Algoritmos y Estructuras de Datos

PARTE II: ALGORÍTMICA (o ALGORITMIA)

Tema 0. Introducción

0.1. Definición y propiedades.

0.2. Análisis de algoritmos.

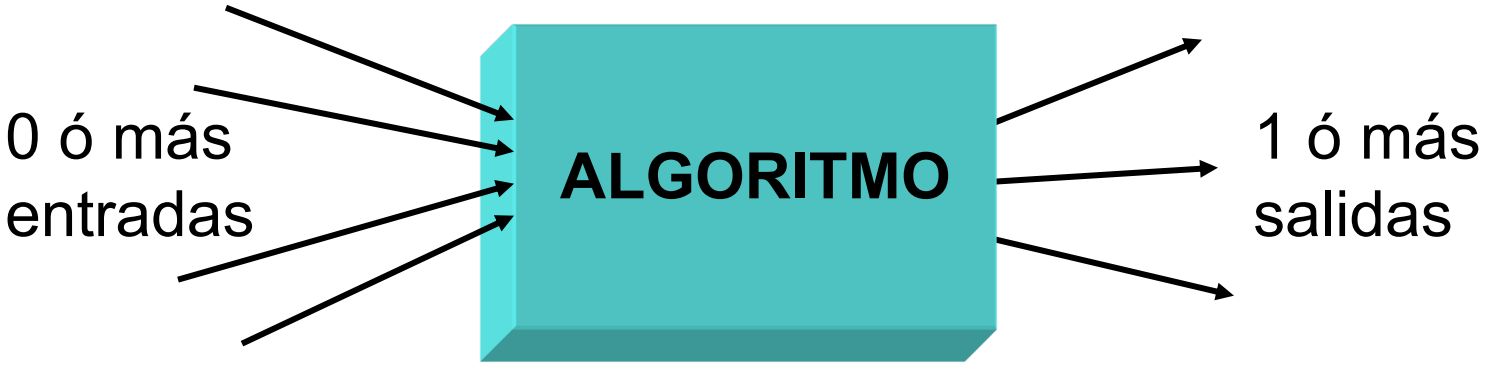
0.3. Diseño de algoritmos.

0.1. Definición y propiedades.

- **Algoritmo:**

  Conjunto de reglas para resolver problema

- Resolver un problema puede verse así:



0.1. Definición y propiedades.

- **Propiedades:**
  - **Definibilidad:** El conjunto debe estar bien definido, sin dejar dudas en su interpretación.
  - **Finitud:** Debe tener un número finito de pasos que se ejecuten en un tiempo finito.
  - **Determinismo:** Algoritmos…
    - **Deterministas:** Para mismos datos de entrada, siempre devuelve mismos datos de salida.
    - **NO deterministas:** Para mismos datos de entrada, pueden devolver diferentes de salida.

