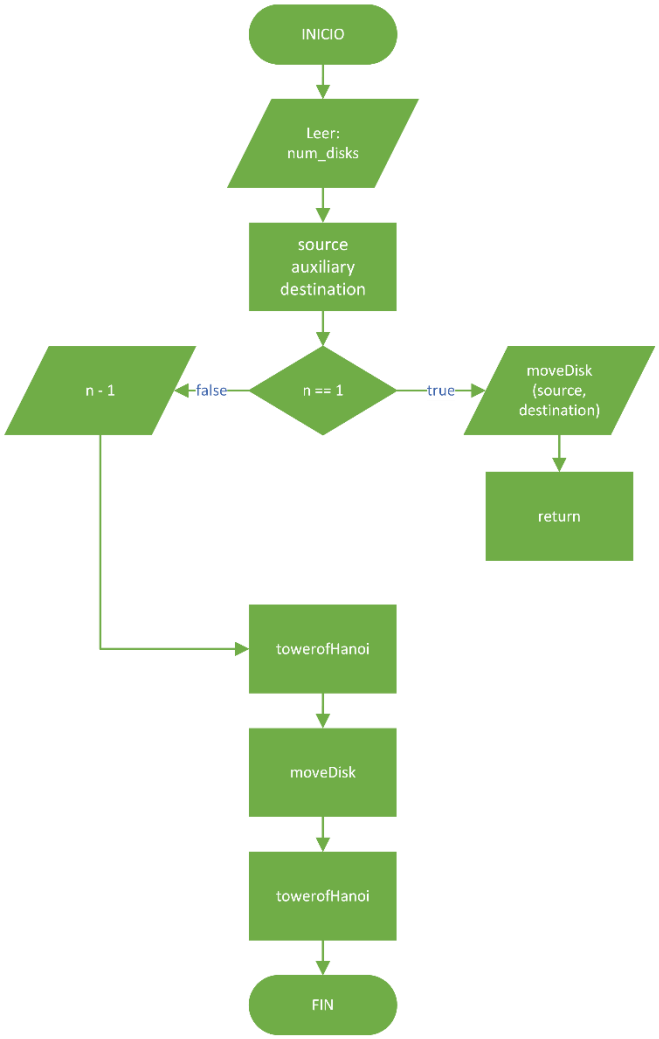
PRÁCTICA 1 – TORRES DE HANÓI

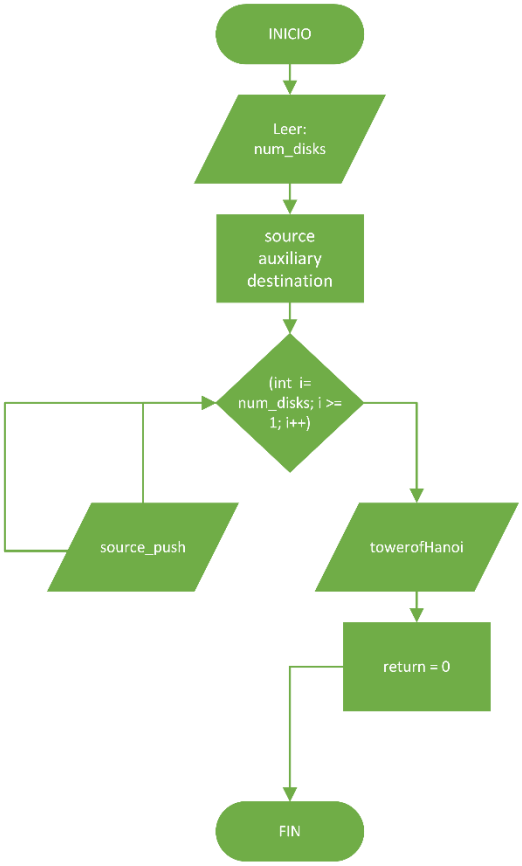
Ana Carolina Arellano Valdez – 738422
Alicia S. Gómez Gálvez - 740144

31 DE OCTUBRE DE 2023

ORGANIZACIÓN Y ARQUITECTURA DE COMPUTADORAS
JUAN PABLO IBARRA ESPARZA

Diagrama de Flujo





Las decisiones que se tomaron al diseñar su programa.

Al principio, quisimos separar las funciones de pop disk y push disk, pensando en una estructura de programación común, en la que solamente se llaman las funciones, sin embargo, tuvimos problemas con los jals, ya que no regresaban o actuaban de la manera esperada, por lo tanto, decidimos hacer el pop y push disk directamente en el main, utilizando el stack para almacenar los datos en memoria, de manera que se recuperen directamente sin tener que hacer saltos a las funciones.

