

ASSEMBLER**Ejercicio**

Ingresa un número y genera la siguiente serie

N: 1	N: 5	N: 0	N: 3	N: 8
1	1		1	1
	22		22	22
	333		333	333
	4444			4444
	55555			55555
				666666
				7777777
				88888888

Ejercicio

Ingresa una cadena y muestra la cantidad de vocales

Hola mundo	python	Java	dwqft	Informatica
4	1	2	0	3

a=61h, e= 65h , i=69h , o=6fh ,u=75h

Ejercicio 1

```
01 ; multi-segment executable file template.
02
03 data segment
04     nro db 0
05     i db 1
06     enter db 10,13,"$"
07 ends
08
09 stack segment
10     dw 128 dup(0)
11 ends
12
13 code segment
14 start:
15     mov ax, data
16     mov ds, ax
17     mov es, ax
18
19
20     mov ah, 1
21     int 21h
22     cmp al, 48
23     je finPrograma
24
25     mov nro, al
26     sub nro, 48
27
28     lea dx, enter
29     mov ah, 9
30     int 21h
31     ; Mostrando nro-1>0
32     mostrando:
33
34     mov ch, 0
35     mov cl, i
36     ;Desplegando cx
37     desplegando:
38
39     mov dl, i
40     add dl, 48
41     mov ah, 2
42     int 21h
43
44     loop desplegando
45     inc i; incrementamos i
46
47     lea dx, enter
48     mov ah, 9
49     int 21h
50
51     dec nro
52     cmp nro, 0
53     jg mostrando
54
55     finPrograma:
56
57     mov ax, 4c00h
58     int 21h
59 ends
60
61 end start
```

Ejercicio 2

```
01 data segment
02     enter db 10,13,"$"
03     cadena db ?
04 ends
05 stack segment
06     dw 128 dup(0)
07 ends
08 code segment
09 start:
10     mov ax, data
11     mov ds, ax
12     mov es, ax
13
14     lea si, cadena
15     leer:
16         mov ah, 1
17         int 21h
18         cmp al, 0dh
19         je FinLectura
20         mov [si], al
21         inc si
22         jmp leer
23     FinLectura:
24
25     dec si
26     mov cx, 0
27     buscando:;Se esta verificando si son vocales
28         cmp [si], 61h
29         je contar
30         cmp [si], 65h
31         je contar
32
33         cmp [si], 69h
34         je contar
35
36         cmp [si], 6fh
37         je contar
38
39         cmp [si], 75h
40         je contar
41
42         jmp noContar
43
44     contar:
45         inc cx
46
47     noContar:
48         dec si
49         cmp si, 0
50         jge buscando
51
52     lea dx, enter
53     mov ah, 9
54     int 21h
55
56     mov dl, cl
57     add dl, 30h
58     mov ah, 2
59     int 21h
60
61     mov ax, 4c00h ; exit to operating system.
62     int 21h
63
64 ends
65
```

Ejer de lic

```
02
03 data segment
04     ; add your data here!
05     msg1 db 10,13,"Numero 1 $"
06     msg2 db 10,13,"Numero 2 $"
07     sol db 10,13,"El numero mayor $"
08     a db 0
09     b db 0
10 ends
11
12 stack segment
13     dw 128 dup(0)
14 ends
15
16 code segment
17 start:
18     ; set segment registers:
19     mov ax, data
20     mov ds, ax
21     mov es, ax
22
23     lea dx, msg1
24     mov ah, 9
25     int 21h                ; output string at ds:dx
26     mov ah, 1
27     int 21h
28     mov a, al
29
30     lea dx, msg2
31     mov ah, 9
32     int 21h                ; output string at ds:dx
33     mov ah, 1
34     int 21h
35     mov b, al
36
37     cmp al, a; cmp b, a
38     jg mayor
39
40     lea dx, sol
41     mov ah, 9
42     int 21h
43
44     mov dl, a
45     mov ah, 2
46     int 21h
47
48     mov ax, 4c00h ; exit to operating system.
49     int 21h
50
51     mayor:
52     lea dx, sol
53     mov ah, 9
54     int 21h
55
56     mov dl, b
57     mov ah, 2
58     int 21h
59
60     mov ax, 4c00h ; exit to operating system.
61     int 21h
62 ends
63 end start ; set entry point and stop the assembler.
64
```