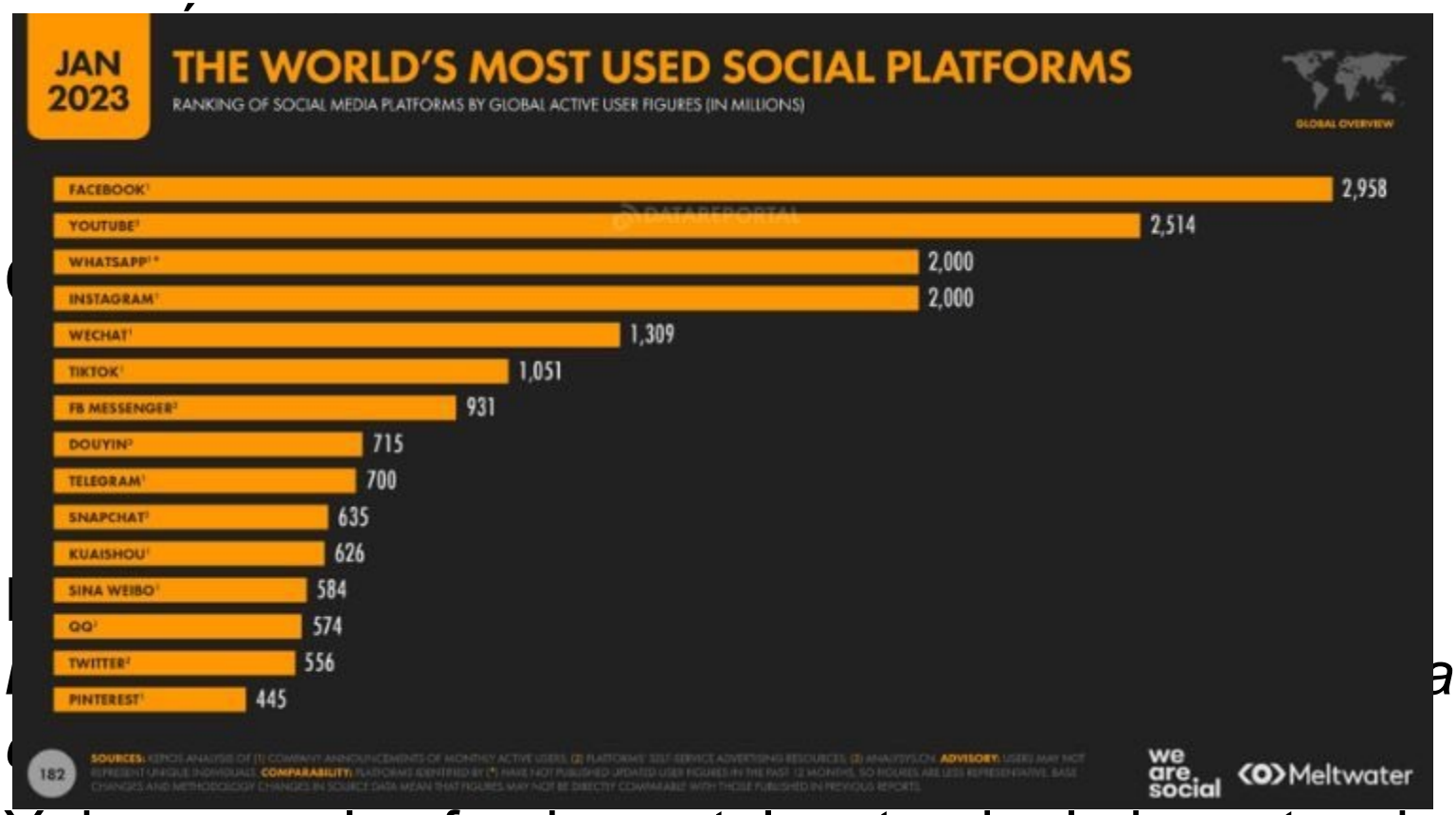
TIP: Estudiaremos los deterministas, pero los no deterministas pueden ser muy interesantes en problemas complejos.

0.1. Definición y propiedades.

- **ALGORITMIA:** Ciencia que estudia técnicas para:
  - ANÁLISIS: **Medir** la eficacia de los algoritmos
  - DISEÑO: **Construir** algoritmos eficientes
- **Objetivo**:
  - Dado un **problema** concreto
  - Encontrar la **mejor forma** de **resolverlo**.
- Formalmente: *Aprender a **analizar**, **comprender** y **resolver** amplia variedad de **problemas** de programación con soluciones **eficientes** y de **calidad**.*
- Y de paso, algo fundamental: entender la importancia de la eficiencia ante problemas de alta complejidad...

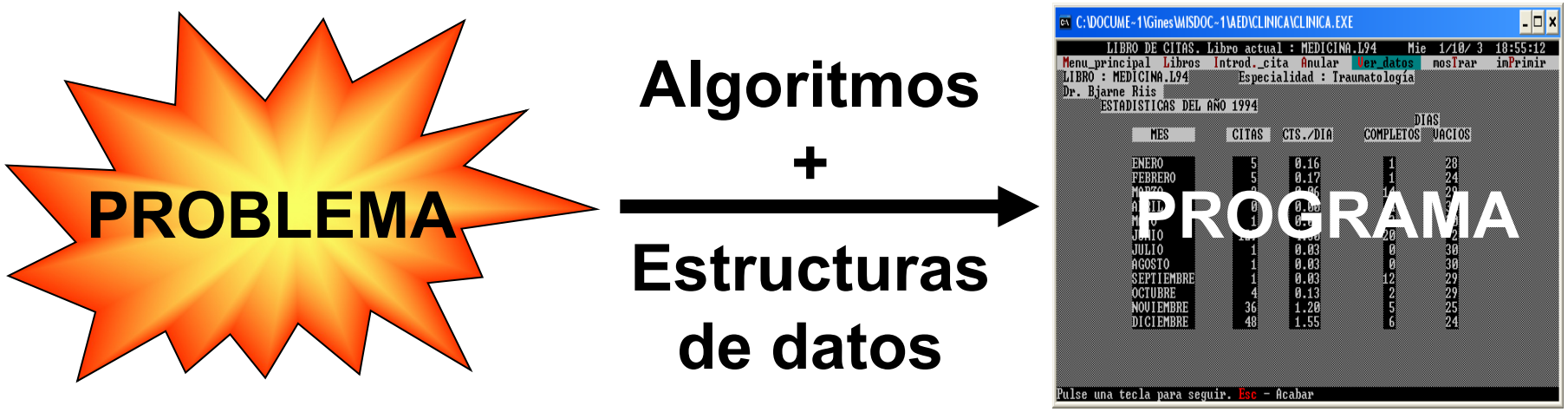
0.1. Definición y propiedades.