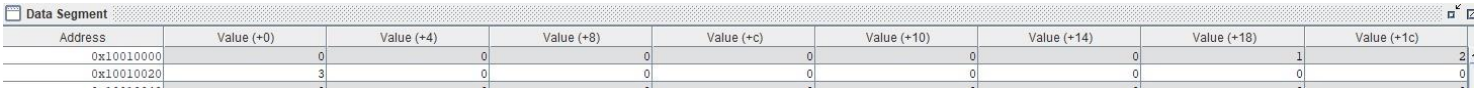
Una simulación en el RARS para 3 discos.

Inicio:

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)
0x10010000	1	2	3
0x10010020	0	0	0
0x10010040	0	0	0

Final:

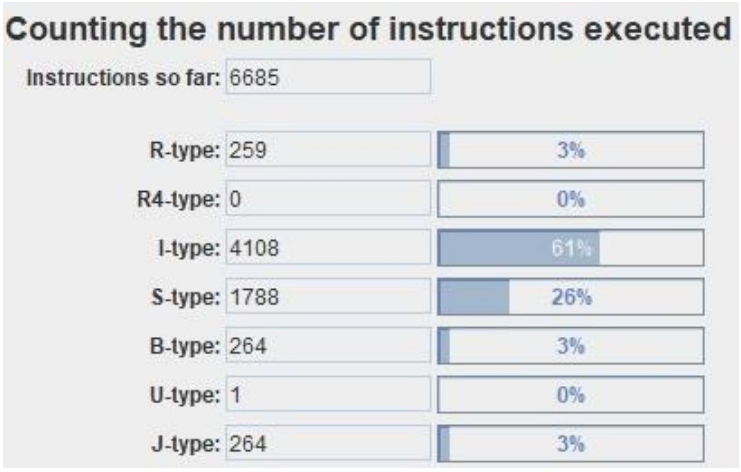

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	0	0	0	0	0	0	1	2
0x10010020	3	0	0	0	0	0	0	0

Análisis del comportamiento del stack para el caso de 3 discos.

Al principio se inicializan los valores de las torres con ayuda de un ciclo for, para luego comenzar a almacenar los datos en el SP. El stack se utiliza para no perder los datos en cada llamada recursiva, así como tener un registro de como es que se están moviendo los discos en cada llamada a hanoi.

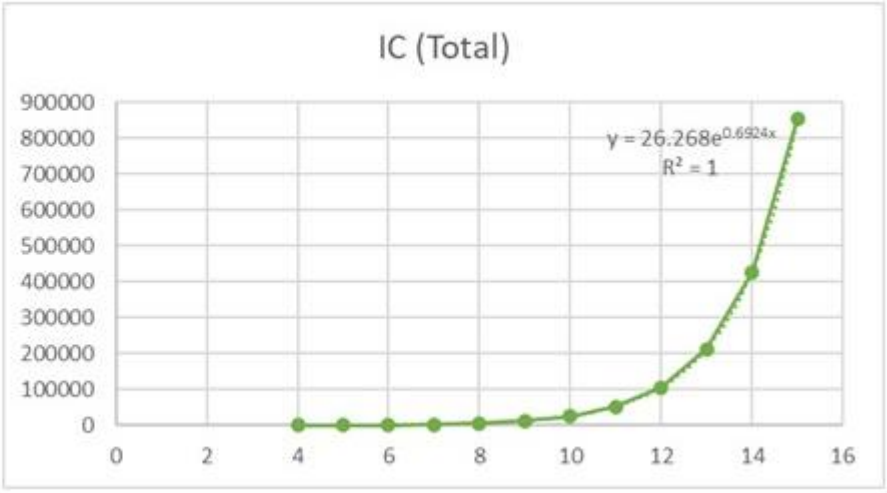
Incluir en el instruction count (IC) y especificar el porcentaje de instrucciones de tipo R, I y J para 8 discos.



R-type: 3% I-type: 61% J-type: 3%

Una gráfica que muestre como se incrementa el IC para las torres de Hanói en los casos de 4 a 15 discos.

# discos	IC (Total)
4	417
5	840
6	1679
7	3350
8	6685
9	13348
10	26667
11	53298
12	106553
13	213056
14	426055
15	852046



Conclusiones

* **Carolina Arellano:** Esta práctica fue un reto un poco complicado, ya que al principio fue un poco complicado visualizar cómo debía de ejecutar el código e implementarlo, sin embargo, fue de gran ayuda tener como base el algoritmo de Fibonacci, ya que con este se ve un poco más cómo es la recursividad implementada en ensamblador. Me pareció bastante interesante ver que no es para nada parecido a la programación en C o cualquier otro lenguaje, pues la lógica se maneja de forma distinta a como normalmente la conocemos.

* **Alicia Gómez:** La verdad es que desde un principio la parte de como traducir el código en C es un poco complicado, pero esta práctica ayudo a que todo quedara mucho más claro, tuvimos un problema de que se ciclaba al momento del segundo disco, no es un lenguaje que quisiera seguir implementando, pero, pues sí ayuda al aprendizaje y manejo de este.