Laboratorio No. 2
Tipos de Algoritmos
(Algoritmos Voraces y Backtracking)

Introducción
En este laboratorio el/la estudiante deberá desarrollar algoritmos utilizando técnicas de diseño y determinando los tipos de algoritmos. A saber: algoritmos voraces, backtracking, programación dinámica, divide y vencerás, algoritmos probabilísticos

Objetivos
Al finalizar este laboratorio, el/la estudiante deberá ser capaz de:

a. Determinar técnicas de diseño para los algoritmos planteados
b. Desarrollar algoritmos voraces
c. Desarrollar algoritmos recursivos utilizando backtracking
d. Aplicar conocimientos discutidos en clase

Contexto

1. Trabaje con un modelo de n capas (domain, controller, test, util).
2. Cree un nuevo proyecto llamado “Laboratory2” utilizando la tecnología javaFX, la cual permitirá trabajar en un entorno gráfico.
3. Defina una clase llamada “Knapsack” e implemente los métodos necesarios para resolver el problema de la mochila (Knapsack Problem) utilizando un tipo de algoritmo voraz. Recuerde que se requiere maximizar las ganancias con los objetos agregados en la mochila sin exceder el peso máximo permitido. **Utilice valor/peso para determinar el orden de selección de cada objeto candidato.**
   El siguiente fragmento de código muestra parte de la solución del problema:

   ```
   private Item[] list; //lista de objetos candidatos para la mochila
   private double capacity; //capacidad máxima en la mochila
   public Item[] solve(); método que resuelve el problema
   ```

   Nota: Item es una clase con los siguientes atributos: name, value, weight, valueWeight

4. Compruebe el funcionamiento de la mochila utilizando una testing class llamada KnapsackTest, de la siguiente forma:
   a. Agregue la siguiente lista a un arreglo de objetos tipo Item:

Nombre	Valor	Peso	Valor/peso
Smart TV 85” 4k	$1000	12 kg	83.3
Smart TV 55” 4k	$700	10 kg	70
Smart TV 32” 4k	$500	7 kg	71.4
PS5	$600	2 kg	300
Libro Java	$30	1 kg	30
Libro ANSI C	$35	1.3 kg	26.9
Samsung Galaxy	$700	0,5 kg	1400
Huawei Smart Phone	$600	0,5 kg	1200
Libro C++	$32	1,2 kg	26.6
Xbox One	$600	2.2 kg	272.72

Drone	$500	3 kg	166.6
Proyector	$200	3 kg	66.6
HP Laptop	$800	3 kg	266,6
MacBook Air	$1100	1.3 kg	846.1
Impresora 3D	$800	4 kg	200
iPhone 15	$1000	0.3 kg	3333.3
iPhone 14	$800	0.3 kg	2666.6
iPhone 13	$600	0.3 kg	2000
Libro Python	$40	0.7 kg	57.1
Libro AI	$80	1.2 kg	66.6

   b. Muestre la solución al problema de la mochila indicando la lista de elementos que se puede agregar y tomando en cuenta las siguientes capacidades máximas para cada mochila: mochila1: valor aleatorio tipo doble entre 10 y 15, mochila2: valor aleatorio doble entre 20 y 25, mochila3: valor aleatorio doble entre 30 y 35, mochila4: valor aleatorio doble entre 40 y 45, mochila5: valor aleatorio doble entre 50 y 55.

5. Utilice backtracking y desarrolle algoritmos recursivos que permitan resolver los siguientes problemas:
   a. El problema de las N-reinas (N Queen Problem). Nombre de la clase: NQueenProblem.
   b. El problema de los movimientos del caballo en un tablero de ajedrez (The Knight Tour Problem). Nombre de la clase: KnightTour.
   c. El problema del laberinto (Rat in a Maze). Nombre de la clase: RatInMaze.
   d. El problema del Sudoku (Sudoku Solver). Nombre de la clase: SudokuSolver.

6. Compruebe el funcionamiento de las N-reinas utilizando una testing class llamada NQuenProblemTest, que permita mostrar la solución para un tablero de 8x8 y otro de 4x4.

