ASSEMBLER

16 Bits = 2Byte

AX= AH AL
BX
CX

MUL reg o memoria

AX * □ = DX AX

2*3=6

DIV reg o memoria

20/3=

XCHG

AX=12h
BX=5h

XCHG AX,BX

AX=5h
BX=12h

Ejercicio

Ingresar un numero determinar si un número es par o impar y mostrar en la pantalla

N: 40 Es par	N: 13 Es impar	N: 2 Es par		

122
1*10+2=12
12*10+2=112

```
al=FF
add al,1; ah=1 ,  al=0
inc al ; ah=1 ,  al=0
```

```
n=FF ; n db 0
add n ,1
n=0
```

```
n=0 ; n db 0
sub n,1
n=FF
```

```
01 data segment
02     msgN db 10,13,"N:$"
03     msgPar db 10,13,"Es Par$"
04     msgImpar db 10,13,"Es impar$"
05     n dw 0
06     val10 dw 10
07     val2 dw 2
08 ends
09 stack segment
10     dw   128  dup(0)
11 ends
12 code segment
13 start:
14     mov ax, data
15     mov ds, ax
16     mov es, ax
17     leer:
18      mov ah , 1
19      int 21h
20      cmp al , 13
21      je finLeer
22      sub al,48
23      mov ah, 0
24      xchg n,ax
25      mul val10
26      add ax, n
27      mov n, ax
28      jmp leer
29      finLeer:
30
31      mov dx, 0
32      mov ax, n
33      div val2
34      cmp dx,0
35      je esPar
36     ; Impar
37      lea dx, msgImpar
38      mov ah, 9
39      int 21h
40      jmp  finPrograma
41
42      esPar:
43        lea dx, msgPar
44        mov ah,9
45        int 21h
46      finPrograma:
47     mov ax, 4c00h ; exit to operating system.
48     int 21h
49 ends
50
51 end start ; set entry point and stop the assembler.
52
```

Ejercicio

Ingresar un número y un carácter y generar lo siguiente

N: 2 C: A A BB i	N: 3 C: F F GG HHH	N: 4 C: B B CC DDD EEEEE		

```
01 data segment
02     msgN db 10,13,"N: $"
03     msgC db 10,13,"C: $"
04     n db 0
05     c db 0
06     i db 1
07     enter db 10,13,"$"
08 ends
09 stack segment
10     dw   128  dup(0)
11 ends
12 code segment
13 start:
14     mov ax, data
15     mov ds, ax
16     mov es, ax
17
18     lea dx,  msgN
19     mov ah,9
20     int 21h
21     mov ah, 1
22     int 21h
23     sub AL, 48
24     mov n, al
25
26     lea dx,  msgC
27     mov ah,9
28     int 21h
29     mov ah, 1
30     int 21h
31     mov c, al
32
33     mov i, 1
34     for:
35       lea dx, enter
36       mov ah,9
37       int 21h
38       mov ch,0
39       mov cl ,i
40       mostrando:
41         mov dl,c
42         mov ah,2
43         int 21h
44       loop mostrando
45       inc c
46       inc i
47     dec n
48     cmp n,0
49     jg for
50
51     mov ax, 4c00h
52     int 21h
53 ends
54 end start ; set entry point and stop the assembler.
```

CH,0
CL,i → CX=CHCL= i

Error
mov cx, i

16Bits (db)8 bits

CX
Etiqueta:

LOOP Etiqueta

jl <
jg >
jle <=
jge =>
je =
jne !=

IF(completo)

```
_____
   memoria dw 0
_____

cmp ax, memoria
jl menor
jg mayor
   ; Codigo cuando sea igual
jmp finIF
menor:
   ;Codigo cuando es menor
jmp finIF
mayor:
   ;Codigo cuando es Mayor
finIF:
```