in den Puffer
EINGABE TEXT:
       DB
              CR, LF, '? $'
EINGABE:
       LD
              DE, EINGABE TEXT ;? in neuer Zeile ausgeben
       LD
              C,STRAUSF
       CALL
              SYSTEM
       LD
              DE, PUFFER
                             ;und Eingabezeile einlesen
       LD
              C,STREINF
       CALL
              SYSTEM
       RET
;Unterprogramm HOLE ZAHL
       wertet die Eingabe aus und erzeugt Binärzahl
       prüft, ob Eingabe zu groß
HOLE ZAHL:
       LD
              HL, PUFFER+1
                             ;HL auf Länge d. Eingabe
       LD
              B,(HL)
                             ;dieselbe nach B
       INC
              HL
                             ;HL auf Eingabe
       EX
                             ;DE auf Eingabe und
              DE, HL
       LD
              HL,0
                             ;HL := 0
HOL SCHLEIFE:
                             ;wiederhole:
       LD
              A,(DE)
                                Zeichen von Eingabe holen
       INC
              DE
                               und Zeiger weiter
              101
       SUB
                                ASCII -> binär wandeln
       RET
              C
                               Fehler falls kleiner 0
       CP
              10
                                oder größer 9
       CCF
       RET
              C
                                (Rücksprung mit Carry bei
```

Fehler)

Schleifenzähler und

**PUSH** 

**PUSH** 

**END** 

START

BC

DE

```
Zeiger retten,
                          DE := HL
      EX
            DE, HL
            B, 10
      LD
      LD
            HL,0
                          HL := 0
                          wiederhole 10 mal:
HOL SCHLEIFE1:
                            HL := HL+DE
      ADD
            HL, DE
      DJNZ
            HOL SCHLEIFE1
                            (d.h. HL := HL*10)
      LD
            E,A
      LD
            D,0
                          HL := HL+Ziffer
            HL, DE
      ADD
      POP
            DE
                           Zeiger und
      POP
            BC
                          Schleifenzähler holen
      DJNZ
                           Zähler (B) dekrementieren
            HOL SCHLEIFE
                        ;bis Zähler=0
      EX
                        ;Ergebnis nach DE
            DE, HL
      LD
            HL, MAX FAKT
      AND
      SBC
            HL, DE
                        ;MAX FACT-Eingabe:
                        ;CY=1 falls Eingabe>MAX FACT
      RET
****************
:Unterprogramm FAKT
      errechnet Fakultät der Eingabe
      Argument wird in DE übergeben
FAKT:
      RET
;Unterprogramm WANDLE DEZ
***************
WANDLE DEZ:
      RET
;Unterprogramm AUSGABE
***************
PRWORDF EQU
            241
AUSGABE:
      LD
            C, PRWORDF
            SYSTEM
      CALL
      RET
```

```
38
```

```
Z80-Assembler
pass 1:
++++++
pass 2:
++++++
no fatal error(s)

SUM=1FE8
CRC=793F
```

>asm

```
TITLE
          Fakultaetenberechnung - Programm 3
; Definition von Systemadressen
****************
SYSTEM EQU
          00005H
                     ;Adresse für Aufruf von
                     ;Systemfunktionen
*******************
; Definition von Systemfunktionen
******************
CONAUSF EQU
          2
                     ; Konsol-Ausgabe
STRAUSF EQU
                     ;String-Ausgabe
STREINF EQU
          10
                     :String-Eingabe
********************
; Definition von ASCII-Konstanten
*******************
CR
     EQU
          ODH
                     ;Cursor an Zeilenanfang
                     ;(Carriage Return)
LF
     EQU
          0AH
                     ;Cursor in neue Zeile
                     ;(Line Feed)
**********************
; Definition von TRUE/FALSE (WAHR/FALSCH)
**********************************
FALSE
     EQU
TRUE
     EQU
          NOT FALSE
```

```
*******************
:Sonstige Konstanten
*****************
                             ;Länge des Eingabepuffers
                     10
MAX LAENGE
              EQU
                             ;größte mögliche Eingabe
              EQU
                     57
MAX FAKT
                             :bei
                     78
                             :maximal 78 Dezimalstellen
DEZ STELL
              EOU
                             (57! = 4*10 \text{ hoch } 76) \text{ bzw.}
              EQU
                     256
                             ;256-Bit Binärzahl
BIN STELL
                             ;(2 \text{ hoch } 256 = 1*10 \text{ hoch } 77)
              EQU
                     BIN STELL/8
BIN BYTE
                             ;Anzahl Bytes d. Binärzahlen
                             ;prüfen, ob Speicher für
CHECK
              EQU
                     TRUE
                             ;Datenbereich frei
                            *********
;Datenbereich
***************
       DSEG
                             ; Datenbereich
       ORG
              1000H
                             ;Beginn bei 1000H
                             ;Puffer für Eingabezeile
PUFFER:
       DB
              MAX LAENGE
                             ;Wert der maximalen Länge
       DS
                             ;Platz f. tatsächliche Länge
              MAX LAENGE
       DS
                             ;Platz für Eingabe
DEZ REG:
       DS
              DEZ STELL/2
                             ;Speicher für Dezimalzahl
BIN REG1:
       DS
              BIN BYTE
                             ;Speicher für Binärzahlen
BIN REG2:
       DS
              BIN BYTE
       IF
              CHECK
                             erzeugt beim assemblieren
       DB
                             ;I-Fehler falls in ge-
       ENDIF
                             ;schütztem Bereich
;Hauptprogramm FAKULTAET
CSEG
                             ;Programmbereich
       ORG
              100H
                             ;Das Programm beginnt bei
                             ;0100H
START:
       CALL
              EINGABE
                             ;Eingabezeile von der
                             ;Tastatur holen
       CALL
              HOLE ZAHL
                             ;Eingabezeile auswerten und
       JR
              C, START
                             ;Neueingabe bei Fehler
```

```
CALL
              FAKT
                             ;Fakultät berechnen
       CALL
              WANDLE - DEZ
                             ;in Dezimalzahl wandeln
              AUSGABE
       CALL
                             :ausgeben
                      ;und zurück zur Eingabe
  R
       START
 *******************
;Unterprogramm EINGABE
       holt eine Eingabezeile von der Tastatur in den Puffer
***************
EINGABE TEXT:
       DB
              CR, LF, '? $'
EINGABE:
       LD
              DE, EINGABE TEXT ;? in neuer Zeile ausgeben
       LD
              C, STRAUSF
       CALL
              SYSTEM
       LD
              DE, PUFFER
                             ;und Eingabezeile einlesen
       LD
              C,STREINF
  LL
       SYSTEM
       RET
;Unterprogramm HOLE ZAHL
       wertet die Eingabe aus und erzeugt Binärzahl
       prüft, ob Eingabe zu groß
 *********************
HOLE ZAHL:
              HL, PUFFER+1
                             ;HL auf Länge d. Eingabe
       LD
                             ;dieselbe nach B
       LD
              B_{\bullet}(HL)
       INC
                             ;HL auf Eingabe
              HL
       EX
              DE, HL
                             ;DE auf Eingabe und
       LD
              HL,0
                             ;HL := 0
HOL_SCHLEIFE:
                             ;wiederhole:
       LD
              A_{\bullet}(DE)
                                Zeichen von Eingabe holen
       INC
              DE
                               und Zeiger weiter
              101
       SUB
                               ASCII -> binär wandeln
       RET
              C
                               Fehler falls kleiner 0
       CP
              10
                               oder größer 9
       CCF
       RET
              C
                                (Rücksprung mit Carry bei
                               Fehler)
              BC
       PUSH
                                Schleifenzähler und
       PUSH
              DE
                                Zeiger retten,
       EX
              DE,HL
                               DE := HL
       LD
              B, 10
       LD
                               HL := 0
              HL,0
HOL SCHLEIFE1:
                               wiederhole 10 mal:
       ADD
              HL, DE
                                 HL := HL+DE
       DJNZ
              HOL SCHLEIFE1
                                 (d.h. HL := HL*10)
```

```
E,A
       LD
       LD
              D, 0
       ADD
              HL, DE
                                HL := HL+Ziffer
       POP
              DE
                                Zeiger und
       POP
              BC
                                Schleifenzähler holen
                                Zähler (B) dekrementieren
       DJNZ
              HOL SCHLEIFE
                              :bis Zähler=0
       EX
              DE,HL
                              Ergebnis nach DE
       LD
              HL, MAX FAKT
       AND
              Α
       SBC
              HL, DE
                              ;MAX FACT-Eingabe:
                              ;CY=1 falls Eingabe>MAX FACT
       RET
:Unterprogramm FAKT
       errechnet Fakultät der Eingabe
       Argument wird in DE übergeben
 *****************
FAKT:
       LD
              HL, BIN REG1
                              ;BIN REG1 := 1
       CALL
              CLEAR
                              ;BIN REG1 wird gelöscht
                              ;und das letzte Byte auf 1
       DEC
              HL
       INC
               (HL)
                              ;inkrementiert
       LD
              B,E
                              ;Argument ins B-Register
FAKT SCHLEIFE:
                              ;wiederhole:
                                BIN REG2 := BIN REG1*B
       CALL
              MULT
       CALL
              MOV12
                                BIN REG1 := BIN REG2
       DJNZ
               FAKT SCHLEIFE
                                 Zähler (B) dekrementieren
                              :bis Zähler=0
       RET
*******************
;Unterprogramm MULT
       multipliziert BIN REG1 mit B, Ergebnis in BIN REG2
***************
MULT:
       LD
              HL, BIN REG2
       CALL
              CLEAR
                              ;BIN REG2 := 0
       LD
               C,B
                              ;Faktor ins C-Register
MULT SCHLEIFE:
                              ;wiederhole:
       SRL
               C
                                 C := C/2, CY := Rest
       CALL
               C, ADDI ERE
                                 falls Rest (Carry)=1:
                                  BIN REG2 := BIN REG2
                                             +BIN REG1
       CALL
               SCHIEBE
                                BIN REG1 := BIN REG1*2
       LD
               A,C
                              ;bis C=0
       OR
              NZ, MULT SCHLEIFE
       JR
       RET
```

```
********************
;Unterprogramm ADDIERE
       addiert BIN REG1 zu BIN REG2, Ergebnis in BIN REG2
   *****
ADDIERE:
       PUSH
             BC
                                  : Vorbereitung:
       LD
             HL, BIN REG2 + BIN BYTE
                                  ; Pointer auf
             DE, BIN REG1 + BIN BYTE
       LD
                                  ;niedrigstes Byte+1
       LD
             B, BIN BYTE
                                  ; Zähler laden
       AND
                                  ;und Carry löschen
ADDIER SCHLEIFE:
                           :wiederhole:
       DEC
                              Zeiger dekrementieren
             HL
       DEC
             DE
       LD
             A,(DE)
                             Byte von BIN REG1 laden
       ADC
             A,(HL)
                              zu BIN REG2 addieren
                             und in BIN REG2 speichern
       LD
              (HL),A
       DJNZ
             ADDIER SCHLEIFE;
                              Zähler (B) dekrementieren
       POP
                           ;bis Zähler=0
       RET
:Unterprogramm SCHIEBE
       schiebt BIN REG1 eine Binärstelle nach links
      *******************
SCHIEBE:
       PUSH
             BC
                                  ; Vorbereitung:
       LD
             HL, BIN REG1 + BIN BYTE
                                  ;Pointer und
       LD
             B, BIN BYTE
                                  :Zähler und
       AND
                                  ; Carry wie b. ADDIERE
SCHIEBE SCHLEIFE:
                                  ;wiederhole:
      DEC
             HL
                                     Zeiger dekrement.,
       RL
             (HL)
                                     byteweise schieben
       DJNZ
             SCHIEBE SCHLEIFE
                                     Zähler (B) dekrem.
       POP
                                  ;bis Zähler=0
      RET
************************
;Unterprogramm CLEAR
       löscht eine Binärzahl
 ***************************
CLEAR:
      PUSH
             BC
             B, BIN BYTE
      LD
                           ; Zähler auf Anzahl Byte
CLEAR1:
                           ;wiederhole:
      LD
             (HL),0
                             ein Byte löschen,
      INC
             HL
                              Zeiger inkrementieren,
      DJNZ
             CLEAR1
                              Zähler (B) dekrementieren
      POP
                           ;bis Zähler=0
      RET
```

