INSTITUTO TECNOLÓGICO DE ESTUDIOS SUPERIORES DE OCCIDENTE
Departamento de Electrónica, Sistemas e Informática.

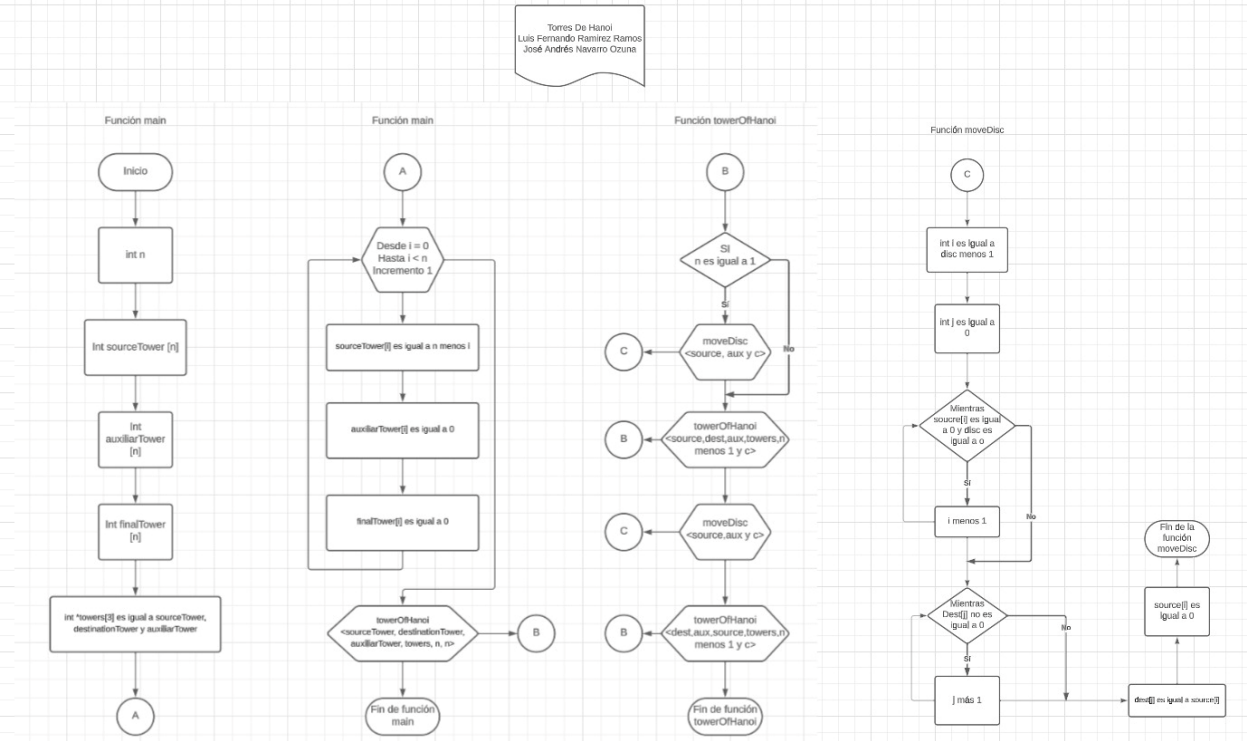
Práctica 1

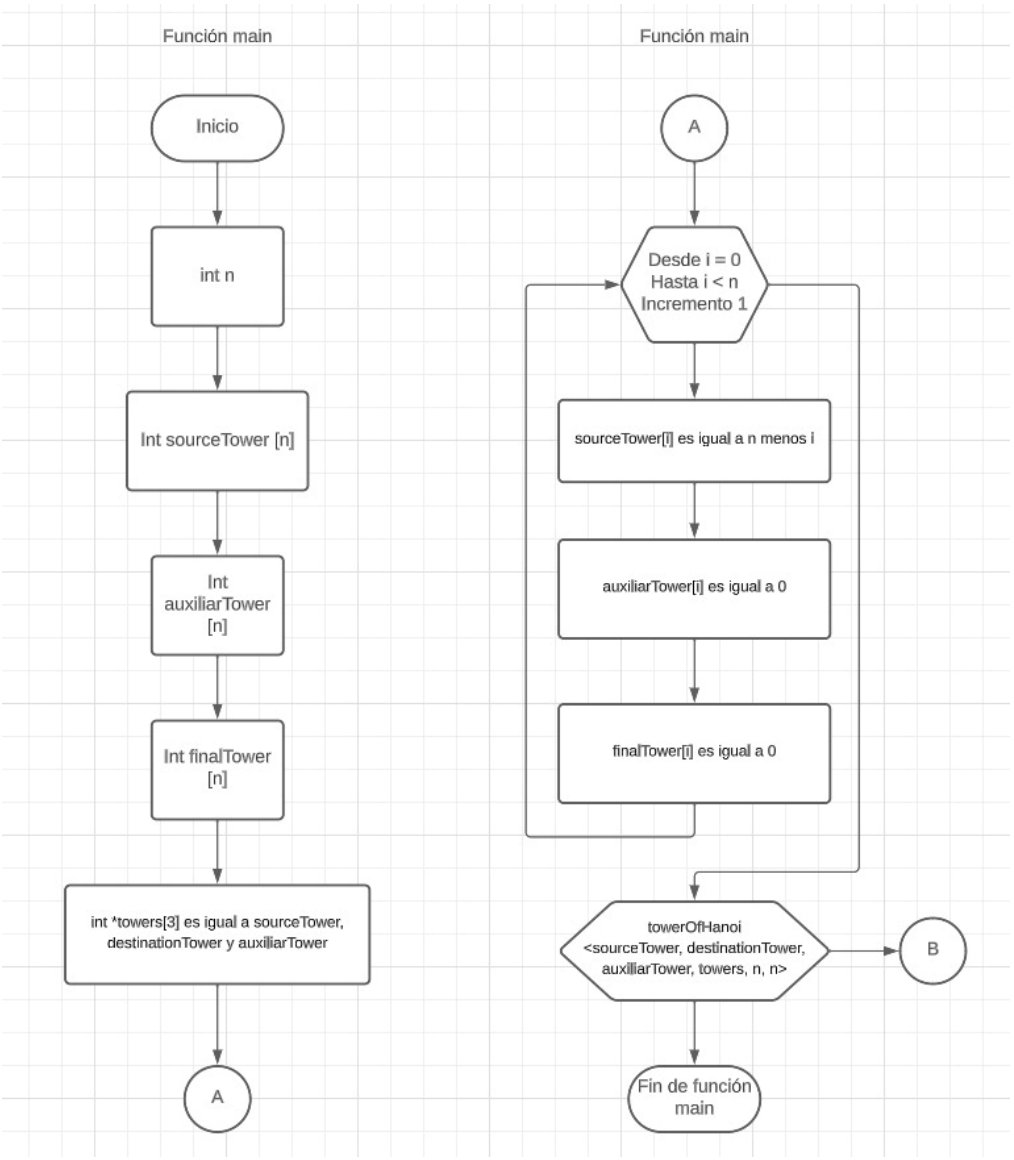
<Torres de Hanoi >

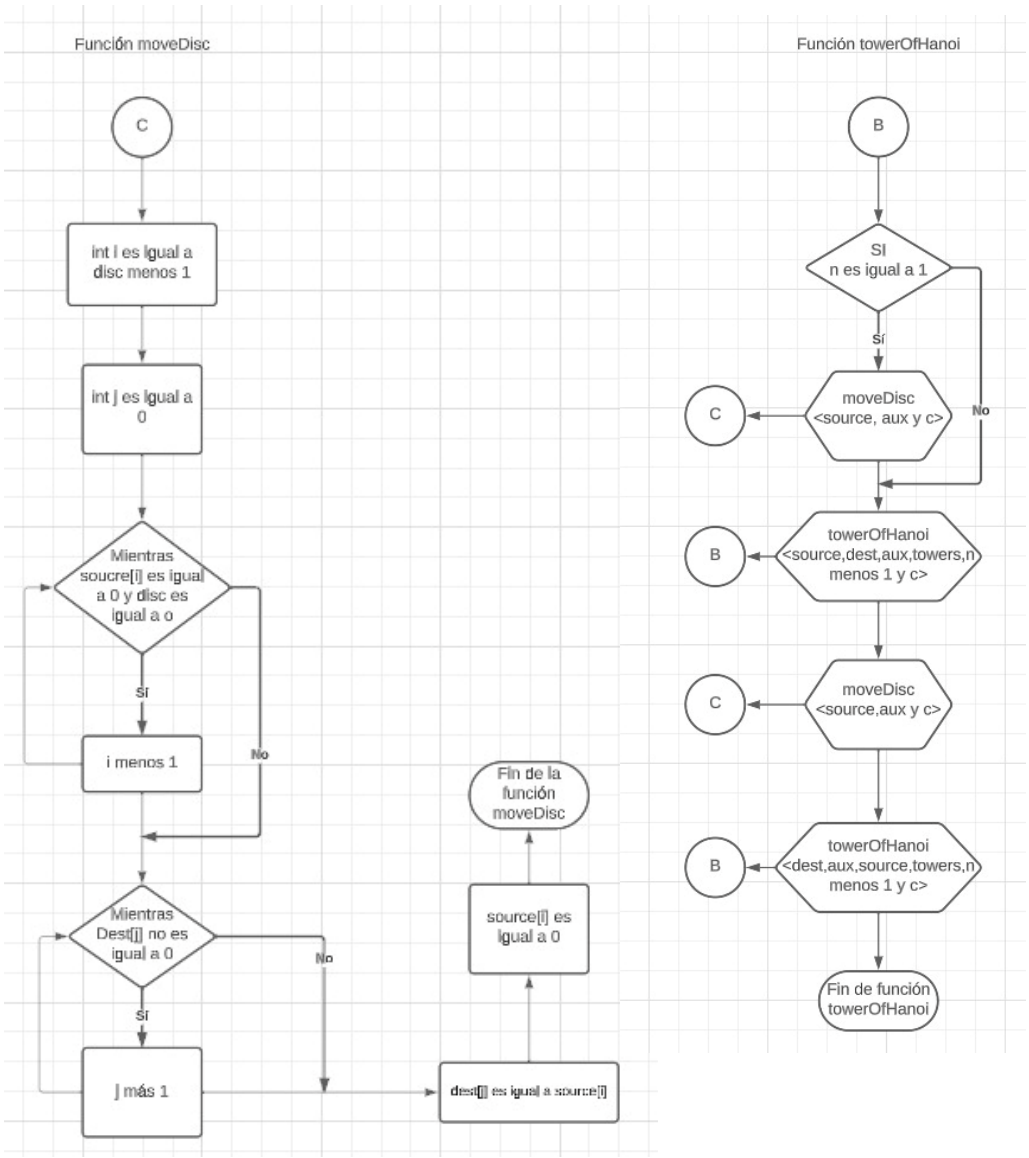
Organización y Arquitectura de Computadoras
José Andrés Navarro Ozuna, 744889
Luis Fernando Ramírez Ramos, 744615
Nombre del profesor: Juan Pablo Ibarra Esparza

Índice

Diagrama de flujo del programa




Función main
Inicio
int n
Int sourceTower [n]
Int auxiliarTower [n]
Int finalTower [n]
int *towers[3] es igual a sourceTower, destinationTower y auxiliarTower
A
Función main
A
Desde i = 0
Hasta i < n
Incremento 1
sourceTower[i] es igual a n menos i
auxiliarTower[i] es igual a 0
finalTower[i] es igual a 0
towerOfHanoi <sourceTower, destinationTower, auxiliarTower, towers, n, n>
B
Fin de función main


