SOLUCIONES A LAS ACTIVIDADES DE LA PRÁCTICA 7: ARITMÉTICA MULTIPRECISIÓN

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

A) Escriba un programa que realice un desplazamiento unitario a la izquierda sobre un número de precisión arbitraria. Evalúe si el resultado es cero y si existe acarreo

```
;-----------------------------------------------
; pr6-ac01.asm
;-----------------------------------------------
; - El programa realiza un desplazamiento unitario
;   a la izquierda sobre un entero de precisión
;   arbitraria.
; - Salva en sendas variables de tamaño byte el
;   acarreo final y si el resultado es nulo.
;-----------------------------------------------

              DOSSEG
              .MODEL SMALL

              .STACK 100h

              .DATA
longitud      EQU    20

enteroX       DW     longitud DUP("21"),0D0Ah,'$'      ;declaro word: 13,10 = 0D0Ah
enteroY       DW     longitud DUP(0),0D0Ah,'$'
banderas      DB     "acarreo: "
acarreo       DB     "0"
              DB     " cero: "
cero          DB     "1",13,10,'$'

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

              MOV    CX, longitud
              XOR    SI, SI
              CLC                              ;clear flag de acarreo
              SAHF                             ;salvo los flags
otro:         MOV    DX, enteroX[SI]           ;cargo una palabra del bignum
              LAHF                             ;recupero el flag de acarreo
              SHL    DX, 1                     ;desplazamiento unitario
              SAHF                             ;salvo los flags
              JZ     salvar                    ;evalúo si el resultado es nulo
              MOV    cero, '0'
salvar:       MOV    enteroY[SI], DX
              ADD    SI, 2
              loop   otro

              LAHF
              JNC    presentar                 ;evalúo si hay acarreo de salida
              MOV    acarreo, '1'
```

```
presentar:    LEA    DX, enteroX
              MOV    AH, 09h
              INT    21h

              LEA    DX, enteroY
              ;MOV   AH, 09h
              INT    21h

              LEA    DX, banderas
              ;MOV   AH, 09h
              INT    21h

              MOV    AX, 4C00h
              INT    21h

              END    inicio
```

B) Suma y resta multiprecisión

```
;-------------------------------------------------
; pr7-ac02.asm
;-------------------------------------------------
; - X e Y son 2 números de longitud 10 bytes y valor constante.
; - También tiene una variable Z de longitud 10
;   bytes para recibir los resultados.
; - Las operaciones a realizar son suma o resta
; - Los operandos pueden ser X, Y ó Z pero el
;   resultado siempre es Z.
; - Mediante unos diálogos se selecciona primero
;   la operación a realizar y luego los operandos.
; - Finalizada la operación, se muestran los
;   operandos y el resultado en pantalla.
; - También se deben mostrar las banderas de cero
;   y acarreo del resultado.
;-------------------------------------------------

              DOSSEG
              .MODEL SMALL

              .STACK 100h

              .DATA
longitud      EQU    10
mensaje0      DB     "Realiza una operación a elegir entre operandos a elegir",10,13
              DB     "y salva el resultado en Z (* opción por defecto)",10,13,'$'
mensaje1      DB     "Seleccione operación (*1:suma 2:resta 3:salir)",10,13,'$'
mensaje2      DB     "Seleccione operando A (*1:X 2:Y 3:Z)",10,13,'$'
mensaje3      DB     "Seleccione operando B (1:X *2:Y 3:Z)",10,13,'$'
enteroX       DB     longitud DUP(30h),10,13,'$'
enteroY       DB     longitud DUP(31h),10,13,'$'
enteroZ       DB     longitud DUP(00h),10,13,'$'
operandoA     DW     enteroX
operandoB     DW     enteroY
A             DB     "  "
operacion     DB     "+ "
B             DB     "  ",10,13,'$'
banderas      DB     "acarreo: "
acarreo       DB     "  "
              DB     "cero: "
cero          DB     "  ",10,13,'$'
```

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

comienzo:   LEA    DX, mensaje0
            MOV    AH, 09h
            INT    21h

