① LECCION 2 - EFICIENCIA. NOTACIÓN O-GRANDE

- **EFICIENCIA**: Qué recursos gasta un algoritmo
  - → Podemos catalogar un algoritmo
- **RECURSOS**: MEMORIA Y TIEMPO de EJECUCIÓN

- EJEMPLO: Asignacion de
  - 50 trabajadores a 50 trabajos
    - ¿Mejor asignacion?
    - Búsqueda Masiva — Mira todas las posibilidades

- $50! \simeq 10^{64}$
- Ordenador:
  - Evalúa 1billon de posibilidad/s
  - Empero 15000 millones de años
  - Aún estará procesando

- No basta con la mejora de hardware

- **ALGORITMO**: Conjunto de pasos o sentencias bien ordenadas que resuelven un problema.
- **IMPLEMENTACIÓN**: Traduccion de las sentencias a un determinado lenguaje de programacion
- **PRINCIPIO DE INVARIANZA**: dos implementaciones de un mismo algoritmo sólo se diferencia en una constante multiplicativa.

- ¿Qué algoritmo escogemos?
  - El más eficiente en recursos.
- EJEMPLO: Para un problema
  - Dos algoritmos A_1 y A_2
  - Eficiencia de A_1: $T_1(n) = 10^{-4} \times 2^n s$
  - Eficiencia de A_2: $T_2(n) = 10^{-2} \times n^3 s$
  - n es el nº de datos de entrada al algoritmo.

n	10	20	30	38
$T_1(n)$	0.1s	2m	>1dia	1 año
$T_2(n)$	10s	80s=1,3m	270s	549s

- **ANÁLISIS ASINTÓTICO**

Análisis de la eficiencia de un algoritmo cuando el nº de datos de entrada tiende a valores muy grandes ($n \to \infty$)

- En la eficiencia de un algoritmo la estructura de datos que escojamos también es relevante.

FUNCION DE EFICIENCIA:-

Se define sobre el nº de datos de entrada del algoritmo $f(n)$ (n: nº de datos de entrada o talla) y devuelve un valor real mayor o igual que 0 midiendo la eficiencia (tiempo o memoria)

$$f: \mathbb{N} \longrightarrow \mathbb{R}_0^+ \qquad n \longrightarrow f(n)$$

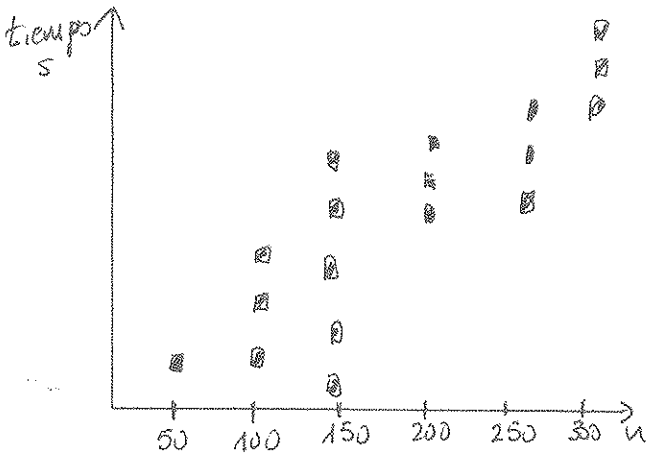
$f(n)$ → tiempo de ejecucion (s)
$f(n)$ → espacio (bytes)

n: tamaño del problema o talla

LECCION 2: continuación

- ¿Como calcular la eficiencia?
  - Teórica — Estudio Asintótico
  - Empírica - Experimentalmente

. EMPÍRICA



- Ejecutamos el algoritmo una vez implementado y obtenemos el tiempo:
  - Para un mismo n (nº de datos de entrada) lo ejecutamos varias veces
  - Lo ejecutamos para diferentes n (distintas tallas)

. En Ctt: como obtener el tiempo

```
#include <ctime>
  ;
  ;
time_t tantes, tdespues;
time (&tantes)
//sentencias
time (&tdespues)
cout << difftime (tdespues, tantes)
```

. PROBLEMAS de la E. EMPÍRICA

1) Los datos que hayamos escogido no son representativos de las características del problema
2) Depende del hardware y librería usada
3) Para poder comparar dos algoritmos debemos ejecutarlos en las mismas condiciones.

. E. TEORICA — Análisis asintótico

CARACTERÍSTICAS

- Independiente del hardware y software
- Representa a todas las entradas
- Usa una descripción a alto nivel del programa.