```
*******************
;Unterprogramm MOV12
       kopiert BIN REG2 nach BIN REG1
   ****************
MOV12:
       PUSH
              BC
                             ;Faktor (für MULT!) retten
       LD
              DE, BIN REG1
                            ;Zie1 = BIN REG1
                            ;Quelle = BIN REG2
       LD
              HL, BIN REG2
       LD
              BC, BIN BYTE
                             ;Anzahl Byte = BIN BYTE
       LDIR
                             ;und schieben
       POP
              BC
       RET
******************
;Unterprogramm WANDLE DEZ
       wandelt BIN REG1 in eine Dezimalzahl (nach DEZ REG)
 ***************
WANDLE DEZ:
       ĹD
              HL, DEZ REG
                             ;alte Dezimalzahl löschen:
       LD
              B, DEZ STELL/2
CLEAR DEZ:
                             :wiederhole:
              (HL),0
                               ein Byte löschen
       LD
       INC
                               Zeiger inkrementieren
              HL
       DJNZ
              CLEAR DEZ
                               Zähler (B) dekrementieren
                             :bis Zähler=0
              DE, BIN STELL
                             ;alle Binärbits wandeln:
       LD
WANDL SCHLEIFE:
                             ;wiederhole:
       CALL
              SCHIEBE
                               BIN REG1 := BIN REG1*2
                               DEZ REG := DEZ REG*2
       LD
              B,DEZ_STELL/2
                                            + Carry
WANDL SCHLEIFE1:
                               wiederhole:
       DEC
              HL
                                 Zeiger dekrementieren
       LD
              A,(HL)
                                 Byte laden
       ADC
              A,(HL)
                                 mit Carry addieren
       DAA
                                 dezimal korrigieren!
       LD
                                 speichern
              (HL),A
       DJNZ
              WANDL SCHLEIFE1
                                 Zähler (B) dekrement.
                               bis Zähler=0
       DEC
              DΕ
                               Zähler (DE) dekremen.
       LD
              A,D
       OR
                             ;bis Zähler=0
              Ε
       JR
              NZ, WANDL SCHLEIFE
       RET
```

```
***************
 ;Unterprogramm AUSGABE
 ********************
FLAG
       EQU
               0
AUSGABE:
       RES
               FLAG, C
                             ;Flag löschen,
       LD
               HL, DEZ REG
                             ; Zeiger und
               B, DEZ STELL/2
       LD
                             ; Zähler setzen
AUSGABE SCHLEIFE:
                             ;wiederhole:
       RLD
                                Dezimaldigit (Halbbyte)
       CALL
               EINE STELLE
                                holen und ausgeben
       RLD
                                dto. für anderes Halb-
       CALL
               EINE STELLE
                                byte (Digit)
       INC
                                Zeiger auf nächstes Byte
       DJNZ
               AUSGABE SCHLEIFE;
                                Zähler (B) dekrementieren
       RET
                             ;bis Zähler=0
EINE STELLE:
                             ;Digit ausgeben:
       AND
               0FH
                             ;rechtes Halbbyte maskieren
       JR
              NZ, SCHREIBE
                             ;Digit=0? nein: ausgeben
       BIT
              FLAG, C
                                ja: Flag gesetzt?
       RET
               Z
                                  nein: zurück (führende
                                     Nullen unterdrücken)
                                  ja: ausgeben (normale
                                                   Nu11)
SCHREIBE:
       SET
              0,C
                             ;Flag setzen
       PUSH
               BC
                             ;Zähler und
       PUSH
              HL
                             ; Zeiger retten,
       ADD
              A, '0'
                             ;Digit -> ASCII
       LD
              E,A
       LD
              C, CONAUSF
                             ;ausgeben
       CALL
              SYSTEM
                             ;mit Systemfunktion
       POP
              HL
                             ; Zeiger und
       POP
              BC
                             ; Zähler wiederherstellen
       RET
       END
              START
>asm
Z80-Assembler
pass 1:
pass 2:
no fatal error(s)
SUM=5F8F
```

CRC=C2B9

## TITLE Primzahlen - Programm 1

```
****************
;Definition von Systemadressen
*****************
SYSTEM EOU
          00005H
                     ;Adresse für Aufruf von
                     ;Systemfunktionen
; Definition von Systemfunktionen
****************
KONAUSF EQU
          2
                     ; Konsol-Ausgabe
DIRCONF EQU
          6
                     ;Direkte Konsol-Ein/Ausgabe
STRAUSF EQU
                     ;Stringausgabefunktion
*****************
; Definition von ASCII-Konstanten
*****************
CR
     EQU
          ODH
                     ;Cursor an Zeilenanfang
                     ;(Carriage Return)
LF
     EQU
          OAH
                     ;Cursor in neue Zeile
                     ;(Line Feed)
CTRL C
          'C' AND 1FH
    EOU
                     ;für Abbruchbedingung
;Definition von TRUE/FALSE (WAHR/FALSCH)
*****************
FALSE
     EQU
TRUE
     EQU
          NOT FALSE
******************
;Sonstige Konstanten
******************
          EQU
                10
DEZ STELL
                     ;maximal 10 Dezimalstellen
                     ;2 \text{ hoch } 32 = 4*10 \text{ hoch } 9
BIN STELL
                     ;32-Bit Binärzahl
          EQU
                32
BIN BYTE
          EQU
                BIN STELL/8
                     ; Anzahl Bytes d. Binärzahlen
```

```
***************
; Hauptprogramm PRIMZAHLEN
***************
START:
      LD
              HL, PRIM
                            ; Setze PRIM = 1
      CALL
              SETZE1
SCHLEIFE:
                            ;wiederhole:
                              wiederhole:
              HL, PRIM
      LD
       CALL
              PLUS2
                                erhöhe PRIM um 2
       CALL
              PRUEFE
                                Prüfe ob PRIM Primzahl
       JR
                              bis Primzahl gefunden
              NC, SCHLEIFE
       CALL
                              Wandle PRIM in Dezimalzahl
              WANDLE DEZ
       CALL
              AUSGABE
                              Gib PRIM auf Konsole aus
      LD
                              Gib CR/LF auf Konsole aus
              DE, NEUE ZEILE
       LD
              C, STRAUSF
       CALL
              SYSTEM
       LD
              C, DIRCONF
       LD
              E, OFFH
              SYSTEM
       CALL
              CTRL C
       CP
              NZ, SCHLEIFE
       JR
       JP
                            ;bis Abbruch mit CTRL-C
NEUE ZEILE:
              CR, LF, '$'
 ********************
;Unterprogramm PRUEFE
       Prüft, ob PRIM eine Primzahl ist
******************
PRUEFE:
              HL, TEILER
       LD
                            ;Setze TEILER = 1
       CALL
              SETZE1
PRUEFE SCHL:
                            ;wiederhole:
      LD
              HL, TEILER
       CALL
              PLUS2
                              erhöhe TEILER um 2
       CALL
              DIV
                              Dividiere PRIM/TEILER
                              Ergebnis: QUOTIENT, REST
       LD
              HL, REST
                            ;bis REST gleich Null
       CALL
              NULL
                               --> keine Primzahl
       RET
              Z
                              --> Carry=0
```

```
:oder TEILER >= QUOTIENT
       LD
               IX, BIN REG
                                  TEILER - QUOTIENT
               DE, TEILER
       LD
                                  Carry=Vorzeichen Ergebnis
       LD
               HL, QUOTIENT
                                  Carry=1: TEILER<QUOTIENT
       CALL
               SUB
                                  Carry=0: TEILER>=QUOTIENT
                                  --> Primzahl
       JR
               C, PRUEFE SCHL
                                  --> Carry:=1
       SCF
       RET
 **********************
:Unterprogramm DIV
       dividiert PRIM durch TEILER
        Ergebnis in QUOTIENT, Rest in REST
            *****************
DIV:
                                ;Setze ZAEHL = 1
        LD
               B, 1
                                ;wiederhole:
DIV SCHL1:
                                   erhöhe ZAEHL um 1
        INC
               В
               HL, TEILER
       LD
                                   schiebe TEILER links
        CALL
                SCHIEBE
        LD
                IX, BIN REG
        LD
                DE, PRIM
        LD
                HL, TEILER
        CALL
                SUB
        JR
                NC, DIV SCHL1
                                ;bis TEILER > PRIM
        LD
                HL, PRIM
        LD
                DE, REST
        CALL
                MOV
                                ;Kopiere PRIM nach REST
                HL, QUOTIENT
        LD
                SETZE1
        CALL
                                ;Setze QUOTIENT gleich Null
        DEC
                (HL)
DIV SCHL2:
                                :wiederhole:
                                   BIN REG :=
                IX, BIN REG
        LD
                                   REST
        LD
                DE, REST
        LD
                HL, TEILER
                                   -TEILER
                                   Ergebnis positiv?
        CALL
                SUB
        LD
                HL, BIN REG
        LD
                DE, REST
                                     ja: BIN REG nach REST
        CALL
                NC, MOV
                                   komplementiere Carry-Flag
        CCF
        LD
                HL, QUOTIENT
        CALL
                SCHIEBE CARRY
                                   Carry-Flag in QUOTIENT
                HL, TEILER
        LD
                SCHIEBE RECHTS
                                   schiebe TEILER rechts
        CALL
                DIV_SCHL2
                                   dekrementiere ZAEHL (B)
        DJNZ
                                ;bis ZAEHL=0
        LD
                HL, TEILER
        CALL
                SCHIEBE CARRY
                                :Teiler wiederherstellen
        RET
```

```
******************
:Unterprogramm SUB
      addiert Register (HL) von Register (DE), Ergebnis
      nach Register (IX)
SUB:
      PUSH
             BC
             BC, BIN_BYTE
      LD
      EX
             DE,HL
      ADD
             HL, BC
                           ;alle Zeiger ans Registerende
             DE, HL
                           ;(+BIN BYTE)
      EX
      ADD
             HL, BC
      ADD
             IX,BC
                           ;Zähler setzen und
      LD
             B,C
      AND
             Α
                           ;Carry löschen
SUB SCHLEIFE:
                           ;wiederhole:
      DEC
             DΕ
      DEC
             HL
      DEC
             IX
                              Zeiger dekrement.,
                           ; Byte laden,
      LD
             A,(DE)
      SBC
             A,(HL)
                           ; subtrahieren und
             (IX),A
                           ; Ergebnis speichern
      LD
      DJNZ
             SUB SCHLEIFE
                           ; Zähler (B) dekrem.
      POP
             BC
                           ;bis Zähler=0
      RET
• ***********************
;Unterprogramm SETZE1
      setzt REGISTER (HL) auf 1
SETZE1:
      PUSH
             BC
      LD
             B, BIN BYTE
                           ; Zähler auf Anzahl Byte
SETZE1 SCHLEIFE:
                           ;wiederhole:
             (HL), 0
      LD
                              ein Byte löschen,
      INC
             HL
                              Zeiger inkrementieren,
             SETZE1 SCHLEIFE;
      DJNZ
                              Zähler (B) dekrementieren
                           ;bis Zähler=0
      DEC
             HL
                           ; Zeiger inkrementieren
      INC
             (HL)
                           ;und Byte auf 1 erhöhen
      POP
             BC
      RET
********************
;Unterprogramm PLUS2
      addiert 2 zu Register (HL)
 **********************
PLUS2:
      PUSH
             BC
      LD
             BC, BIN BYTE
                           ; Zeiger ans Registerende
```

