SOLUCIONES AL EXAMEN DE

LABORATORIO DE ESTRUCTURAS DE LOS COMPUTADORES

CURSO 95/96. FEBRERO 1.996

Parte de Ensamblador.

4º)Escribir un programa en ensamblador que pida un número de una cifra por teclado y saque como resultado el doble de signos menos por pantalla seguido de ese número de signos más.

Signo	Código ASCII
+	2Bh
-	2Dh

Ejemplo:
c:\> **ejer1**
6
----------++++++ (son 12 signos - y 6 signos +)
c:\>

Solución:

```
;El programa lo realizaremos de la siguiente manera: leemos la tecla; convertimos la
;tecla de su código ASCII al número correspondiente, el número lo almacenamos en una
;posición de mem. etiquetada como NUMERO. Lo multiplicamos por dos, y visualiza-
;mos los menos, recuperamos el número y visualizamos el más.

GRUPO_SEG       GROUP     SEG_CODIGO,SEG_DATOS
    ASSUME      CS:GRUPO_SEG, DS:GRUPO_SEG

SEG_CODIGO      SEGMENT   PUBLIC
    ORG 100h
PRINCIPAL       PROC      FAR
    MOV AH,01             ;Cargamos en AH el 1 para la fn. 1 de int 21h
    INT 21h               ;Leemos el número, queda su cod. ASCII en AL
    SUB AL,30h            ;Restamos 30h para convertirlo al número
    CMP AL,9              ;Si el número está entre 0-9 ya lo tenemos
    JBE NUM_CONVERTIDO    ;Número entre 0-9 seguimos el programa
    SUB AL,7              ;Restamos 7 si su rango está de A-F
NUM_CONVERTIDO:           ;Continuamos con el prg.
    MOV NUMERO,AL         ;Almacenamos en mem el número leído
    MOV AH,2              ;Ponemos en AH el dos para calcular el doble
    MUL AH                ;AL*AH -> AL . AL tiene el doble del número
    MOV CX,AX             ;CX hará de contador en el bucle.
    MOV DL,MENOS          ;Cargamos en DL el cod. ASCII a visualizar
    MOV AH,02             ;En AH ponemos la fn. dos de la int 21h
IMP:
    INT 21h               ;Visualizamos el dígito que hay en DL
    LOOP IMP              ;Restamos 1 a CX, si <> 0 a IMP, sino seguimos
    MOV CL,NUMERO         ;Recuperamos el número en el contador
    XOR CH,CH             ;Borramos la parte alta (Podría no hacer falta)
    MOV DL,MAS            ;En DL el cod ASCII del signo más
    MOV AH,02             ;Cargamos la fn 2, int21h (no haría falta)
```

```
IMP2:
    INT 21h                     ;Visualizamos el dígito que hay en DL
    LOOP IMP2                   ;Restamos 1 a CX,si <> 0 a IMP2, sino seguimos
    INT 20h                     ;Fin del prg., devolvemos el control al DOS
PRINCIPAL       ENDP

SEG_CODIGO      ENDS

SEG_DATOS       SEGMENT
    MAS     EQU 2Bh             ;No se reserva mem. al compilar, donde se en-
    MENOS   EQU 2Dh             ;cuentre MAS o MENOS se sustituye por el cod.
                                ;correpondiente
    NUMERO      DB      0       ;Reservamos mem. para almacenar el NUMERO
                                ;leído del teclado
SEG_DATOS ENDS

        END   PRINCIPAL
```

2º)Escribir un programa en ensamblador que lea un número de una cifra y que saque la suma de ese número con todos los anteriores hasta el cero.

