

Universidad Autónoma de Baja California

Ingeniería en Computación



Facultad de Ciencias Químicas e Ingeniería

Organización y Arquitectura de Computadoras

Práctica 11

Zavala Román Irvin Eduardo

Grupo: 551

03/11/2021

Periodo 2021-2

Objetivo

Seleccionar las instrucciones de control de flujo del programa adecuadas en la manipulación de cadenas, para desarrollar aplicaciones de sistemas basados en microprocesador, mediante el análisis de su funcionalidad, de forma responsable y eficiente.

Desarrollo

1. Cree un programa llamado P11.asm que contenga la siguiente rutina:

substr: almacena en una cadena una copia de una porción de otra cadena. Recibe en ESI la dirección de una cadena fuente, en EDI la dirección de la cadena destino, en BX la posición inicial a copiar y en CX la cantidad de caracteres.

Si la cadena es más corta que los caracteres solicitados en CX, el procedimiento copia todos los posibles. Si la posición en BX es mayor que la longitud de la cadena, el procedimiento retorna un -1 en EAX, de lo contrario retorna 0.

Ejemplo:

```
mov esi, cadena ; si la cadena es "Hola mundo"
```

```
mov edi, destino
```

```
mov bx, 1 ; copiar a partir de la posición 1
```

```
mov cx, 5 ; copiar 5 caracteres
```

```
call substr ; destino es "olá m", retorná EAX = 0
```

Resultado

El código en NASM de la implementación de substr queda de la siguiente manera:

```
%include "pc_io.inc"

section .data
    msg1: db "Ingresa una cadena:",0
    msg2: db "Se va a crear una cadena desde: ",0
    msg3: db "Y se van a copiar esta cantidad de caracteres: ",0
section .bss
    fuente resb 256
    destino resb 256
    cad resb 256
section .text
    global _start:

_start:
    call clrscr
    mov edx,msg1
    call puts

    mov eax, 3    ;Leemos
    mov ebx, 0
    mov ecx, fuente
    mov edx, 256
    int 80h

    mov ebx, 1
    mov ecx, 5

    mov edx, msg2
    call puts
    mov eax, ebx
    mov esi,cad
    call printHex
    call salto

    mov edx, msg3
    call puts
    mov eax, ecx
    mov esi,cad
    call printHex
    call salto

    mov esi, fuente
    mov edi, destino
    call substr
```

```
mov esi,cad
call printHex ;Imprimo eax pa ver si retorno bien 0 o -1
call salto
```

```
mov edx,destino
call puts
call salto
```

```
mov eax, 1 ;FIN
mov ebx,0
int 80h
```

substr:

```
push edx
push ecx
push esi
push edi
```

```
mov edx, 0
add ecx,1
```

.buscar_inicio:

```
cmp byte[esi], 0
je .error
```

```
inc esi
inc edx
```

```
cmp edx,ebx
jne .buscar_inicio
mov edx, 1
mov eax, 0
```

.copiar:

```
cmp byte[esi],0
je .correcto
mov al, [esi]
mov [edi],al
```

```
inc esi
inc edi
inc edx
```

```
cmp edx,ecx
jne .copiar
jmp .correcto
```

```

.error:
    mov eax, -1
    jmp .fin
.correcto:
    mov eax, 0
.fin:

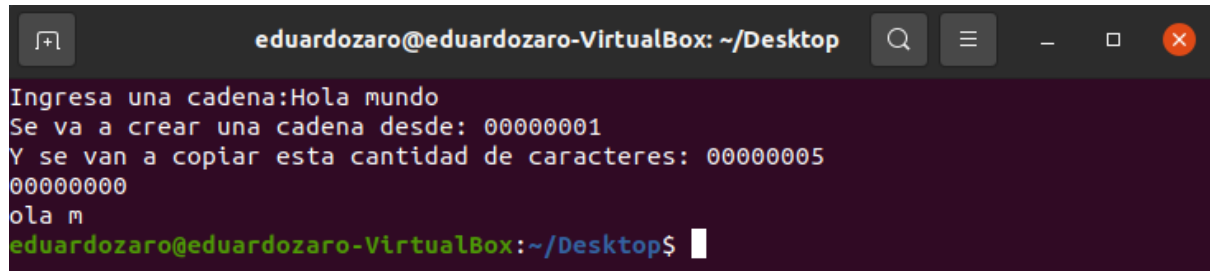
    pop edi
    pop esi
    pop ecx
    pop edx
    ret

printHex:
    pushad
    mov edx, eax
    mov ebx, 0fh
    mov cl, 28
.nxt:
    shr eax, cl
.msk:
    and eax, ebx
    cmp al, 9
    jbe .menor
    add al, 7
.menor:
    add al, '0'
    mov byte [esi], al
    inc esi
    mov eax, edx
    cmp cl, 0
    je .print
    sub cl, 4
    cmp cl, 0
    ja .nxt
    je .msk
.print:
    mov eax, 4
    mov ebx, 1
    sub esi, 8
    mov ecx, esi
    mov edx, 8
    int 80h
    popad
    ret
salto:
    push eax
    mov al, 10 ;SALTO
    call putchar
    int 80h

```

```
pop eax
ret
```

Y la salida de este programa es el siguiente:

A screenshot of a terminal window titled 'eduardozaro@eduardozaro-VirtualBox: ~/Desktop'. The terminal shows the following text: 'Ingresa una cadena:Hola mundo', 'Se va a crear una cadena desde: 00000001', 'Y se van a copiar esta cantidad de caracteres: 00000005', '00000000', 'ola m', and the prompt 'eduardozaro@eduardozaro-VirtualBox:~/Desktop\$' with a cursor. The window has standard Linux window controls (minimize, maximize, close) and a search icon.

```
eduardozaro@eduardozaro-VirtualBox: ~/Desktop
Ingresa una cadena:Hola mundo
Se va a crear una cadena desde: 00000001
Y se van a copiar esta cantidad de caracteres: 00000005
00000000
ola m
eduardozaro@eduardozaro-VirtualBox:~/Desktop$
```

Conclusión

La implementación de esta función que existe en lenguajes de mayor nivel se me hace bastante útil ya que varias funciones que suelo usar en C por ejemplo podrían ser implementados en NASM, ampliando mi capacidad en este lenguaje.

Dificultades en el desarrollo

Nada más allá de simples confusiones con NASM.