## SOLUCIONES A LAS ACTIVIDADES DE LA PRÁCTICA 4: MACROS

**NOTA:** las soluciones a las actividades son propuestas orientativas ya que a la hora de programar no existe una única solución; plagiar la solución no tiene ningún sentido, utilícela para corregir algún error o mejora su versión

### B) Crea una macro que multiplique por 10 usando desplazamientos.

```
;-----------------------------------------------
; pr4-ac02.asm
;-----------------------------------------------
; - Macro que multiplica por 10 un factor byte
;-----------------------------------------------

              TITLE  pr4-ac02
              DOSSEG
              .MODEL SMALL

              INCLUDE macros.inc

;-----------------------------------------------
; @factorx10: multiplica 'factor' por 10
; parámetros:
; factor -> número sin signo de tamaño byte
; devuelve: producto en AX
; registros: AX, BX
;-----------------------------------------------
@factorx10          MACRO factor

              IF TYPE (factor) EQ 1      ;; si factor es byte
                     XOR AH, AH
                     MOV AL, factor      ;;salvo el factor en el acumulador
                     SHL AX, 1           ;;multiplico por 2
                     MOV BX, AX          ;;salvo el resultado en BX
                     SHL AX, 1           ;;multiplico por 4
                     SHL AX, 1           ;;multiplico por 8
                     ADD AX, BX          ;;factorx8 + factorx2 = factorx10
              ELSE
                     .ERR
                     %OUT parametro ilegal
              ENDIF
              ENDM

              .STACK 100h
              .DATA
factor        DB 237
producto      DW 0

              .CODE
inicio:       @iniciarDS
              @factorx10 factor
              mov producto, AX
              @fincodigo
              END
```

## C) Crea una macro que divida entre 256 usando desplazamientos.

```
;-----------------------------------------------
; pr4-ac03.asm
;-----------------------------------------------
; - Macro que divide entre 256 un factor word
;-----------------------------------------------

              TITLE  pr4-ac03.asm
              DOSSEG
              .MODEL SMALL

              INCLUDE macros.inc

;-----------------------------------------------
; @dividendo%256: divide 'dividendo' entre 256
; parámetros:
; dividendo -> número sin signo de tamaño word
; devuelve: cociente en AL y resto en AH
; registros: AX
;-----------------------------------------------
@dividendo%256      MACRO dividendo

              IF TYPE (dividendo) EQ 2   ;;si dividendo es word
                     MOV AX, dividendo   ;;cargo el dividendo
                     XCHG AL, AH         ;;igual a desplazar 8; AX/256
              ELSE
                     .ERR
                     %OUT parametro ilegal
              ENDIF
              ENDM

              .STACK 100h
              .DATA
dividendo     DW 15862
cociente      DB 0
resto         DB 0

              .CODE
inicio:       @iniciarDS
              @dividendo%256 dividendo
              mov cociente, AL
              mov resto, AH
              @fincodigo
              END
```

## D) Utiliza macros para obtener los 2 caracteres ASCII de la codificación hexadecimal de un entero sin signo de tamaño byte.

```
;-----------------------------------------------
; pr4-ac04.asm
;-----------------------------------------------
; - Macro que devuelve los dígitos de una
;   codificación hexadecimal de 1 byte
; - Macro que devuelve los caracteres ASCII de
;   dígitos hexadecimales
;-----------------------------------------------
```

```
            TITLE pr4-ac04.asm
            DOSSEG
            .MODEL SMALL

;-------------------------------------------------
; @entero2hex: obtiene 2 dígitos hexadecimales de
; un entero de tamaño byte
; parámetros:
; entrada -> operando de tamaño byte
; salida -> buffer de memoria de tamaño word
; devuelve: 2 dígitos hexadecimales en el buffer
; registros: AX, CL
;-------------------------------------------------
@entero2hex         MACRO entrada, salida

            ;;si entrada byte y salida word
            IF (TYPE entrada EQ 1) AND (TYPE salida EQ 2)
                  MOV    AL, entrada
                  MOV    AH, AL
                  AND    AL, 0Fh       ;;dígito de menor peso
                  MOV    byte ptr salida, AL
                  MOV    CL, 4
                  SHR    AH, CL        ;;dígito de mayor peso
                  MOV    byte ptr salida[1], AH
            ELSE
                  .ERR
                  %OUT parametro ilegal
            ENDIF
            ENDM

;-------------------------------------------------
; @digito2ASCII: obtiene el ASCII de un dígito
; parámetros:
; entrada -> operando de tamaño byte
; salida -> posición de memoria
; devuelve: el ASCII del dígito
; registros: AL
;-------------------------------------------------
@digito2ASCII       MACRO entrada, salida
            local numero, escribir

            ;;si entrada y salida bytes
            IF (TYPE entrada EQ 1) AND (TYPE salida EQ 1)
                  MOV    AL, entrada
                  CMP    AL, 10
                  JB     numero
                  ADD    AL, 'A'-10
                  JMP    escribir
numero:           ADD    AL, 030h
escribir:         MOV    byte ptr salida, AL
            ELSE
                  .ERR
                  %OUT parametro ilegal
            ENDIF
            ENDM
```

```
                .STACK 100h
                .DATA
dato            DB 0A7h
wbuffer         LABEL word
buffer          DB 0, 0
;bbuffer        LABEL byte
;buffer         DW 0
cadenaASCII     DB 0, 0

                .CODE
inicio:         MOV    AX, @DATA
                MOV    DS, AX

                @entero2hex dato, wbuffer
                ; el dígito de menor peso va el último en la cadena
                @digito2ASCII buffer[0], cadenaASCII[1]
                @digito2ASCII buffer[1], cadenaASCII[0]

                ;@entero2hex dato, buffer
                ;@digito2ASCII bbuffer[0], cadenaASCII[1]
                ;@digito2ASCII bbuffer[1], cadenaASCII[0]

                MOV    AX, 4C00h
                INT    21h

                END
```