- Orden de Eficiencia, T(n)

Un algoritmo tiene un orden de eficiencia T(n), si ∃ una implementación del algoritmo y su tiempo de ejecución f(n) está acotado superiormente por c·T(n) siendo c una constante c ≥ 1 y n el tamaño del problema.

$$f(n) \leq c \cdot T(n)$$

- Ordenes de Eficiencia más comunes

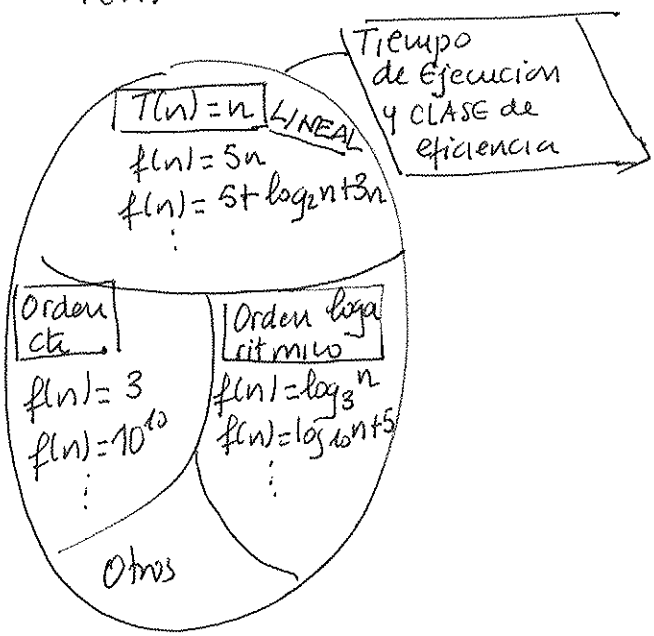
- Orden lineal — $T(n) = n$
- Orden cte — $T(n) = a$ a es una cte
- Orden logarítmico — $T(n) = \log(n)$
- Orden cuadrático — $T(n) = n^2$
- Orden exponencial — $T(n) = c^n$ c es una ct

LECCIÓN 2: continuación

- E. TEÓRICA de un ALGORITMO

1) Descubrir la función de eficiencia $f(n)$ del algoritmo

2) Dada $f(n)$ deducir cual es su orden de eficiencia $T(n)$



EJEMPLO 2: Orden parcial

$$f(n) = \begin{cases} n^2 & \text{si } n \text{ es par} \\ n^4 & \text{si } n \text{ es impar} \end{cases}$$

$g(n) = n^3$

¿Quién es mayor $f(n)$ o $g(n)$?

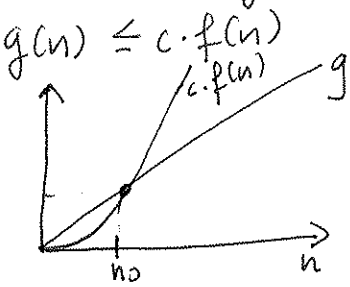
NOTACIÓN ASINTÓTICA

- $5n^2 + 3n + 10 \simeq n^2$

DEFINICIÓN O-grande (tiempo de ejecución en el peor caso) eg: Búsqueda secuencial.

Diremos que un algoritmo con tiempo de ejecución $g(n) \in O(f(n))$ si $\exists\, c \in \mathbb{R}_0^+$ y $n_0 \in \mathbb{N}$ tq $\forall n \geq n_0$

$g(n) \leq c \cdot f(n)$



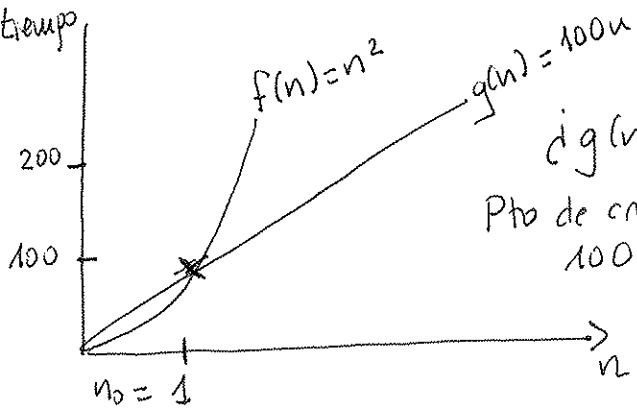
Ejemplos

- $n^2 + 5n \in O(n^2)$ p.e $c = 2$ $n_0 = 5$
- $3n^3 + 2n^2 \in O(n^3)$ $c = 4$ $n_0 = 2$
- $3^n \notin O(2^n)$

COMPARACIÓN DE EFICIENCIA ENTRE DOS ALGORITMOS

- CARACTERÍSTICAS
  - Debe $\exists$ independencia de lo que ocurra en un nº finito de valores para n (tamaño del problema)
  - Independiente de la función que representa al orden de eficiencia.

- COMPARAR = los ordenes de eficiencia
  = comparar los perfiles de crecimiento



¿$g(n) \leq f(n)$? $\Rightarrow$ $g(n) \leq c \cdot f(n)$ $n \geq n_0$ $n_0 = 1$ $c = 100$

Pto de cruce $100n = n^2 \Rightarrow$ $n^2 - 100n = 0$

$n = 100$