# Guía de Trabajos Prácticos – Programación Lógica 4

1. Escriba un programa en prolog que determine si una casilla del tablero de ajedrez se encuentra amenazada. En caso de estarlo, indicar quien la amenaza y desde que posición. Las piezas a tener en cuenta serán:
   - reina: mueve en diagonal y en linea recta
   - alfil: mueve siempre en diagonal
   - torre: mueve siempre en linea recta

   La forma de indicar la posición de las piezas rivales será:
   "pieza"(F, C).

   En donde:
   - "pieza" es una de las piezas descriptas arriba.
   - F es la fila en donde se encuentra la pieza
   - C es la columna en donde se encuentra la pieza

   Ejemplo:
   reina(2, 3).
   alfil(6, 3).
   torre(8, 5).

   amenaza(4, 5, R).
   R = reina(2, 3).
   R = alfil(6, 3).
   R = torre(8, 5).

2. Se desea programar un sistema de recomendación de películas para Netflix. Cuando un usuario inicia la aplicación, el sistema debe recomendarle una lista de películas, según lo que éste haya visto previamente. Para eso, consulta en toda la base de conocimientos la listas de películas de los usuarios de Netflix y organiza la recomendación en función de los que hayan visto las mismas películas que el usuario en cuestión, basado en un nivel de similaridad.

   Por ejemplo, si tenemos 3 usuarios en Netflix, y en la base de conocimientos tenemos que las películas que vio el usuario José fueron 'Misterio a Bordo', 'The Perfection', 'Otro día para matar','Barreras','Yo soy Sam','Inspection'; las que vio el usuario María fueron 'Quizás para siempre','A pesar de todo','La casa del Lago','Barreras'; y las que vio el usuario Pedro son 'Barreras', 'Otro día para matar', 'Yo soy Sam','A pesar de todo'; si Pedro inicia su sesión, el sistema debe recomendar películas en función de su lista de películas vistas y de la lista de otro usuario con cuya similitud de películas vistas sea mayor a 2. En esta caso, se debería obtener la/s película/s que Pedro no vio: 'A pesar de todo'.

   Se solicita, en base a este ejemplo, crear la recomendación de la/s película/s para el usuario Pedro.

3. Implemente en prolog los predicados que se describen a continuación teniendo en cuenta las siguientes consideraciones:
   - No se deben usar predicados predefinidos
   - Se estará trabajando con conjuntos
   - Los mismos se representarán como listas de elementos
   - Los conjuntos no pueden tener elementos repetidos, condición que hay que validar

en los conjuntos de entradas y asegurar en el de salida

a. union(C1, C1, U)
   El predicado union/3 ejecuta la unión entre los conjuntos C1 y C2 y devuelve el resultado en el conjunto U.
   union([1, 2, 3], [3, 4, 5], U).
   U = [1, 2, 3, 4, 5].
b. interseccion(C1, C1, U)
   El predicado interseccion/3 ejecuta la intersección entre los conjuntos C1 y C2, y devuelve el resultado en el conjunto U.
   interseccion([1, 2, 3], [3, 4, 5], U).
   U = [3].
c. diferencia(C1, C1, U)
   El predicado diferencia/3 ejecuta la operación diferencia de conjuntos entre los conjuntos C1 y C2, y devuelve el resultado en el conjunto U.
   diferencia([1, 2, 3], [3, 4, 5], U).
   U = [1, 2].

IMPORTANTE: No utilizar predicados predefinidos y verificar que los conjuntos de entrada sean válidos, es decir que no contengan elementos repetidos, en caso de ser inválidos alguno de los dos, el predicado debe fallar.

4. Dada una lista de itinerarios de colectivos para viajes de larga distancia, se pide encontrar las combinaciones posibles para viajar sabiendo el origen, el destino y el horario al que se necesita arribar al destino con hasta dos horas de espera en cada punto intermedio o en el arribo final.
   Los viajes están representados por el predicado viaje/4, en donde los argumentos con :
   - Origen
   - Destino
   - Hora de salida
   - Duración del viaje

   Se debe devolver una lista donde cada elemento de la misma representa un viaje como una lista con [Origen, Destino, HoraPartida, HoraLlegada].
   Recordar la restricción de que no puede pasar más de dos horas entre la llegada a un destino y la partida hacia otro, o hasta la hora que hemos puesto como hora de arribo a destino final.

