

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:

1 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

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

```

4     while inicio < fin:
5         if arr[inicio] != arr[fin]:
6             return False
7         inicio = inicio + 1
8         fin = fin - 1
9     return True
10
11 # Funcion para imprimir el sufijo minimo que necesita ser
12 # agregado al inicio
13 # para hacer el arreglo un palindromo.
14 def imprimirSufijoParaPalindromo(arr, n):
15     # Encuentra la posicion inicial desde donde el sufijo debe
16     # ser un palindromo
17     while i < n:
18         if esPalindromo(arr, i, n):
19             break
20         i = i+1
21
22 # Imprimir el sufijo necesario en orden inverso, que debe ser
23 # agregado al inicio
24 i = i-1
25 while i >= 0:
26     print(arr[i], end=" ")
27
28 arr = [1, 0, 1, 1]
29 n = len(arr)
30
31 print("El sufijo necesario para convertir el arreglo en un
32 palindromo es:", end=" ")
33 imprimirSufijoParaPalindromo(arr, n)
34 print()

```

Listing 1: Función en Python

3 Código en Ensamblador

```

1  %include "io.inc"
2
3  section .data
4  arr dd 1, 0, 1, 1, 0, 0    ; Definir el arreglo.
5  n dd 6                    ; Longitud del arreglo.
6
7  section .bss
8  ; Reserva de espacio para variables si es necesario.
9
10 section .text
11 global CMAIN
12
13 CMAIN:
14 mov ebp, esp; for correct debugging
15 mov edi, arr    ; Primer argumento de '
16     imprimirSufijoParaPalindromo', la direccion de arr.

```

```

16  mov esi, [n]           ; Segundo argumento de '
17      imprimirSufijoParaPalindromo', el tamaño de arr.
18  call _imprimirSufijoParaPalindromo
19
20  xor eax, eax
21  ret
22
23  _esPalindromo:
24  mov ecx, esi           ; copiamos n a ecx.
25  dec ecx               ; fin = n - 1.
26  mov ebx, eax           ; copiamos inicio a ebx (inicio de
27      comparaci n).
28
29  palindromo_loop:
30  cmp ebx, ecx           ; Comparamos inicio con fin.
31  jge .esPalindromoRetornoVerdadero ; Si inicio >= fin, es
32      palindromo.
33  mov eax, [edi + ebx*4] ; Valor en inicio.
34  cmp eax, [edi + ecx*4] ; Valor en fin.
35  jne .esPalindromoRetornoFalso
36  inc ebx               ; Incrementamos inicio.
37  dec ecx               ; Decrementamos fin.
38  jmp palindromo_loop
39  .esPalindromoRetornoFalso:
40  mov eax, 0            ; Retornar falso (0).
41  ret
42  .esPalindromoRetornoVerdadero:
43  mov eax, 1            ; Retornar verdadero (1).
44  ret
45
46  _imprimirSufijoParaPalindromo:
47  xor ecx, ecx           ; i = 0.
48
49  find_palindromo_prefix_loop:
50  cmp ecx, esi           ; Comparamos i con n.
51  je .find_palindromo_prefix_done ; si i == n hemos terminado.
52  mov edx, esi           ; Argumento n para esPalindromo.
53  mov eax, ecx           ; Argumento inicio para
54      esPalindromo.
55  push ecx
56  call _esPalindromo
57  pop ecx
58  test eax, eax          ; Verificamos si esPalindromo
59      retorno verdadero.
60  jz .increment_i        ; Si no, incrementamos i y
61      continuamos.
62  jmp .find_palindromo_prefix_done ; Si hemos terminado.
63
64  .increment_i:
65  inc ecx
66  jmp find_palindromo_prefix_loop
67
68  .find_palindromo_prefix_done:
69  dec ecx ; Ajustamos para el índice correcto del último elemento
70      no palindromo.
71  print_prefix:
72  cmp ecx, -1 ; Verificamos si hemos terminado con todos los

```

```

66     elementos.
67     jle .done_printing
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