Examen de Laboratorio de Estructura de Computadores
I.T. Informática de Gestión / Sistemas. 29 de enero de 2009
Departamento de Automática – ATC –

Apellidos, Nombre:

Grupo de laboratorio: Día____; Hora_____Gestión ☐ Sistemas ☐ Libre Elección ☐

Test (3 puntos)

Pregunta correcta = **0,3** Pregunta no contestada = **0** Pregunta incorrecta = **- 0,1**

1. En una arquitectura Von Neumann, el movimiento de información entre los diferentes elementos se realiza mediante:

☐ Los registros.

☐ La Unidad Aritmética.

☐ La unidad de Entrada/Salida.

☐ Los buses de datos, direcciones y control.

2. Un programa en lenguaje máquina se puede ejecutar:

☐ En cualquier máquina o microprocesador de cualquier familia y fabricante.

☐ Sólo en la máquina o microprocesador de la familia compatible para la que está escrito.

☐ Con el ensamblador adecuado, en cualquier máquina o microprocesador.

☐ Ninguna de las respuestas anteriores es correcta.

3. ¿Cuál de las secuencias de operaciones siguientes es suficiente para crear un fichero ejecutable a partir de un fichero fuente en ensamblador?

☐ Ensamblar, enlazar y depurar.

☐ Basta con ensamblar.

☐ Ensamblar y enlazar.

☐ Ensamblar y depurar.

4. En Windows 2000 una carpeta compartida en red y los permisos de dicha carpeta en el ordenador local que la contiene:

☐ Tienen que ser iguales.

☐ Tienen que ser diferentes.

☐ Pueden ser iguales o diferentes.

☐ Solamente se pueden establecer si eres el Administrador del sistema.

5. ¿Cual de las siguientes afirmaciones sobre el juego de instrucciones del i8086/88 es **incorrecta**?:

☐ El formato de una instrucción nos indica el significado de cada bit de la instrucción

☐ Todas las instrucciones deben tener código de operación.

☐ Sólo algunas instrucciones tienen operando fuente y operando destino

☐ Todas las instrucciones tienen el mismo tamaño en bytes

6. ¿En Windows cuál de las siguientes afirmaciones es **incorrecta**?

☐ Podemos eliminar un fichero seleccionándolo y dando a la tecla suprimir. Va a parar a la papelera y podremos recuperarlo.

☐ Podemos eliminar una carpeta seleccionándola y dando a la tecla suprimir. Va a parar a la papelera y podremos recuperarlo.

☐ Las dos respuestas anteriores son ciertas.

☐ Podemos eliminar una carpeta seleccionándola y pulsando las teclas Mayúsculas + Suprimir. Va a parar a la papelera y podremos recuperarlo.

7. Ejecutando la utilidad IPCONFIG.EXE con el parámetro /ALL obtenemos:

☐ La dirección IP del PC donde se ejecuta.

☐ La dirección MAC del PC donde se ejecuta.

☐ Las dos respuestas anteriores son ciertas.

☐ La dirección IP de cualquier PC de la red local a la que se esté conectado.

8. En una sesión de MS-DOS si se quiere borrar un grupo de nueve ficheros con nombres Fichero1.txt, Fichero2 txt, … Fichero9.txt, ejecutamos la orden:

☐ DEL *.txt

☐ DEL Fichero?.*

☐ DEL Fichero?. txt

☐ Las dos respuestas anteriores son ciertas.

9. Entre las funciones de un sistema operativo se encuentra:

☐ Proporcionar la interfaz de usuario.

☐ Planificar recursos.

☐ Facilitar la entrada-salida.

☐ Las tres respuestas anteriores son ciertas.

10. Las siglas PCI, ISA, hacen referencia a

☐ Los tipos de memoria que se pueden instalar en un PC

☐ Los conectores o *slots* de expansión de algunos tipos de bus de un PC

☐ El conector o *socket* donde se inserta el microprocesador

☐ Las tres respuestas anteriores son ciertas.