            LEA    DX, mensaje1
            MOV    AH, 09h
            INT    21h

            MOV    AH, 07h             ;entrada de carácter sin eco
            INT    21h

            MOV    operacion, '+'
            CMP    AL, '3'
            JNE    seguir0
            JMP    salir
seguir0:    CMP    AL, '2'
            JNE    seguir1
            MOV    operacion, '-'

seguir1:    LEA    DX, mensaje2
            MOV    AH, 09h
            INT    21h

            MOV    AH, 07h             ;entrada de carácter sin eco
            INT    21h

            MOV    operandoA, OFFSET enteroX
            MOV    A, 'X'
            CMP    AL, '2'
            JNE    seguir2
            MOV    operandoA, OFFSET enteroY
            MOV    A, 'Y'
seguir2:    CMP    AL, '3'
            JNE    seguir3
            MOV    operandoA, OFFSET enteroZ
            MOV    A, 'Z'

seguir3:    LEA    DX, mensaje3
            MOV    AH, 09h
            INT    21h

            MOV    AH, 07h             ;entrada de carácter sin eco
            INT    21h

            MOV    operandoB, OFFSET enteroY
            MOV    B, 'Y'
            CMP    AL, '1'
            JNE    seguir4
            MOV    operandoB, OFFSET enteroX
            MOV    B, 'X'
seguir4:    CMP    AL, '3'
            JNE    seguir5
            MOV    operandoB, OFFSET enteroZ
            MOV    B, 'Z'
```

```
seguir5:    MOV   CX, longitud/2
            XOR   SI, SI
            MOV   cero, '1'           ;la bandera de cero a '1' indica que es cero
            CLC
            PUSHF                     ;pongo el CF a 0 y guardo estado en la pila
bucle:      MOV   BX, operandoA       ;leo componente de operandos
            MOV   AX, [BX+SI]
            MOV   BX, operandoB
            MOV   DX, [BX+SI]
            CMP   operacion, '-'      ;sumo o resto según esté configurado
            JE    resta
            POPF                      ;cargo el estado (por el acarreo)
            ADC   AX, DX
            JMP   comun
resta:      POPF
            SBB   AX, DX
comun:      PUSHF                     ;guardo estado en la pila (por el acarreo)
            JZ    salvar
            MOV   cero, '0'
salvar:     MOV   word ptr enteroZ[SI], AX   ;salvo la componente del resultado
            ADD   SI, 2
            LOOP  bucle

            MOV   acarreo, '0'
            POPF                      ;recupero las banderas de la última operación
            JNC   seguir6
            MOV   acarreo, '1'

seguir6:    MOV   AH, 09h
            LEA   DX, A               ;presento la operación que realizo
            INT   21h

            MOV   DX, operandoA       ;escribo el enteroX
            INT   21h

            MOV   DX, operandoB       ;escribo el enteroY
            INT   21h

            LEA   DX, enteroZ         ;escribo el enteroZ
            INT   21h

            MOV   AH, 09h
            LEA   DX, banderas        ;presento las banderas
            INT   21h

            JMP   comienzo

salir:      MOV   AX, 4C00h
            INT   21h

            END   inicio
```

C) Suma de números de precisión arbitraria en BCD

```
;-----------------------------------------------
; pr7-ac03.asm
;-----------------------------------------------
; - El programa pide 2 enteros decimales de
;   longitud arbitraria no superior a 80 caracteres.
; - La entrada es bloqueante, carácter a carácter
;   y sin eco. Se comprueba si el carácter es
;   numérico y si lo es se hace eco y se guarda.
;   El carácter ENTER finaliza la captura.
; - Cada vez que se introduce un carácter no
;   numérico se emite un pitido.
; - Los caracteres ASCII se convierten a BCD y se
;   suman ambos operandos.
; - Finalmente se presenta el resultado
;-----------------------------------------------

