

Modifica el código de prueba para incluir instrucciones de SHL, SHR, ROR y ROL (las 4 para cada inciso). Tu debes de realizar los desplazamientos y rotación a tu consideración, para imprimir cada uno de los caracteres que se piden (cada uno en un programa por separado). Para cada caso documenta tus resultados en el taller. Para tu repositorio solo es necesario que subas el código del inciso g. Adelante puedes encontrar el código de prueba y el enlace correspondiente al mismo código.

A:

```
1 section .data
2     char db 0
3     nl db 10
4
5 section .text
6     global _start
7
8 _start:
9     mov al, 0b00010001    ; 17 decimal
10
11     shl al, 2             ; 17 << 2 = 68
12     shr al, 1             ; 68 >> 1 = 34
13     shl al, 1             ; 34 << 1 = 68
14     sub al, 3             ; 68 - 3 = 65 ('A')
15
16     ; ROL y ROR solo de demostración (sin cambiar resultado)
17     rol al, 1
18     ror al, 1             ; regresa a 65
19
20     mov [char], al
21
22     mov eax, 4
23     mov ebx, 1
24     mov ecx, char
25     mov edx, 1
26     int 0x80
27
28     mov eax, 4
29     mov ebx, 1
30     mov ecx, nl
31     mov edx, 1
32     int 0x80
33
34     mov eax, 1
35     xor ebx, ebx
36     int 0x80
```

Output:

A

0:

```
1 section .data
2     char db 0
3     nl db 10
4
5 section .text
6     global _start
7
8 _start:
9     mov al, 6                ; 6 decimal
10
11     shl al, 3                ; 6 << 3 = 48 ('0')
12
13     ; Rotaciones demostrativas
14     rol al, 1
15     ror al, 1                ; regresa a 48
16
17     mov [char], al
18
19     mov eax, 4
20     mov ebx, 1
21     mov ecx, char
22     mov edx, 1
23     int 0x80
24
25     mov eax, 4
26     mov ebx, 1
27     mov ecx, nl
28     mov edx, 1
29     int 0x80
30
31     mov eax, 1
32     xor ebx, ebx
33     int 0x80
```

Output:

0

g:

```
1 ▾ section .data
2   | char db 0
3   | nl db 10
4
5 ▾ section .text
6   | global _start
7
8 ▾ _start:
9   | mov al, 0b00011001    ; 25 decimal
10
11  | shl al, 2              ; 25 << 2 = 100
12  | add al, 3              ; 103 ('g')
13
14  | ; Rotación demostrativa
15  | rol al, 1
16  | ror al, 1              ; regresa a 103
17
18  | mov [char], al
19
20  | mov eax, 4
21  | mov ebx, 1
22  | mov ecx, char
23  | mov edx, 1
24  | int 0x80
25
26  | mov eax, 4
27  | mov ebx, 1
28  | mov ecx, nl
29  | mov edx, 1
30  | int 0x80
31
32  | mov eax, 1
33  | xor ebx, ebx
34  | int 0x80
```

Output:

g

=:

```
1 ▾ section .data
2   | char db 0
3   | nl db 10
4
5 ▾ section .text
6   | global _start
7
8 ▾ _start:
9   | mov al, 0b00011110    ; 30 decimal
10
11  | shl al, 1              ; 30 << 1 = 60
12  | add al, 1              ; 61 ('=')
13
14  | ; Rotaciones demostrativas
15  | rol al, 1
16  | ror al, 1              ; regresa a 61
17
18  | mov [char], al
19
20  | mov eax, 4
21  | mov ebx, 1
22  | mov ecx, char
23  | mov edx, 1
24  | int 0x80
25
26  | mov eax, 4
27  | mov ebx, 1
28  | mov ecx, nl
29  | mov edx, 1
30  | int 0x80
31
32  | mov eax, 1
33  | xor ebx, ebx
34  | int 0x80
```

Output:

=