```
;(+BIN BYTE) (kein Carry!)
       ADD
               HL, BC
                              ; Zähler setzen
               B,C
       LD
                              :Summand = 2
               A, 2
       LD
                              :wiederhole:
PLUS2 SCHLEIFE:
                                 Zeiger dekrementieren
               HL
       DEC
                                 ein Byte addieren
       ADC
               A, (HL)
                                 und wieder speichern
       LD
               (HL),A
                                 Akku=0 für folgende Bytes
               A,0
       LD
                                 Zähler (B) dekrementieren
       DJNZ
               PLUS2 SCHLEIFE
                               :bis Zähler=0
       POP
       RET
*******************
;Unterprogramme SCHIEBE, SCHIEBE CARRY
       schiebt Register (HL) eine Binärstelle nach links
*****************
SCHIEBE CARRY:
               BC
       . PUSH
       PUSH
               AF
                               ;Carry retten
       LD
               BC, BIN BYTE
                               ; Zeiger ans Registerende
       ADD
               HL, BC
                               ;(+BIN BYTE)
       P<sub>O</sub>P
               AF
                               ;Carry restaurieren
                               ;und schieben
               SCHIEBE1
        JR
SCHIEBE:
       PUSH
               BC
        LD
               BC, BIN BYTE
                               ; Zeiger ans Registerende
                               ;(+BIN BYTE) (kein Carry!)
        ADD
               HL, BC
SCHIEBE1:
                               ; Zähler (B) setzen
       LD
               B,C
                               ;wiederhole:
SCHIEBE SCHL:
        DEC
                                  Zeiger dekrement.,
               HL
                                 byteweise schieben
        RL
               (HL)
               SCHIEBE SCHL
                                  Zähler (B) dekrem.
        DJNZ
                               ;bis Zähler=0
        POP
        RET
 ***************
;Unterprogramm SCHIEBE RECHTS
        schiebt Register (HL) eine Binärstelle nach rechts
SCHIEBE RECHTS:
        PUSH
               BC
        LD
               B, BIN BYTE
                               ; Zähler setzen
        AND
                               ;Carry löschen
                               ;wiederhole:
SCHIEBE R:
        RR
               (HL)
                                 byteweise schieben
        INC
                                  Zeiger inkrement.,
               HL
                SCHIEBE R
                                  Zähler (B) dekrem.
        DJNZ
                               ;bis Zähler=0
        POP
        RET
```

```
**********************
:Unterprogramm MOV
       kopiert Register (HL) nach Register (DE)
        ********************
MOV:
       PUSH
               BC
               BC, BIN BYTE
       LD
                              ;Anzahl Byte = BIN BYTE
       LDIR
                              ;und schieben
       POP
               BC
       RET
;Unterprogramm NULL
       Prüft, ob Register (HL) gleich Null
NULL:
       PUSH
               BC
       LD
               B, BIN BYTE
                              ;Anzahl Byte = BIN BYTE
       XOR
                              ;A=0
NULL SCHLEIFE:
                              :wiederhole:
                                 ODER-Verkn mit Byte
       OR
               (HL)
       INC
               HL
                                 Zeiger inkrementieren
                                 Zähler (B) dekrement.
       DJNZ
               NULL SCHLEIFE
                              :bis Zähler=0
       AND
                              ;Bedingungsflags setzen
               Α
               BC
       POP
       RET
;Unterprogramm WANDLE DEZ
       wandelt PRIM in eine Dezimalzahl (nach DEZ REG)
******************
WANDLE DEZ:
       LD
               HL, PRIM
       LD
               DE, BIN REG
       CALL
               MOV
                              ;PRIM nach BIN REG
       LD
               HL, DEZ REG
                              ;alte Dezimalzahl löschen:
       LD
               B DEZ STELL/2
CLEAR DEZ:
                              :wiederhole:
       LD
               (HL),0
                                 ein Byte löschen
       INC
               HL
                                 Zeiger inkrementieren
       DJNZ
               CLEAR DEZ
                                 Zähler (B) dekrementieren
                              :bis Zähler=0
       LD
                              ;alle Binärbits wandeln:
               DE, BIN STELL
WANDL SCHLEIFE:
                              ;wiederhole:
       LD
               HL, BIN REG
       CALL
               SCHIEBE
                                 BIN REG := BIN REG*2
                                 DEZ REG := DEZ REG*2
       LD
               B, DEZ STELL/2
                                              <del>T</del> Carry
```

```
5
```

```
wiederhole:
WANDL SCHLEIFE1:
                                    Zeiger dekrementieren
       DEC
               HL
                                    Byte laden
       LD
               A,(HL)
                                    mit Carry addieren
       ADC
               A,(HL)
                                    dezimal korrigieren!
        DAA
                                    speichern
        LD
                (HL), A
                                    Zähler (B) dekrement.
               WANDL SCHLEIFE1
        DJNZ
                                  bis Zähler=0
                                  Zähler (DE) dekremen.
        DEC
               DE
        LD
               A,D
                                :bis Zähler=0
        OR
                E
               NZ, WANDL SCHLEIFE
        JR
        RET
****************
;Unterprogramm AUSGABE
*****************
FLAG
        EQU
                0
AUSGABE:
        RES
                                ;Flag löschen,
                FLAG, C
        LD
                HL, DEZ REG
                                :Zeiger und
                B, DEZ STELL/2
                                ; Zähler setzen
        LD
AUSGABE SCHLEIFE:
                                :wiederhole:
                                  Dezimaldigit (Halbbyte)
        RLD
                                  holen und ausgeben
        CALL
                EINE STELLE
        RLD
                                  dto für anderes Halb-
        CALL
                                  byte (Digit)
                EINE STELLE
                                  Zeiger auf nächstes Byte
        INC
                HL
                                  Zähler (B) dekrementieren
        DJNZ
                AUSGABE SCHLEIFE;
        RET
                                ;bis Zähler=0
EINE STELLE:
                                ;Digit ausgeben:
                                ;rechtes Halbbyte maskieren
        AND
                0FH
        JR
                NZ, SCHREIBE
                                ;Digit=0? nein: ausgeben
        BIT
                FLAG, C
                                   ja: Flag gesetzt?
        RET
                                    nein: zurück (führende
                                        Nullen unterdrücken)
                                    ja: ausgeben (normale
                                                       Nu11)
SCHREIBE:
        SET
                0,C
                                ;Flag setzen
        PUSH
                BC
                                ; Zähler und
        PUSH
                HL
                                ; Zeiger retten,
                                ;Digit -> ASCII
        ADD
                A, 0'
        LD
                E,A
        LD
                C, KONAUSF
                                ;ausgeben
        CALL
                SYSTEM
                                ;mit Systemfunktion
        POP
                                ;Zeiger und
                HL
        POP
                BC
                                ;Zähler wiederherstellen
        RET
```

```
****************
; Datenbereich
********************
DEZ REG:
                      ;Speicher für Dezimalzahl
     DS
          DEZ STELL/2
BIN REG:
     DS
          BIN BYTE
PRIM:
     DS
          BIN BYTE
                      ;Speicher für Binärzahlen
TEILER:
     DS
          BIN BYTE
QUOTIENT:
     DS
          BIN BYTE
REST:
     DS
          BIN BYTE
     END
           START
Z80-Assembler
pass 1:
pass 2:
no fatal error(s)
SUM=817C
CRC=2533
```

## TITLE Primzahlen - Programm 2

```
*******************
;Definition von Systemadressen
SYSTEM EQU
         00005H
                  ;Adresse für Aufruf von
                  ;Systemfunktionen
;Definition von Systemfunktionen
*******************
KONAUSF EQU
         2
                  ;Konsol-Ausgabe
DIRCONF EQU
         6
                  ;Direkte Konsol-Ein/Ausgabe
STRAUSF EQU
         9
                  ;Stringausgabefunktion
```

```
*******************
;Definition von ASCII-Konstanten
.*******************
CR
     EQU
           ODH.
                       ;Cursor an Zeilenanfang
                       ;(Carriage Return)
ĹF
     EQU
           OAH
                       ;Cursor in neue Zeile
                       ;(Line Feed)
           'C' AND 1FH
                       ;für Abbruchbedingung
CTRL C
     EOU
; Definition von TRUE/FALSE (WAHR/FALSCH)
******************
FALSE
     EOU
TRUE
     EQU
           NOT FALSE
*****************
;Sonstige Konstanten
*****************
DEZ STELL
           EQU
                 10
                       ;maximal 10 Dezimalstellen
                       ;2 \text{ hoch } 32 = 4*10 \text{ hoch } 9
BIN STELL
           EQU
                 32
                       ;32-Bit Binärzahl
BIN BYTE
           EQU
                 BIN STELL/8
                       ; Anzahl Bytes d. Binärzahlen
;Hauptprogramm PRIMZAHLEN
*****************
START:
      LD
           HL, PRIM
                       ;Setze PRIM = 1
      CALL
           SETZE1
SCHLEIFEO:
                       ;wiederhole:
           B,7
                          Zähler für 7 Zahlen/Zeile
     LD
SCHLEIFE:
                         wiederhole:
                           wiederhole:
      PUSH
           BC
                            Zähler retten
      LD
           HL, PRIM
      CALL
           PLUS2
                            erhöhe PRIM um 2
           PRUEFE
      CALL
                            Prüfe ob PRIM Primzahl
      POP
           BC
                            Zähler restaurieren
```

4/56 LISTINGS

```
JR
               NC, SCHLEIFE
                               ;
                                    bis Primzahl gefunden
        PUSH
               BC
                                    Zähler retten
       CALL
               WANDLE DEZ
                                    Wandle PRIM in Dezimalzahl
       CALL
                                    Gib PRIM auf Konsole aus
               AUSGABE
       POP
               BC
                                    Zähler restaurieren
       DJNZ
               SCHLEIFE
                                    und dekrementieren
                                  bis Zähler (B) gleich Null
       LD
               DE, NEUE ZEILE
                                  Gib CR/LF auf Konsole aus
               C,STRAUSF
       LD
       CALL
               SYSTEM
       LD
               C, DIRCONF
       LD
               E, OFFH
       CALL
               SYSTEM
               CTRL C
       CP
        JR
               NZ, SCHLE IFEO
        JP
               0
                               ;bis Abbruch mit CTRL-C
NEUE ZEILE:
       DB
               CR, LF, '$'
******************
;Unterprogramm PRUEFE
       Prüft, ob PRIM eine Primzahl ist
   ******************
PRUEFE:
       LD
               HL, TEILER
                               ;Setze TEILER = 1
       CALL
               SETZE1
PRUEFE SCHL:
                               ;wiederhole:
       LD
               HL, TEILER
       CALL
                                  erhöhe TEILER um 2
               PLUS2
       CALL
               DIV
                                  Dividiere PRIM/TEILER
                                  Ergebnis: QUOTIENT, REST
       LD
               HL, REST
                               ;bis REST gleich Null
       CALL
               NULL
                                  --> keine Primzahl
       RET
               Z
                                  --> Carry=0
       LD
               IX.BIN REG
                               ;oder TEILER >= QUOTIENT
                                  TEILER - QUOTIENT
       LD
               DE, TEILER
       LD
               HL, QUOTIENT
                                  Carry=Vorzeichen Ergebnis
       CALL
               SUB
                                  Carry=1: TEILER<QUOTIENT
                                  Carry=0: TEILER>=QUOTIENT
       JR
               C, PRUEFE SCHL
                                  --> Primzahl
       SCF
                                  --> Carry:=1
```

RET

