## **Semana 4 – Control de Lectura**

### Notación Asintótica

### 3.1 Introducción

* Este capítulo tiene como objetivo evaluar la eficiencia de los algoritmos.
* Se enfoca en cuantificar el uso de recursos (tiempo, memoria) que consume un algoritmo dependiendo del tamaño de entrada.
* La notación asintótica permite estudiar el comportamiento del algoritmo cuando el tamaño del problema es muy grande, sin depender de detalles como:
  + Lenguaje de programación
  + Implementación específica
  + Velocidad del hardware

Idea clave: No se busca medir tiempos exactos, sino entender cómo crece el uso de recursos cuando el problema escala.

### 3.2 Una notación para “el orden de…”

* Se define una función matemática:  
  f: ℕ → ℝ⁺, donde f(n) representa el uso de recursos en función del tamaño n.
* Se introduce la notación Big O: O(g(n)), que significa que f(n) está acotada superiormente por una constante multiplicada por g(n) para valores grandes de n.
* Esta notación se centra en el comportamiento a gran escala, ignorando detalles pequeños o casos particulares.
* Se presenta el concepto de función dominante: si una función crece más rápido que otra, domina su comportamiento asintótico.

### Ejemplo y propiedades

* Ejemplo:  
  f(n) = 27n³ + 111n² + 112  
  Al aplicar notación asintótica, se simplifica a:  
  f(n) ∈ O(n³)  
  Se eliminan constantes y términos de menor orden.
* Propiedades importantes:
  + Si f(n) ∈ O(g(n)) y g(n) ∈ O(h(n)), entonces f(n) ∈ O(h(n)) (transitividad).
  + Combinación de funciones:
    - Suma: se conserva el término de mayor crecimiento.
    - Producto: se multiplican los órdenes.

### Importancia práctica

* Permite comparar algoritmos de manera objetiva y estandarizada, sin importar la plataforma.
* Es una herramienta para predecir eficiencia en escenarios grandes.
* Aunque no proporciona valores exactos de tiempo, permite anticipar el costo computacional conforme el problema escala.

### Notaciones comunes y funciones frecuentes

* Constantes: 1, 10
* Logarítmicas: log n, log₂ n
* Polinómicas: n, n², n³
* Exponenciales: 2ⁿ, eⁿ
* Factoriales: n!

También se introducen notaciones menos comunes:

* o(g(n)): crecimiento estrictamente menor que g(n)
* ω(g(n)): crecimiento estrictamente mayor que g(n)