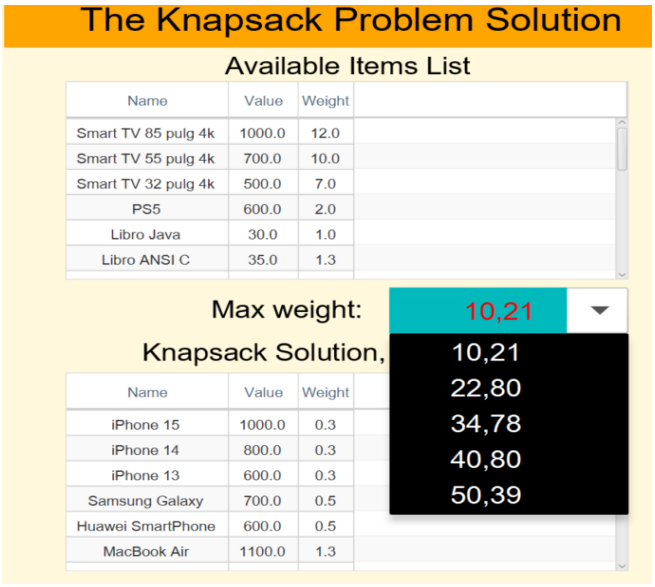
7. Compruebe el funcionamiento del Tour del Caballero utilizando una testing class llamada KnightTourTest, que permita mostrar la solución para un tablero de 8x8.

8. Compruebe el funcionamiento de la Rata en el Laberinto utilizando una testing class llamada RatInMazeTest, que permita crear un algoritmo que genere valores aleatorios de 0 y 1 y los coloque en un tablero de tamaño n x n. Muestre la solución para 5 laberintos con valores aleatorios de la siguiente forma: Maze1 de 4x4, Maze2 de 8x8, Maze3 de 10x10, Maze4 de 12x12, Maze5 de 16x16.

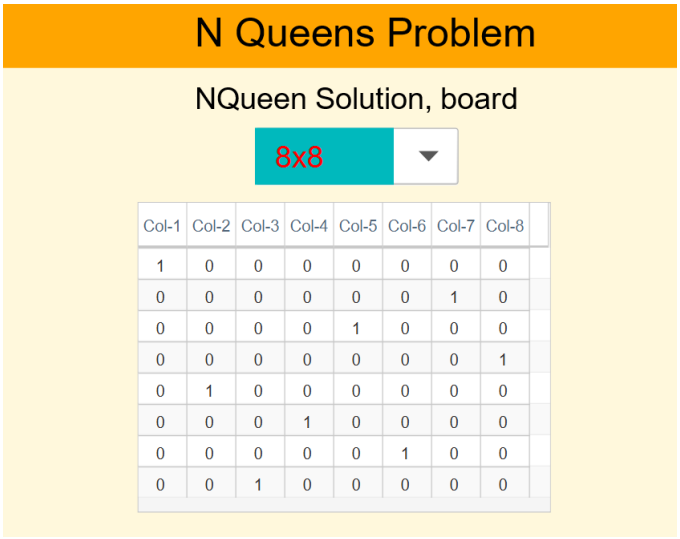
9. Compruebe el funcionamiento del Sudoku utilizando una testing class llamada SudokuSolverTest, que permita crear un algoritmo que genere números aleatorios entre 1 y 9 en cada celda, asegurándose que no se repitan en la misma fila, columna o bloque de 3x3. Muestre la solución para 5 tableros con datos iniciales diferentes.

10. Utilice la tecnología “javaFX” para crear un entorno gráfico que muestre un menú principal y la solución a los algoritmos anteriores, de la siguiente forma:

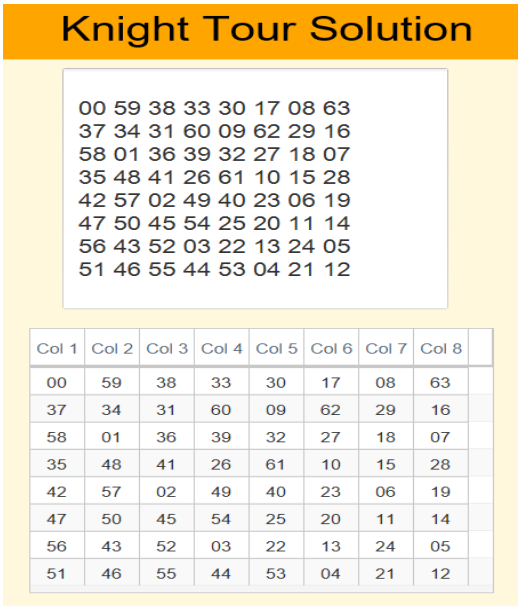
    a. Knapsack Problem: Utilice 2 objetos gráficos tipo TableView para mostrar la lista de ítems y la solución al problema de la mochila. Para la solución agregue un objeto tipo comboBox que permita seleccionar las capacidades máximas permitidas de acuerdo a lo siguiente: mochila1: valor aleatorio tipo doble entre 10 y 15, mochila2: valor aleatorio doble entre 20 y 25, mochila3: valor aleatorio doble entre 30 y 35, mochila4: valor aleatorio doble entre 40 y 45, mochila5: valor aleatorio doble entre 50 y 55.