```
55
```

```
****************
;Unterprogramm DIV
       dividiert PRIM durch TEILER
       Ergebnis in QUOTIENT, Rest in REST
 *********************
DIV:
               B, 1
       LD
                               ; Setze ZAEHL = 1
DIV SCHL1:
                               ;wiederhole:
               В
       INC
                                  erhöhe ZAEHL um 1
       LD
               HL, TEILER
       CALL
               SCHIEBE
                                  schiebe TEILER links
       LD
               IX, BIN REG
       LD
               DE, PRIM
       LD
               HL, TEILER
       CALL
               SUB
       JR
               NC, DIV SCHL1
                               ;bis TEILER > PRIM
       LD
               HL, PRIM
       LD
               DE, REST
       CALL
               MOV
                               ;Kopiere PRIM nach REST
       LD
               HL, QUOTIENT
       CALL
               SETZE1
       DEC
               (HL)
                               ;Setze QUOTIENT gleich Null
DIV_SCHL2:
                               ;wiederhole:
       LD
               IX, BIN REG
                                  BIN REG :=
       LD
               DE, REST
                                  REST
       LD
               HL, TEILER
                                  -TEILER
       CALL
               SUB
                                  Ergebnis positiv?
       LD
               HL, BIN REG
               DE, REST
       LD
       CALL
               NC, MOV
                                    ja: BIN REG nach REST
       CCF
                                  komplementiere Carry-Flag
       LD
               HL, QUOTIENT
       CALL
               SCHIEBE CARRY
                                  Carry-Flag in QUOTIENT
       LD
               HL, TEILER
       CALL
               SCHIEBE RECHTS
                                  schiebe TEILER rechts
       DJNZ
               DIV SCHL2
                                  dekrementiere ZAEHL (B)
                               ;bis ZAEHL=0
       LD
               HL, TEILER
       CALL
               SCHIEBE CARRY
                               ;Teiler wiederherstellen
       RET
```

```
F
```

```
;Unterprogramm SUB
      addiert Register (HL) von Register (DE), Ergebnis
      nach Register (IX)
  ******************
SUB:
             BC
      PUSH
             BC, BIN BYTE
      LD
      EX
             DE, HL
      ADD
             HL, BC
                           ;alle Zeiger ans Registerende
      EX
             DE, HL
                           ;(+BIN BYTE)
      ADD
             HL, BC
             IX,BC
                           ;Zähler setzen und
      ADD
      LD
             B,C
      AND
                           ;Carry löschen
             Α
                           ;wiederhole:
SUB SCHLEIFE:
      DEC
             DE
      DEC
             HL.
      DEC
             IX
                             Zeiger dekrement.,
             A,(DE)
                             Byte laden,
      LD
                             subtrahieren und
      SBC
             A,(HL)
                             Ergebnis speichern
      LD
             (IX),A
             SUB SCHLEIFE
                             Zähler (B) dekrem.
      DJNZ
                           ;bis Zähler=0
      POP
             BC
      RET
*******************
;Unterprogramm SETZE1
      setzt REGISTER (HL) auf 1
***************
SETZE1:
      PUSH
             BC
                           ;Zähler auf Anzahl Byte
      LD
             B,BIN BYTE
SETZE1 SCHLEIFE:
                           ;wiederhole:
             (HL),0
      LD
                             ein Byte löschen
       INC
                              Zeiger inkrementieren,
      DJNZ
             SETZE1 SCHLEIFE ;
                             Zähler (B) dekrementieren
                           ;bis Zähler=0
      DEC
                           ; Zeiger inkrementieren
             HL
       INC
             (HL)
                           ;und Byte auf 1 erhöhen
      POP
             BC
      RET
*******************
;Unterprogramm PLUS2
       addiert 2 zu Register (HL)
PLUS2:
      PUSH
             BC
             BC, BIN BYTE
                           ; Zeiger ans Registerende
      LD
```

RET

```
ADD
              HL, BC
                             ;(+BIN BYTE) (kein Carry!)
       LD
              B,C
                             ; Zähler setzen
       LD
              A, 2
                             :Summand = 2
PLUS2 SCHLEIFE:
                             ;wiederhole:
       DEC
              HL
                               Zeiger dekrementieren
       ADC
              A,(HL)
                               ein Byte addieren
                               und wieder speichern
       LD
              (HL),A
                               Akku=0 für folgende Bytes
       LD
              A, 0
       DJNZ
              PLUS2 SCHLEIFE
                               Zähler (B) dekrementieren
       POP
                             :bis Zähler=0
       RET
 ******************
;Unterprogramme SCHIEBE, SCHIEBE CARRY
       schiebt Register (HL) eine Binärstelle nach links
SCHIEBE CARRY:
       PUSH
              BC
              AF
       PUSH
                             ;Carry retten
              BC, BIN BYTE
       LD
                             ; Zeiger ans Registerende
       ADD
              HL, BC
                             :(+BIN BYTE)
       P<sub>O</sub>P
              AF
                             ;Carry restaurieren
       JR
              SCHIEBE1
                             ;und schieben
SCHIEBE:
       PUSH
              BC
       LD
              BC, BIN BYTE
                             ; Zeiger ans Registerende
       ADD
              HL, BC
                             ;(+BIN BYTE) (kein Carry!)
SCHIEBE1:
                             ;Zähler (B) setzen
       LD
              B,C
SCHIEBE SCHL:
                             ;wiederhole:
       DEC
              HL
                               Zeiger dekrement.,
       RL
              (HL)
                               byteweise schieben
       DJNZ
              SCHIEBE SCHL
                               Zähler (B) dekrem.
       POP
                             ;bis Zähler=0
       RET
;Unterprogramm SCHIEBE RECHTS
       schiebt Register (HL) eine Binärstelle nach rechts
 ********************
SCHIEBE RECHTS:
       PUSH
              BC
              B, BIN BYTE
       LD
                             ; Zähler setzen
       AND
                             ;Carry löschen
SCHIEBE R:
                             :wiederhole:
       RR
              (HL)
                               byteweise schieben
       INC
              HL
                               Zeiger inkrement.
       DJNZ
              SCHIEBE R
                               Zähler (B) dekrem.
       POP
                             ;bis Zähler=0
```

```
*******************
;Unterprogramm MOV
       kopiert Register (HL) nach Register (DE)
 ******************
MOV:
       PUSH
              BC
              BC, BIN BYTE
       LD
                            ;Anzahl Byte = BIN BYTE
                            ;und schieben
       LDIR
       POP
              BC
       RET
;Unterprogramm NULL
       Prüft, ob Register (HL) gleich Null
*****************
NULL:
       PUSH
              BC
       LD
              B, BIN BYTE
                            ;Anzahl Byte = BIN BYTE
       XOR
                            ;A=0
NULL SCHLEIFE:
                            ;wiederhole:
       OR
              (HL)
                               ODER-Verkn. mit Byte
       INC
              HL
                               Zeiger inkrementieren
      DJNZ
              NULL SCHLEIFE
                               Zähler (B) dekrement.
                            ;bis Zähler=0
       AND
                            ;Bedingungsflags setzen
              Α
       POP
              BC
       RET
;Unterprogramm WANDLE DEZ
       wandelt PRIM in eine Dezimalzahl (nach DEZ REG)
WANDLE DEZ:
      LD
              HL, PRIM
       LD
              DE, BIN REG
       CALL
              MOV
                            ;PRIM nach BIN REG
              HL, DEZ REG
      LD
                            ;alte Dezimalzahl löschen:
       LD
              B, DEZ STELL/2
CLEAR DEZ:
                            ;wiederhole:
       LD
              (HL), 0
                               ein Byte löschen
       INC
                               Zeiger inkrementieren
       DJNZ
                               Zähler (B) dekrementieren
              CLEAR DEZ
                            ;bis Zähler=0
      LD
              DE, BIN STELL
                            ;alle Binärbits wandeln:
WANDL SCHLEIFE:
                            :wiederhole:
       LD
              HL, BIN REG
       CALL
              SCHIEBE
                               BIN REG := BIN REG*2
                               DEZ REG := DEZ REG*2
       LD
              B,DEZ STELL/2
                                           + Carry
```

```
wiederhole:
WANDL SCHLEIFE1:
                                    Zeiger dekrementieren
       DEC
               HL
                                    Byte laden
               A,(HL)
       LD
                                    mit Carry addieren
       ADC
               A_{\bullet}(HL)
                                    dezimal korrigieren!
       DAA
       LD
               (HL),A
                                    speichern
        DJNZ
               WANDL SCHLEIFE1
                                    Zähler (B) dekrement.
                                  bis Zähler=0
                                  Zähler (DE) dekremen.
        DEC
               DE
        LD
               A, D
        OR
                                ;bis Zähler=0
               Ε
               NZ, WANDL SCHLEIFE
        JR
        RET
****************
:Unterprogramm AUSGABE
*****************
FLAG
        EQU
               0
AUSGABE:
        RES
               FLAG, C
                                ;Flag löschen,
        LD
               HL, DEZ REG
                                :Zeiger und
                B, DEZ STELL/2
                                ; Zähler setzen
        LD
AUSGABE SCHLEIFE:
                                ;wiederhole:
        RLD
                                  Dezimaldigit (Halbbyte)
        CALL
                EINE STELLE
                                  holen und ausgeben
        RLD
                                  dto. für anderes Halb-
        CALL
                EINE STELLE
                                  byte (Digit)
                                   Zeiger auf nächstes Byte
        INC
               HL
        DJNZ
                AUSGABE SCHLEIFE;
                                   Zähler (B) dekrementieren
        RET
                                ;bis Zähler=0
EINE STELLE:
                                ;Digit ausgeben:
        AND
                0FH
                                ;rechtes Halbbyte maskieren
        JR
                                ;Digit=0? nein: ausgeben
                NZ, SCHREIBE
        BIT
                FLAG, C
                                   ja: Flag gesetzt?
        JR
                                     ja: ausgeben (normale
                NZ, SCHREIBE
                                                        Nu11)
                E,' '
        LD
                                    nein: Zwischenraum aus-
        PUSH
                BC
                                        geben
        PUSH
               HL
        LD
                C, KONAUSF
        CALL
                SYSTEM
        POP
                HL
        POP
                BC
        RET
SCHREIBE:
        SET
                0,C
                                ;Flag setzen
        PUSH
                BC
                                ; Zähler und
        PUSH
                HL
                                ; Zeiger retten
                A, '0'
        ADD
                                ;Digit -> ASCII
```

```
LD
            E,A
      LD
            C, KONAUSF
                         ;ausgeben
      CALL
            SYSTEM
                         ;mit Systemfunktion
      POP
            HL
                         ; Zeiger und
      PC<sub>P</sub>
            BC
                         ; Zähler wiederherstellen
      RET
******************
;Datenbereich
***************
DEZ REG:
      DS
            DEZ STELL/2
                        ;Speicher für Dezimalzahl
BIN REG:
            BIN BYTE
      DS
PRIM:
                        ;Speicher für Binärzahlen
      DS
            BIN BYTE
TEILER:
            BIN BYTE
      DS
QUOTIENT:
      DS
            BIN BYTE
REST:
      DS
            BIN BYTE
            START
      END
Z80-Assembler
pass 1:
pass 2:
no fatal error(s)
SUM=8D36
CRC=1EDB
```

Str.

