

# Algoritmos y Estructuras de Datos I

## Memoria Proyecto Prácticas

Curso 2022-2023

**Grupo de prácticas:** *PCEO*










<b>Autores:</b>
<i>Jose Guillén Riquelme</i>
<i>Gonzalo Lucena Vázquez</i>
<b>Usuario juez on-line:</b>
<i>D23</i>

# Índice

<b>1. Prueba de aceptación en Mooshak</b>	<b>3</b>
<b>2. Análisis y diseño del problema</b>	<b>4</b>
<b>3. Listado del código</b>	<b>7</b>
<b>4. Informe de desarrollo</b>	<b>8</b>
<b>5. Conclusiones y valoraciones personales</b>	<b>9</b>

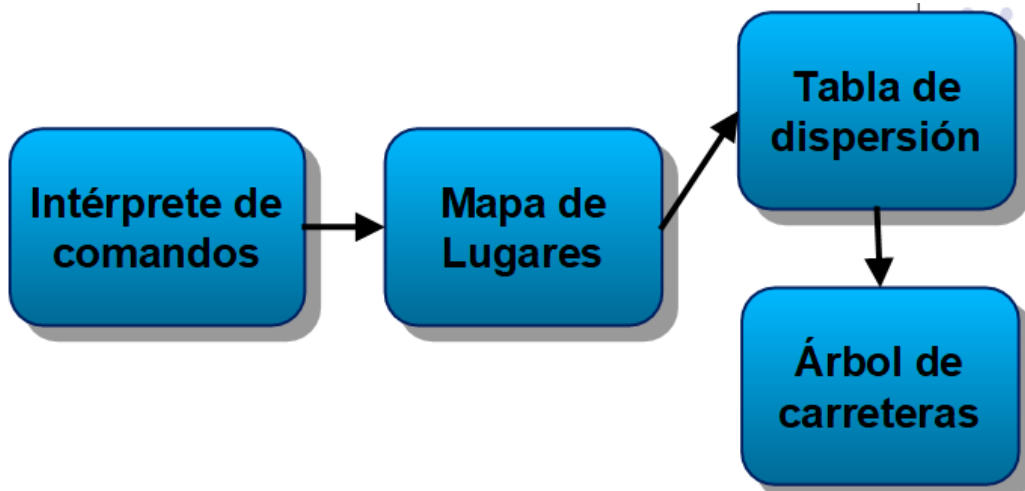
# 1. Prueba de aceptación en Mooshak

(Del 301 y 302 se tomaría el último enviado)

<a href="#">582</a>	144:08:58		G4 LUCENA VAZQUEZ, GONZALO	001	C++	1 Accepted	final
<a href="#">586</a>	146:18:03		G4 LUCENA VAZQUEZ, GONZALO	002	C++	1 Accepted	final
<a href="#">2754</a>	1403:19:45		G4 LUCENA VAZQUEZ, GONZALO	301	Archivo_TAR	2 Accepted	final
<a href="#">1976</a>	1140:25:03		G4 LUCENA VAZQUEZ, GONZALO	301	Archivo_TAR	2 Accepted	final
<a href="#">1931</a>	1104:18:35		G4 LUCENA VAZQUEZ, GONZALO	201	Archivo_TAR	3 Accepted	final
<a href="#">1536</a>	730:27:19		G4 LUCENA VAZQUEZ, GONZALO	005	Archivo_TAR	2 Accepted	final
<a href="#">954</a>	362:53:55		G4 LUCENA VAZQUEZ, GONZALO	004	C++	2 Accepted	final
<a href="#">939</a>	341:57:13		G4 LUCENA VAZQUEZ, GONZALO	003	C++	1 Accepted	final
<a href="#">2989</a>	2162:57:55		G4 LUCENA VAZQUEZ, GONZALO	302	Archivo_TAR	4 Accepted	final