- **ALGORITMIA:** Ciencia que estudia técnicas para:
  -
  -
  - *ación*
- Y de paso, algo fundamental: entender la importancia de la eficiencia ante problemas de alta complejidad...



0.1. Definición y propiedades.

Los **algoritmos NO** son el único componente en la resolución de un problema de programación...

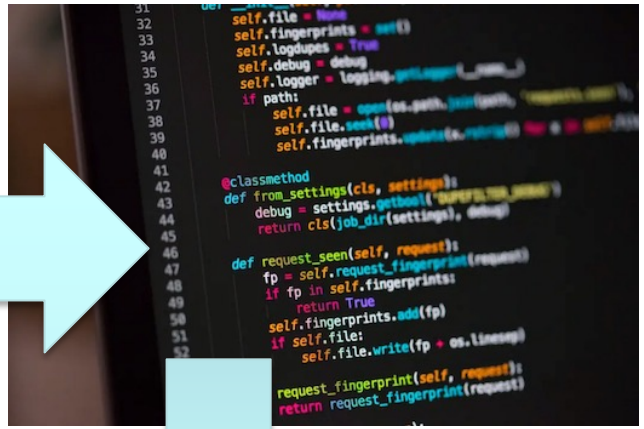


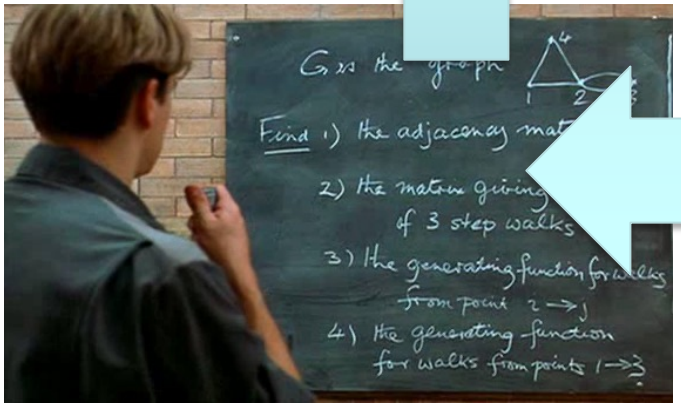
Algoritmos + Estructuras de Datos = Programas

- **Estructura de datos:** Parte estática, almacenada.
- **Algoritmo:** Parte dinámica, manipulador.

0.1. Definición y propiedades.

Vale ya lo he pillado, ¡¡ahora a programar!!





0.1. Definición y propiedades.

MÉTODO CIENTÍFICO		En INFORMÁTICA
1. Observación	⟷	1. **Análisis** del problema
2. Hipótesis	⟷	2. **Diseño** programa (alg. y estr.)
3. Experimentación	⟷	3. **Implementación** (programación)
4. Verificación	⟷	4. **Verificación** y pruebas

Otra idea sobre cómo resolver un problema:

- **Refinamiento por pasos sucesivos:**
  - Escribir la estructura de la solución en pseudocódigo, de manera muy genérica.
  - Especificar pasos cada vez con más detalle/precisión.
  - Repetir refinamiento hasta llegar a una implementación.

0.2. Análisis de algoritmos.

ALGORITMIA = ANÁLISIS + DISEÑO

- **0.2 Análisis de algoritmos:** Estudio de los recursos que necesita la ejecución de un algoritmo.

  >>No confundir con análisis de un problema.

- **0.3 Diseño de algoritmos:** Técnicas generales para la construcción de algoritmos.

  Por ejemplo, *divide y vencerás*:

  dado un problema, divídelo, resuelve los subproblemas y luego junta las soluciones.

0.2. Análisis de algoritmos.

- **Análisis de algoritmos.** Normalmente estamos interesados en el estudio del tiempo de ejecución.

- Dado un algoritmo, usaremos las siguientes notaciones:
  - t(..): Tiempo de ejecución del algoritmo.
  - O(..): Orden de complejidad.
  - o(..): O pequeña del tiempo de ejecución.
  - Ω(..): Cota inferior de complejidad.
  - Θ(..): Orden exacto de complejidad.