```
. ***********************
;Definition von Systemfunktionen
**************
DRUCKF EQU
            5
                        ;Drucker-Ausgabe
Definition von ASCII-Konstanten
******************
CR
      EQU
            0DH
                        ;Cursor an Zeilenanfang
                        ;(Carriage Return)
      EOU
            0AH
LF
                        ;Cursor in neue Zeile
                        ;(Line Feed)
******************
; Definition von TRUE/FALSE (WAHR/FALSCH)
*******************
FALSE
      EOU
            0
TRUE
      EOU
            NOT FALSE
******************
;Sonstige Konstanten
DEZ STELL
            EQU
                        ;maximal 10 Dezimalstellen
                        ;2 \text{ hoch } 32 = 4*10 \text{ hoch } 9
            EQU
                  32
                        ;32-Bit Binärzahl
BIN STELL
BIN BYTE
            EQU
                  BIN STELL/8
                        ; Anzahl Bytes d. Binärzahlen
**********************
;Hauptprogramm für Multitasking
      ändert BIOS-Einsprung für Konsolstatus und -eingabe
    **********************
HAUPT:
      LD
            HL, (WSTART+1)
                        ;Adresse Warmstart in BIOS-
      REPT
            3
                        ;tabelle + 3 = Adresse
      INC
            HL
                        ; Konsolstatus
      ENDM
      LD
            DE, BIOS ST
                        ; Zeiger DE auf interne
                        :Sprungtabelle
      LD
                        ; Zähler für 6 Byte setzen
            B, 6
HAUPT SCHLEIFE:
                        ;wiederhole:
            C,(HL)
      LD
                          Byte aus BIOS-Tabelle
      LD
            A_{\bullet}(DE)
                          merken, ersetzen durch
      LD
            (HL),A
                          Byte aus interner Tab.
      LD
                          und in interner Tab.
            A,C
      LD
            (DE),A
                          speichern
      INC
            DE
                          Zeiger für interne und
      INC
            HL
                          Biostabelle erhöhen
      DJNZ
            HAUPT SCHLEIFE
                          Zähler (B) dekrementieren
                        ;bis Zähler=0
```

```
LD
                SP, STACK
                                ;Stack und Stackpointer für
                                :Primzahlenprogramm vorber-
                                :reiten:
        LD
               HL, START
                                :Startadresse auf Stack
        PUSH
               HL
                                   (RET=Sprung nach START!)
        PUSH
                AF
                                ;4 Registerpaare auf Stack
        PUSH
                BC
        PUSH
                DE
        PUSH
               HL
        LD
                (SP PRIM), SP
                                :Stackpointer ablegen
        JP
                WSTART
                                :Warmstart
*********************
;Unterprogramme STATUS und INPUT
        ersetzen BIOS-PROGRAMME für Konsolstatus und -eingabe
************************
BIOS ST:
                                ;Diese beiden Sprungbefehle
               STATUS
        JP
                                ;werden von Haupt gegen die
BIOS IN:
                                ;beiden der BIOS-Sprung-
        JP
                INPUT
                                ;tabelle getauscht
STATUS:
        CALL
               BIOS ST
                                ;Konsolstatus abfragen
        OR
               Α
                                ; Rücksprung wenn Zeichen
        RET
               NZ
                                :bereit
TRANSFER PRIM:
                                ;sonst Primzahlenprogramm:
       PUSH
               BC
                                ;Register retten
       PUSH
               DE
        PUSH
               HL
        PUSH
               IX
        LD
                (SP SYS),SP
                               ;Stackpointer (System) retten
       LD
               SP, (SP PRIM)
                               ;Stackpointer (PRIM) holen
        POP
               HL
                               ;4 Registerpaare laden
       POP
               DE
       POP
               BC
       POP
               AF
       RET
                               ;und Rücksprung an letzte
                               ;Adresse
TRANSFER SYS:
       PUSH
               AF
                               ;4 Registerpaare
       PUSH
               BC
                               ;für PRIM retten
       PUSH
               DE
       PUSH
               HL
       LD
               (SP PRIM), SP
                               ;dto. Stackpointer für PRIM
       LD
               SP, (SP SYS)
                               ;Stackpointer für System
       POP
               IX
       POP
               HL
                               ;und Register für System
       POP
               DE
                               ;holen
       POP
               BC
       XOR
               Α
                               ;=keine Taste bereit
       RET
                               ;zurück ins System
```

```
INPUT:
                             ;wiederhole:
       CALL
              STATUS
                             ; Status abfragen (mit PRIM)
                             ;bis Zeichen bereit
       JR
              Z.INPUT
       JP
              BIOS IN
                             :Zeichen holen
; Hauptprogramm PRIMZAHLEN
*****************
START:
       LD
              HL, PRIM
                             ;Setze PRIM = 1
       CALL
              SETZE1
SCHLEIFEO:
                             ;wiederhole:
       LD
              B,7
                                Zähler für 7 Zahlen/Zeile
SCHLEIFE:
                                wiederhole:
                                  wiederhole:
       PUSH
              BC
                                    Zähler retten
       LD
              HL, PRIM
       CALL
                                    erhöhe PRIM um 2
              PLUS2
       CALL
              PRUEFE
                                    Prüfe ob PRIM Primzahl
       POP
              BC
                                    Zähler restaurieren
       JR
              NC, SCHLEIFE
                                  bis Primzahl gefunden
       PUSH
              BC
                                  Zähler retten
       CALL
                                  Wandle PRIM in Dezimalzahl
              WANDLE DEZ
       CALL
              AUSGABE
                                  Gib PRIM auf Drucker aus
       POP
              BC
                                  Zähler restaurieren
       DJNZ
              SCHLEIFE
                                  und dekrementieren
                                bis Zähler (B) gleich Null
       LD
              E,CR
                                Gib CR/LF auf Drucker aus
       LD
              C, DRUCKF
       CALL
              SYSTEM
       LD
              E,LF
       LD
              C, DRUCKF
       CALL
              SYSTEM
       JR
              SCHLEIFE0
                             ;bis RESET
```

```
;Unterprogramm PRUEFE
       Prüft, ob PRIM eine Primzahl ist
PRUEFE:
       LD
              HL, TEILER
                             :Setze TEILER = 1
       CALL
              SETZE1
PRUEFE SCHL:
                             ;wiederhole:
       LD
              HL, TEILER
       CALL
              PLUS2
                                erhöhe TEILER um 2
       CALL
              DIV
                                Dividiere PRIM/TEILER
                                Ergebnis: QUOTIENT, REST
                             ;bis REST gleich Null
       LD
              HL.REST
       CALL
                                --> keine Primzahl
              NULL
       RET
                                --> Carry=0
              Ζ
       LD
                             ;oder TEILER >= QUOTIENT
              IX, BIN REG
       LD
              DE, TEILER
                                TEILER - QUOTIENT
              HL, QUOTIENT
       LD
                                Carry=Vorzeichen Ergebnis
       CALL
              SUB
                                Carry=1: TEILER<QUOTIENT
                                Carry=0: TEILER>=QUOTIENT
                               --> Primzahl
       JR
              C, PRUEFE SCHL
       SCF
                                --> Carry:=1
       RET
**********************
;Unterprogramm DIV
       dividiert PRIM durch TEILER
       Ergebnis in QUOTIENT, Rest in REST
******************
DIV:
       LD
              B, 1
                             :Setze ZAEHL = 1
DIV SCHL1:
                             ;wiederhole:
       INC
              В
                                erhöhe ZAEHL um 1
       LD
              HL, TEILER
       CALL
              SCHIEBE
                                schiebe TEILER links
       LD
              IX, BIN REG
       LD
              DE, PRIM
       LD
              HL, TEILER
       CALL
              SUB
       JR
              NC, DIV SCHL1
                             ;bis TEILER > PRIM
       LD
              HL, PRIM
       LD
              DE, REST
       CALL
              MOV
                             ;Kopiere PRIM nach REST
              HL, QUOTIENT
       LD
       CALL
              SETZE1
       DEC
              (HL)
                             ;Setze QUOTIENT gleich Null
```

```
6
```

```
DIV SCHL2:
                             ;wiederhole:
       LD
              IX, BIN REG
                                BIN REG :=
       LD
                                REST
              DE, REST
       LD
                                -TEILER
              HL, TEILER
       CALL
              SUB
                                Ergebnis positiv?
       LD
              HL, BIN REG
              DE, REST
       LD
       CALL
              NC, MOV
                                  ja: BIN REG nach REST
       CCF
                                komplementiere Carry-Flag
       LD
              HL, QUOTIENT
       CALL
              SCHIEBE CARRY
                                Carry-Flag in QUOTIENT
       LD
              HL, TEILER
       CALL
              SCHIEBE RECHTS
                                schiebe TEILER rechts
              DIV SCHL2
                                dekrementiere ZAEHL (B)
       DJNZ
                             ;bis ZAEHL=0
       LD
              HL, TEILER
       CALL
              SCHIEBE CARRY
                             ;Teiler wiederherstellen
       RET
****************
;Unterprogramm SUB
       addiert Register (HL) von Register (DE), Ergebnis
       nach Register (IX)
SUB:
       PUSH
              BC
       LD
              BC, BIN BYTE
       EX
              DE,HL
       ADD
              HL, BC
                             ;alle Zeiger ans Registerende
                             ;(+BIN BYTE)
       EX
              DE,HL
       ADD
              HL, BC
              IX,BC
       ADD
                             ; Zähler setzen und
       LD
              B,C
       AND
              Α
                             ;Carry löschen
SUB SCHLEIFE:
                             ;wiederhole:
       DEC
              DE
       DEC
              HL
       DEC
              IX
                                Zeiger dekrement.,
       LD
              A, (DE)
                                Byte laden,
       SBC
              A,(HL)
                                subtrahieren und
               (IX),A
                                Ergebnis speichern
       LD
                                Zähler (B) dekrem.
       DJNZ
              SUB SCHLEIFE
       POP
               BC
                             ;bis Zähler=0
       RET
;Unterprogramm SETZE1
       setzt REGISTER (HL) auf 1
****************
SETZE1:
       PUSH
                             ; Zähler auf Anzahl Byte
       LD
              B, BIN BYTE
```

```
F
```

```
SETZE1 SCHLEIFE:
                               ;wiederhole:
       LD
               (HL),0
                                 ein Byte löschen.
        INC
               HL
                                  Zeiger inkrementieren,
               SETZE1 SCHLEIFE;
                                  Zähler (B) dekrementieren
       DJNZ
                               ;bis Zähler=0
       DEC
               HL
                               ; Zeiger inkrementieren
       INC
               (HL)
                               ;und Byte auf 1 erhöhen
       POP
               BC
       RET
**************************************
;Unterprogramm PLUS2
       addiert 2 zu Register (HL)
 *********************
PLUS2:
       PUSH
               BC
       LD
               BC, BIN BYTE
                               ; Zeiger ans Registerende
       ADD
               HL, BC
                               ;(+BIN BYTE) (kein Carry!)
       LD
               B,C
                               ; Zähler setzen
       LD
               A, 2
                               Summand = 2
PLUS2 SCHLEIFE:
                               ;wiederhole:
       DEC
               HL
                                  Zeiger dekrementieren
       ADC
               A,(HL)
                                 ein Byte addieren
       LD
               (HL),A
                                 und wieder speichern
       LD
               A,0
                                 Akku=0 für folgende Bytes
                                 Zähler (B) dekrementieren
       DJNZ
               PLUS2 SCHLEIFE
       POP
               BC
                               ;bis Zähler=0
       RET
* **********************************
;Unterprogramme SCHIEBE, SCHIEBE CARRY
       schiebt Register (HL) eine Binärstelle nach links
 **************************
SCHIEBE CARRY:
       PUSH
               BC
       PUSH
               AF
                               ;Carry retten
       LD
               BC, BIN BYTE
                               ; Zeiger ans Registerende
       ADD
               HL, BC
                               ;(+BIN BYTE)
       POP
               AF
                               ;Carry restaurieren
               SCHIEBE1
       JR
                               ;und schieben
SCHIEBE:
       PUSH
               BC
       LD
               BC, BIN BYTE
                               ; Zeiger ans Registerende
       ADD
               HL, BC
                               ;(+BIN BYTE) (kein Carry!)
SCHIEBE1:
       LD
               B,C
                               ; Zähler (B) setzen
SCHIEBE SCHL:
                               ;wiederhole:
       DEC
               HL
                                 Zeiger dekrement.,
       RL
               (HL)
                                 byteweise schieben
       DJNZ
               SCHIEBE SCHL
                                 Zähler (B) dekrem.
       POP
                               ;bis Zähler=0
               BC
       CALL
               TRANSFER SYS
       RET
```

RET

