UNIVERSIDAD SAN FRANCISCO DE QUITO
COLEGIO: CIENCIAS E INGENIERíAS
Semestre: 202510 – Primer Semestre 2025/2026
Horario: LI 14:30 – 15:50 (Aula – M312)

TAREA I

TEMA:
Métodos de Búsqueda, Juegos y Heurística

DESCRIPCIÓN:
En esta tarea se van a familiarizar con algoritmos de búsqueda, diferentes estrategias heurísticas y la relación que tiene con la evaluación de juegos.

EJERCICIOS:

1. **8-Tile:**
   El 8-tile es un juego clásico que consiste en tener ocho baldosas en un tablero 3x3 donde cada baldosa tiene un número asociado del 1 al 8 y donde existe un espacio en blanco (vacío). El tablero permite deslizar cada baldosa de izquierda a derecha, de arriba hacia abajo y viceversa.

   La idea de este juego es conseguir distintas configuraciones del tablero sea que se inicia desde un tablero completamente ordenado, o de uno aleatorio.
   Su trabajo consiste en generar el árbol (o grafo) de posibles jugadas en este tablero para encontrar el camino (path) de jugadas que llevan desde un estado inicial del tablero al estado final (objetivo) del tablero. Para esto, explorarán distintas estrategias de búsqueda como son: breadth-first search, depth-first search y best-first search (la base del algoritmo A*). Para el caso de best-first search, exploren tres heurísticas: distancia euclideana, distancia manhattan y el número de baldosas correctamente ubicadas en el tablero.
   a. Breadth-first search.
   b. Depth-first search.
   c. Best-first search
      i. Número de baldosas bien ubicadas
      ii. Distancia Euclideana
      iii. Distancia Manhattan

   Para cada estrategia de búsqueda para un estado inicial definido y uno final deseado comparen:
   a. El número de estados visitados.
   b. La profundidad y/o el ancho del grafo/árbol generado.
   c. Muestre el árbol y el camino utilizado para encontrar la solución (¿es la misma solución la que encuentran las diferentes estrategias de búsqueda?)
   d. ¿Qué sucede con el uso de Best-first search? ¿Cuál heurísitca es mejor, por qué? ¿Qué característica tiene que tener una heurística para funcionar?

2. **8-Queens.**
   El problema de las 8 reinas en un tablero de ajedrez (8x8) es un problema clásico de computación y usualmente se utiliza para probar el concepto de backtracking (si no están familiarizados con el concepto, agreguen una pequeña definición en su informe). En esta ocasión, utilizando un lenguaje imperativo (C++, Java, C, Python, etc) cree un programa que permita resolver el problema genérico de ubicar N reinas (de ajedrez) en un tablero de dimensiones NxN.
   a. Tomando en cuenta un orden absoluto para las ubicaciones del tablero, por ejemplo:

1	2	3	4	…	N
N+1	N+2	…	…	…	2N
2N+1	…	…	…	…	3N
…	…	…	…	…	…
…	…	…	…	…	…
…	…	…	…	…	NxN

   Genere todas las soluciones posibles a este problema para tableros de:
   - 5x5 (5 Reinas)
   - 8x8 (8 Reinas)
   - 11x11 (11 Reinas)
   - 27x27 (27 Reinas)

   Reporte el número total de respuestas posibles y en el caso de los tableros 5x5 y 8x8 incluya las respuestas en uno o varios archivos de texto. ¿Qué sucede para los tableros de 11X11, 27x27 o de tamaño superior?

3. **4-en raya (tic-tac-toe modificado):**
   El 4-en raya es un juego inspirado en el clásico tic-tac-toe (o tres en raya). En particular, este juego se construye en una matriz 4x4. Y dos jugadores X o O toman turnos para intentar conseguir cuatro Xs o cuatro Os en línea. A diferencia del tres en raya, al ser un tablero par, no existe una ubicación central del tablero y las estrategias de juego pueden variar simétricamente sin un punto de ventaja.

   Su trabajo consiste en utilizar BestFirst Search utilizando dos heurísticas (Distancia Manhattan y Distancia Euclidiana) para encontrar el camino entre las siguientes jugadas (considere que X juega primero):

   a. Estados:

Inicial

			O
	X		
		O	
			X

Final

			O
X	X	O	
	O	O	
X	O	X	X

b. Estados:

Inicial

Final

X	X	X	O
O	O	O	X
	X		
O			

c. Estados:

Inicial

X			O
	X	O	O
	X	X	X
			O

Final

X	X	X	O
X	X	O	O
O	X	X	X
O	O	O	O

Reporte para cada caso el total de estados generados, el total de estados visitados y el camino (jugadas).

Para el caso (b) muestre que no es posible que exista un ganador.

4. **Programas Lógicos:** para los siguientes problemas, trabaje utilizando PROLOG.
   a. Problema del Farmer, Wolf, Goat, Cabbage: El Problema del Farmer, Wolf, Goat, Cabbage (Granjero, Lobo, Cabra, Col) es un famoso acertijo lógico que plantea el desafío de cómo un granjero puede llevar a su lobo, su cabra, su col y él mismo al otro lado de un río, utilizando un bote que solo puede llevar al granjero y a una de las otras dos cosas a la vez. Sin embargo, hay una restricción: si el granjero deja solo al lobo con la cabra en cualquier orilla, el lobo se comerá a la cabra, y si el granjero deja solo a la cabra con la col, esta comerá la col.
   Su trabajo consiste en modelar este problema y encontrar el conjunto de pasos que no violen la restricción, de forma que el granjero pueda cruzar de una orilla del río a la otra.

   b. Problema de los Misioneros y Caníbales: El problema de los misioneros y caníbales es otro acertijo lógico que plantea un desafío similar al del ejercicio anterior. Este problema involucra a tres misioneros y tres caníbales que se encuentran en un lado de un río y deben cruzarlo utilizando un bote que tiene una capacidad máxima de dos personas. Sin embargo, hay una restricción importante: en cualquier lado del río, si los misioneros son superados en número por los caníbales, los caníbales se comerán a los misioneros.
   Su trabajo consiste en modelar este problema y encontrar el conjunto de pasos que no violen la restricción, de forma que los tres caníbales y los tres misioneros puedan cruzar exitosamente de una orilla del río a la otra.

5. **Verificación:**
   Como parte de esta tarea, verifique las soluciones de los problemas de programación lógicos utilizando su lenguaje de programación imperativo de preferencia (de preferencia utilice backtracking).

6. **Redes Semánticas:**
   Revise la documentación de la W3C sobre redes semánticas (https://www.w3.org/2001/sw/wiki/Main_Page) y construya un grafo semántico (red semántica) del enunciado de la tarea. Utilice la especificación RDF en formato OWL. Incluya una visualización sencilla de la ontología creada (utilice Protégé).

ENTREGABLES:

Para esta tarea ustedes van a entregar:

1. Un pdf con los comentarios relevantes (respuestas y discusión) de los ejercicios prácticos.
2. Además, incluirán el código fuente de cada solución desarrollada y los archivos relevantes que hayan sido generados para resolver los ejercicios.
3. Finalmente, recuerden que dentro de la siguiente semana de entregar la tarea tendrán que defender la misma en horario de oficina del profesor. La defensa consiste en el 80% de su nota en esta actividad.

PREGUNTAS Y CONSULTAS:

Si tienen preguntas y consultas, no duden en escribir al e-mail:
driofrioa@usfq.edu.ec
Si tienen preguntas urgentes, pueden escribirme en Telegram bajo el usuario:
@danielriofrio