|  |
| --- |
|  |

|  |
| --- |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |

2011-2012

**PRÁCTICA DE ALGORITMIA**

Índice

[INTRODUCCIÓN 2](#_Toc326496875)

[ESTRUCTURAS DE LA PRÁCTICA 3](#_Toc326496876)

[Lista Principal 3](#_Toc326496877)

[Árboles 4](#_Toc326496878)

[Principal 4](#_Toc326496879)

[ESTUDIO DE EFICIENCIA 5](#_Toc326496880)

[Prueba 1 “150.000 Muestras” 6](#_Toc326496881)

[Prueba 2 “100.000 Muestras” 7](#_Toc326496882)

[Prueba 3 “50.000 Muestras” 8](#_Toc326496883)

[Prueba 4 “10.000 Muestras” 9](#_Toc326496884)

[CONCLUSIÓN 10](#_Toc326496885)

# INTRODUCCIÓN

Para la realización de la práctica de algoritmia se ha pedido realizar una estructura con una lista y dos árboles utilizados como índices, en donde se podrán realizar búsquedas diferentes. También se requiere hacer un estudio de eficiencia sobre dos métodos de ordenación estudiados en clase (mezcla y burbuja) y uno de inserción ordenado para la recuperación de los datos.

También se ha utilizado una pila, para almacenar las posiciones que han sido borradas, y para el mejor entendimiento de cada estructura utilizada, se explicará brevemente cada una de ellas más adelante en el apartado que corresponda, describiendo que es y cómo funciona.

La práctica está compuesta por un Menú que contiene varias opciones y una adicional que es la 0 para SALIR, en algunas de estas opciones para realizar la búsqueda del alumno ya sea para eliminarlo, imprimirlo u otra cosa, se pedirá que inserte diferentes datos.

# ESTRUCTURAS DE LA PRÁCTICA

## Lista Principal



Consiste en una lista que contiene un vector de la estructura alumnos para guardar los datos necesarios para la práctica, al ser una estructura estática y ordenada por índices, para hacer la eliminación de alumnos se ha tomado la decisión de hacer borrados lógicos, para ello se ha añadido una pila que contiene las posiciones borradas.

La lista.- Es una estructura lineal en la que tiene un único antecesor y sucesor (en el caso de la práctica se ha implementado la versión estática

La pila.- Es una estructura lineal que sigue la política LIFO (Last Input, First Output) último entrar y primero en salir, que es similar a una pila de platos. En el caso de la práctica se ha implementado una pila estática y las operaciones básicas que realiza son: Crear un pila vacía, devolver verdadero o falso para saber si la pila está vacía, devolver el elemento que se encuentra en la cima o TOPE,PUSH añade un elemento y POP lo elimina de la pila.

## Árboles



Son estructuras para generar el índice, cada árbol contiene varios “Nodo\_arbol” y cada uno de estos nodos contienen una “listaInt” donde se almacenan los IDs de los alumnos de la lista principal.

Los árboles.- Son Estructuras no lineales en la que los nodos no tienen una relación uno a uno, sino que cada nodo puede tener de 0 a N sucesores y antecesores. Cada nodo tiene un único predecesor, o también 2 o más sucesores, excepto el nodo raíz que no tiene predecesor.  
Un árbol cuyos nodos tienen “N” elementos se les llama “árbol binario” con dos sucesores, que es lo que se ha implementado en la práctica. Los recorridos que realiza un árbol binario son 3: orden previo, simétrico y posterior y las operaciones básicas de un árbol binario o métodos a implementar son: insertar, pertenece, encontrar, padre de y borrar.

## Principal



Para reagrupar todas las estructuras y sobre todo facilitar el paso de parámetros entre funciones se ha creado Principal, que contiene 2 árboles para hacer de índices uno por edad y otro por orden alfabético y una lista de valores.

# ESTUDIO DE EFICIENCIA

Para el estudio de eficiencia hemos recuperado los datos de 3 maneras diferentes, hemos insertado alumnos con valores de nombre, edad, DNI, y nota generados aleatoriamente, los nombre no son reales son cadenas de texto generadas al azar, igual que los DNI.

La ordenación se realiza primero por la edad del alumno y seguido por el nombre.

1º Estrategia: Insertar de manera ordenada y después recuperar los datos.

2º Estrategia: Insertar de manera aleatoria y luego, antes de recuperar los datos ordenarlo por el método de burbuja, coste teórico 2^n.

3º Estrategia: Insertar de manera aleatoria y luego, antes de recuperar los datos ordenarlo por el método de mezcla directa, coste teórico log2n.

**MÉTODO DE** BURBUJA (**bubble sort).-** Es un algoritmo de ordenamiento que funciona revisando cada elemento de la lista que va a ser ordenada con el siguiente, intercambiándolos de posición si están en el orden equivocado. La idea es imaginar que los elementos suben como burbujas a las primeras posiciones.

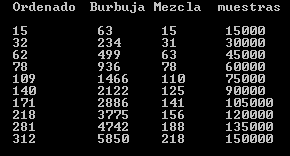
**BUSCAR METODO DE MEZCLA.-** Este algoritmo consiste básicamente en dividir en partes iguales la lista de números y luego mezclarlos comparándolos, dejándolos ordenados.

Si se piensa en este algoritmo recursivamente, podemos imaginar que va a dividir la lista hasta tener un elemento en cada una, luego lo compara con el que está a lado y según corresponda, lo situará en donde se corresponda.

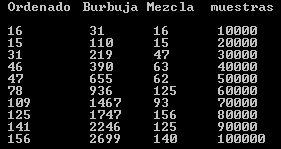
**INSERCIÓN ORDENADA.-** Consiste en ordenar los elementos del array según se van insertando, y así se evitará tener que ordenar después.

Para la realización de las pruebas hemos obviado imprimir el resultado, por tratarse de métodos ya probados que funcionan correctamente, ya que el mayor tiempo de las pruebas el programa se lo pasaría imprimiendo por pantalla.

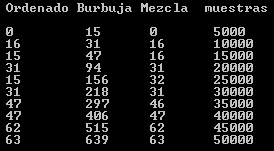
## Prueba 1 “150.000 Muestras”



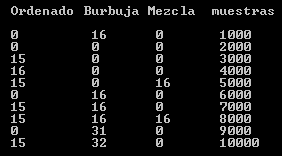
## Prueba 2 “100.000 Muestras”



## Prueba 3 “50.000 Muestras”



## Prueba 4 “10.000 Muestras”



# CONCLUSIÓN

De las tres estrategias utilizadas, como se muestra en las gráficas la primera en descartar sería la ordenación por burbuja “**estrategia 2**”, siendo ésta muy costosa para realizar búsquedas cuando hay muchas muestras , luego nos quedarían 2 estrategias, claramente cuando existen muchas muestras por ejemplo unas 150.000 el método de ordenación por mezcla es más rápido que insertar ordenado, y cuanto mayor sea el numero de muestras esta diferencia se va incrementando, pero cuando existen pocas muestras por ejemplo unas 1000, el método insertar ordenado es eficiente puesto que ya podríamos recuperar los datos siempre ordenados. Dependiendo del uso que se le vaya a dar a la aplicación se debe utilizar la **“estrategia 3**” ordenar por mezcla cuando exista un número elevado de muestras, o insertar ordenado **“estrategia 1**” cuando se requiera un número bajo de muestras.