```
67
```

```
******************
;Unterprogramm SCHIEBE RECHTS
      schiebt Register (HL) eine Binärstelle nach rechts
******************
SCHIEBE RECHTS:
      PUSH
            BC
      LD
             B, BIN BYTE
                          ; Zähler setzen
                          ;Carry löschen
      AND
             Α
SCHIEBE R:
                          ;wiederhole:
      RR
             (HL)
                          : byteweise schieben
      INC
             HL
                            Zeiger inkrement.,
      DJNZ
             SCHIEBE R
                            Zähler (B) dekrem.
      POP
                          ;bis Zähler=0
             BC
      RET
****************
:Unterprogramm MOV
      kopiert Register (HL) nach Register (DE)
MOV:
      PUSH
             BC
             BC, BIN BYTE
      LD
                          ;Anzahl Byte = BIN BYTE
      LDIR
                          ;und schieben
      POP
             BC
      RET
*******************
;Unterprogramm NULL
      Prüft, ob Register (HL) gleich Null
***************
NULL:
      PUSH
             BC
             B, BIN BYTE
      LD
                          ;Anzahl Byte = BIN BYTE
      XOR
                          :A=0
NULL SCHLEIFE:
                          ;wiederhole:
                          ; ODER-Verkn. mit Byte
      OR
             (HL)
      INC
                            Zeiger inkrementieren
             HL
      DJNZ
             NULL SCHLEIFE
                             Zähler (B) dekrement.
                          ;bis Zähler=0
      AND
             Α
                          ;Bedingungsflags setzen
      POP
             BC
```

```
;Unterprogramm WANDLE DEZ
       wandelt PRIM in eine Dezimalzahl (nach DEZ REG)
 *********************
WANDLE DEZ:
       LD
              HL, PRIM
       LD
              DE, BIN REG
       CALL
              MOV
                             ;PRIM nach BIN REG
              HL, DEZ REG
                             ;alte Dezimalzahl löschen:
       LD
       LD
              B, DEZ STELL/2
CLEAR DEZ:
                             ;wiederhole:
       LD
              (HL), 0
                               ein Byte löschen
       INC
                                Zeiger inkrementieren
              HL
       DJNZ
              CLEAR DEZ
                                Zähler (B) dekrementieren
                             :bis Zähler=0
       LD
              DE, BIN STELL
                             ;alle Binärbits wandeln:
                             ;wiederhole:
WANDL SCHLEIFE:
              HL, BIN REG
       LD
              SCHIEBE
                                BIN REG := BIN REG*2
       CALL
                                DEZ REG := DEZ REG*2
       LD
              B, DEZ STELL/2
                                             + Carry
WANDL SCHLEIFE1:
                                wiederhole:
                                  Zeiger dekrementieren
       DEC
              HL
       LD
              A,(HL)
                                  Byte laden
       ADC
              A,(HL)
                                  mit Carry addieren
                                  dezimal korrigieren!
       DAA
       LD
              (HL),A
                                  speichern
                                  Zähler (B) dekrement.
       DJNZ
              WANDL SCHLEIFE1
                                bis Zähler=0
       DEC
              DE
                                Zähler (DE) dekremen.
       LD
              A,D
       OR
                             ;bis Zähler=0
              Ε
       JR
              NZ, WANDL SCHLEIFE
       RET
:Unterprogramm AUSGABE
****************
FLAG
       EQU
              0
AUSGABE:
       RES
              FLAG, C
                             ;Flag löschen,
       LD
              HL, DEZ REG
                             ; Zeiger und
       LD
              B, DEZ STELL/2
                             ; Zähler setzen
```

DS

```
:wiederhole:
AUSGABE SCHLEIFE:
                                    Dezimaldigit (Halbbyte)
        RLD
                                    holen und ausgeben
        CALL
                EINE STELLE
                                    dto. für anderes Halb-
        RLD
        CALL
                EINE STELLE
                                    byte (Digit)
                                    Zeiger auf nächstes Byte
        INC
                HL
                AUSGABE SCHLEIFE;
                                    Zähler (B) dekrementieren
        DJNZ
        RET
                                 :bis Zähler=0
EINE STELLE:
                                 ;Digit ausgeben:
        AND
                0FH
                                 ;rechtes Halbbyte maskieren
        JR
                NZ, SCHREIBE
                                 ;Digit=0? nein: ausgeben
                                    ja: Flag gesetzt?
        BIT
                FLAG, C
                                      ja: ausgeben (normale
        JR
                NZ, SCHREIBE
                                                          Nu11)
                E.' '
        LD
                                      nein: Zwischenraum aus-
        PUSH
                BC
                                          geben
        PUSH
                HL
        LD
                C, DRUCKF
        CALL
                SYSTEM
        POP
                HL
        POP
                BC
        RET
SCHREIBE:
        SET
                0,C
                                 ;Flag setzen
        PUSH
                 BC
                                 ; Zähler und
        PUSH
                                 ; Zeiger retten,
                HL
                 A, '0'
                                 ;Digit -> ASCII
        ADD
        LD
                E,A
        LD
                 C, DRUCKF
                                 ;ausgeben
        CALL
                 SYSTEM
                                 ;mit Systemfunktion
        POP
                                 ; Zeiger und
                HL
                                 ;Zähler wiederherstellen
        POP
                 BC
        RET
                    ************
SP PRIM:
                                 ;Stackpointer für Primzahlen-
                 2
        DS
                                 ;programm
                                 ;Stackpointer für System
SP SYS:
        DS
                 2
                                 ;programme
DEZ REG:
                 DEZ_STELL/2
                                 ;Speicher für Dezimalzahl
        DS
BIN REG:
        DS
                 BIN BYTE
PRIM:
        DS
                 BIN BYTE
                                 ;Speicher für Binärzahlen
TEILER:
                 BIN BYTE
```

QUOTIENT:

DS

BIN BYTE

REST:

DS

BIN BYTE

DS

100

; viel Platz für Stack!

STACK:

**END** 

HAUPT

Z80-Assembler

pass 1:

pass 2:

<del>┞</del>╋╋╋╋╋╋╋╋╋╇╇╇╇╇╇╇╇╇╇╇╇╇╇╇╇╇╇╇╇╇╇╇

no fatal error(s)

SUM=BDF6

CRC=6074

## TITLE BUCHSTABENWANDLUNG - Programm 1

**************			
;Dieses Programm wandelt alle Buchstaben im Textspeicher von ;Klein- auf Großschreibung, außer in Kommentaren oder in An-			
;führungsstrichen stehenden Strings ;************************************			
		·	
***************************************			
; Definition von Systemadressen ; ************************************			
SYSTEM	EQU	00005Н	;Adresse für Aufruf von ;Systemfunktionen
***************			
; Definition von Systemfunktionen			
·*************************************			
WSTARTF	EQU	0	;Warmstart
DIRCONF	EQU	6	;Direkte Konsol Ein/Ausgabe
STRAUSF	EQU	9	; Ausgabe von Strings auf CON:
TXTBEGF	EQU	247	;Startadresse des Text- ;speichers abfragen
	•		, spereners abriagen
.*****************			
;Definition von ASCII-Konstanten			
**************************************			
CR	EQU	ODH	;Cursor an Zeilenanfang ;(Carriage Return)
LF	EQU	OAH	;Cursor in neue Zeile ;(Line Feed)
EOF	EQU	1AH	;(End Of File)
			,(LAIC OI TITE)
***************			
; Definition von Flags in Register E			
KOM	EQU	0	;Kommentar: Bit 0
STR	EQU	1	;String: Bit 1

```
; Hauptprogramm BUCHSTABENWANDLUNG
START:
        JP
                START1
TEXT:
        DB
                'Klein-Grossbuchstabenwandlung '
                'fuer Programmtext', CR, LF, '$'
        DB
START1:
        LD
                SP, (SYSTEM+1)
                                ;Stackpointer laden
        LD
                                ;Meldung ausgeben
                DE, TEXT
        LD
                C, STRAUSF
        CALL
                SYSTEM
        CALL
                VORBEREITUNG
                                ;alles vorbereiten
        JR
                C, FEHLER
                                :Abbruch bei Fehler
        RES
                KOM, E
                                ;Flag für Kommentar Nullsetzen
        RES
                STR, E
                                ;Flag für String Nullsetzen
SCHLEIFE:
        CALL
                HOLE ZEICHEN
                                ;ein Zeichen holen,
                C.FEHLER
                                :Abbruch bei Fehler und
        JR
                                ;höchstes Bit wegmaskieren
        AND
                7FH
        BIT
                KOM, E
                                ; im Kommentar?
        JR
                NZ, KOMMENTAR
                                ; ja: weiter dort
        BIT
                STR,E
                                ; im String?
        JR
                NZ, STRING
                                ; ja: dto.
PRUEFE KOMM:
                                ; nein:
        CP
                                ;Semikolon?
        JR
                NZ, PRUEFE STR
                                ; nein: weiter
KOMM ANFANG:
                                ; ja: Kommentar beginnt
        SET
                KOM, E
                                ;Flag setzen
        JR
                WANDLE NICHT
                                ;und weiter ohne wandeln
PRUEFE STR:
                1111
        CP
                                ;Anführungsstrich?
        JR
                NZ, WANDLE
                                ; nein: weiter mit wandeln
STRING ANFANG:
                                ; ja: String beginnt
        SET
                STR, E
                                ;Flag setzen
        JR
                WANDLE NICHT
                                ;und weiter ohne wandeln
KOMMENTAR:
        CP
                CR
                                ; Zeile zu Ende (CR)?
        JR
                NZ, WANDLE NICHT; nein: weiter im Komm.
```

```
; ja: Kommentar zu Ende
KOMM ENDE:
                            ;Flag rücksetzen
              KOM, E
       RES
                            ;und weiter mit wandeln
       JR
              WANDLE
STRING:
       CP
              CR
                            :Zeile zu Ende (CR)?
                            ; ja: String endet
       JR
              Z,STRING ENDE
       CP
                            ;Anführungsstrich?
              NZ, WANDLE NICHT; nein: weiter im String
       JR
STRING ENDE:
                            ;Flag rücksetzen
       RES
              STR, E
                            und weiter mit wandeln
WANDLE:
       CP
              60H
                            ; kleiner Buchstabe ?
              C, WANDLE NICHT
                              nein: nicht wandeln
       JR
                               ja: wandeln
       SUB
WANDLE NICHT:
       CALL
              SCHREIBE ZEICHEN ; Zeichen wegschreiben
                            ;Abbruch bei Fehler
       JR
              C, FEHLER
       CP
                            ;Textende?
              EOF
       JR
              NZ, SCHLEIFE
                            ; nein: weiter
FERTIG:
       CALL
              ABSCHLUSS
                            ; ja: fertig
FEHLER:
       LD
              C, WSTARTF
                            ;Warmstart
       CALL
              SYSTEM
*****************
;Unterprogramm VORBEREITUNG
       - lädt IX und IY mit Textspeicheranfang
VORBEREITUNG:
              C, TXTBEGF
                            ;Textspeicheranfang
       LD
       CALL
              SYSTEM
                            ;holen nach HL
       PUSH
                            ;und HL
              HL
       POP
              IX
                             ;nach IX
              HL
       PUSH
       POP
              IY
                             ;und IY
       AND
              Α
                             ;Carry Nullsetzen
       RET
;Unterprogramm Abschluss
```

ABSCHLUSS:

RET

```
********************
:Unterprogramm HOLE ZEICHEN
*********************
HOLE ZEICHEN:
          A_{\bullet}(IX)
     LD
     INC
          ΙX
                    ;Carry Nullsetzen
     AND
     RET
*******************
;Unterprogramm SCHREIBE ZEICHEN
SCHREIBE ZEICHEN:
          (IY),A
     LD
     INC
          IY
     AND
          Α
                    ;Carry Nullsetzen
     RET
PROGRAMM ENDE:
     END
          START
```

## TITLE BUCHSTABENWANDLUNG - Programm 2

```
***********************************
; Dieses Programm wandelt alle Buchstaben einer Datei von
;Klein- auf Großschreibung, außer in Kommentaren oder in An-
;führungsstrichen stehenden Strings
**********************
;Definition von Systemadressen
* ************************************
SYSTEM EQU
           00005H
                        ;Adresse für Aufruf von
                        ;Systemfunktionen
***********************************
; Definition von Systemfunktionen
                 *************
WSTARTF EQU
           0
                        ;Warmstart
DIRCONF EQU
           6
                        ;Direkte Konsol Ein/Ausgabe
STRAUSF EQU
           9
                        ; Ausgabe von Strings auf CON:
```

