**EL ORDEN DE EFICIENCIA DE LOS ALGORITMOS OBJETIVO**

El analisis de algoritmos una herramienta para hacer la evaluación del diseño de un algoritmo, permite establecer la calidad de un programa y compararlo con otros que puedan resolver el mismo problema, sin necesidad de desarrollarlos.

**El análisis de algoritmos se basa en:**

* El análisis de las características estructurales del algoritmo que respalda el programa.
* La cantidad de memoria que utiliza para resolver un problema.
* La evaluación del diseño de las estructuras de datos del programa, midiendo la eficiencia de los algoritmos para resolver el problema planteado.

**Eficiencia de los algoritmos**

**O(1)** -quiere decir que da igual cuántos elementos haya en la lista, la operación siempre tarda lo mismo.

Es **constante** la operación no depende del tamaño de los datos. Es el caso ideal, pero a la vez probablemente el menos frecuente.

**O(log(n)) –** es **logarítmica** y por regla general se asocia con algoritmos que "trocean" el problema para abordarlo

**O(n)-** el tiempo necesario para ejecutar la función es función directa y lineal del número de elementos que le pasemos.

Es **lineal** y el tiempo de ejecución es directamente proporcional al tamaño de los datos. Crece en una línea recta.

**O(n\*log(n)) –** en este caso se trata de funciones similares a las anteriores, pero que rompen el problema en varios trozos por cada elemento, volviendo a recomponer información tras la ejecución de cada "trozo".

**O(n2) –** Es **cuadrática** y es típico de algoritmos que necesitan realizar una iteración por todos los elementos en cada uno de los elementos a procesar.

**O(2n) –** Son algoritmos muy raros pues en condiciones normales no debería ser necesario hacer algo así.

Es **exponencial y s**e trata de funciones que duplican su complejidad con cada elemento añadido al procesamiento.

**O(n!) - explosión combinatoria.** Un algoritmo que siga esta complejidad es un algoritmo totalmente fallido. Una explosión combinatoria se dispara de tal manera que cuando el conjunto crece un poco, lo normal es que se considere computacionalmente inviable. Solo se suele dar en algoritmos que tratan de resolver algo por la mera fuerza bruta. No deberías verlo nunca en un software "real".

Gráfico

Descripción generada automáticamente

|  |  |
| --- | --- |
| **NOTACION BIG O** | **ALGORITMO DE ELEMPLO** |
| O (log n) | Búsqueda binaria |
| O(n) | Búsqueda simple |
| O (n\*log n) | Ordenación rápida (quicksort) |
| O(n2) | Ordenación por selección |
| O(n!) | Vendedor viajero |

**RESUMEN**

* Constante: O (1)
* Lineal: O(n)
* Logarítmica: O (n log n)
* Cuadrática: O(n^2)
* Exponencial: O(2^n)
* Factorial: O(n!)