    b. N Queen Problem: Utilice un objeto gráfico tipo ComboBox para seleccionar el tablero (4x4 o 8x8), un TextArea y un TableView para mostrar la solución según corresponda-.

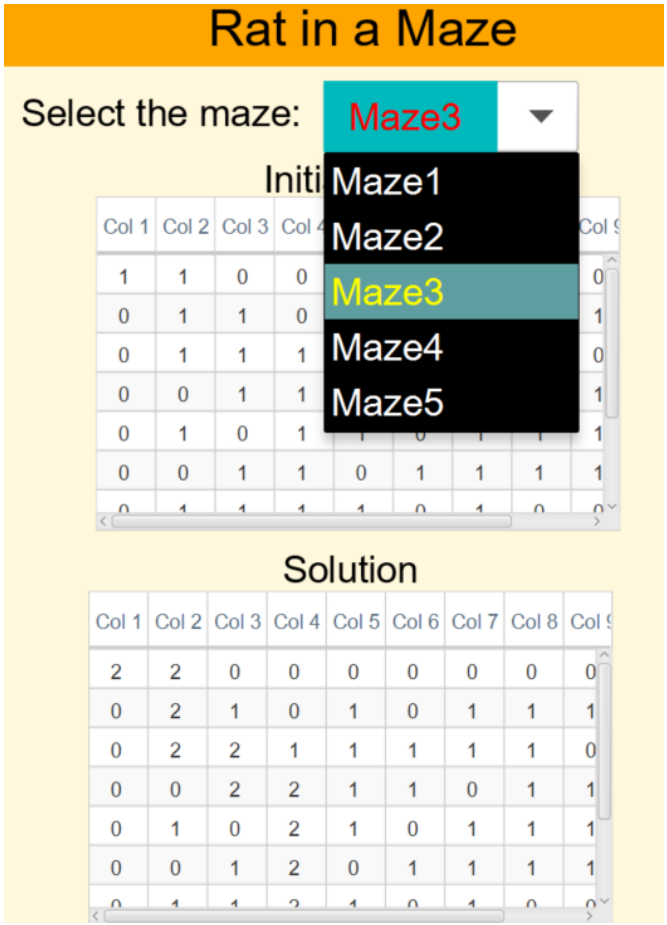


c. The Knight Tour: Similar al caso anterior.



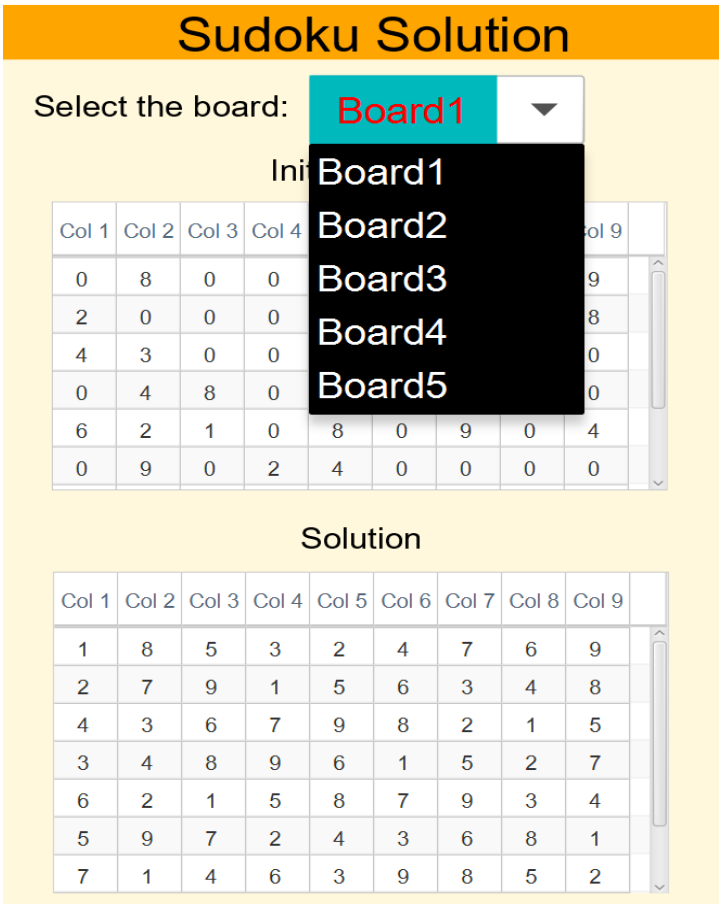
Col 1	Col 2	Col 3	Col 4	Col 5	Col 6	Col 7	Col 8
00	59	38	33	30	17	08	63
37	34	31	60	09	62	29	16
58	01	36	39	32	27	18	07
35	48	41	26	61	10	15	28
42	57	02	49	40	23	06	19
47	50	45	54	25	20	11	14
56	43	52	03	22	13	24	05
51	46	55	44	53	04	21	12

