



Apellidos, Nombre:

Grupo de laboratorio:

Gestión

Sistemas

Libre Elección

Test (3 puntos)

Pregunta correcta= **0,3** Pregunta no contestada= **0** Pregunta incorrecta (tipo test)= **-0,15**

1) ¿Cual de las siguientes afirmaciones es **correcta**?

- a) En lenguaje máquina se representan las instrucciones con nombres simbólicos o mnemónicos
- b) **En lenguaje ensamblador cada instrucción se corresponde con una instrucción máquina**
- c) En lenguaje de alto nivel cada instrucción se corresponde con una instrucción máquina
- d) Todas son correctas

2) ¿Cual de las siguientes afirmaciones es **incorrecta**?

- a) Los buses interconectan la memoria principal con la unidad de control
- b) Los buses interconectan la unidad aritmética con la unidad de control
- c) **Los buses interconectan los diferentes registros**
- d) Existen bus de datos, bus de direcciones y bus de control

3) ¿Cual de las siguientes instrucciones **no modifican necesariamente** la secuencia normal de ejecución de un programa?

- a) JMP dir1
- b) **JNE dir1**
- c) CALL dir1
- d) RET

4) ¿Cual de las siguientes afirmaciones es **incorrecta**?:

- a) El formato de una instrucción nos indica el significado de cada bit de la instrucción
- b) Todas las instrucciones deben tener código de operación.
- c) **Todas las instrucciones deben tener operando fuente y operando destino**
- d) No todas las instrucciones tienen el mismo tamaño en bytes

5) Señale la opción **correcta** para Windows 2000:

- a) Para determinar la información de red debemos ejecutar el comando PING.EXE desde una sesión de MSDOS
- b) Para determinar la información de red debemos ejecutar el comando PINGCONFIG.EXE desde una sesión de MSDOS
- c) Para determinar la información de red debemos ejecutar el comando IPPING.EXE desde una sesión de MSDOS
- d) **Para determinar la información de red debemos ejecutar el comando IPCONFIG.EXE desde una sesión de MSDOS**



Apellidos, Nombre:

Grupo de laboratorio:

Gestión

Sistemas

Libre Elección

6) Para MSDOS. Señale la opción **correcta**:

- a) No podemos borrar directorios que contengan otros directorios sin borrar estos previamente
- b) El comando *REMOVE* sirve para borrar ficheros
- c) Desde el Símbolo de sistema (MSDOS) podemos compartir directorios
- d) **Los ficheros borrados con *ERASE* no se pueden recuperar**

7) ¿Cual de las siguientes afirmaciones es **correcta**?

- a) Tras ejecutar el comando *RENAME fich1 fich2* tendremos dos ficheros idénticos con los nombres fich1 y fich2
- b) Tras ejecutar el comando *MOVE fich1 fich2* tendremos dos ficheros idénticos con los nombres fich1 y fich2
- c) **Tras ejecutar el comando *COPY fich1 fich2* tendremos dos ficheros idénticos con los nombres fich1 y fich2**
- d) Son ciertas a) y c).

8) ¿Cual de las siguientes afirmaciones es **correcta**?

- a) Desde Windows podemos eliminar un fichero seleccionándolo y dando a la tecla suprimir, pero irá a parar a la papelera y podremos recuperarlo
- b) Desde Windows podemos eliminar una carpeta seleccionándola y dando a la tecla suprimir, pero irá a parar a la papelera y podremos recuperarlo
- c) Desde Windows podemos eliminar un fichero permanentemente pulsando Mayúsculas + Suprimir una vez seleccionado.
- d) **Todas son correctas**

9) Para Windows 2000. Señale la opción **incorrecta**.