   Ejemplo:

   camino('Paraná', 'Córdoba', 21, Tramos).
   Tramos = [['Paraná', 'Santa Fe', 13, 14], ['Santa Fe', 'Córdoba', 16, 21]].

   camino('Paraná', 'Resistencia', 13, Tramos).
   Tramos = [['Paraná', 'Corrientes', 3, 12], ['Corrientes', 'Resistencia', 12, 13]].
   Tramos = [['Paraná', 'Santa Fe', 13, 14], ['Santa Fe', 'Córdoba', 16, 21], ['Córdoba', 'Tucumán', 22, 6], , ['Tucumán', 'Resistencia', 7, 12]].

   El cuerpo de conocimientos es el siguiente:
   viaje('Paraná', 'Santa Fe', 7, 1).
   viaje('Paraná', 'Santa Fe', 13, 1).
   viaje('Paraná', 'Santa Fe', 18, 1).
   viaje('Santa Fe', 'Paraná', 4, 1).
   viaje('Santa Fe', 'Paraná', 13, 1).
   viaje('Santa Fe', 'Paraná', 19, 1).

```
viaje('Paraná', 'Corrientes', 3, 9).
viaje('Paraná', 'Corrientes', 8, 9).
viaje('Paraná', 'Corrientes', 16, 9).
viaje('Corrientes', 'Paraná', 0, 9).
viaje('Corrientes', 'Paraná', 10, 9).
viaje('Corrientes', 'Paraná', 18, 9).
viaje('Santa Fe', 'Rosario', 5, 3).
viaje('Santa Fe', 'Rosario', 9, 3).
viaje('Santa Fe', 'Rosario', 20, 3).
viaje('Rosario', 'Santa Fe', 3, 3).
viaje('Rosario', 'Santa Fe', 11, 3).
viaje('Rosario', 'Santa Fe', 18, 3).
viaje('Santa Fe', 'Córdoba', 6, 5).
viaje('Santa Fe', 'Córdoba', 16, 5).
viaje('Santa Fe', 'Córdoba', 22, 5).
viaje('Córdoba', 'Santa Fe', 8, 5).
viaje('Córdoba', 'Santa Fe', 14, 5).
viaje('Córdoba', 'Santa Fe', 20, 5).
viaje('Santa Fe', 'Resistencia', 3, 9).
viaje('Santa Fe', 'Resistencia', 12, 9).
viaje('Santa Fe', 'Resistencia', 19, 9).
viaje('Resistencia', 'Santa Fe', 6, 9).
viaje('Resistencia', 'Santa Fe', 12, 9).
viaje('Resistencia', 'Santa Fe', 16, 9).
viaje('Corrientes', 'Resistencia', 8, 1).
viaje('Corrientes', 'Resistencia', 12, 1).
viaje('Corrientes', 'Resistencia', 16, 1).
viaje('Resistencia', 'Corrientes', 9, 1).
viaje('Resistencia', 'Corrientes', 11, 1).
viaje('Resistencia', 'Corrientes', 17, 1).
viaje('Resistencia', 'Tucumán', 2, 5).
viaje('Resistencia', 'Tucumán', 8, 5).
viaje('Resistencia', 'Tucumán', 19, 5).
viaje('Tucumán', 'Resistencia', 4, 5).
viaje('Tucumán', 'Resistencia', 7, 5).
viaje('Tucumán', 'Resistencia', 12, 5).
viaje('Tucumán', 'Resistencia', 18, 5).
viaje('Córdoba', 'Tucumán', 5, 8).
viaje('Córdoba', 'Tucumán', 11, 8).
viaje('Córdoba', 'Tucumán', 22, 8).
viaje('Tucumán', 'Córdoba', 7, 8).
viaje('Tucumán', 'Córdoba', 12, 8).
viaje('Tucumán', 'Córdoba', 18, 8).
```