11. Rat in a Maze: Dos tableView, uno para mostrar el laberinto inicial y otro para mostrar la solución. Utilice el algoritmo creado para la clase de testeo y muestre 5 laberintos con valores aleatorios de la siguiente forma: Maze1 de 4x4, Maze2 de 8x8, Maze3 de 10x10, Maze4 de 12x12, Maze5 de 16x16.

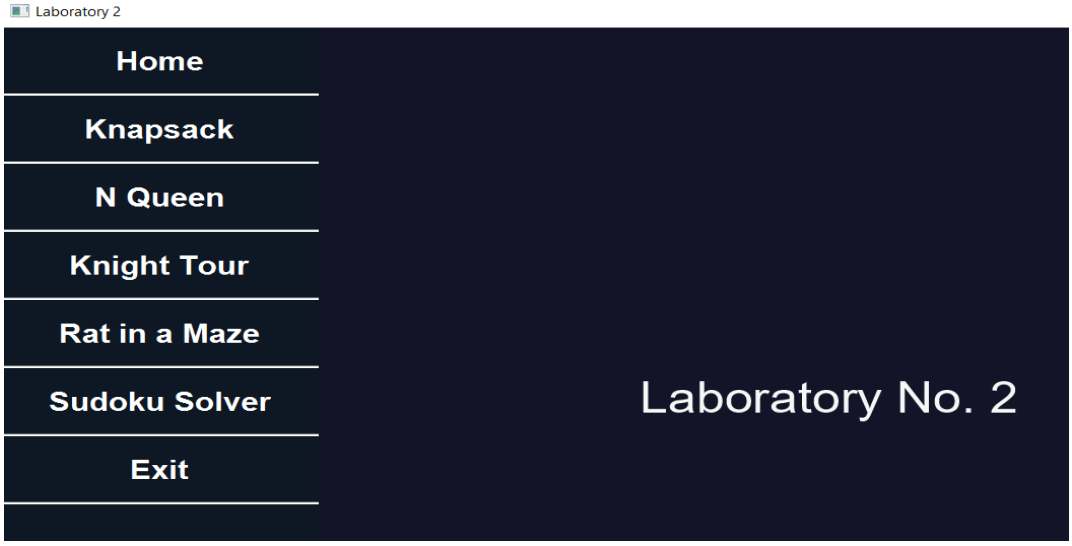


Col 1	Col 2	Col 3	Col 4	Col 5	Col 6	Col 7	Col 8	Col 9
2	2	0	0	0	0	0	0	0
0	2	1	0	1	0	1	1	1
0	2	2	1	1	1	1	1	0
0	0	2	2	1	1	0	1	1
0	1	0	2	1	0	1	1	1
0	0	1	2	0	1	1	1	1
0	1	1	2	1	0	1	0	0

a. Sudoku Solver: Utilice el algoritmo creado para la clase de testeo y muestre la solución para 5 tableros.



Un ejemplo del menú gráfico es el siguiente:



Resuelva y publique el laboratorio en el entorno del curso de la plataforma de mediación virtual (METICS). Verifique la fecha límite para el envío del informe.
URL: https://mv1.mediacionvirtual.ucr.ac.cr/course/view.php?id=7513