```
; ------------------------------------------------------------------------
; Quelldatei: sieb11.asm
; Zieldatei : sieb11.com
; Funktion  : Assemblerprogramm fr das Sieb des Erasthostenes bis 500000
; Sprache   : TASM 3.2
; Autor     : HD.Kirmse
; (c)1995     Gymnasium II Saalfeld
; ------------------------------------------------------------------------

; Vereinbarungen :

     CRLF      EQU   0D0Ah                  ; Carriage Return + Line Feed
     CR        EQU   0Dh                    ; Carriage Return


 MODEL TINY

 PROG          SEGMENT   WORD

               ASSUME  CS : PROG , DS : PROG , ES : PROG

               ORG   100h

 START:        JMP   BEGIN

     primz     DW    2                      ; Primzahl - Variable
     text      DB    'Geben Sie die gr”áte Zahl ein ( < 500000 ) : ','$'

     n         DD    ?                      ; Variable n
     h         DW    ?                      ; Hilfsvariable frs Wurzelziehen
     g         DW    ?                      ; g := wurzel (n)


 BEGIN:

               MOV   DI,OFFSET x
               MOV   AL,1
               MOV   CX,64200               ; von 64200 ist 200 zur Sicherh.
               REP   STOSB

               CALL cls                     ; Bildschirm l”schen
               CALL eingabe                 ; n eingeben
               CALL grenze                  ; g bestimmen ( wurzel(n) )

     repeat:                                ; wiederhole :
               CALL streiche_vielfache
               CALL naechste_primzahl

               MOV   AX,g
               CMP   primz,AX               ;   vergleiche primz - g
     JBE       repeat                       ; wenn primz <= g

               CALL ausgabe                 ; alle Primzahlen ausgeben
               CALL readln                  ; warten auf RETURN
               CALL cls                     ; Bildschirm l”schen

               MOV   AH,4Ch
               INT   21h                    ; zu DOS
               RET

; ------------------------------------------------------------------------

 wurzel        PROC                         ; nur fr PROC grenze !

               COMMENT                      *

               zieht die Quadratwurzel der Zahl n vom Typ "word"
               wobei die L”sung wieder vom Typ word ist und nach g zurck
               geliefert wird. Ist n eine Quadratzahl, so ist das Ergebnis
               exakt, andernfalls ist es entweder ist es eine benachbarte
               Zahl, die allerdings nicht den Rundungsregeln gengt.
               Es werden die Register AX, BX, CX und DX ver„ndert.
```

```
                *

                MOV   CX,128                    ; Startwert fr x1
                MOV   BX,CX                     ; Korrekturglied
                SHR   BX,1                      ; ist immer die H„lfte
    @1:         MOV   AX,CX                     ; aktuelles w ( wurzel )
                XOR   DX,DX                     ; wegen Multiplikation
                MUL   AX                        ; x = w * w    ( quadrat )
                CMP   AX,h                      ; Vergleich x mit n
                JE    @4                        ; wenn gleich -> Ende
                CMP   AX,h                      ; Vergleich x mit n
                JA    @2                        ; gr”áer -> @2
                ADD   CX,BX                     ; sonst   :  + Korrekturwert
                JMP   @3                        ; berspring: wenn kleiner
    @2:         SUB   CX,BX                     ;            - Korrekturwert
    @3:         SHR   BX,1                      ; neuer Korrekturwert
                JNZ   @1                        ; wenn  Korrekturwert <> 0
    @4:         MOV   g,CX                      ; Funktionsergebnis nach g
                RET                             ; fr Rcksprung

wurzel          ENDP

grenze          PROC

                COMMENT                         ;

                Um die  Implementation der Wurzel einfach zu halten, wird
                folgendermaáen vorgegangen :
                Ist x > 0FFFFh , wird x durch 10000h dividiert , dann
                die Wurzel gezogen , sicherheitshalber inkrementiert und
                mit 100h multipliziert. ( sqrt a*b = sqrt a * sqrt b ! )

                ;

                MOV   AX,WORD PTR n + 2         ; Highanteil von n laden
                OR    AX,AX                     ; fr Nullflag
                JNZ   @5_                       ; springe wenn <> 0
                MOV   AX,WORD PTR n             ; sonst lade Lowanteil
                MOV   h,AX                      ; dient zur Parameterbergabe !