```
STREINF EQU
                          ; Eingabe von Strings mit CON:
             10
OPENF
                          ;Datei oeffnen
      EQU
             15
CLOSEF
      EQU
             16
                          :Datei schliessen
DELETEF EQU
             19
                          ;Datei löschen
READF
             20
                          :aus Datei lesen
      EOU
WRITEF
      EQU
             21
                          ;in Datei schreiben
             22
MAKEF
      EQU
                          ;Datei anlegen
SETDMAF EQU
             26
                          ;DMA-Adresse setzen
MAKEFOB EOU
             250
                          ;FCB aus String erzeugen
****************
;Definition von ASCII-Konstanten
CR
      EQU
             ODH
                           ;Cursor an Zeilenanfang
                           ;(Carriage Return)
                           ;Cursor in neue Zeile
LF
      EOU
             OAH
                           ;(Line Feed)
                           ;Markierung für Textende
EOF
      EQU
             1AH
                           ;(End Of File)
; Definition von Flags in Register E
*****************
KOM
      EQU
             0
                           :Kommentar: Bit 0
STR
      EQU
                                    Bit 1
                           ;String:
*******************
; Hauptprogramm BUCHSTABENWANDLUNG
******************
      JP
START:
             START1
TEXT:
      DB
             'Klein-Grossbuchstabenwandlung '
             'fuer Programmtext', CR, LF, '$'
      DB
START1:
      LD
             SP, (SYSTEM+1)
                          ;Stackpointer laden
             DE, TEXT
      LD
                           ;Meldung ausgeben
             C, STRAUSF
      LD
             SYSTEM
      CALL
```

	CALL JR	VORBEREITUNG C, FEHLER	;alles vorbereiten ;Abbruch bei Fehler		
	RES RES	KOM,E STR,E	;Flag für Kommentar Nullsetzen ;Flag für String Nullsetzen		
SCHLEIF	<b></b>				
COLLETT	CALL JR AND	HOLE_ZEICHEN C,FEHLER 7FH	;ein Zeichen holen, ;Abbruch bei Fehler und ;höchstes Bit wegmaskieren		
	BIT JR	KOM,E NZ,KOMMENTAR	;im Kommentar? ; ja: weiter dort		
	BIT JR	STR,E NZ,STRING	;im String? ; ja: dto.		
PRUEFE KOMM: ; nein:					
TROLL L	CP	1:1	;Semikolon?		
	JR	NZ, PRUEFE STR	; nein: weiter		
KOMM_AN	FANG:	_	; ja: Kommentar beginnt		
	SET	KOM,E	;Flag setzen		
	JR	WANDLE_NICHT	;und weiter ohne wandeln		
מממוממו	own .				
PRUEFE_S	CP	1111	;Anführungsstrich?		
	JR	NZ,WANDLE	; nein: weiter mit wandeln		
STRING A		WZ, WANDEL	; ja: String beginnt		
01K1K0_1	SET .	STR,E	;Flag setzen		
	JR	WANDLE NICHT	;und weiter ohne wandeln		
KOMMENT			- 11		
	CP	CR	; Zeile zu Ende (CR)?		
	JR	NZ,WANDLE_NICHI	; nein: weiter im Komm.		
KOMM ENDE: ; ja: Kommentar zu Ende					
KOMI_LIN	RES	KOM,E	;Flag rücksetzen		
	JR	WANDLE	;und weiter mit wandeln		
STRING:	~	<b>an</b>	- 11		
	CP	CR	; Zeile zu Ende (CR)?		
	JR CP	Z,STRING_ENDE	; ja: String endet ;Anführungsstrich?		
	JR	NZ.WANDLE NICHT	; nein: weiter im String		
STRING			,		
_	RES	STR,E	;Flag rücksetzen		
			;und weiter mit wandeln		
WANDLE:	OD.	COLL	tal alman Dark art 1 = 0		
	CIP JR	60H C,WANDLE NICHT	; kleiner Buchstabe ? ; nein: nicht wandeln		
	SUB	20H	; ja: wandeln		
			, , ,		

WANDLE NICHT:

CALL

```
:Abbruch bei Fehler
       JR
               C, FEHLER
       CP .
               EOF
                               :Textende?
                               ; nein: weiter
       JR
               NZ, SCHLEIFE
FERTIG:
       CALL
               ABSCHLUSS
                               ; ja: fertig
FEHLER:
       LD
               C, WSTARTF
                               :Warmstart
               SYSTEM
       CALL
******************
;Unterprogramm VORBEREITUNG
        - lädt Datei in Speicher
        - lädt IX und IY mit Textanfang
*****************
                       'Name der Eingabe-Datei ? $'
VORB FRAGE:
               DB
VORB FEHLER1:
               DB
                       'Datei nicht gefunden!', CR, LF, '$'
VORB FEHLER2:
               DB
                        'Speicher voll!', CR, LF, '$'
               DW
                       0
SATZ ZAHL:
                               ;Anzahl gelesene Datensätze
                       16,0
PUFFER:
               DB
                               ;Eingabe-Puffer für Datei-
                       16
               DS
                               ;name, max. 16 Zeichen
EINGABE FCB:
                       36,0
                               ;Dateisteuerblock Eingabe
               DS
VORBEREITUNG:
       LD
               DE, VORB FRAGE
                               ;nach Dateiname fragen,
       LD
               C,STRAUSF
        CALL
               SYSTEM
       LD
               DE, PUFFER
                               ;Name in Puffer lesen
       LD
                C,STREINF
        CALL
               SYSTEM
       LD
               DE, PUFFER+2
                               ;und File-Control-Block
       LD
                IY, PUFFER
                               ;erzeugen
       LD
                IX, EINGABE FCB
       LD
               C, MAKEFCB
        CALL
               SYSTEM
                               ;(Carry=0 => Fehler)
       CCF
                               ;Carry komplementieren
        RET
               C
                               ;Rücksprung mit Carry=1
                               ;bei Fehler
```

SCHREIBE ZEICHEN ; Zeichen wegschreiben

```
E
```

```
**************
;Unterprogramm Abschluss
       - speichert Datei auf Diskette
**************
                       'Name der Ausgabe-Datei ? $'
ABSCH FRAGE:
               DB
ABSCH FEHLER1:
               DB
                       'Datei existiert!', CR, LF, '$'
                       'Inhaltsverzeichnis voll!',CR,LF,'$'
ABSCH FEHLER2:
               DB
ABSCH FEHLER3:
               DB
                       'Diskette voll!', CR, LF, '$'
AUSGABE FCB:
               DS
                       36,0
                               ;Dateisteuerblock Ausgabe
ABSCHLUSS:
       LD
               DE, ABSCH FRAGE
                               ;nach Dateiname fragen
       LD
               C, STRAUSF
       CALL
               SYSTEM
       LD
               DE, PUFFER
                               ;Name in Puffer lesen
       LD
               C, STREINF
       CALL
               SYSTEM
       LD
               DE, PUFFER+2
                               ;und File-Control-Block
       LD
               IY, PUFFER
                               ;erzeugen
       LD
               IX, AUSGABE FCB
       LD
               C, MAKEFCB
       CALL
               SYSTEM
                               ;(Carry=0 => Fehler)
       RET
               NC
                               ;Rücksprung bei Fehler
       LD
               DE, AUSGABE FCB
                               ;Datei öffnen
       LD
               C, OPENF •
               SYSTEM
       CALL
                               ;A=OFFH falls Datei
       CP
               OFFH
                                      nicht existiert!
       JR
               Z, ABSCHLUSS1
       LD
               DE, AUSGABE FCB
                               ;falls Datei existiert:
       LD
               C,CLOSEF
                               ;Datei schliessen,
       CALL
               SYSTEM
       LD
               DE, ABSCH FEHLER1 ; Fehlertext ausgeben
               C,STRAUSF
       LD
       CALL
               SYSTEM
       RET
                               ;und Rücksprung
ABSCHLUSS1:
       LD
               DE, AUSGABE FCB
                               ;Datei anlegen
       LD
               C, MAKEF
       CALL
               SYSTEM
                               ;A=OFFH bei Fehler
       CP
               OFFH
       JR
               NZ, ABSCHLUSS2
```

```
LD
                 DE, ABSCH FEHLER2 ; Fehlertext ausgeben
        LD
                 C, STRAUSF
        CALL
                 SYSTEM
        RET
                                  ;und Rücksprung
ABSCHLUSS2:
        LD
                 DE, PROGRAMM ENDE ; Anfangsadresse zum
                                   :Abspeichern
        LD
                 BC, (SATZ ZAHL)
                                   und Anzahl Datensätze
                                  :laden
SCHREIB SCHLEIFE:
        LD
                                  ;BC=0: kein Datensatz mehr
                A,B
        OR
                C
                                  ;zu schreiben
                 Z, SCHREIB ENDE
        JR
        LD
                HL, 128
                                  ;DE+128 ergibt die Adresse
        ADD
                HL,DE
                                  ;für den nächsten Durchlauf
        PUSH
                HL.
                                  ;diese Adresse retten
        DEC
                 BC
                                  ;Zähler dekrementieren
        PUSH
                BC
                                  ;und retten,
        LD
                 C,SETDMAF
                                  ;momentane Transfer-
        CALL
                SYSTEM
                                  ;Adresse (DMA-Adr.) setzen
        LD
                DE, AUSGABE FCB
                                  ; Record (Datensatz) auf
        LD
                C.WRITEF
                                  ;Diskette schreiben
        CALL
                SYSTEM
                                  ;A=O falls erfolgreich,
                                  ;sonst Diskette voll
        POP
                BC
                                  ;Zähler restaurieren und
        POP
                DE
                                  ;nächste DMA-Adresse in DE
                Α
        AND
        JR
                Z, SCHREIB SCHLEIFE ; und weiter falls nicht voll
        LD
                DE, AUSGABE FCB ; sonst Datei löschen,
        LD
                C, DELETEF
        CALL
                SYSTEM
        LD
                DE, ABSCH FEHLER3 ; Fehlertext ausgeben
        LD
                C.STRAUSF
        CALL
                SYSTEM
        RET
                                  ;und Rücksprung
SCHREIB ENDE:
        LD
                DE, AUSGABE FCB
                                  ;Datei schliessen
        LD
                C,CLOSEF
        CALL
                SYSTEM
        RET
                                  ;und Rücksprung
```

```
F
```

```
****************
:Unterprogramm HOLE ZEICHEN
. *********************
HOLE ZEICHEN:
     LD
          A,(IX)
     INC
          IX
          A
                    ;Carry Nullsetzen
     AND
     RET
****************
;Unterprogramm SCHREIBE ZEICHEN
***************
SCHREIBE ZEICHEN:
     LD
          (IY),A
     INC
          IY
                    ;Carry Nullsetzen
     AND
          Α
     RET
PROGRAMM ENDE:
     END
          START
```

## TITLE BUCHSTABENWANDLUNG - PROGRAMM 3

\*\*\*\*\*\*\*\*\*\*\*\*\*\*

