# Práctica 2
# Representación y búsqueda en la librería AIMA

**Fecha de entrega: 31 de octubre del 2017**

El objetivo de esta práctica es conocer las técnicas de representación de problemas y los algoritmos de búsqueda disponibles en la librería Java asociada al libro AIMA[1].

## 1. Librería Java de AIMA

La librería Java de AIMA se puede descargar de https://github.com/aimacode/aima-java/releases. Al descomprimir el zip descargado aparecerán los siguientes directorios:

- aima-core: Algoritmos implementados.
- aima-gui: Demos de aplicaciones desarrolladas utilizando aima-core.
- aimax-osm: Aplicaciones y librerías relacionadas con Open Street Map (OSM). Framework para desarrollar sistemas de navegación.
- release: archivos jar de la librería que usaremos en la práctica 3.

La librería AIMA contiene un framework (clases genéricas Java) que separa la representación del problema de los algoritmos de búsqueda. También contiene demos de varios problemas clásicos: NReinas, 8-Puzzle, etc.

La librería implementa algoritmos de búsqueda ciega y de búsqueda heurística. Entre otros podréis encontrar los siguientes algoritmos implementados:

- Búsqueda ciega:
  - Primero en anchura: BreathFirstSearch.java
  - Primero en profundidad: DepthFirstSearch.java
  - Coste uniforme: UniformCostSearch.java
  - Profundidad limitada: DepthLimitedSearch.java
  - Profundización iterativa: IterativeDeepingSearch.java
- Búsqueda heurística:
  - Voraz: GreedyBestFirstSearch.java
  - A*: AStarSearch.java

Para más información acerca de la librería podéis consultar el archivo: README.md en el repositorio GitHub de aima-java (https://github.com/aimacode/aima-java).

[1] Russell, S., Norvig, P., Artificial Intelligence: A Modern Approach (3rd Edition).

## 2. Desarrollo de la práctica

1. Importad en Eclipse los proyectos aima-core, aima-gui y aimax-osm (puedes seguir las instrucciones para Eclipse en https://github.com/aima-java/aima-java/wiki/AIMA3e-Using-Eclipse-IDE).
2. Ejecutad IntegratedAimaApp (aima.gui.swing.applications).
3. Probad la aplicación ejecutando distintos ejemplos, en ejecución completa y paso a paso. Seleccionad distintos problemas, estados iniciales y algoritmos.
4. Puzle de 8:
   a. Ejecutad el problema del puzle de 8 con el ejemplo extremo, probando la búsqueda en anchura, el método voraz (con ambas heurísticas) y el algoritmo A* (con las dos heurísticas disponibles).
   b. Analizad el coste de las soluciones encontradas. ¿Son óptimas? En los casos en los que el algoritmo no ha encontrado la solución óptima analizad qué es lo que ha ocurrido.
   c. Analizad el coste de memoria de cada uno de los algoritmos. ¿A qué se deben las diferencias en este aspecto entre los distintos algoritmos?
   d. A la vista de los resultados obtenidos, ¿qué algoritmo es mejor? Razonad vuestra respuesta.
   e. Explorad el código de la librería y localizad la definición del estado y de los operadores en el problema del puzle de 8.
   f. Explorad el código de la librería y localiza el test de estado objetivo.
   g. Explorad el código de la librería y localizad la definición de la heurística Manhattan.
5. Búsqueda de caminos:
   a. Ejecutad la búsqueda de caminos en el mapa de Rumanía, desde Arad a Eforie, utilizando la búsqueda en anchura, coste uniforme, voraz y el algoritmo A*. Todos con TreeSearch y GraphSearch.
   b. Analizad el coste de las soluciones encontradas. ¿Son óptimas? En los casos en los que el algoritmo no ha encontrado la solución óptima analizad qué es lo que ha ocurrido.
   c. Analizad el coste de memoria de cada uno de los algoritmos. ¿A qué se deben las diferencias en este aspecto entre los distintos algoritmos?
   d. A la vista de los resultados obtenidos, ¿qué algoritmo es mejor? Razonad vuestra respuesta.
   e. Explorad el código de la librería y localizad la definición del estado y de los operadores en el problema de la búsqueda de caminos.
6. Explorad la librería y buscad los dos esquemas generales de búsqueda: TreeSearch y GraphSearch. Mirad el código con detenimiento, ¿en qué se diferencian estos dos esquemas?