## 2. Análisis y diseño del problema.

- a. ¿Qué clases se han definido y qué relación existe entre ellas? Incluir una representación gráfica de las clases.
- ➔ Primero se creó el intérprete de comandos. Tras ello se definieron las clases Lugar, la cual se usaría explícitamente para la clase MapaLugares. Posteriormente se crearía la clase TablaHash para ir almacenando y, por último, las clases Carretera y ArbolAVL. He aquí una representación gráfica:



- b. ¿Qué módulos existen, qué contienen y cuál es la relación de uso entre ellos? Poner una representación gráfica de los ficheros y sus dependencias (includes).
- ➔ Existe un módulo para el main, otro para el intérprete de comandos, otro para el mapa, otro para la tabla hash y otro para el árbol AVL.
- ÁrbolAVL: Contiene la clase Lugar, la clase Nodo y la clase Arbol. No depende de otros ficheros.
  - TablaHash: Contiene la clase Carretera (necesaria en el uso de la tabla hash) y la clase TablaHash. Depende del árbol AVL.
  - MapaLugares: Contiene la clase MapaLugares. Incluye el fichero de la tabla hash.
  - Intérprete: Contiene la clase Intérprete. Incluye el fichero de cabecera MapaLugares.
  - Main: Fichero cpp que depende de MapaLugares.

La representación gráfica podría ser la de arriba, pero con las flechas en sentido contrario, ya que hemos usado módulos justamente para dichas clases.

- c. ¿Contiene el makefile todas las dependencias existentes?
- ➔ Sí, todas especificadas.
- d. ¿Qué tipo de tablas de dispersión se ha usado y por qué? Justificar la decisión.
- ➔ Se ha usado la dispersión abierta con tablas de tamaño fijo ya que hemos pensado en dar prioridad a la sencillez que tiene este formato antes que la adaptación que se

*podría llegar a tener mediante tablas con amaño variable. En cualquier caso, siempre nos planteamos usar la dispersión abierta.*

- e. *¿Qué función de dispersión se ha usado? ¿Se han probado varias? Justificar la elección de la función de dispersión, en función de su eficiencia.*  
 ➔ *Se ha decidido usar la función de dispersión djb2a, ya que pudimos comprobar que se adaptaba muy bien a esta actividad mientras que otras, que a priori pueden ser más sencillas (como por ejemplo, la de suma posicional).*
- f. *Si se hace reestructuración de la tabla, explicar cómo se hace.*  
 ➔ *No se hace.*
- g. *¿Cómo se libera la tabla de dispersión?*  
 ➔ *Mediante un recorrido y usando el método clear en las listas de la tabla.*
- h. *¿Qué tipo de árboles se han implementado y por qué?*  
 ➔ *Se han usado los árboles AVL porque, para esta actividad, no sería adecuado el uso de tries (ya que su propósito está más basado en la búsqueda de palabras y aquí no estaríamos manejando prefijos) y tampoco el de árboles B (porque sería matar moscas a cañonazos, ya que estos se emplearían principalmente en unidades con almacenamiento enorme y en esta actividad lo estaríamos desaprovechando).*
- i. *¿Cómo es la definición del tipo árbol y del tipo nodo?*  
 ➔ *Hacemos declaración adelantada de la clase ArbolAVL y luego, al definir la clase Nodo, hacemos que sea friend class con la del árbol para que pueda acceder a la parte privada del nodo.*
- j. *¿Cómo referencian las carreteras al tipo lugar?*  
 ➔ *En vez de usar un puntero a lugar para el destino, hemos optado por usar un string.*
- k. *¿Cómo se listan los adyacentes de un nodo?*  
 ➔ *Hemos utilizado un recorrido inOrden del árbol (de esta manera se recorre alfabéticamente) y luego imprimimos cada elemento. Hemos necesitado usar un parámetro entero por referencia para saber qué elemento se imprimía el último, y para poder así modificar su salida y eliminar la coma del final.*
- l. *Si se han implementado árboles AVL, ¿cómo se hace el balanceo?*  
 ➔ *Mediante una función recursiva que va haciendo el balanceo en los nodos hijos de ser necesario y con otra que se encarga de hacer las rotaciones pertinentes cuando nota el desequilibrio en las alturas.*
- m. *¿Cómo se liberan los árboles?*  
 ➔ *Mediante el destructor, el cual básicamente elimina la raíz.*