Programación en Ensamblador (7 puntos)

1. Sabiendo que ***mem8*** y ***mem16*** son datos definidos en memoria de 8 y 16 bits respectivamente y que una instrucción se considera incorrecta si genera mensajes del tipo *Severe Errors* y *Warning Errors* al ensamblar, se pide:

a) Indica qué instrucciones son **incorrectas** (pueden ser varias)

- ☐ `mov al,bl`
- ☐ `mov mem16,ax`
- ☐ `mov bl,mem16`
- ☐ `mov cl,mem8`
- ☐ `mov ch,129`
- ☐ `mov bx,mem8`
- ☐ `mov 127, bl`
- ☐ `mov al,ah`
- ☐ `mov dh,256`
- ☐ `mov ax,ax`

b) Indica qué instrucciones son **incorrectas** (pueden ser varias):

- ☐ `mov al,[si][di]`
- ☐ `mov ax,[di+4]`
- ☐ `mov cl,[bp][di]+4`
- ☐ `mov ch,[bx][di+3]`
- ☐ `mov ax,[dx][si]`
- ☐ `mov bl,[bx]`
- ☐ `mov dl,[bp][bx]`
- ☐ `mov di,mem16[si]`

2. A partir de las definiciones de datos siguientes:

```
Num1      DB 254
Num2      DB 34
Cociente  DB ?
Resto     DB ?
```

Escribe el fragmento de código necesario para realizar correctamente la división sin signo de **`Num1`** entre **`Num2`**, guardando el resultado en las variables **`Cociente`** y **`Resto`**.

Respuesta:

```
xor ah,ah           ;División por byte, dividendo en AX
mov al, Num1        ;ponemos el byte en AL y cero en AH
div Num2
mov Cociente, al
mov Resto, ah
```

3. El código fuente que se muestra a continuación es de un programa que pide una cadena de hasta 8 caracteres y a continuación la imprime en la pantalla.

```
DOSSEG
.MODEL SMALL
.STACK 100h
.DATA
  Texto DB "Introduce una frase de hasta 8 caracteres: ",13,10,'$'
  Otralinea DB 13,10,'$'
  MaximoMas1   DB 9
  CaracteresLeidos DB 0
  Cadena DB 9 DUP (0)

.CODE
Inicio:
       mov ax, @data
       mov ds, ax
       EscribirMsj Texto
       PedirTexto MaximoMas1
       EscribirMsj Otralinea
       mov al, CaracteresLeidos
       xor ah,ah
       push ax
       lea ax, Cadena
       push ax
       call ImprimirCad
       mov ah, 4Ch
       int 21h
END Inicio
```

a) Indica el contenido de la pila después de ejecutarse la instrucción anterior a la instrucción **`call ImprimirCad`**, sabiendo que la variable **`MaximoMas1`** está almacenada en la dirección 4184:0041, se han leído 7 caracteres incluido el **`<enter>`** y suponiendo que la estructura de la pila al comenzar la ejecución del programa es la siguiente:

SS:SP →4184:0100	
4184:00FF	**00**
4184:00FE	**06**
4184:00FD	**00**
4184:00FC	**43**
4184:00FB	
4184:00FA	
4184:00F9	

b) Escribe la macro **`EscribirMsj`** que imprime las cadenas **`Texto`** y **`OtraLinea`**.

```
EscribirMsj MACRO DirTexto
                    mov ah, 9
                    lea dx, DirTexto
                    int 21h
            ENDM
```

c) Escribe la macro **`PedirTexto`** que lee del teclado la cadena utilizando las definiciones de datos: **`MaximoMas1`**, **`CaracteresLeidos`** y **`Cadena`**.

```
PedirTexto  MACRO DirBuffer
                    mov ah, 0Ah
                    lea dx, DirBuffer
                    int 21h
            ENDM
```

d) Escribe el procedimiento **`ImprimirCad`** que recoge de la pila los parámetros necesarios para imprimir en la pantalla la cadena leída previamente.

