Codigo 4: Revertir cadena de caracteres

```
section .data
   texto db “rojem le ser nebuR roseforp lE”, 0
   len   equ $ - texto
   newline db 10

section .bss
   temp_char resb 1          ; Reservamos 1 byte para imprimir

section .text
   global _start

_start:
   mov esi, len              ; ESI apunta al tamaño de la cadena
   dec esi                   ; Ajustamos para empezar desde el último carácter

print_rev:
   mov al, [texto + esi]     ; Cargar carácter en AL
   mov [temp_char], al       ; Guardar en buffer temporal

   mov eax, 4                ; sys_write
   mov ebx, 1                ; stdout
   mov ecx, temp_char        ; dirección del carácter
   mov edx, 1                ; 1 byte
   int 0x80

   dec esi
   cmp esi, -1
   jg print_rev

   ; salto de línea
   mov eax, 4
   mov ebx, 1
   mov ecx, newline
   mov edx, 1
   int 0x80

   ; salir
   mov eax, 1
   xor ebx, ebx
   int 0x80
```

Este programa imprime el mensaje rojem le ser nebuR roseforp lE en orden inverso, carácter por carácter. Para lograrlo, primero define el texto original como una cadena de caracteres en la sección .data. También calcula automáticamente la longitud del texto usando la directiva equ, que permite determinar cuántos bytes ocupa la cadena. Además, define un

salto de línea para usarlo al final del programa y reserva un byte en la sección .bss para almacenar temporalmente cada carácter que se va a imprimir.

El código comienza en la etiqueta _start, que es el punto de entrada del programa. Allí se carga en el registro esi la longitud del texto, y luego se decrementa en uno para apuntar al último carácter de la cadena (recordando que los índices comienzan en 0). A partir de este punto, se inicia un bucle que recorre el texto de atrás hacia adelante.

Dentro del bucle print_rev, el programa accede al carácter actual usando el índice esi y lo almacena en el registro al. Luego, guarda ese carácter en el búfer temporal temp_char, el cual fue reservado previamente. Esto es necesario porque el servicio de impresión (sys_write) requiere una dirección de memoria con los datos a imprimir, no puede imprimir directamente desde un registro.

Después de guardar el carácter, el programa prepara los registros necesarios para realizar la llamada al sistema que escribe en la pantalla: eax se configura con el número de servicio 4 (que representa sys_write), ebx con el descriptor de archivo 1 (que representa la salida estándar o pantalla), ecx con la dirección del carácter a imprimir, y edx con la cantidad de bytes a escribir, en este caso 1. Luego se ejecuta int 0x80 para hacer efectiva la impresión.

El índice esi se reduce en uno en cada iteración, y el bucle se repite mientras el índice sea mayor que -1. Esto asegura que todos los caracteres del texto, desde el último hasta el primero, sean impresos uno por uno. Una vez que termina el bucle, el programa imprime un salto de línea para que la salida sea más legible.

Finalmente, el programa termina limpiamente usando el servicio sys_exit con el número 1 en eax y 0 en ebx como código de salida. De esta manera, el programa muestra cómo recorrer una cadena al revés y cómo imprimir carácter por carácter usando llamadas al sistema, sin depender de funciones externas o librerías de alto nivel.