- a) Para acceder al administrador de dispositivos debemos ejecutar previamente la aplicación Sistema
- b) Las propiedades del Sistema (dentro de panel de control) nos da información del sistema operativo del computador
- c) Las propiedades del Sistema (dentro de panel de control) nos da información de la memoria RAM del PC
- d) **No podemos acceder al administrador de dispositivos desde Windows 2000**

10) Señale la opción **correcta**:

- a) La placa base no necesita estar conectada a la fuente de alimentación
- b) **La memoria RAM no necesita estar conectada a la fuente de alimentación**
- c) El disco duro no necesita estar conectado a la fuente de alimentación
- d) El CD-ROM no necesita estar conectado a la fuente de alimentación



Apellidos, Nombre:

Grupo de laboratorio:

Gestión

Sistemas

Libre Elección

Ensamblador (7 puntos)

Cada pregunta vale 0,7 puntos

Dado el siguiente programa en ensamblador:

```
Dosseg
```

```
.model small
```

```
.stack 100h
```

```
.data
```

```
LF equ 0Ah
```

```
CR equ 0Dh
```

```
caracter_primer db 0
```

```
caracter_ulti db 255
```

```
Maxmas1 DB 255
```

```
CaracLeidos DB 0
```

```
Almacena DB 255 DUP(0)
```

```
Fin_cadena DB '$'
```

```
.code
```

```
escribe_caracter proc
```

```
mov ah,02
```

```
int 21h
```

```
ret
```

```
escribe_caracter endp
```

```
Inicio:
```

```
mov ax, @data
```

```
mov ds, ax
```

```
xor cx,cx
```

```
mov dl, [caracter_primer]
```

```
mov cl, [caracter_ulti]
```

```
imprime_loop:
```

```
    call escribe_caracter
```

```
    inc dl
```

```
    loop imprime_loop
```

```
mov ah, 4Ch
```

```
int 21h
```

```
end Inicio
```

1) ¿Qué se muestra en pantalla al ejecutar el programa?

Muestra los 256 caracteres del código ascii



Apellidos, Nombre:

Grupo de laboratorio:

Gestión

Sistemas

Libre Elección

-
- 2) Cambia el procedimiento `escribe_caracter` por una macro que haga exactamente lo mismo. Cambia también la llamada al procedimiento por la llamada a la macro.

```
escribe_caracter macro  
mov ah,02  
int 21h  
endm
```

Y la llamada ahora es: **escribe_caracter**
en vez de : **call escribe_caracter**

- 3) Realiza los cambios necesarios en el procedimiento `escribe_caracter` para que cada carácter mostrado en pantalla esté en una línea diferente y en el comienzo de línea de la pantalla.

```
escribe_caracter proc  
mov ah,02  
mov bl,dl  
int 21h  
mov dl, LF  
int 21h  
mov dl, CR  
int 21h  
mov dl,bl  
ret  
escribe_caracter endp
```

- 4) Cambia el procedimiento `escribe_caracter` y lo que consideres necesario del programa para que lo que se muestra en pantalla en vez de eso se guarde en memoria a partir de la dirección **Almacena**

```
escribe_caracter proc  
mov Almacena[si],dl  
ret  
escribe_caracter endp
```

Además antes del bucle `imprime_loop` inicializamos `si=0` y dentro del bucle vamos incrementando `si`:

```
xor si,si  
imprime_loop:  
    call escribe_caracter  
    inc dl  
    inc si  
    loop imprime_loop
```



Apellidos, Nombre:

Grupo de laboratorio:

Gestión

Sistemas

Libre Elección

- 5) Realiza el procedimiento `saca_cadena` que saque por pantalla lo almacenado en memoria en el apartado anterior. Añade en el programa la llamada a este nuevo procedimiento en el lugar adecuado.

```
saca_cadena proc
lea dx, Almacena
mov ah, 9h
int 21h
ret
saca_cadena endp
```

Esta solución tiene un problema, como lo que se imprime es el código ascii y estamos utilizando la `int 21h` con `ah=9` imprime el código hasta llegar al carácter \$ es decir solo se muestra en pantalla 37 caracteres (el 36 es \$).

