

GUIÓN DE LA PRÁCTICA 2

OBJETIVO:

- Algoritmos de ordenación y su estudio comparativo

1. TRES ALGORITMOS DE ORDENACIÓN “MALOS”

Se proporcionan ficheros Java para trabajar con los algoritmos siguientes vistos en teoría: *Inserción Directa*, *Selección* y *Burbuja*. **El código específico de cada uno de los métodos debe ser incluido por el alumno en los ficheros.**

Son algoritmos *malos* porque son cuadráticos ($O(n^2)$) en sus casos mejor, peor y medio (excepto el de *Inserción* que en el caso mejor es lineal $O(n)$).

Para probar que las implementaciones funcionan correctamente se proporciona una clase **OrdenacionPruebas** que recibe un argumento n que corresponde con el número de elementos a ordenar. Intenta comprender con detalle el funcionamiento de todos los algoritmos analizando los tiempos para diferentes tamaños del problema.

Para medir los tiempos ya se ha creado otra clase denominada **OrdenacionTiempos** en la que se incluye el código necesario para medir los tiempos respectivos en el caso ordenación sobre un vector ordenado, vector inversamente ordenado y vector aleatorio.

2. EL ALGORITMO DE ORDENACIÓN MEJOR: “QUICKSORT”

En este caso considera el algoritmo de ordenación Rápido o Quicksort. Estúdialo en detalle, ya que es un algoritmo más elaborado que los otros. Completa el código cuando sea necesario y analiza los tiempos para diferentes tamaños del problema, concluyendo si los tiempos obtenidos son los esperados desde el punto de vista de la complejidad en cada caso.

- **RapidoCentral.java** → En este caso utilizamos como pivote el elemento central en cada iteración (ya está disponible el código completo).
- **RapidoFatal.java** → Utiliza una selección de pivote mala, revisa exactamente lo que hace, puedes compararlo con la clase anterior para ver las diferencias (Normalmente es una mala idea) **Describe en el documento en que consiste este método de selección, cuándo funciona mal y cuando no y que efecto tiene en el tiempo de ejecución.**
- **RapidoMedianaTres.java** → En este caso se elige la mediana a tres como pivote utilizando para ello el primer, el último y el elemento central en cada iteración. Además de completar el código.

3. TRABAJO PEDIDO

Se le pide programar lo necesario para rellenar 4 tablas de tiempos (para Inserción, Selección, Burbuja y Quicksort con pivote utilizando mediana a tres). Cada tabla será así:

Tiempos MÉTODO x:

<i>N</i>	<i>t ordenado</i>	<i>t inverso</i>	<i>t aleatorio</i>
10 000
20 000
40 000
80 000
160 000
320 000
640 000
1 280 000

Hasta “heap overflow” o casque			

Las clases que programe las incluirá dentro del paquete `alg<dnipropio>.p2`. Si utiliza Eclipse llamar al proyecto `prac02_Ordena<UOpropio>`

Se entregará, por separado, un documento con las 4 tablas cubiertas, y los ficheros fuente de las clases que haya tenido que programar o modificar junto con una explicación coherente de los resultados.

Esto se realizará a través de la tarea que se habilitará en el campus virtual. El plazo límite es un día antes de la próxima sesión de prácticas.