Ejemplo:
c:\> **sumador**
7
1C
c:\>

c:\> **sumador**
A
37
c:\>

Solución:

```
;El programa se realizará de la siguiente manera: leemos la tecla, convertimos del
;código ASCII a número, y mediante un bucle sumamos todos. Posteriormente
;visualizamos los dos bytes de la suma.

GRUPO_SEG       GROUP       SEG_CODIGO, SEG_DATOS
    ASSUME      CS:GRUPO_SEG, DS:GRUPO_SEG

SEG_CODIGO      SEGMENT     PUBLIC
    ORG 100h
PRINCIPAL       PROC        FAR
    MOV AH,01               ;Cargamos la fn uno de la int 21H
    INT 21h                 ;Leemos la tecla
    SUB AL,30h              ;Restamos 30H al cód. ASCII
    CMP AL,9                ;Si el nº está entre 0-9 ya lo tenemos.
    JBE NUM_CONVERTIDO      ;Nº entre 0-9 seguimos el programa.
    SUB AL,7                ;Si el número está entre A-F le restamos 7
```

```
NUM_CONVERTIDO:                 ;Ya tenemos el número en AL.
    XOR AH,AH                   ;Borramos AH.
    MOV CX,AX                   ;CX será el contador, le introducimos el valor
SUMAR_NUM:                      ;Comenzamos el bucle para sumar
    ADD SUMA,CL                 ;Sumamos en la var. de mem. SUMA el contador
    LOOP SUMAR_NUM              ;Restamos uno a CX,si es <>0 volvemos al bucle
    MOV DL,SUMA                 ;La suma total la pasamos a DL para imprimir
    MOV DH,DL                   ;Hacemos una copia de seguridad
    MOV CL,4                    ;CL tiene el nº de bits a desplazar para visualizar el
                                ;nibble alto
    SHR DL,CL                   ;Ponemos el nibble alto a cero y en el bajo el alto
    CALL IMPRIMIR               ;Imprim. nibble alto (pasado por param. en DL)
    MOV DL,DH                   ;Recuperamos el nº a visualizar
    AND DL,0Fh                  ;Anulamos nibble alto y dejamos el bajo
    CALL IMPRIMIR               ;Visualizamos el nibble bajo
    INT 20h                     ;Fin y devolvemos el control al DOS
PRINCIPAL   ENDP

IMPRIMIR    PROC        NEAR
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; Este procedimiento imprime un dígito, cuyo valor será el contenido en DL al llamar al
; ;procedimiento. Esto implica DL entre 0 y F.                                          ;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
    PUSH DX                     ;Almacenamos DX en pila
    PUSH AX                     ;Almacenamos AX en pila
    MOV AH,02                   ;Cargamos en AH la fn. 2 de int 21H
    ADD DL,30h                  ;Sumamos a DL 30h para transformarlo a ASCII
    CMP DL,39h                  ;Si el nº está entre 0-9 ya tenemos cod ASCII
    JBE IMP                     ;Imprimimos si nº entre 0-9
    ADD DL,7                    ;Si está entre A-F sum. 7 para el cod. ASCII
IMP:
    INT 21h                     ;Visualizamos el número
    POP AX                      ;Recuperamos el valor que AX tenía al entrar en
                                ;proc.
    POP DX                      ;Recuperamos el valor que AX tenía al entrar en
                                ;proc.
    RET                         ;Volvemos
IMPRIMIR    ENDP
SEG_CODIGO      ENDS
SEG_DATOS       SEGMENT
    SUMA    DB  0               ;Guardamos un byte para el total de la suma
SEG_DATOS       ENDS
        END PRINCIPAL
```

3º)Escribir un programa en ensamblador que lea un número de cuatro cifras por teclado y devuelva como resultado:

- 0 si no es un número capicúa.
- 1 si es un número capicúa.

Ejemplo:

c:\> **capicúa**
2125
0
c:\>
c:\> **capicúa**
2112
1
c:\>

- **Nota:** se sugiere el uso de la pila para su resolución.