Es mejor la siguiente solución donde se imprime carácter a carácter:

```
saca_cadena proc
xor si,si
mov cx,255
repite_loop:
    mov dl, Almacena[si]
    inc si
    mov ah, 2h
    int 21h
    loop repite_loop
ret
saca_cadena endp
```

Llamada una vez terminado el bucle `imprime_loop`:

```
call saca_cadena
```

Desde el Code View y con la opción **mixed** en la que se puede ver intercalada una línea en ensamblador y su correspondiente código máquina visualizamos lo siguiente:

ESCRIBE_CARACTER:

```
17:  mov ah,02
4FE9:0010 B402      MOV    AH,02
18:  int 21h
4FE9:0012 CD21      INT    21
19:  ret
4FE9:0014 C3        RET
```

INICIO:

```
24:  mov ax, @data
4FE9:0015 B8EC4F      MOV    AX,4FEC
25:  mov ds, ax
```



Apellidos, Nombre:

Grupo de laboratorio:

Gestión

Sistemas

Libre Elección

```
4FE9:0018 8ED8      MOV    DS,AX
27:   xor cx,cx
4FE9:001A 33C9      XOR    CX,CX
28:   mov dl, [caracter_primer]
4FE9:001C 8A160000    MOV    DL,Byte Ptr [CHARACTER_PRIMER (0000)
29:   mov cl, [caracter_ulti]
4FE9:0020 8A0E0100    MOV    CL,Byte Ptr [CHARACTER_ULTI (0001)]
```

IMPRIME_LOOP:

```
32:   call escribe_caracter
4FE9:0024 E8E9FF      CALL   ESCRIBE_CHARACTER (0010)
33:   inc dl
4FE9:0027 FEC2      INC    DL
34:   loop imprime_loop
4FE9:0029 E2F9      LOOP  IMPRIME_LOOP (0024)
```

```
36:   mov ah, 4Ch
4FE9:002B B44C      MOV    AH,4C
37:   int 21h
4FE9:002D CD21      INT    21
4FE9:002F 0000      ADD    Byte Ptr [BX+SI],AL
4FE9:0031 FFFF      ???   DI
4FE9:0033 0000      ADD    Byte Ptr [BX+SI],AL
```

6) ¿Cuál es la dirección física de memoria donde se encuentra la instrucción de la línea 29: **mov cl, [caracter_ulti] 4FE9:0020**

CS * 10h + IP → 4FE9 * 10h + 0020 =4FEB0h

7) ¿Cuál es el tamaño en bytes que ocupa la instrucción anterior?

Código máquina: 8A0E0100 → 4 Bytes

8) ¿En que dirección efectiva (dentro del data segment) se encuentra la variable **caracter_ulti**?

Se puede ver que es la 2ª variable del data segment.

Además se puede ver en el código máquina 8A0E0100.

También en el desplazamiento respecto al comienzo del data segment [CHARACTER_ULTI (0001)]

Luego la dirección efectiva es: 0001.



Apellidos, Nombre:

Grupo de laboratorio:

Gestión

Sistemas

Libre Elección

Dados los siguientes valores de registros

AX = 000F BX = 0002 CX = 00FF DX = 0000 SP = 00FE BP = 0000 SI = 0000
DI = 0000 DS = 4FEC ES = 4FD9 SS = 4FFD CS = 4FE9 IP = 0014

NV UP

EI PL

ZR NA

PE CY

9) ¿Qué registros se modifican y con que valores si ejecuto **DIV BX**?

Dividimos DX AX : BX

El cociente se guarda en AX y el resto en DX, luego:

AX=0007

DX=0001

El resto de registros quedan igual

Nota: (ip tambien se modificará incrementándose en tantos bytes como tenga la instrucción)

10) ¿Qué registros se modifican y con que valores si ejecuto **ADC AX, BX**?

Suma con acarreo y el resultado se guarda en AX.

El flag de carry esta a 1 (CY) luego sumamos $000F + 0002 + 1 = 0012 h = AX$

Nota: (ip tambien se modificará incrementándose en tantos bytes como tenga la instrucción)

Nota: El flag de carry pasará a 0 (NC)