Instituto Tecnológico de Costa Rica Área de Ingeniería en Computadores Lenguajes, Compiladores e Intérpretes

Módulo: Lenguajes

Profesor:
Marco Rivera Meneses

Tarea
Funcional #1

Estudiante:
Julio Varela Venegas

Grupo:2

I Semestre

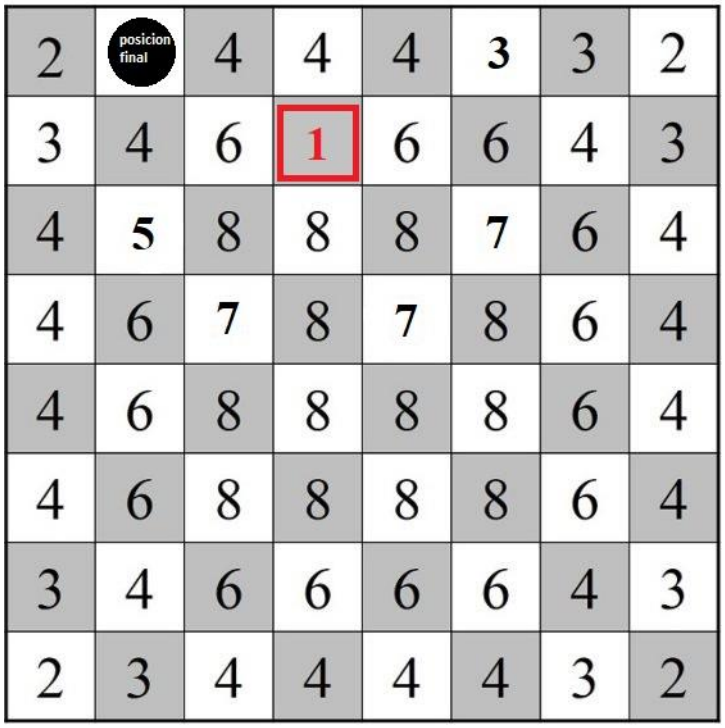
2023

Descripción de algoritmos desarrollados

El problema del caballo tiene varias soluciones matemáticas y lógicas planteadas a nivel de implementación de algoritmos, el que traté de desarrollar para esta tarea se basa en una distribución de números en cada casilla del tablero, de manera que cada casilla tenga la cantidad de movimientos posibles para el caballo colocado en esa misma posición, luego el caballo se moverá a la casilla con el número menor, dejando el número de movimientos que lleva en la casilla anterior, de manera que este proceso lo llevará a cabo hasta que no queden más movimientos por hacer, un ejemplo ilustrativa es el siguiente:

2	3	4	4	4	4	3	2
3	4	6	Posicion inicial	6	6	4	3
4	6	8	8	8	8	6	4
4	6	8	8	8	8	6	4
4	6	8	8	8	8	6	4
4	6	8	8	8	8	6	4
3	4	6	6	6	6	4	3
2	3	4	4	4	4	3	2

2	2	4	4	4	3	3	2
3	4	6	Posicion inicial	6	6	4	3
4	5	8	8	8	7	6	4
4	6	7	8	7	8	6	4
4	6	8	8	8	8	6	4
4	6	8	8	8	8	6	4
3	4	6	6	6	6	4	3
2	3	4	4	4	4	3	2



Este método se debe aplicar una y otra vez hasta que se haya cumplido con el total de posiciones en el tablero, que para este caso serían un total de 64 ya que el tablero es de 8x8.

Esta función principal que llama a las demás deberá ejecutarse recursivamente hasta lograr llegar al punto de parada que fue el mencionado en la idea anterior.

Funciones implementadas:

