

# Assembler listing:TESTEXE.ASM

```
;*****;
;*          T E S T E X E . A S M          *;
;*-----*
;* Task           : Simple EXE program: can be assembled using *;
;*               : either Turbo Assembler (TASM) or Microsoft's *;
;*               : Macro Assembler (MASM) *;
;*-----*
;* Author        : MICHAEL TISCHER *;
;* Developed on   : 06/07/1987 *;
;* Last update    : 04/07/1995 *;
;*-----*
;* Assembly      : MASM:      masm testexe; *;
;*               :           link testexe; *;
;*               *;
;*               TASM:      tasm testexe *;
;*               tlink testexe *;
;*****;
```

== Stack ==

```
stackseg segment para STACK 'STACK' ;Definition of stack segment
```

```
dw 256 dup (?) ;The stack comprises 256 words
```

```
stackseg ends ;End of stack segment
```

== Data ==

```
data segment para 'DATA' ;Definition of data segment
```

```
!-- All data, buffers and variables can be stored here -----
```

```

        ;...
        ;...
        ;...

data      ends                                ;End of data segment

;== Code =====

code      segment para 'CODE'                ;Definition of CODE segment

        assume cs:code, ds:data, ss:stackseg

                                ;CS defines the code segment, DS
                                ;DS the data segment and SS the stack
                                ;segment. ES can be accessed freely

prog      proc far                          ;This procedure is the main routine,
                                ;and is accessed right after the start
                                ;of the program

        ;-- CS and SS are already initialized. DS must be initialized
        ;-- manually, because it pints to the PSP (like ES)

        mov  ax,data                      ;Load segment address of data segment
        mov  ds,ax                       ;into the DS register

        call setfree                      ;Release memory not needed

        ;-- Place additional main program code here -----

        ;...

```

```

        i...
        i...

        ;--- End program here using DOS function 4Ch -----

mov     ax,4C00h           ;Load function number and error code 00
int     21h               ;DOS call

        ;--- Program execution stops here because of DOS call -----

prog     endp              ;End of PROG procedure

;-- Subroutines -----
;-- This is the area provided for subroutines within the program -----

a_proc   proc near

        i...
        i...
        i...
        ret

a_proc   endp

b_proc   proc near

        i...
        i...
        i...
        ret

b_proc   endp

```

```

;-- SETFREE: Release unused memory -----
;-- Input   : ES = Address of PSP
;-- Output  : none
;-- Registers: AX, BX, CL and FLAGS are affected
;-- Info    : Since the stack segment is always the last segment in an
;             EXE file, ES:0000 points to the beginning of the program
;             in memory, and SS:SP points to the end. This allows easy
;             calculation of the program's length.

```

```

setfree  proc near

```

```

    mov  bx,ss           ;Compute distance between
    mov  ax,es           ;PSP and stack
    sub  bx,ax

    mov  ax,sp           ;Compute stack length
    add  ax,15           ;in paragraphs
    mov  cl,4
    shr  ax,cl           ;Stack length

    add  bx,ax           ;Add two values of current length

    mov  ah,4ah          ;Reserve this memory only
    int  21h            ;DOS call

    ret                 ;Return to caller

```

```

setfree  endp

```

```

;== End =====

```

code

ends

end prog

;End of CODE segment

;Begin execution with PROG procedure