7. Explorad la librería y buscad el esquema de búsqueda primero en anchura. Mirad el código con detenimiento, ¿en qué momento se comprueba si un nodo es objetivo? ¿antes de añadirlo a la frontera o antes de expandirlo? ¿por qué crees que se hace así?
8. Analizad el código de uno de los algoritmos de búsqueda informada para ver cómo está implementado en la librería.

## 3. Memoria

La memoria que debéis de entregar debe de contener el **nombre de los dos integrantes del grupo** y, al menos, lo siguiente:

1. Puzle de 8. Esta sección debe de tener, al menos, los siguientes apartados:
    a. Tabla comparativa de los resultados obtenidos con los 5 algoritmos.
    b. Optimalidad de las soluciones encontradas. Explicad qué algoritmos encuentran la solución óptima y cuáles no y analizad por qué ocurre esto.
    c. Coste de memoria. Analizad las diferencias, respecto al coste de memoria, de los 5 algoritmos y explicad a qué se deben estas diferencias.
    d. Indicad cuál es el mejor de los 5 algoritmos para la resolución del puzle de 8. Justificad vuestra respuesta.
    e. Indicad donde se definen los estados y los operadores del puzle de 8 y explicad como están implementados.
    f. Indicad donde se define el test de estado objetivo en el puzle de 8 y explicad como está implementado.
    g. Indicad donde se define la heurística Manhattan y explicad como está implementada.
2. Búsqueda de caminos. Esta sección debe de tener, al menos, los siguientes apartados:
    a. Tabla comparativa de los resultados obtenidos con los 4 algoritmos (cada uno con TreeSearch y GraphSearch).
    b. Optimalidad de las soluciones encontradas. Explicad qué algoritmos encuentran la solución óptima y cuáles no y analizad por qué ocurre esto.
    c. Coste de memoria. Analizad las diferencias, respecto al coste de memoria, de los 4 algoritmos y explicad a qué se deben estas diferencias.
    d. Indicad cuál es el mejor de los 4 algoritmos para la búsqueda de caminos entre dos ciudades. Justificad vuestra respuesta.
    e. Indicad donde se definen los estados y los operadores de la búsqueda de caminos y como están implementados.

3. Esquemas generales de búsqueda. Indicad en que paquete se encuentran los esquemas generales de búsqueda TreeSearch y GraphSearch y explicad la diferencia entre ambos.
4. Algoritmo primero en anchura. Indicad en que paquete se encuentra el algoritmo de búsqueda primero en anchura e indicad en qué momento se comprueba si un estado es objetivo y por qué creéis que se ha tomado esa decisión.
5. Algoritmo de búsqueda informada. Indicad cual es el algoritmo qué habéis elegido, en que paquete y clase se encuentra su implementación y cómo esta implementado.

## 4. Entrega de la práctica

La práctica se entregará a través del Campus Virtual. Se habilitará una nueva tarea: **Entrega de la Práctica 2** que permitirá subir un pdf que contendrá la memoria de la práctica.

El fichero subido deberá tener el siguiente nombre: **Practica2GXX.pdf**, siendo XX el nombre del grupo. Por ejemplo, *Practica2G01.pdf*.

Uno sólo de los miembros del grupo será el encargado de subir la práctica.

La fecha límite para entregar la Práctica 2 será el martes **31 de octubre a las 23:55**. Se recomienda no dejar la entrega para el último momento para evitar problemas de última hora.

No se corregirá ninguna práctica que no cumpla estos requisitos.