La **abstracción en algoritmos y programación** es la habilidad de simplificar un problema complejo identificando únicamente los aspectos esenciales e ignorando los detalles irrelevantes.
A continuación, se presenta una guía práctica de ejercicios organizados por niveles, diseñados para reforzar el pensamiento abstracto en la resolución de problemas algorítmicos.

Bloque 1: Abstracción de Datos (Modelado de Entidades)

Objetivo: Identificar los datos indispensables para resolver un problema e ignorar los datos irrelevantes o "ruido".

Ejercicio 1: Filtrado de Atributos

Imagina que debes diseñar el modelo de datos para un sistema de asistencia de un curso propedéutico. Analiza la siguiente lista de características de un estudiante y clasifícalas en **Relevantes (R)** o **Irrelevantes (I)** para el módulo de registro:

1. Nombre completo del alumno.
2. Marca del teléfono inteligente con el que ingresa.
3. Horario asignado (ej. 09:00 - 11:00 hrs).
4. Color de vestimenta.
5. Materia inscrita (Algoritmos o Álgebra).
6. Distancia de su casa a la universidad.
7. Estado del registro (Presente, Retardo, Falta).

Ejercicio 2: Definición de Modelo Abstracto

Diseña la estructura de datos abstracta para el objeto RegistroAsistencia. Completa la tabla especificando el Nombre del Campo, el Tipo de Dato y la Justificación.

Nombre del Campo	Tipo de Dato Sugerido	Justificación
nombre	Texto (String)	Identifica de manera única al alumno registrado.
materia	Texto / Enum	Permite clasificar la asistencia entre Algoritmos y Álgebra.

Nombre del Campo	Tipo de Dato Sugerido	Justificación
horario	Texto / Rango	Define el grupo o bloque horario correspondiente.
fecha	Fecha (YYYY-MM-DD)	Registra el día exacto en que se realiza la sesión.
horaRegistro	Hora (HH:MM:SS)	Esencial para calcular la puntualidad del alumno.
estado	Texto ("Presente", "Retardo", "Falta")	Resultado de la evaluación del registro de asistencia.

Bloque 2: Abstracción Procedimental (Caja Negra y Funciones)

Objetivo: Separar QUÉ debe hacer un procedimiento de CÓMO lo realiza internamente.

Ejercicio 3: Definición de Interfaces (Entradas, Proceso y Salida)

Considera el requerimiento: *"Determinar automáticamente si un alumno está Presente o en Retardo al momento de enviar su registro"*.
Completa el modelo para la función evaluarEstadoAsistencia:

- **Entradas (Inputs):** Identifica los parámetros indispensables de entrada.
- **Proceso abstracto:** Describe la regla o condición matemática básica.
- **Salida (Output):** Define el valor que retorna la función.

Ejercicio 4: Generalización de Procesos Repetitivos

Analiza los siguientes casos específicos:

- **Caso A:** Un alumno llega a las 09:08 a la clase de las 09:00. Tolerancia = 10 min. → Estado: *Presente*.
- **Caso B:** Un alumno llega a las 11:15 a la clase de las 11:00. Tolerancia = 10 min. → Estado: *Retardo*.
- **Caso C:** Un alumno llega a las 09:22 a la clase de las 09:00. Tolerancia = 10 min. → Estado: *Falta*.

Tarea: Escribe un algoritmo o regla abstracta generalizada en pseudocódigo utilizando

variables generales (horaInicio, horaLlegada, minutosTolerancia) que funcione para cualquier materia u horario.

Bloque 3: Modelado de Problemas del Mundo Real a Algoritmos

Ejercicio 5: Extracción de Lógica de un Enunciado

Lee la siguiente solicitud en lenguaje natural:

> "Hola, fíjate que en el propedéutico de Algoritmos el profesor inicia puntual a las 9 am. El salón es cómodo pero a veces el internet se pone lento. Los alumnos llegan caminando o en autobús. Queremos que el sistema identifique la hora actual; si el alumno presiona el botón entre 9:00 y 9:10 se consideró Presente, si presiona entre 9:11 y 9:20 se marque Retardo, y después de las 9:20 sea Falta. No importa si vino en camión o caminando."

- **Tarea 1:** Identifica y elimina las frases o detalles irrelevantes para el algoritmo.
- **Tarea 2:** Escribe la estructura lógica condicional limpia (SI ... SINO SI ... SINO) basada únicamente en las variables necesarias.