```
; Dieses Programm wandelt alle Buchstaben einer Datei von
;Klein- auf Großschreibung, außer in Kommentaren oder in An-
; führungsstrichen stehenden Strings
. *******************
; Definition von Systemadressen
             ***********
SYSTEM EQU
           00005H
                      ;Adresse für Aufruf von
                      :Systemfunktionen
; Definition von Systemfunktionen
*****************
           0
WSTARTF EQU
                      ;Warmstart
DIRCONF EQU
           6
                      ;Direkte Konsol Ein/Ausgabe
STRAUSF EQU
                      ; Ausgabe von Strings auf CON:
OPENF
     EQU
           15
                      ;Datei oeffnen
```

CLOSEF	EQU	16	;Datei schliessen		
DELETEF	EQU	19	;Datei löschen		
READF	EQU	20	;aus Datei lesen		
WRITEF	EQU	21	;in Datei schreiben		
MAKEF	EQU	22	;Datei anlegen		
RENAMEF	EQU	23	;Datei umbenennen		
SETDMAF	EQU	26	;DMA-Adresse setzen		
;*************************************					
CR	EQU	ODH	;Cursor an Zeilenanfang		
LF	EQU	OAH	;(Carriage Return) ;Cursor in neue Zeile		
EOF	EQU	1AH	;(Line Feed) ;Markierung für Textende ;(End Of File)		
;*************************************					
KOM STR	EQU EQU	0 1	;Kommentar: Bit 0 ;String: Bit 1		
;*************************************					
START:	JP	START1			
TEXT: START1:	DB DB	'Klein-Grossbuchstabenwandlung ' 'fuer Programmtext', CR, LF, '\$'			
	LD	SP,(SYSTEM+1)	;Stackpointer laden		
	LD LD CALL	DE,TEXT C,STRAUSF SYSTEM	;Meldung ausgeben		
	CALL JR	VORBEREITUNG C,FEHLER	;alles vorbereiten ;Abbruch bei Fehler		
	RES RES	KOM,E STR,E	;Flag für Kommentar Nullsetzen ;Flag für String Nullsetzen		

SCHLEIFE:

```
;ein Zeichen holen,
        CALL
                 HOLE ZEICHEN
        JR
                 C, FEHLER
                                  :Abbruch bei Fehler und
                                  ; höchstes Bit wegmaskieren
        AND
                 7FH
        BIT
                 KOM, E
                                  ; im Kommentar?
        JR
                 NZ, KOMMENTAR
                                  ; ja: weiter dort
                                  ; im String?
        BIT
                 STR, E
        JR
                 NZ, STRING
                                  ; ja: dto.
PRUEFE KOMM:
                                  ; nein:
        CP
                                  ;Semikolon?
        JR
                 NZ, PRUEFE STR
                                  ; nein: weiter
KOMM ANFANG:
                                   ; ja: Kommentar beginnt
        SET
                                  ;Flag setzen
                 KOM, E
                                  ;und weiter ohne wandeln
        JR
                 WANDLE NICHT
PRUEFE STR:
                 1 1 1 1
        CP
                                   ;Anführungsstrich?
        JR
                                  ; nein: weiter mit wandeln
                 NZ, WANDLE
STRING ANFANG:
                                   ; ja: String beginnt
        SET
                 STR, E
                                   ;Flag setzen
        JR
                 WANDLE NICHT
                                   und weiter ohne wandeln
KOMMENTAR:
        CP
                 CR
                                   ; Zeile zu Ende (CR)?
        JR
                 NZ, WANDLE NICHT; nein: weiter im Komm.
KOMM ENDE:
                                   ; ja: Kommentar zu Ende
        RES
                                   ;Flag rücksetzen
                 KOM, E
        JR
                 WANDLE
                                   ;und weiter mit wandeln
STRING:
        CP
                 CR
                                   ; Zeile zu Ende (CR)?
        JR
                                   ; ja: String endet
                 Z,STRING ENDE
                 1 1 1 1
        CP
                                   ;Anführungsstrich?
        JR
                 NZ, WANDLE NICHT; nein: weiter im String
STRING ENDE:
        RES
                 STR, E
                                   ;Flag rücksetzen
                                   und weiter mit wandeln
WANDLE:
        CP
                 60H
                                   ; kleiner Buchstabe ?
        JR
                 C, WANDLE NICHT
                                     nein: nicht wandeln
        SUB
                 20H
                                      ja: wandeln
WANDLE NICHT:
        CALL
                 SCHREIBE ZEICHEN ; Zeichen wegschreiben
        JR
                 C, FEHLER
                                   ;Abbruch bei Fehler
        CP
                                   ;Textende?
        JR
                 NZ, SCHLEIFE
                                   ; nein: weiter
```

```
FERTIG:
        CALL
               ABSCHLUSS
                                ; ja: fertig
FEHLER:
                                ; ".$$$" FCB löschen falls
       LD
               DE, AUSGABE FCB
                C, DELETEF
       LD
                                ; vorhanden
        CALL
               SYSTEM
        LD
               C, WSTARTF
                                ; Warmstart
        CALL
                SYSTEM
 ***********************
:Unterprogramm VORBEREITUNG
        - lädt Datei in Speicher
        - lädt IX und IY mit Textanfang
 ******************
                        'Datei nicht gefunden!',CR,LF,'$'
VORB FEHLER1:
                DB
VORB FEHLER2:
                DB
                        'Inhaltsverzeichnis voll!',CR,LF,'$'
STANDARD FCB
               EOU
                        005CH
                                :Standard-Dateisteuerblock
                                ;vom Betriebssystem
VORBEREITUNG:
        LD
                HL, STANDARD FCB ; Dateiname für
                DE, EINGABE FCB
       LD
                                ;Eingabedatei
        LD
                BC, 12
                                ;(Laufwerk+Name+Typ=12)
       LDIR
       LD
                HL, STANDARD FCB ; Dateiname für
                DE, AUSGABE FCB
                                :Ausgabedatei (".$$")
       LD
                                ;(Laufwerk+Name=9)
        LD
                BC,9
       LDIR
        LD
                HL, STANDARD FCB : Dateiname für
                DE, BACKUP FCB
                                ;Sicherungsdatei (".BAK")
        LD
        LD
                BC,9
                                ;(Laufwerk+Name=9)
       LDIR -
        LD
                DE, EINGABE FCB
                                ;Datei öffnen
        LD
                C, OPENF
        CALL
                SYSTEM
                                ;A=OFFH falls Datei
        CP
                OFFH
                                       nicht gefunden
        JR
                NZ, VORBEREITUNG1
        LD
                DE, VORB FEHLER1 ; Textausgabe "Datei
        LD
                C, STRAUSF
                                           nicht gefunden"
                SYSTEM
        CALL
        SCF
                                ;und mit Carry=1 für Fehler
        RET
                                ;zurück ins Hauptprogramm
VORBEREITUNG1:
        LD
                                ;Datei löschen falls
                DE, AUSGABE FCB
                C, DELETEF
        LD
                                        ; vorhanden
```

CALL

SYSTEM

```
Œ
```

```
LD
               DE, AUSGABE FCB ; Datei anlegen
       LD
               C, MAKEF
       CALL
               SYSTEM
                               :A=OFFH bei Fehler
       CP
               OFFH
               Z, VORBEREITUNG2
       JR
                              ;ok: Carry löschen und zurück
       AND
                               ins Hauptprogramm, sonst
       RET
VORBEREITUNG2:
               DE, VORB FEHLER2 ; Fehlertext ausgeben
       LD
       LD
               C.STRAUSF
       CALL
               SYSTEM
       SCF
                               ;und mit Carry=1 für Fehler
       RET
                               zurück ins Hauptprogramm
*****************
;Unterprogramm Abschluss
       - speichert Datei auf Diskette
BACKUP FCB:
               0,'
                          BAK' ;Dateisteuerblock Sicherungs-
       DB
       DS
               24,0
                               ;datei (backup)
ABSCHLUSS:
       CALL
               SCHREIBE DS
                               ;letzten Datensatz schreiben
       RET
                               und zurück falls Fehler
               C
       LD
               DE, AUSGABE FCB
                              ;sonst Ausgabedatei
       LD.
               C,CLOSEF
                               :schliessen
       CALL
               SYSTEM
       LD
               DE, BACKUP FCB
                               ;alte Backup-Datei löschen
               C.DELETEF
       LD
                               ;falls vorhanden
       CALL
               SYSTEM
       LD
               HL, BACKUP FCB+1 ; Eingabedatei in ".BAK" umbe-
       LD
               DE, EINGABE FCB+16+1
                                              ;nennen
               BC, 11
                               ;(Name+Typ=11)
       LD
       LDIR
       LD
               DE, EINGABE FCB
               C, RENAMEF
       LD
       CALL
               SYSTEM
       LD
               HL, EINGABE FCB+1 ; Ausgabedatei ".$$"
                                ;in alte Eingabedatei umbe-
       LD
               DE, AUSGABE FCB+16+1
                                              ;nennen
       LD
               BC, 11
                               ;(Name+Typ=11)
       LDIR
       LD
               DE, AUSGABE FCB
               C, RENAMEF
       LD
       CALL
               SYSTEM
       RET
                               ;und Rücksprung
```

```
*******************
:Unterprogramm HOLE ZEICHEN
******************
EINGABE FCB:
       DS
               36,0
                               ;Dateisteuerblock Eingabe
EINGABE ZEIGER:
        DB
               0
                               ; Zeiger im Eingabepuffer
EINGABE PUFFER:
       DS
               128
                               ;Puffer für einen Datensatz
HOLE ZEICHEN:
       LD
               A, (EINGABE ZEIGER)
       LD
               C,A
                               ; Pufferposition berechnen
       LD
               B,0
       LD
               HL, EINGABE PUFFER
        ADD
               HL, BC
       INC
               Α
                               ; Zeiger inkrementieren,
       AND
                               ;höchstes Bit wegmaskieren
               7FH
       LD
               (EINGABE ZEIGER), A ; und Zeiger speichern
       LD
               A,C
       AND
                               ;wenn Zeiger auf Null, dann
               Α
       PUSH
               HL
       CALL
               Z, LESE DS
                               ;zuerst Datensatz einlesen
       POP
               HL
       RET
               C
                               ;zurück bei Fehler
       LD
               A,(HL)
                               ; Zeichen aus Puffer laden
                               ;Carry=Null (kein Fehler)
       AND
               Α
                               ;und zurück
       RET
LESE DS:
                               ;Datensatz lesen:
       PUSH
               DE
       LD
               DE, EINGABE PUFFER
       LD
               C, SETDMAF
                               ;Transferadresse setzen
       CALL
               SYSTEM
       LD
               DE, EINGABE FCB
                               ;und Datensatz einlesen
       LD
               C, READF
       CALL
               SYSTEM
       POP
               DE
               Α
        AND
               Z
       RET
                               ;zurück falls ok, sonst
       SCF
                               ;mit Carry zurück
       RET
```

F

```
E
```

```
******************
;Unterprogramm SCHREIBE ZEICHEN
AUSGABE FCB:
       DB
              0,'
                         $$$'; Dateisteuerblock Ausgabe
       DS
               24,0
AUSGABE_ZEIGER:
                              ; Zeiger im Ausgabepuffer
AUSGABE PUFFER:
       DS
               128
                              ;Puffer für einen Datensatz
SCHREIBE ZEICHEN:
       PUSH
               AF
                              ; Zeichen retten
       LD
               A, (AUSGABE ZEIGER)
               C,A
       LD
                              ;Pufferposition berechnen
       LD
               B, 0
       LD
               HL, AUSGABE PUFFER
       ADD
               HL, BC
       POP
               AF
                              ; Zeichen holen und
       PUSH
               AF
                              ;wieder retten,
               (HL),A
       LD
                              ; Zeichen in Puffer legen
       LD
               A,C
       INC
               Α
                              ; Zeiger inkrementieren,
       AND
               7FH
                              ; höchstes Bit wegmaskieren
               (AUSGABE ZEIGER), A ; und Zeiger speichern
       LD
       AND
                              ;wenn Zeiger auf Null, dann
       CALL
               Z, SCHREIBE DS
                              ;Datensatz wegschreiben
       POP
               BC
                              ; Zeichen holen, aber Flags
       LD
               A,B
                              :lassen!
       RET
SCHREIBE DS:
                              ;Datensatz schreiben:
       PUSH
               DE
       LD
               DE, AUSGABE PUFFER
       LD
               C, SETDMAF
                              ;Transferadresse setzen
               SYSTEM
       CALL
       LD
               DE, AUSGABE FCB
                              ;und Datensatz schreiben
```

LD C,WRITEF
CALL SYSTEM

POP DE
AND A
RET Z ;zurück falls ok, sonst
SCF ;mit Carry zurück
RET

PROGRAMM\_ENDE:

END START

F