              DOSSEG
              .MODEL SMALL

              .STACK 100h

              .DATA
longitud      EQU    12     ;no debe ser superior a 99

mensaje1      DB     10, 13, "Introduce 2 enteros decimales de tamaño "
              DB     10, 13, "arbitrario no superior a "
              DB     (longitud/10)+30h, ((longitud mod 10)mod 10)+30h
              DB     " caracteres",13,10,'$'
COMMENT #
mensaje1      DB     10, 13, "Introduce 2 enteros decimales de tamaño "
              DB     10, 13, "arbitrario no superior a 80 caracteres",13,10,'$'
#
mensaje2      DB     10, 13, "Entero ",'$'
mensaje3      DB     ":",13,10,'$'
enteroX       DB     longitud DUP(0)
enteroY       DB     longitud DUP(0)
suma          DB     longitud+1 DUP(0), '$'

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

              MOV    AH, 09h             ;emito mensaje1
              LEA    DX, mensaje1
              INT    21h

              MOV    BX, OFFSET enteroX  ;primer entero

lee_operando: MOV    AH, 09h             ;emito mensaje2
              LEA    DX, mensaje2
              INT    21h

              MOV    DL, 'X'
              CMP    BX, OFFSET enteroX
              JZ     XoY
```

```
            MOV    DL, 'Y'

XoY:        MOV    AH, 02H             ;escribo X o Y
            INT    21h

            MOV    AH, 09h             ;emito mensaje3
            LEA    DX, mensaje3
            INT    21h

            MOV    DL, 20h             ;ASCII del espacio en blanco
            MOV    AH, 02H             ;dejo en espacio en blanco
            INT    21h


;COMIENZA LA CAPTURA DE DÍGITOS

            XOR    DI, DI
            MOV    CX, longitud

lee_tecla:  MOV    AH, 08h             ;leo carácter sin eco
            INT    21h

            CMP    AL, 0               ;compruebo si es carácter especial
            JZ     lee_tecla

            CMP    AL, 13              ;compruebo si es ENTER
            JZ     fin_oper

            CMP    AL, 30h             ;compruebo que es una cifra
            JB     lee_tecla           ;si no es menor que 30h ('0')
            CMP    AL, 39h             ;no mayor que 39h ('9')
            JA     lee_tecla

            MOV    DL, AL
            MOV    AH, 02H             ;escribo el dígito en la pantalla
            INT    21h

            PUSH   AX                  ;salvo el dígito en la pila
            INC    DI
            LOOP   lee_tecla


fin_oper:   CMP    DI, 0               ;no se ha leído ningún carácter
            JZ     otro_oper

            MOV    CX, DI
            MOV    DI, longitud
salva_oper: POP    AX
            DEC    DI
            AND    AL, 0Fh
            MOV    [BX+DI], AL
            LOOP   salva_oper

otro_oper:  CMP    BX, OFFSET enteroY
            JZ     operar
            MOV    BX, OFFSET enteroY  ;segundo entero
            JMP    lee_operando
```

```
operar:       MOV    CX, longitud
              MOV    SI, longitud
              XOR    AX, AX

;REALIZO LA SUMA BCD

otro_digito:  CLC                       ;paso el posible acarreo en AH
              CMP    AH, 0              ;al flag de acarreo
              JZ     sumar_digito
              STC
              MOV    AX, 0
sumar_digito: MOV    AL, enteroX[SI-1]
              ADC    AL, enteroY[SI-1]
              AAA
              OR     AL, 30h
              MOV    suma[SI], AL       ;guardo el dígito BCD como ASCII
              DEC    SI
              LOOP   otro_digito

              OR     AH, 30h            ;último acarreo
              MOV    suma[SI], AH


              MOV    AH, 09h            ;emito CR, LF
              LEA    DX, mensaje3+1
              INT    21h

              MOV    AH, 09h            ;emito CR, LF
              LEA    DX, mensaje3+1
              INT    21h

              MOV    CX, longitud
              XOR    DI, DI

buscar_ceros: CMP    suma[DI], 30h
              JNZ    presentar
              INC    DI
              LOOP   buscar_ceros

presentar:    MOV    AH, 09h            ;presento la cadena del resultado
              LEA    DX, suma[DI]
              INT    21h

              MOV    AX, 4C00h
              INT    21h

              END    inicio
```