Laboratorio 12 – Teoría de la Computación

Noviembre 2025

Problema 1

Recursión y ciclos en PF

En PF **pura**, los *ciclos* (loops) se modelan con **recursión**: una función se llama a sí misma hasta alcanzar un caso base. No se muta estado; el "contador" viaja como parámetro.
Idea mínima:

- **Recursión** ⇒ repetición definida por caso base y paso recursivo (p. ej., procesar lista: cabeza + recursión en cola).
- **Ciclos imperativos** ⇒ usan variables mutables e índices; en PF se *emulan* con recursión o combinadores de orden superior (`map`/`fold`/`filter`).

Micro–ejemplo (conceptual): sumar una lista

$$\text{suma}([]) = 0, \quad \text{suma}(x : xs) = x + \text{suma}(xs).$$

pd:Esto reemplaza un `for` imperativo sin estado mutable.

Cuándo se usa PF y cuándo no

Cuándo sí (ejemplo). Transformaciones de datos puras (pipelines de listas/tablas) donde la inmutabilidad evita efectos colaterales y facilita pruebas y paralelización. *Ejemplo:* ordenar, filtrar y mapear campos antes de serializar resultados.

Cuándo no (ejemplo). Lógica muy dependiente de I/O mutable o efectos paso a paso (p. ej., bucles con muchas escrituras a archivos/sockets o UIs muy *stateful*). En estos casos, un estilo imperativo/híbrido puede ser más directo.