Capítulo 1 PRELIMINARES

Nombre: Jean Carlos Iñiguez

* **¿Qué es un algoritmo?**

Un **algoritmo** es un conjunto de reglas precisas para realizar un cálculo o resolver un problema, ya sea a mano o, más comúnmente, en una computadora. El término proviene del matemático persa **al-Khwarizmi**. Los métodos escolares para sumar o dividir, o incluso el algoritmo de **Euclides** para hallar el máximo común divisor, son ejemplos clásicos.

Los algoritmos deben seguirse de forma **mecánica**, sin depender de intuición, creatividad o juicios subjetivos. Por eso, una receta de cocina solo sería un algoritmo si las instrucciones son claras y precisas. Si incluye términos vagos como “a gusto” o “hasta que esté listo”, ya no se considera un algoritmo.

Se aceptan, sin embargo, algoritmos que involucran **aleatoriedad controlada**. Por ejemplo, elegir un número del 1 al 6 con **probabilidades conocidas** y definidas (como lanzar un dado justo) es válido. Estos se llaman **algoritmos probabilistas** y se tratan en el Capítulo 10.

En general, se espera que un algoritmo proporcione siempre una **respuesta correcta** al aplicarse bien. Aunque en ciertos casos, como el cálculo de la raíz cuadrada de 2 (número irracional), nos conformamos con una **aproximación**, ya que no existe una solución exacta en notación decimal finita.

* **Notación Para los Programas**

Para evitar ambigüedades, el libro no utiliza lenguaje natural como el español para describir algoritmos, ya que resulta poco preciso. En su lugar, los algoritmos se presentan en forma de programas estructurados, sin adherirse a un lenguaje de programación específico (como Pascal), con el fin de destacar los conceptos esenciales y no los detalles técnicos.

**Características de la notación utilizada:**

* Se usan frases en español y notación matemática (álgebra, teoría de conjuntos, símbolos como +, ∪, etc.) para mayor claridad.
* Una sola instrucción puede representar múltiples instrucciones reales en un lenguaje de programación.
* No se espera que los algoritmos puedan ejecutarse directamente; deben ser traducidos a un lenguaje de programación concreto.

**Simplicidad y claridad ante todo:**

* Se omiten declaraciones de tipos básicos (enteros, reales, booleanos), salvo en casos importantes como funciones o procedimientos recursivos.
* Las variables se asumen como locales, salvo que el contexto indique otra cosa.
* Se evita el uso extensivo de begin y end, usando sangrías para definir bloques.
* La instrucción devolver indica el final de un procedimiento o función, y en este último caso, también el valor a retornar.
* Tipos de parámetros y tipos de retorno solo se indican si ayudan a entender mejor el algoritmo.
* **Notación Matemática**

Esta sección ofrece una revisión concisa de la notación matemática usada en el libro. Aunque se asume que el lector ya está familiarizado con muchos conceptos, se recomienda leerla para entender bien los **símbolos** empleados, algunos de los cuales no son universales (como [i..j], ∨, ∃, lg, ⌊x⌋, +, ℝ₀⁺).

**1.4.1 Cálculo Proposicional**

* Solo hay **dos valores de verdad**: **verdadero** y **falso**.
* Una **variable booleana** solo puede tomar uno de esos valores.
* Principales **operadores lógicos**:
  + **Conjunción**: p ∧ q (verdadero solo si ambos lo son)
  + **Disyunción**: p ∨ q (verdadero si al menos uno lo es)
  + **Negación**: ¬p (verdadero si p es falso)
  + **Implicación**: p ⇒ q (“si p entonces q”)
  + **Equivalencia**: p ⇔ q (p y q tienen el mismo valor de verdad)

Se pueden construir expresiones booleanas combinando variables, constantes, operadores lógicos y paréntesis.

**1.4.2 Teoría de Conjuntos**

* Un **conjunto** es una colección no ordenada de elementos **distintos**.
* **Conjunto finito**: tiene un número limitado de elementos.
* **Conjunto infinito**: no tiene límite en la cantidad de elementos.
* **Cardinalidad** |X|: número de elementos del conjunto X.
* **Conjunto vacío** ∅: conjunto sin elementos, |∅| = 0.
* **Notación**: se usa { } para listar elementos.  
  Ejemplo: {2, 3, 5, 7} es el conjunto de primos menores de 10.
* Se permiten **puntos suspensivos** si no hay ambigüedad.  
  Ejemplo: {1, 2, 3, ..., 10}
* **Técnica de demostración: Inducción Matemática**

La **inducción matemática** es una herramienta fundamental en algoritmia, utilizada para demostrar propiedades sobre la corrección y eficiencia de los algoritmos. Aunque el nombre es similar, no debe confundirse con el **razonamiento inductivo**, típico del método científico, donde se infieren reglas generales a partir de casos particulares.

* **Inducción vs. Deducción**:
  + **Inducción científica**: parte de ejemplos para inferir reglas generales, pero puede llevar a conclusiones incorrectas si no se consideran todos los casos.
  + **Deducción**: parte de reglas generales para deducir casos particulares; sus conclusiones son válidas si se aplica correctamente.

Se presentan ejemplos donde la inducción lleva a errores:

* Un polinomio p(n)=n2+n+41p(n) = n^2 + n + 41p(n)=n2+n+41 genera números primos para muchos valores de nnn, pero no para todos (p.ej., p(40)=1681p(40) = 1681p(40)=1681, que es compuesto).
* La **conjetura de Euler** de 1769, que afirmaba que no existían enteros positivos A, B, C y D que satisfagan A4+B4+C4=D4A^4 + B^4 + C^4 = D^4A4+B4+C4=D4, fue refutada en 1987 por Elkies, quien encontró un contraejemplo con números de varias cifras.

Estos casos muestran que el razonamiento inductivo puede fallar, por lo que en matemáticas se prefiere el uso de **inducción matemática formal**, que ofrece pruebas válidas siempre que se aplique correctamente.