Solución:

```
;El programa se ha realizado de la siguiente manera: leeremos la mitad de los dígitos, y
;los almacenaremos en la pila. Después según vayamos leyendo del teclado la otra
;mitad, sacamos el dígito que está en lo alto de la pila, y se comparan (deben ser
;iguales). En cuanto uno sea distinto se pone una posición de mem. a cero, sino
;permanecerá a uno (eso implica que son capicúas).

GRUPO_SEG       GROUP       SEG_CODIGO, SEG_DATOS
    ASSUMECS:GRUPO_SEG, DS:GRUPO_SEG

SEG_CODIGO      SEGMENT     PUBLIC
    ORG 100h
PRINCIPAL       PROC        NEAR
    XOR AX,AX                   ;Inicializamos AX a cero
    XOR DX,DX                   ;Inicializamos DX a cero
    MOV AL,NUM_CARACS           ;En AL cargamos el número de carac. a leer
    MOV CX,2                    ;Ponemos en CX el dos para calcular la mitad.
    DIV CL                      ;Calculamos la mitad de los carac. a leer
    MOV MITAD_CARACS,AL         ;Almacenamos en mem. el número que nos dice la
                                ;mitad de caracteres a leer
    MOV CL,MITAD_CARACS         ;Cargamos el contador
PEDIR_1a_MITAD:         ;Bucle para pedir la 1ª mitad de caracteres
    MOV AH,01                   ;Cargamos en AH la función uno de int 21h
    INT 21h                     ;Leemos el carácter
    XOR AH,AH                   ;Borramos AH
    PUSH AX                     ;Guardamos el carácter leído en pila
    LOOP PEDIR_1a_MITAD         ;Si CX <> 0 pedimos otro carac. sino seguimos
    MOV CL,MITAD_CARACS         ;Cargamos el contador para leer la 2ª mitad
PEDIR_2a_MITAD:         ;Bucle para pedir la 2ª mitad
    MOV AH,01                   ;Cargamos en AH el 1 para la fn. 1 de int 21h
    INT 21h                     ;Pedimos el número
    XOR AH,AH                   ;Borramos AH para la comparación
    POP DX                      ;Sacamos el dígito de la pila a DX
```

```
    CMP AX,DX                   ;Comparamos el número leído con el que le
                                ;corresponde
    JE DIGITOS_IGUALES          ;Si son iguales pedimos otro carac. ya que la
                                ;posición de mem ya está inicializada a uno
                                ;(capicúa)
    AND CAPICUA,00h             ;Si son distintos ponemos CAPICÚA a cero
DIGITOS_IGUALES:
    LOOP PEDIR_2a_MITAD         ;Si CX <> 0 pedimos otro carac. sino seguimos
    MOV DL,CAPICUA              ;El valor de capicúa a DL para visualizarlo
    ADD DL,30h                  ;Sumamos 30h para poner su cod. ASCII
    MOV AH,02h                  ;Movemos el 2 para la fn. 2 de int 21h
    INT 21h                     ;Visualizamos un 0 o un 1 según sea capicúa o no
    INT 20h                     ;Terminamos y devolvemos control a DOS
PRINCIPAL       ENDP
SEG_CODIGO      ENDS
SEG_DATOS SEGMENT
    NUM_CARACS      DB  4       ;Posición con el num. de carac. a leer
    MITAD_CARACS    DB  ?       ;Posición con el num. mitad de carac. a leer
    CAPICÚA         DB  1       ;Inicializamos a que los nºs son capicúa
SEG_DATOS ENDS
    END     PRINCIPAL
```

Parte de MS-DOS.

1º)Realizar un comando de MS-DOS, llamado ***enlazar.bat*** que reciba como parámetro el nombre de un fichero con extensión ASM, lo compile y lo enlace. El funcionamiento es como sigue:

Enlazar, sin parámetros, visualiza un mensaje de ayuda.

Enlazar /? visualiza un mensaje de ayuda.