Función moveDisc
C
int l es igual a disc menos 1
int j es igual a 0
Mientras soucre[i] es igual a 0 y disc es igual a o
Sí
No
i menos 1
Mientras Dest[j] no es igual a 0
No
Sí
j más 1
dest[j] es igual a source[i]
source[i] es igual a 0
Fin de la función moveDisc
Función towerOfHanoi
B
SI n es igual a 1
Sí
No
C
moveDisc <source, aux y c>
B
towerOfHanoi <source,dest,aux,towers,n menos 1 y c>
C
moveDisc <source,aux y c>
B
towerOfHanoi <dest,aux,source,towers,n menos 1 y c>
Fin de función towerOfHanoi

Decisiones tomadas al diseñar el programa

Antes de empezar el algoritmo en .asm, lo programamos en C desde 0 para poder comprender al 100% el funcionamiento de las torres de Hanoi. Se construyó el código haciendo la separación entre el main, la función que realizaba el algoritmo y una función dedicada a mover cada disco, la cual era llamada constantemente dentro de la función del algoritmo. Una vez que tuvimos el código en C y comprendimos cómo funciona el algoritmo, lo utilizamos como base para empezar a crear el .asm.

En un principio, intentamos separar el código de la misma forma, la función principal, un salto hacia una función del algoritmo, un salto hacia una función que moviera los discos y un salto al final del código, pero terminamos solo haciendo un salto a la solución del algoritmo y uno al final del proceso. La función del algoritmo se encarga de hacer todos los movimientos necesarios, tanto cargar los valores necesarios, como realizar el push y el pop.

Simulación para 3 discos

Inicio, después de inicializar la torre origen:

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	0x00000001	0x00000002	0x00000003	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010020	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010040	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000

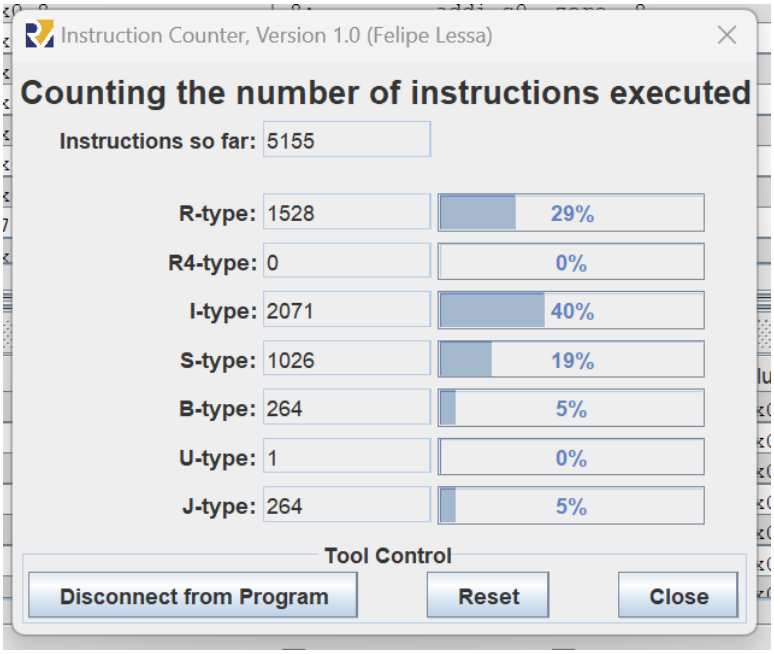
Final:

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000001	0x00000002
0x10010020	0x00000003	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010040	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000

Análisis del comportamiento del stack

En este algoritmo, el stack es utilizado para almacenar los valores necesarios entre cada llamada recursiva que se ejecuta.