- n.** ¿Se usan variables globales en el programa final?*
- ➔ Sí, una variable llamada mapa de la clase MapaLugares.*

### 3. Listado del código

Nombre Fichero	Descripción
Makefile	Fichero Makefile para la compilación del proyecto
MapaLugares.h	Fichero de cabecera para la clase MapaLugares
MapaLugares.cpp	Fichero de implementación para la clase MapaLugares
ArbolAVL3.h	Fichero de cabecera para la clase del árbol
ArbolAVL3.cpp	Fichero de implementación de la clase del árbol
TablaHash.h	Fichero de cabecera para la clase TablaHash
TablaHash.cpp	Fichero de implementación para la clase TablaHash
Intérprete.h	Fichero de cabecera para el intérprete de comandos
Intérprete.cpp	Fichero de implementación para el intérprete de comandos
Main.cpp	Fichero de implementación del main.

## 4. Informe de desarrollo

Normalmente el trabajo lo hemos realizado juntos mediante llamadas o en las propias sesiones de laboratorio. En caso de que uno estuviese muy ocupado con asignaturas que el otro no tuviese pues el otro avanzaba o investigaba por su cuenta.

A grandes rasgos, los ejercicios 001 y 002 nos costaron, más que nada, por la adaptación a la nueva actividad, pero aún así conseguimos realizarlos por nuestra cuenta sin tener que fijarnos en las soluciones subidas al aula virtual. El 003 también tuvo su dificultad pero, estando ya más “rodados”, se pudo llevar mejor. Con el 004 no tuvimos mucho problema, sin embargo, el 005 si nos llevó más tiempo y nos quedamos algo atascados principalmente porque tuvimos problemas a la hora de manejar las condiciones de recorrido de la lista y con los punteros. En el 201 la mayoría del tiempo se nos fue a la hora de implementar la función de dispersión, ya que empleamos mucho en investigar cómo hacerlo. Y por último, la 301 se llevó a cabo con relativa facilidad, pero la 302 sí que nos costó mucho más principalmente con fallos que nos iban surgiendo continuamente mientras adaptábamos las funciones a la clase de los árboles y tener que corregirlos.

Entrega	Análisis y diseño	Implementación y validación	Total
001	2h	1h	3h
002	2h	1h	3h
003	2h	1h	3h
004	3h	2h	5h
005	4h	3h	7h
201	6h	2h	8h
301	2h	1h	3h
302	3h	Unas 30h	Unas 33h aprox.
Total (h)	24h	41h	65h
Total (%)	37%	63%	100%



## 5. Conclusiones y valoraciones personales

Gonzalo: Bajo mi parecer, hemos sabido llevar bien la ejecución de esta práctica en términos generales y creo que hemos aprendido mucho de ellas. Esto nos ha hecho crecer y mejorar a la hora de gestionar los fallos y tratar de corregirlos. En definitiva, esta actividad ha sido un gran ejercicio de paciencia, en especial afrontando el ejercicio 302 que ha sido todo un quebradero de cabeza.

Jose: Creo que el trabajo ha resultado ser una buena manera de mejorar en el manejo de C++ y de trabajar en parejas. De hecho era muy habitual quedarse atascado sin saber solucionar ciertas cosas y en estas ocasiones tu compañero te podía aportar la idea que a ti no se te ocurría. Por eso creo que la actividad ha sido muy productiva pero también un poco desesperante ya que tuvimos algunos problemas para realizar el ejercicio 302.