




Tema 23



Diseño de Algoritmos

Técnicas Descriptivas

1. Introducción al diseño algorítmico

 Un algoritmo es una secuencia **ordenada y finita de pasos** para resolver un problema.

Propiedades clave:

-  Precisión
-  Determinismo
-  Efectividad
-  Finitud

 Aplicación: IA, videojuegos, Big Data, ciberseguridad, sistemas embebidos...

2. Construcción y estructura de algoritmos

Elementos básicos:

- Instrucciones: asignación, entrada/salida, operadores
- Control de flujo: secuencia, condicionales, bucles
- Modularidad: uso de funciones o procedimientos
- Validación de datos

Buenas prácticas:

- Evitar redundancias
- Comentar el código
- Dividir en pasos o módulos lógicos

3. Representación de algoritmos

3.1 Pseudocódigo

- Lenguaje intermedio entre lenguaje natural y programación
- Legible y sin sintaxis rígida

3.2 Diagramas de flujo

- Representación visual del algoritmo
- Elementos: decisiones, bucles, operaciones
- Muy útil en la fase inicial

3.3 Nassi-Shneiderman

- Alternativa estructurada al diagrama de flujo
- Favorece el diseño modular

3.4 Tablas de decisión

- Útiles cuando hay muchas condiciones
- Representación compacta de reglas lógicas

4. Herramientas de apoyo al diseño

Entornos visuales:

- Scratch, Blockly, App Inventor, MakeCode

IA / LLMs:

- ChatGPT, GitHub Copilot: generan, explican, depuran

Prototipado interactivo:

- Figma, Adobe XD para simular flujos algorítmicos en UI/UX

5. Metodología descriptiva de diseño

1. **Análisis:** entradas, salidas, restricciones
2. **Estructuración:** secuencia, bucles, condicionales
3. **Modularidad (Top-Down):** divide en funciones
4. **Estructuras de datos:** listas, arrays, pilas, árboles...
5. **Pruebas:**
 - Caja negra: entradas/salidas
 - Caja blanca: lógica interna

6. Representación de estrategias algorítmicas

 Patrón → Representación

- Divide y vencerás → recursividad estructurada
- Dinámica → tablas de subresultados
- Voraz → decisiones locales óptimas
- Backtracking → árbol de decisiones
- Heurísticos/evolutivos → esquemas de evolución

 Elección de estructura de datos = claridad y eficiencia

7. Eficiencia algorítmica

Notación Big-O

- $O(1)$: acceso directo
- $O(n)$: recorrido lineal
- $O(n \log n)$: ordenaciones rápidas
- $O(n^2)$: burbuja, fuerza bruta
- $O(n!)$: permutaciones, backtracking

 Considera:

- Peor / mejor / promedio
- Complejidad temporal vs espacial

8. Ética y limitaciones

⚠ Sesgos y errores:

- Datos mal etiquetados → decisiones injustas
- Falta de explicabilidad → pérdida de confianza

🎯 El programador es responsable de los efectos de los algoritmos en salud, justicia, finanzas...

9. Conclusión

💡 Diseñar algoritmos es:

- Pensar con **lógica estructurada**
- Resolver problemas de forma **eficiente y modular**
- Enlazar problema → lógica → código → experiencia

🚀 Tendencias actuales (IA, visual, cuántica) están redefiniendo cómo pensamos los algoritmos.