IC y % de instrucciones tipo R, I y J para 8 discos



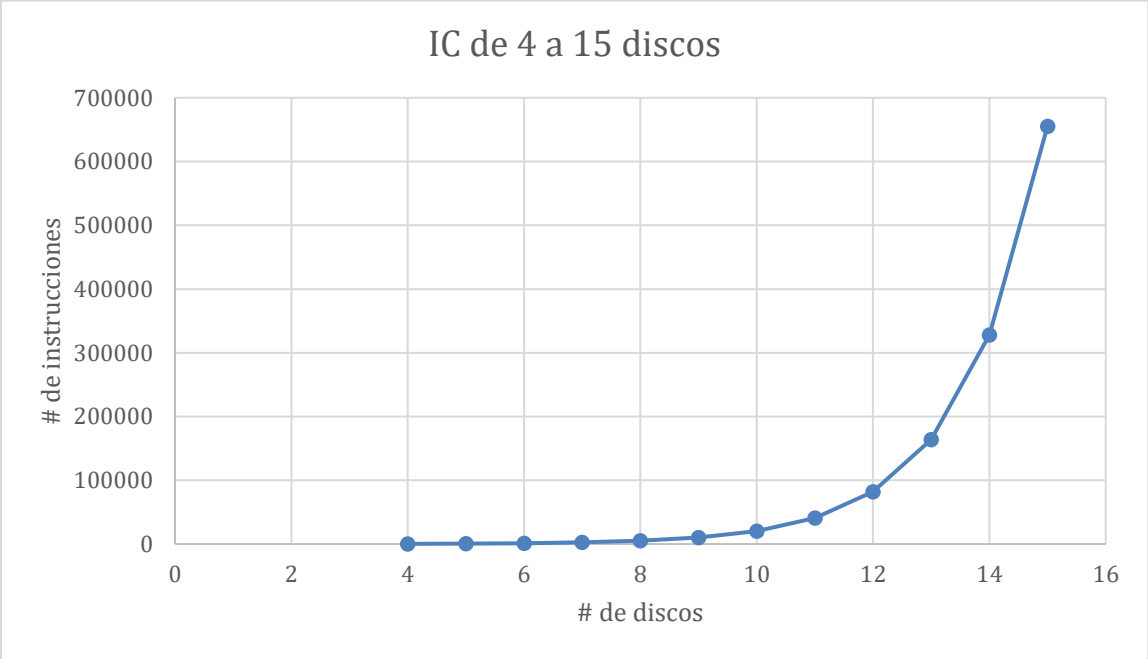
Tipo R: 29%

Tipo I: 40%

Tipo J: 5%

Gráfica del incremento del IC de 4 a 15 discos

# de discos	IC
4	327
5	654
6	1301
7	2588
8	5155
9	10282
10	20529
11	41016
12	81983
13	163910
14	327757
15	655444



Conclusiones

Luis Fernando Ramírez Ramos: Si antes creía tener un alto nivel de comprensión sobre la recursión y sobre algoritmos como el de las Torres de Hanoi, esta práctica me hizo dudarlo completamente y salir con un entendimiento sobre el tema bastante sólido. Dentro de la universidad, nunca me había enfrentado a un reto tan complicado y que me hiciera pensar que no lo lograría resolver. El hecho de tener que combinar 2 temas que, en mi opinión, son sumamente abstractos, como la recursión y el lenguaje ensamblador, hizo que también influyera en mi cierto grado de intimidación, pues nunca había trabajado con nada similar. Fue una práctica bastante compleja, en la cual fue necesario emplear gran nivel de creatividad y de resolución de problemas.

José Andrés Navarro Ozuna: Personalmente nunca había tratado con este problema de las Torres de Hanoi, siento que fue una práctica la cual tuvo sus complicaciones, pero pudimos sacarlas adelante. Esta práctica me dejó bastante claro tanto como lo sencillo que podemos manipular la memoria con RISC-V como el funcionamiento y la importancia de la recursión para estos problemas que hace trabajar nuestro cerebro poniéndonos un reto mejorando nuestra creatividad y pensamiento crítico.