- **`solucion`:** Esta función es el punto de entrada principal. Toma un parámetro `n` que representa el tamaño de la matriz y `posicion` que indica la posición inicial del caballo. Si `n` es menor o igual a 4, muestra un mensaje indicando que no se puede crear la matriz. De lo contrario, llama a la función `crear_matriz` con los parámetros proporcionados.
- **`incrementar-contador`**: Esta función incrementa la variable global `contador` en 1.
- **`usar-contador`**: Esta función llama a `incrementar-contador` y luego muestra el valor actual de `contador`.
- **`crear_matriz`**: Esta función crea una matriz de tamaño `nxn` y la llena con valores calculados llamando a la función `Dar_Valor_Casillas`. La posición indicada por el parámetro `posicion` recibe el valor actual del contador. Retorna la matriz resultante.
- **`Dar_Valor_Casillas`**: Esta función calcula el valor correspondiente para cada casilla de la matriz. Utiliza las funciones `P_U_fila`, `S_P_fila` y `llenar_medio` para determinar los valores basados en la posición de la casilla en la matriz.
- **`P_U_fila`, `S_P_fila` y `llenar_medio`**: Estas funciones auxiliares son utilizadas por `Dar_Valor_Casillas` para asignar los valores a las casillas según su posición en diferentes filas de la matriz.
- **`posiciones_caballo`**: Esta función toma el tamaño del tablero `tamano` y la posición inicial del caballo `ficha`. Calcula todas las posibles posiciones a las que puede moverse el caballo y las devuelve como una lista.
- **`restar-indices`**: Esta función toma dos listas: `lista1` que representa la matriz original y `lista2` que contiene las posiciones a restar en la matriz. Modifica la matriz original restando 1 a los valores de las casillas correspondientes a las posiciones indicadas en `lista2`.

- **`encontrar-tablero`**: Esta función toma una lista `lista1` que representa una matriz y una sublista que contiene la posición `(i, j)` de una casilla. Actualiza el valor de la casilla en la posición `(i, j)` de la matriz restando 1.
- **`imprimir-matriz:`** Esta función imprime por consola la matriz modificada.
- **`realizar-movs`**: Esta función muestra el primer elemento de `lista2` por consola.
- **`PDC-Sol:`** Esta función principal toma el tamaño del tablero `n` y una lista `lista` que contiene la posición inicial del caballo. Llama a la función `solucion` para obtener la matriz original, luego llama a `posiciones_caballo` para obtener las posiciones posibles del caballo. A continuación, llama a `restar-indices` para modificar la matriz original, imprime la matriz modificada y muestra el primer movimiento del caballo.

Problemas sin solución:

El movimiento lógico de la ficha dentro del tablero de manera recursiva fue algo a lo que no se le pudo dar conclusión debido al tiempo por lo tanto quedó como un problema no resuelto.
Hizo falta la lógica suficiente junto con otras funciones extras para poder comparar las casillas restadas y que la ficha se moviera hacia la menor de todas dejando el número de movimientos que llevaba en la casilla anterior.

Problemas encontrados:

No es considerado un problema del todo, pero algunas de las funciones implementadas llevan funciones dentro como **let** o condicionales como un **if** que son considerados una mala practica para el tipo de programación que estamos viendo, ya que no se consideran totalmente funcionales.

Plan de actividades:

La tarea fue entregada el 5 de mayo, pero la inicié hasta el lunes 15 de mayo, debido a un examen de circuitos que tuve el día 13, empecé con la idea de terminar la primera función designada que debía encontrar una solución para el tablero y una posición determinadas, sin embargo solo logré realizar la matriz con los números correspondientes a cada casilla como se mostró en las imágenes anteriores (cantidad de movimientos), mostrar la primera posición de la ficha sobre el tablero y restar los índices a cada casilla alcanzable, sin embargo me faltó realizar la comparación entre los números alcanzables y posicionar la ficha en la posición con el menor índice.

Conclusiones

Al estar en un nuevo entorno de programación como lo es el funcional, es muy complicado para los estudiantes tener una comprensión adecuada del lenguaje y sus funcionalidades.

Con respecto a la elaboración de un algoritmo complejo, su implementación puede llegar a ser un desafío para aquellos que no están acostumbrados. Además, se requiere un conocimiento sobre las funciones de car, cdr y sus respectivas extensiones como caar, cddr y combinaciones.

Recomendaciones

Realizar una exhaustiva búsqueda sobre las librerías para el manejo y cumplimiento de las interfaces gráficas.

Bibliografía

Algorithms that Backtrack. (26 de 09 de 2003). Obtenido de Section 28: http://htdp.org/2003-09-26/Book/curriculum-Z-H-35.html#node_chap_28

Guzmán, J. E. (2005). Introducción a la programación con Scheme. En J. E. Guzmán, *Introducción a la programación con Scheme* (págs. 376-388). Cartago: Editorial Tecnológica de Costa Rica.

Bitácora

15/05/23. Día de inicio en la tarea

16/05/23. Logré imprimir la matriz con los índices para cada casilla, sin importar el tamaño de esta lo hacía correctamente

17/05/23. Logré restar los índices de las casillas alcanzables

18/05/23. Intenté elaborar alguna función que me pudiera extraer los números de las casillas y compararlos con los demás, aunque no logré finalizar.