**WATER SORT PUZZLE**

**Grupo:** BC-09

**Integrantes:**

* Juan Caravantes Algaba
* Cristian Ballesteros Serrano
* Gregorio Barchino Rodríguez-Caro

**1. DESCRIPCIÓN DEL PROBLEMA**

El problema consiste en implementar un programa de búsqueda para resolver un Water Sort Puzzle. Para ello hay que aplicar los conceptos estudiados en el Tema 2 de la teoría sobre búsquedas. El estado inicial de puzle se introducirá en un fichero json y habrá que generar un archivo con la solución al puzle siguiendo la estrategia de búsqueda indicada.

**2. CÓMO SE HA RESUELTO**

**ENTREGABLE 1**

Al principio implementamos la lógica del artefacto usando JSONArrays, ya que pensábamos que era la única estructura de almacenamiento de información que podíamos usar, es decir, que no podíamos usar pilas, listas, etc. Eso nos llevó a crear la clase Botella, que era una implementación nuestra de una pila.

Luego nos enteramos de que sí podíamos usar pilas y listas, así que eliminamos la clase Botella y pasamos a usar un array de pilas. En este momento fue también cuando implementamos la funcionalidad de Accion y ES\_Posible.

A continuación, fue cuando se subieron los tests para leer estados de un fichero, y nos dimos cuenta de que nuestra implementación del método Acción no era del todo acertada, así que creamos la clase Juego, que contenía el JSON y nuestro array de pilas. Dentro de esta clase se realizaba la acción.

Lo siguiente fue implementar la clase para leer el fichero con los estados, así que eso hicimos. Esta clase nos leía la línea de fichero y nos devolvía un array de pilas, que era el estado que se había leído. El problema es que solo podía leer una línea si estaba bien hecha, es decir, las líneas con errores no las leía bien.

Después de esto, como el código no estaba muy optimizado debido al uso de un array para guardar las pilas, reacondicionamos el código para usar arraylist en vez de array. Eso hizo que el código quedase más limpio.

Para terminar, lo que hicimos fue cambiar la clase para leer el fichero de forma que pudiera leer cualquier línea, y en caso de ser errónea, que lo informara.

**ENTREGABLE 2**

Antes de comenzar con la tarea 2, actualizamos la clase para leer fichero para que nos construyese el json también.

Una vez finalizado eso, comenzamos haciendo un pequeño esbozo de la clase WP, y a continuación nos repartimos las tareas a realizar:

* Gregorio realizaría la función objetivo
* Juan y Cristian realizarían la función sucesores

Gregorio terminó su parte antes de Juan y Cristian terminar la suya, por lo que se unió a ayudarles. Para crear la función sucesores, creamos la clase Sucesor, que tendría los 3 atributos que constituyen un sucesor: acción, estado resultado, y coste.

Para terminar, añadimos a la clase Leer un método para leer los ficheros que contendrían la definición de los problemas. Además, reacondicionamos el método leerEstado() para esto.

**Entregable 3**

Creamos la clase Nodo con las especificaciones que se nos dieron.

Para la frontera, usamos simplemente una cola con prioridad de las que vienen implementadas en java. Para hacerla funcional, implementamos la interfaz Comparable en la clase Nodo y creamos el método compareTo().

En lo que respecta a las comprobaciones de la implementación del árbol de búsqueda, comentar que todo funcionó correctamente a la primera.

A continuación, comenzamos a implementar el algoritmo de búsqueda en el Espacio de Estados. La estructura de visitados la implementamos con un arraylist con los estados visitados. Para comprobar si un estado está en la lista de visitados, usamos el método contains() que trae la clase ArrayList, y para que contains() funcione, hemos creado el método equals() en la clase Estado, y lo que hace, fundamentalmente, es comprobar si los toString() de los dos estados son iguales.

Por último, cuando fuimos a intentar resolver los problemas proporcionados por el profesor, nos dimos cuenta de que no conseguíamos encontrar una solución, ya que se quedaba en bucle infinito. Para solucionar esto tuvimos que solucionar dos errores:

* El primero era que, a la hora de establecer el atributo valor de un nodo usando la estrategia en profundidad, este no se asignaba correctamente. El error se daba en cómo hacíamos la asignación: valor = 1 / (1 - profundidad). Como valor se trata de un double, lo que ocurría era que el valor de “valor” siempre era 0 (exceptuando el nodo inicial, que tenía correctamente el valor 1). Esto era porque el valor de la división no lo trataba como un número decimal, sino como un entero. Para solucionar eso, simplemente hubo que cambiar el 1 del numerador a 1.0, quedando así: 1.0 / (1 - profundidad).
* El segundo era que los criterios que seguía el método ES\_Valido() para ver si una acción era válida estaban equivocados, así que los corregimos.

Una vez solucionados estos errores, el algoritmo ya funcionaba, o eso creíamos… Tras esto creamos el método exportarSolucion() para exportar el camino al estado solución en un fichero, y pensábamos que ya habíamos acabado, pero nos quedaba corregir una cosa más.

