Apuntes/Notación Big-Oh

Decimos que T(n) = O(f(n)) (que el tiempo de ejecución de un algoritmo es de tal orden) si existen dos constantes, c > 0 y n°0, tales que se cumpla **T(n) ≤ c*f(n)** *para todo n ≥ n°0*

f(n) representa una cota superior de T(n)

Lo que nos da esta notación es nada mas una cota **asintótica superior** con respeto a la función de nuestro algoritmo, pero no una **asintóticamente ajustada**, por lo que en algunos casos podemos hacer declaraciones muy imprecisas sobre el orden del algoritmo

Por ejemplo, es absolutamente correcto decir que la búsqueda binaria se ejecuta en un tiempo O(n), ya que el *tiempo de ejecución del algoritmo*, en función de una entrada de datos, no crece mas rápido que una constante multiplicada por n

Reglas

Si tenemos un T1(n) = O(f(n)) y un T2(n) = O(g(n)), entonces...

1. T1(n) + T2(n) = Max(O(f(n)), O(g(n)))
2. T1(n) * T2(n) = O(f(n) * g(n))
3. Si T(n) es un polinomio de grado k → T(n) = O(n^k)
4. T(n) = log^k(n) → O(n) para cualquier k (n crece mas rápido que cualquier logarítmo)
5. T(n) = cte → O(1)
6. T(n) = cte * f(n) → T(n) = O(f(n))