Enlazar nombre_fichero_ASM llama al programa MASM con ese nombre de fichero y posteriormente, lo enlaza LINK. (No se debe dar la extensión del fichero)

El mensaje de ayuda que se debe visualizar en los dos primeros casos es:

> Objetivo: compila y enlaza un fichero fuente en ensamblador.
> Sintaxis: ***enlazar [nombre_fichero_ASM]***

Ejemplos:

- ***Enlazar*** visualiza el mensaje de ayuda
- ***Enlazar /?*** visualiza el mensaje de ayuda.
- ***Enlazar*** test1 compila y enlaza el fichero fuente test1.asm

Solución:

```
@ ECHO OFF
IF "%1" == "" GOTO AYUDA
MASM %1 ;
LINK %1 ;
EXE2BIN %1.EXE %1.COM
GOTO FIN
```

```
:AYUDA
ECHO Objetivo:      compila y enlaza un fichero fuente en ensamblador
ECHO Sintaxis:      enlazar [nombre_fichero_ASM]
:FIN
```

2º)Realizar un comando de MS-DOS, llamado ***mover.bat*** que funcione como sigue:

Mover, sin parámetros, visualiza un mensaje de ayuda.

Mover /? visualiza un mensaje de ayuda.

Mover [F1 D1 [F2 D2 ....]] ***/B [F4 D1 D2 D3]***

El mensaje de ayuda que se debe visualizar en los dos primeros casos es:

> Objetivo: mueve un fichero a otro directorio
> Sintaxis: ***Mover [F1 D1 [F2 D2 ....***]] ***/B [F4 D3 D4 D5]***
>
> F1 es el fichero de origen 1, F2 es el fichero de origen 2, ...
> D1 es el directorio destino 1, D2 es el directorio destino 2, ...
> /B indica que el fichero origen F4 se copiará en los destinos D3 D4 D5

Ejemplos

- ***Mover*** visualiza el mensaje de ayuda.
- ***Mover /?*** visualiza el mensaje de ayuda.
- ***Mover test1.asm fuentes test2.obj objetos*** mueve el fichero *test1.asm* al directorio *fuentes* y el fichero *test2.obj* al directorio *objetos*.
- ***Mover test1.asm fuentes /B test2.asm fuentes c:\segurida d:\backup*** mueve el fichero *test1.asm* al directorio *fuentes* y el fichero *test2.asm* a los directorios *fuentes, segurida* del disco c: y al *backup* del disco d:

Solución:

```
@ ECHO OFF
IF "%1" == "" GOTO AYUDA
IF “%1” == “/?” GOTO AYUDA
:BUCLE1
IF “%1” == “/B” GOTO BUCLE2
IF "%1" == "" GOTO FIN
COPY %1 %2
DEL %1
SHIFT
SHIFT
GOTO BUCLE1
:BUCLE2
SHIFT
IF "%1" == "" GOTO FIN
COPY %1 %2
COPY %1 %3
COPY %1 %4
DEL %1
GOTO FIN
:AYUDA
ECHO Objetivo:      mueve un fichero a otro directorio
ECHO Sintaxis:      Mover [F1 D1 [F2 D2 ...] /B [F4 D1 D2 D3]
```

```
ECHO                F1 es el fichero de origen 1, F2 es el fichero de origen 2 ...
ECHO                D1 es el directorio destino 1, D2 es el directorio destino 2 ...
ECHO        /B indica que el fichero origen F4 se copiará en los destinos ECHO
      D3, D4, D5
:FIN
```

3º)Realizar un comando de MS-DOS, llamado contar***.bat*** que funcione como sigue:

Contar /? visualiza un mensaje de ayuda.

Contar, visualiza el número nn de ficheros ejecutables (EXE).

Ejemplos

Contar /? muestra el mensaje de ayuda.

Contar muestra el número 20.

Nota: se supone que en directorio actual hay 20 ficheros ejecutables (EXES)

Solución:

```
@ ECHO OFF
IF "%1" == "/?" GOTO AYUDA
DIR *.* | FIND /C /I "EXE"
GOTO FIN
:AYUDA
ECHO El comando Contar devuelve el número de ficheros ejecutables.
ECHO  Sintaxis: CONTAR
:FIN
```