                CALL  wurzel                    ; Ergebnis nach g
                JMP   @6_                       ; fertig
    @5_:        MOV   h,AX                      ; Parameterbergabe
                CALL  wurzel                    ; Ergenis in g ist zu klein
                MOV   AX,g                      ; noch multiplizieren mit 100h
                INC   AX                        ; aufrunden
                MOV   CL,8                      ; um 8 Bit verschieben
                SHL   AX,CL                     ; nach links
                MOV   g,AX                      ; Ergebnis speichern
    @6_:        RET

grenze          ENDP

; ---------------------------------------------------------------------------

cls             PROC

                COMMENT                         *

                Die L”schroutine nutzt den Nebeneffekt des
                Bildschirml”schens beim Initialisieren des
                Bildschirmmodus. Damit dieser nicht ver„ndert
                wird, wird er gelesen und wieder gesetzt.

                *

                MOV   AH,15                     ; Videomodus lesen
                INT   10h                       ; BIOS-Interrupt Bildschirm
                MOV   AH,0                      ; und wieder setzen
                INT   10h
                RET

cls             ENDP
```

```
; ------------------------------------------------------------------------

 writeZ       PROC                          ; Routine fr 8 stellige Zahl

              COMMENT                       *

              Diese Routine schreibt eine Zahl vom Typ "word" .
              Die Ausgabe entspricht in Pascal "print(x : n)" .
              Dabei ist n = 8 die Anzahl der zu verwendeten Ziffern
              und x die auszugebende Zahl, die rechtsbndig
              ausgegeben wird. Es wird die Zahl durch 10000 geteilt und
              Ergebnis und Rest nacheinander bearbeitet.
              x muá in AX vorliegen .
              Es werden die Register AX, BX, CX, DX und DI ver„ndert.

              *

              ???

              MOV  DI,8                     ; gewnschte Anzahl Ziffern
              MOV  BX,10                    ; Divisor - Zehnersystem
              XOR  CX,CX                    ; Vorbereitung Schleifenz„hler

     @5:      XOR  DX,DX                    ; wegen Division
              DIV  BX                       ; div 10
              PUSH DX                       ; Rest = Ziffer vom Ergebnis
              INC  CX                       ; Anzahl aktualisieren
              OR   AX,AX                    ; (Nullflag) bin ich fertig ?
              JNE  @5                       ; wenn <> 0 -> weiter

              XOR  BX,BX                    ; BX vorbereiten
              MOV  BL,0FFh - 15             ; Initialisierung fr Leerzeichen
     @6:      CMP  DI,CX                    ; mssen Leerzeichen sein ?
              JBE  @7                       ; springe bei n <= Anzahl
              PUSH BX                       ; "Leerzeichen" speichern
              INC  CX                       ; Anzahl aktualisieren
              JMP  @6

     @7:      MOV  AH,02                    ; Funktionsnummer INT 21h
     @8:      POP  DX                       ; hole Zeichen
              ADD  DL,'0'                   ; konvertiere zu ASCII
              INT  21h                      ; Ausgabe
              LOOP @8                       ; insgesamt n mal

              RET

 writeZ       ENDP

; ------------------------------------------------------------------------

 readW        PROC

              COMMENT                       *

              Diese Routine liest eine Zahl vom Typ "word" ein .
              Der Abschluá der Eingabe bildet "RETURN" .
              Es werden die Register AX, BX, CX, DX und DI ver„ndert.

              *

              XOR  BX,BX
              XOR  DX,DX
              MOV  CX,10                    ; lade Teiler
              MOV  DI,5                     ; lade Schleifenz„hler
              PUSH DX

     @9:      MOV  AH,01                    ; Funktionsnummer
              INT  21h                      ; Zeichen entgegenehmen
              CMP  AL,CR                    ; war es ein RETURN ?
              JE   @10                      ; zum Ende
              SUB  AL,48                    ; konvertiere ASCII -> Zahl
              MOV  BL,AL                    ; Zwischenspeichern fr Addition
              POP  AX
```

```
                MUL   CX                        ; alter Wert * 10
                ADD   AX,BX                     ; eingegebener Wert dazu
                PUSH  AX                        ; auf Stack speichern
                DEC   DI                        ; Schleifenz„hler aktualisieren
                JNZ   @9

      @10:      POP   AX                        ; Ergebnis in AX
                RET

 readW          ENDP

; ---------------------------------------------------------------------------

 streiche_vielfache PROC

                COMMENT                         *

                Diese Routine setzt alle Vielfachen von "primz" Null .
                Es werden die Register AX, BX, CX und DX ver„ndert.

