Menor sufijo a agregar para convertir en Palindromo (Problema 64)

March 29, 2024

1 Descripcion del Ejercicio

Un palíndromo es una palabra, frase, número o secuencia que se lee igual hacia adelante que hacia atrás, sin importar la dirección en la que se lea. Los palíndromos pueden ser de una sola palabra, como "11011" o "12321".

Dado un patrón en forma de bits. Devolver el menor sufijo que habría que agregarle a dicho patrón para que fuera palíndromo.

Salida

Para la salida debe imprimir el menor sufijo, separando cada dígito por un espacio.

Por ejemplo, para $L = [1, 0, 1, 1]$ debería imprimir:

¿ 0 1

Logisim

Se dispondrá en *INPUT* los datos de entrada a partir de la dirección 0. La entrada se estructura de la siguiente forma:

- w_0: n (tamaño de la lista L) - $w_{1:n}$: L

SASM

En la sección `.data` se deben definir los valores de entrada de la siguiente forma:

- `n` : un número de tamaño `dd` que representa al tamaño de la lista L - `array` : un array de números de tamaño `dd` que representa L

Por ejemplo, un posible encabezado podría ser:

```
section .data
n dd 4
array dd 1, 0, 1, 1
```

2 Pseudocódigo

```
# Funcion para saber si es palindromo
def esPalindromo(arr, inicio, n):
  fin = n - 1
```

```
    while inicio < fin:
      if arr[inicio] != arr[fin]:
        return False
      inicio = inicio + 1
      fin = fin - 1
  return True

  # Funcion para imprimir el sufijo minimo que necesita ser
    agregado al inicio
  # para hacer el arreglo un palindromo.
  def imprimirSufijoParaPalindromo(arr, n):

  # Encuentra la posicion inicial desde donde el sufijo debe
  # ser un palindromo
    while i < n:
      if esPalindromo(arr, i, n):
        break
      i = i+1

  # Imprimir el sufijo necesario en orden inverso, que debe ser
    agregado al inicio
  i = i-1
  while i>=0:
    print(arr[i], end =" ")

  arr = [1, 0, 1, 1]
  n = len(arr)

  print("El sufijo necesario para convertir el arreglo en un
    palindromo es:", end=" ")
  imprimirSufijoParaPalindromo(arr, n)
  print()


```

Listing 1: Función en Python

3 Código en Ensamblador

```
%include "io.inc"

section .data
arr dd 1, 0, 1, 1, 0, 0    ; Definir el arreglo.
n dd 6                  ; Longitud del arreglo.

section .bss
; Reserva de espacio para variables si es necesario.

section .text
global CMAIN

CMAIN:
mov ebp, esp; for correct debugging
mov edi, arr        ; Primer argumento de '
  imprimirSufijoParaPalindromo', la direccion de arr.
```

```
16  mov esi, [n]          ; Segundo argumento de '
      imprimirSufijoParaPalindromo', el tama o de arr.
17  call _imprimirSufijoParaPalindromo
18
19  xor eax, eax
20  ret
21
22  _esPalindromo:
23  mov ecx, esi                    ; copiamos n a ecx.
24  dec ecx                         ; fin = n - 1.
25  mov ebx, eax                    ; copiamos inicio a ebx (inicio de
      comparaci n).
26
27  palindromo_loop:
28  cmp ebx, ecx                    ; Comparamos inicio con fin.
29  jge .esPalindromoRetornoVerdadero ; Si inicio >= fin, es
      palindromo.
30  mov eax, [edi + ebx*4]          ; Valor en inicio.
31  cmp eax, [edi + ecx*4]          ; Valor en fin.
32  jne .esPalindromoRetornoFalso
33  inc ebx                         ; Incrementamos inicio.
34  dec ecx                         ; Decrementamos fin.
35  jmp palindromo_loop
36  .esPalindromoRetornoFalso:
37  mov eax, 0                       ; Retornar falso (0).
38  ret
39  .esPalindromoRetornoVerdadero:
40  mov eax, 1                       ; Retornar verdadero (1).
41  ret
42
43  _imprimirSufijoParaPalindromo:
44  xor ecx, ecx                    ; i = 0.
45
46  find_palindromo_prefix_loop:
47  cmp ecx, esi                    ; Comparamos i con n.
48  je .find_palindromo_prefix_done ; si i == n hemos terminado.
49  mov edx, esi                    ; Argumento n para esPalindromo.
50  mov eax, ecx                    ; Argumento inicio para
      esPalindromo.
51  push ecx
52  call _esPalindromo
53  pop ecx
54  test eax, eax                   ; Verificamos si esPalindromo
      retorno verdadero.
55  jz .increment_i                 ; Si no, incrementamos i y
      continuamos.
56  jmp .find_palindromo_prefix_done ; Si hemos terminado.
57
58  .increment_i:
59  inc ecx
60  jmp find_palindromo_prefix_loop
61
62  .find_palindromo_prefix_done:
63  dec ecx ; Ajustamos para el indice correcto del ultimo elemento
      no palindromo.
64  print_prefix:
65  cmp ecx, -1 ; Verificamos si hemos terminado con todos los
```

```
    elementos.
66  jle .done_printing
67
68  PRINT_DEC 4, [arr + ecx*4]
69
70  dec ecx ; Mover al siguiente elemento para imprimir.
71  jmp print_prefix
72
73  .done_printing:
74  ret
75
```

Listing 2: Código Tribonacci en Ensamblador