0.2. Análisis de algoritmos.

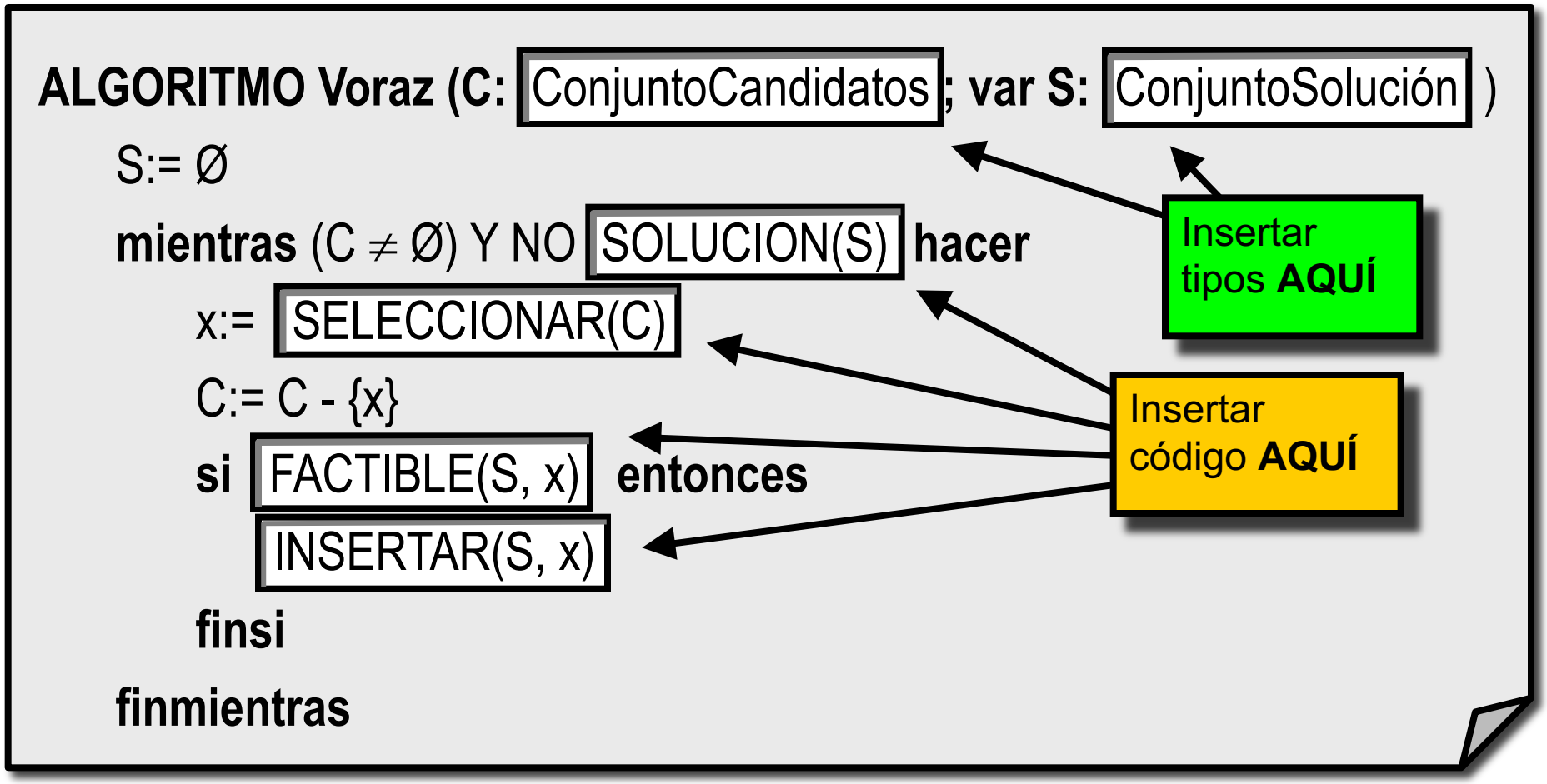
- **Ejemplo.** Analizar el tiempo de ejecución y el orden de complejidad del siguiente algoritmo.

```
Hanoi (N, A, B, C: integer)
        if  N=1 then
              Mover (A, C)
        else begin
              Hanoi (N-1, A, C, B)
              Mover (A, C)
              Hanoi (N-1, B, A, C)
        end
```

- Mecanismos:
  - Conteo de instrucciones.
  - Uso de ecuaciones de recurrencia.
  - Medida del trabajo total realizado.

0.3. Diseño de algoritmos.

- Técnicas generales, aplicables a muchas situaciones.
- **Esquemas algorítmicos.** Ejemplo:



0.3. Diseño de algoritmos.

- **Técnicas de diseño** de algoritmos:
  (T2) Divide y vencerás
  (T3) Algoritmos voraces
  (T4) Programación dinámica
  (T5) Backtracking

- **Dado un problema:** seleccionar la técnica, seguir el proceso/esquema algorítmico, obtener el algoritmo y comprobarlo.
- **Recordar:** No empezar tecleando código a lo loco >> análisis/diseño/implementar/verificar.