**RENDIMIENTO Y TIEMPO DE EJECUCION DE ALGORITMOS**

* **ALGORITMO DE ORDENACION QUICKSHORT:** Este algoritmo se basa en determinar uno de los valores de la lista como pivote, y a partir de él definir los demás valores como mayore o menores. Una vez haber hecho esa división, se aplica de forma recursiva en ambos la misma técnica, hasta ordenar toda la lista.

Para definir su notación O Grande, considerando su caso promedio, se puede decir que tiene dos factores que la definen:

* La altura de pila de llamadas: Cuantas veces tiene que iterar sobre la lista par poder ordenar todos los números. En este caso y considerando que el pivote es el valor medio (éste nos definirá el promedio de O Grande), es decir que se dividirán los demás valores en parte iguales entre mayores y menores, el tiempo que necesitará para realizar todas las iteraciones necesarias será de O(log n).
* Tiempo que necesita cada nivel (iteración): Pero en cada una de esas iteraciones deberá recorrer todos los valores de la lista, por lo que en cada iteración necesitará un tiempo de O(n)

Estos dos factores darán a Quickshort un tiempo de O(n log n).

* **BÚSQUEDA SECUENCIAL:** Sabiendo que la notación O Grande establece el tiempo de ejecución del caso peor, el de la búsqueda secuencial será de O(n), ya que tiene que recorrer todos los elementos de la lista que están en posición anterior al caso buscado. Aunque realmente es dependiente de la posición del elemento en la lista, con la notación O Grande no se determina el tiempo real necesario, sino que la relación que hay entre número de elementos y progresión de la búsqueda. En este caso se define como O(n), porque avanza de elemento en elemento hasta encontrar el buscado.

Como se puede comprobar en el programa de esta tarea, necesita tantas iteraciones como la posición que ocupa el elemento buscado en la lista.

* **BUSQUEDA BINARIA:** En el caso de la búsqueda binaria en cambio, el número de iteraciones que necesitano es proporcional al número de elementos que hay en la lista, ya que busca el valor medio y determina si el elemento está en la mitad menor o en la mitad mayor, en cada una de sus iteraciones. Esto quiere decir que si n es el número de elementos y x el número de iteraciones:

n = 2x

Por lo que: x = log n

Como el tiempo que toma el algoritmo es dependiente del número de iteraciones necesarias, se puede decir que este algoritmo tomará un tiempo de O(log n).

Estos algoritmos en listas de pocos elementos no tienen un efecto apreciable en tiempo, aunque se puedan intuir considerando el número de iteraciones que realiza cada uno. Estos algoritmos tienen un efecto más muy importante en listas de muchos valores donde los tiempos de ejecución de la búsqueda secuencia será lineal y logarítmica en el caso de la secuencial.