Solución 1 imprimendo carácter a carácter

```
ImprimirCad PROC
                push bp
                mov bp,sp
                push ax
                push cx
                push dx
                push si
                mov cx,[bp+6]           ;Cantidad de caracteres a cx
                mov si, [bp+4]          ;Dirección de Cadena a SI
                mov ah, 2
                bucle:
                        mov dl,[si]
                        int 21h
                        inc SI
                loop bucle
                pop si
                pop dx
                pop cx
                pop ax
                pop bp
                ret 4
ImprimirCad ENDP
```

Solución 2 imprimendo la cedana con el DOS

```
ImprimirCad PROC
                push bp
                mov bp,sp
                push ax
                push dx
                push si
                mov bx,[bp+6]               ;Cantidad de caracteres
                mov si,[bp+4]               ;Dirección inicio cadena
                mov BYTE PTR [bx][si],'$'   ;Ponemos '$' al final
                mov dx,si
                mov ah,9                ;Servicio DOS para
                int 21h                 ;imprimir cadena apuntada por DX
                pop si
                pop dx
                pop ax
                pop bp
                ret 4
ImprimirCad ENDP
```

4. A continuación se muestran el código fuente y el código máquina de un programa en ensamblador tal y como lo muestra *CodeVIEW*.

```
1:      DOSSEG
2:      .MODEL SMALL
3:      .STACK 100h
4:      .DATA
5:      Cadena DB 'Este es el texto del que se cuentan las letras#'
6:      Letras DB 0
7:
8:      .CODE
9:
10:      mov ax,@DATA
11:      mov ds,ax
12:
13:      xor si,si
14:      Lazo:
15:              cmp Cadena[si],'#'
16:              je Salir
17:              cmp Cadena[si],' '
18:              je Espacio
19:              inc Letras
20:              Espacio:
21:              inc si
22:              jmp Lazo
23:      Salir:
24:      mov ah,4Ch
25:      int 21h
26:      END
```

```
10:      mov ax,@DATA
4009:0010 B80C40          MOV       AX,400C
11:      mov ds,ax
4009:0013 8ED8            MOV       DS,AX
13:      xor si,si
4009:0015 33F6            XOR       SI,SI
LAZO:
15:              cmp Cadena[si],'#'
4009:0017 80BC000023      CMP       Byte Ptr [SI+CADENA],23
16:              je Salir
4009:001C 740E            JZ        SALIR (002C)
17:              cmp Cadena[si],' '
4009:001E 80BC000020      CMP       Byte Ptr [SI+CADENA],20
18:              je Espacio
4009:0023 7404            JZ        ESPACIO (0029)
19:              inc Letras
4009:0025 FE062F00        INC       Byte Ptr [LETRAS (002F)]
ESPACIO:
21:              inc si
4009:0029 46              INC       SI
22:              jmp Lazo
4009:002A EBEB            JMP       LAZO (0017)
SALIR:
24:      mov ah,4Ch
4009:002C B44C            MOV       AH,4C
25:      int 21h
4009:002E CD21            INT       21
```

a) En qué dirección efectiva dentro del segmento de datos está el primer carácter del texto almacenado en **`Cadena`**:

Se encuentra en la dirección efectiva 0000

b) En qué dirección efectiva dentro del segmento de datos está la variable **`Letras`**:

Se encuentra en la dirección efectiva 002F

c) Indica el cambio a realizar en el código máquina de la instrucción de la línea 16 (740E) para que salte a la instrucción de la línea 21. Especificar también la orden que habría que dar en CodeVIEW para realizar el cambio:

Hay que cambiar 0E por 0B en la dirección 4009:001D.
En CodeVIEW habría que ordenar EB 0x4009:0x001D B

d) Indica la dirección donde se encuentra el segmento de datos:

400C

PUNTUACIÓN DE LA PROGRAMACIÓN EN ENSAMBLADOR
Ejercicio 1: 1,4 puntos (0,2 puntos por opción correcta; -0,2 puntos por opción incorrecta).
Ejercicio 2: 1 punto.
Ejercicio 3: a) 0,5 puntos; **b)** 0,75 puntos; **c)** 0,75 puntos; **d)** 1,6 puntos
Ejercicio 4: 1 punto. 0,25 puntos cada apartado.

PÁGINA EN BLANCO PARA BORRADOR