**TAREA\_45**

búsqueda-secuencial | búsqueda-binaria | notación-Big-O

**Algoritmo de búsqueda secuencial**: Se realiza la búsqueda de un elemento en un conjunto de elementos comparando el elemento a buscar con cada uno de los elementos del conjunto para determinar si es igual. En este tipo de búsqueda es necesario recorrer todos los elementos del conjunto para determinar si el elemento a buscar está presente o no en el conjunto.

**Algoritmo de búsqueda binaria**: En este tipo de algoritmos se utiliza la técnica divide y vencerás, se busca el elemento central del conjunto donde se va a realizar la búsqueda (pivote) y en base a él, el conjunto se divide en dos subconjuntos (parte inferior: desde el inicio al pivote, y parte superior: desde el pivote al final), se compara el elemento de búsqueda con el pivote y según sea mayor o menor, nos quedamos con la parte superior o inferior del conjunto. Esta operación se repite de manera iterativa hasta que el conjunto de búsqueda sea indivisible. Es imprescindible que el conjunto donde se realiza la búsqueda esté ordenado.

**Notación-Big-O**: Se trata de una herramienta para saber como de eficiente es un algoritmo. Para determinar esta eficiencia necesitamos saber como de complejo es dicho algoritmo, es decir, como se comporta (en términos de tiempo de ejecución y espacio de memoria requerido) a medida que el tamaño de los datos de entrada crece. Puede tener diferentes órdenes de complejidad:

**Orden constante**: aunque el tamaño del input aumente el tiempo de ejecución no aumenta. **O(1).**

**Orden lineal**: en este caso el tiempo de ejecución aumenta de manera lineal con respecto al tamaño del input -> **O(n).**

**Orden logarítmico**: la relación entre el crecimiento del número de datos y el del tiempo de ejecución es de tipo logarítmico ->**O(log n).**

**Orden exponencial**: el tiempo de ejecución aumenta de manera exponencial con respecto al aumento del tamaño del input-> **2n**

**Orden cuadrático**: el tiempo de complejidad aumenta de manera cuadrática con respecto al tamaño de los datos de entrada -> **O(n2)**, este tipo de algoritmos no son muy eficientes cuando el tamaño de los datos es muy grande.

Tablas con notación Big-O asociada a diferentes estructuras de datos y diferentes algoritmos de búsqueda aquí: <https://www.bigocheatsheet.com/>