                *

                MOV   DX,primz                  ; primz
                MOV   AX,DX                     ; v
                XOR   CL,CL                     ; Null setzen

      @11:      ADD   AX,DX                     ; v := v + primz  in AX
                MOV   BX,OFFSET x               ; Adresse von x[0]
                ADD   BX,AX                     ; Adresse von x[v]
                MOV   [BX],CL                   ; x[v] := 0

                CMP   AX,n                      ; vergleiche v mit n
                JBE   @11                       ; springe, wenn v <= n

                RET

 streiche_vielfache ENDP

; ---------------------------------------------------------------------------

 naechste_primzahl  PROC

                COMMENT                         *

                Diese Routine incrementiert "primz" solange, bis x[primz] <> 0.
                Es werden die Register BX, CX und DX ver„ndert.

                *

                MOV   DX,0FF00h                 ; Schleifenz„hler (merken)
                MOV   CX,DX                     ; und Lowbyte = 0 !
                MOV   BX,OFFSET x
                ADD   BX,primz                  ; Primzahl in DS:[BX]
      @12:      INC   BX                        ; Adresse fr n„chstes Byte
                CMP   DL,[BX]                   ; ist dieses Byte = 0 ?
                LOOPE @12                       ; dann weiter
                SUB   DX,CX                     ; Anzahl Durchl„ufe in DX
                ADD   primz,DX                  ; neue Primzahl
                RET

 naechste_primzahl  ENDP

; ---------------------------------------------------------------------------

 writeln        PROC                            ; = PASCALprozedur: writeln

                MOV   AH,02                     ; fr Ausgabe eines Zeichen
                MOV   DX,CRLF                   ; DL = LF (Line Feed)
                INT   21h                       ; DOS - Interrupt
                XCHG  DH,DL                     ; DL = CR (Carriage Return)
                INT   21h                       ; DOS - Interrupt
                RET
```

```
writeln     ENDP

; ---------------------------------------------------------------------

readln      PROC                        ; = PASCALprozedur: readln

    @13:    MOV  AH,01                  ; fr Eingabe eines Zeichen
            INT  21h                    ; DOS - Interrupt
            CMP  AL,CR                  ; ist es ein Return ?
            JNE  @13                    ; wenn nicht, -> nochmal
            RET

readln      ENDP

; ---------------------------------------------------------------------

ausgabe     PROC

            COMMENT                     *

            Diese Routine besteht aus einer Schleife, die mit LODSB
            das Feld x byteweise in AL l„dt. Dabei wird BX hochgez„hlt.
            Wenn AL <> 0 gibt BX die Primzahl an. Demzufolge wird BX
            ausgegeben. Dazu wird die Prozedur "writeW" aufgerufen.
            Auáerdem mssen die Register BX und CX gerettet werden.
            Es werden die Register AX, BX, CX und SI ver„ndert.

            *

            MOV  CX,n                   ; gr”átm”gliche Zahl
            MOV  BX,1                   ; Initialisierung Z„hler
            SUB  CX,2                   ; Feld 0 und 1 entf„llt
            MOV  SI,OFFSET x            ; fr LODSB
            ADD  SI,2                   ; auf 2. Element
    @14:    LODSB                       ; Feldelement in AL
            INC  BX                     ; Z„hler korrigieren
            OR   AL,AL                  ; um Nullflag zu setzen
            JZ   @15                    ; wenn 0 , keine Primzahl
            PUSH BX                     ; Register sichern
            PUSH CX
            MOV  AX,BX                  ; Zahl wird in AX erwartet
            CALL writeZ                 ; ausgeben
            POP  CX                     ; Register restaurieren
            POP  BX
    @15:    LOOP @14                    ; bis CX = 0
            RET

ausgabe     ENDP

; ---------------------------------------------------------------------

eingabe     PROC

            COMMENT                     *

            Zuerst wird ein String ('Geben Sie die gr”áte ...') ausgegeben.
            Danach erfolgt die Eingabe von n. Mit dem 2maligen Aufruf
            der Prozedur "writeln" wird eine Leerzeile erreicht.
            Es werden die Register AX und DX ver„ndert.

            *

            MOV  AH,09                  ; fr Ausgabe eines Strings
            MOV  DX,OFFSET text         ; Adresse in DS:DX
            INT  21h                    ; DOS - Interrupt

            CALL readW                  ; word eingeben
            MOV  n,AX                   ; und speichern

            CALL writeln                ; Cursor in n„chster Zeile
            CALL writeln                ; jetzt Leerzeile

            RET
```

```
eingabe     ENDP

    x       DB    ?

PROG        ENDS


END         START
```