Esta cosa era que el algoritmo, en realidad, no terminaba de funcionar bien, y esto era debido a que nos faltó definir bien que estado era solución o no, es decir, la función objetivo no funcionaba como debía. El error se hallaba en que había más de una botella con un mismo tipo de líquido, cuando, para que fuera solución, todo el líquido de un mismo tipo tenía que estar en la misma botella. Tras arreglar esto ya todo funcionaba como debía y conseguimos acabar la tarea 3.

**Entregable 4**

El principal contenido de esta entrega era la creación de la función heurística. Esta la implementamos en la clase Nodo y, siguiendo el guion de la entrega y el ejemplo mostrado en ella, conseguimos crearla sin complicaciones.

A continuación, simplemente añadimos al switch del método setValor() de Nodo los case para las búsquedas Greedy y A.

Tras eso cambiamos la manera de comparar si dos estados eran iguales. Hasta ahora lo hacíamos comparando los toString de dos estados en el método equals(), pero como nos indicó el profesor, lo indicado era comparar los md5 de los mismos, así que eso hicimos. Para ello buscamos en internet cómo hacerlo, y con ello implementamos el método calcularMD5(). Lo único que quedaba era en el método equals() comparar los md5 en lugar de los toString.

Para finalizar revisamos el código y eliminamos el código comentado, además de eliminar los métodos que no se usaban (como varios getters y setters) y cambiar los métodos que usaban varios return para que solo usaran uno.

**3. MANUAL DE USUARIO**

Para utilizar el software hay que extraer el zip dentro de un proyecto Java en tu IDE de Java preferido. Para ejecutarlo simplemente tienes que abrir el fichero *Ejecutable.java* y lanzarlo. Una vez hecho eso se te pedirá que introduzcas el nombre de uno de los problemas ejemplos que vienen en la carpeta *Problemas* y que escojas una de las 5 estrategias de búsqueda. Una vez completada la búsqueda se te mostrará por pantalla el camino al nodo solución encontrado. Adicionalmente se te habrá creado un fichero en la carpeta *Soluciones* con el nombre del problema y la búsqueda utilizada que contendrá el mismo camino que se te mostró por pantalla.

**4. OPINIONES Y REPARTO DEL TRABAJO**

**Cristian** (Participación: Como todo el mundo)

El laboratorio ha estado muy bien estructurado, ha sido bastante fácil de seguir y cuando nos quedábamos un poco pillados el profesor nos ayudaba. Si tuviera que decir algo negativo al respecto sería que había ocasionas en las que nos mareaban un poquito, ya que nos daban unas indicaciones respecto a cómo implementar ciertos métodos, y a la semana siguiente eran otros. Pero en general ha sido una de las mejores experiencias en lo que llevo de tercero.

**Gregorio** (Participación: Más que el resto, pero no mucho más)

El laboratorio ha sido uno de los mejores en lo que llevo de carrera, principalmente debido a que está muy bien estructurado y explicado, lo que hace que muy rara vez te sientas perdido y no sepas como atacar el problema. Y esto resalta mucho sobre todo cuando tenemos laboratorios de otras asignaturas en los que simplemente hacemos un “Hola mundo” con un lenguaje nuevo, y luego nos dan un guion para realizar Netflix. Laboratorios como ese hacen que valores aún más laboratorios como este.

El laboratorio iba muy de la mano con la parte de teoría, lo que ha hecho que a la hora de tener que implementar el algoritmo de búsqueda haya sido muy sencillo, ya que teníamos las bases muy bien asentadas.

Comentar también que el hecho de que no haya entregas obligatorias semanales ni nada por el estilo ha ayudado bastante, sobre todo a nivel de estrés, ya que si en alguna otra asignatura se presentaba alguna deadline podía hacer un momento a un lado este laboratorio sin ser penalizado.

Y para concluir, el hecho de que pudiéramos usar el lenguaje que quisiéramos para desarrollar la práctica ha estado bastante bien. Se agradece que no nos introduzcan cada dos por tres un nuevo lenguaje de programación que seguramente no volvamos a usar en nuestra vida…

Mi experiencia, sinceramente, ha sido que este laboratorio es el mejor del primer cuatrimestre de tercero, y seguramente lo pondría en mi top 5 de lo que llevo de carrera.

**Juan** (Participación: Como todo el mundo)

El laboratorio ha sido interesante y bastante dinámico, nos han incentivado a trabajar en equipo utilizando GitHub y hacer la repartición de tareas. Había veces en las que utilizábamos esta mecánica y otras en las que no, pero en general ha sido una experiencia nueva. A la hora de presentar las prácticas todo estaba detallado y la carga de trabajo era adecuada teniendo en cuenta que hay más laboratorios a parte de este.

El profesor era receptivo y nos aclaraba las dudas sin ningún tipo de problema, aparte de que el hecho de que haya varías oportunidades para poder realizar la prueba del laboratorio hace que la presión sea menor y se pueda trabajar a un ritmo que se adecue bastante bien a los horarios.

Se nos ha dado mucha libertad, podíamos elegir el lenguaje de programación que usaríamos, no había entregas, se nos ha permitido trabajar sin presiones y al ritmo que cada uno necesitaba. El problema a resolver propuesto era interesante y, aparte, iba coordinado con el avance de la asignatura en su gran mayoría. Siento que es uno de los mejores laboratorios que he tenido ya que siento que he aprendido algo de verdad y que puedo utilizar en el ámbito laboral o personal.