

Assembler listing: DUMPA.ASM

```

;*****;
;*                               D U M P A                               *;
;*-----*
;*   Task : Filter which reads characters from the standard input      *;
;*           device and outputs these characters as Hex and ASCII       *;
;*           dumps on the standard output device.                      *;
;*-----*
;*   Author      : Michael Tischer                                     *;
;*   Developed on : 08/01/87                                           *;
;*   Last update  : 04/07/95                                           *;
;*-----*
;*   Assembly    : MASM DUMPA;                                         *;
;*               LINK DUMPA;                                           *;
;*               EXE2BIN DUMPA DUMPA.COM                               *;
;*               - or -                                               *;
;*               TASM DUMPA                                           *;
;*               TLINK /T DUMPA                                         *;
;*-----*
;*   Call        : DUMPA [<Input] [>Output]                           *;
;*****;

```

== Constants =====

```

NUL      equ 0                ;ASCII code for NUL character
BEL      equ 7                ;ASCII code for Bell
BS       equ 8                ;ASCII code for Backspace
TAB      equ 9                ;ASCII code for Tab
LF       equ 10               ;ASCII code for Linefeed
CR       equ 13               ;ASCII code for Carriage Return
EOF      equ 26               ;ASCII code for End of File
ESCAPE   equ 27               ;ASCII code for Escape

```

```

;== Program starts here =====
code      segment para 'CODE'      ;Definition of CODE segment
        org 100h

        assume cs:code, ds:code, es:code, ss:code

;-- Start routine -----
dump      label near

        ;-- Read 9 bytes from standard input device -----
xor  bx,bx                ;Standard input has the handle 0
mov  cx,9                 ;Read 9 characters
mov  dx,offset ninebyte  ;Buffer address
mov  ah,3Fh              ;Function code for handle reading
int  21h                 ;Call DOS function
or   ax,ax               ;Characters read?
jne  dodump              ;Yes --> Process line
jmp  dumpend             ;No --> DUMPEND

dodump:   mov  dx,ax       ;Record number of characters read

        ;-- Fill output buffer with spaces -----
mov  cx,15               ;15 words (30 bytes)
mov  ax,2020h            ;ASCII code of " " to AH and AL
mov  di,offset dumpbuf   ;Output buffer address
cld                      ;Increment on string commands

```

```

rep stosw                ;Fill buffer with spaces

;-- Construct output buffer -----
mov  cx,dx                ;Get number of characters read in
mov  di,offset dumpbuf+31 ;Position ASCII codes in buffer
mov  bx,offset ninebyte   ;Pointer to input buffer
mov  si,offset dumpbuf    ;Position for hex codes in buffer

bytein: mov  ah,[bx]        ;Read byte
        push si            ;Store SI on stack
        mov  si,offset sotab ;Address of special character table
        mov  dx,offset sotext-6 ;Address of special character text
sotest: add  dx,6           ;Next entry in special text
        lodsb             ;Load code from special char table
        cmp  al,255        ;Reached end of table?
        je   noso         ;Yes --> No special character
        cmp  ah,al         ;Do codes match?
        jne  sotest        ;No --> Test next table element

;-- Code was a special character -----

push cx                ;Store counter
mov  si,dx             ;Copy DX to SI
lodsb                 ;Read number of char control codes
mov  cl,al             ;Transfer number of characters to CL
rep  movsb             ;Copy designation into buffer
pop  cx               ;Get counter
pop  si               ;Return SI from stack
mov  al,ah             ;Copy character to AL
jmp  short hex         ;Calculate hex code

```

```

noso:      pop     si                ;Return SI from stack
           mov     al,ah             ;Copy character to AL
           stosb                    ;Store in buffer

hex:        mov     al,ah            ;Character code to AL
           and     ah,1111b          ;Mask upper 4 bits in AH
           shr     al,1              ;Shift AL right 4 bits
           shr     al,1
           shr     al,1
           shr     al,1
           or      ax,3030h          ;Convert AH and AL into ASCII codes
           cmp     al,"9"            ;Is AL a letter ?
           jbe     nobal             ;No --> No correction
           add     al,"A"-"1"-9      ;Correct AL
nobal:      cmp     ah,"9"            ;Is AH a letter ?
           jbe     hexout            ;No --> No correction
           add     ah,"A"-"1"-9      ;Correct AH
hexout:     mov     [si],ax          ;Store hex code in buffer
           add     si,3              ;Point to next position

           inc     bx                ;Set pointer to next byte
           loop    bytein            ;Process next byte

           mov     al,219            ;Set separator
           stosb

           mov     ax,LF shl 8 + CR  ;CR and LF terminate buffer
           stosw                    ;Write in buffer

           ;-- Send dump to the standard output device -----

           mov     bx,1              ;Standard output is handle 1

```

```

        mov     cx,di                ;Determine number of characters
        sub     cx,offset dumpbuf    ;to be transmitted
        mov     dx,offset dumpbuf    ;Buffer address
        mov     ah,40h               ;Function code for handle
        int     21h                  ;Call DOS function
        jmp     dump                  ;Read in next 9 bytes

dumpend  label near

        mov     ax,4C00h              ;Function number for ending program
        int     21h                  ;End program with end code

;== Data =====

ninebyte db 9 dup (?)                ;The 9 bytes read in
dumpbuf  db 30 dup (?), 219          ;the output buffer
        db 49 dup (?)

sotab    db NUL,BEL,BS,TAB           ;Control character table
        db LF,CR,EOF,ESCAPE
        db 255

sotext   equ this byte               ;Text of special characters
        db 5,"<NUL>"                 ;Null
        db 5,"<BEL>"                 ;Bell
        db 4,"<BS> "                 ;Backspace
        db 5,"<TAB>"                 ;Tab
        db 4,"<LF> "                 ;Linefeed
        db 4,"<CR> "                 ;Carriage Return
        db 5,"<EOF>"                 ;End of File
        db 5,"<ESC>"                 ;Escape

```

```
== End =====
```

```
code      ends      ;End of CODE segment
          end  dump
```