Nota: Se empieza por especificar un problema. Luego se procede a escribir **un** algoritmo que cumpla la especifcación. Una vez escrito el algoritmo se escribe el programa que implementa el algoritmo.

- Tipo de datos:
  - Enteros $\mathbb{Z}$
  - Reales $\mathbb{R}$
  - Bool $\mathbb{B} = \{\textbf{true}, \textbf{false}\}$
  - Char : Hay función char(z:$\mathbb{Z}$) y ord(c:Char) $c : char, \texttt{ord}\,('\texttt{c}') + 1 = \texttt{ord}\,('\texttt{d}'), char()$
  - Enum: `enum` *Nombre* $\{\texttt{CONSTANTE1}, \texttt{CONSTANTE2}, \texttt{CONSTANTE3}, \ldots\}$.
    $\begin{cases} \texttt{ord}\,(\texttt{CONSTANTE1}) = 0 \\ Nombre(2) = \texttt{CONSTANTE3} \end{cases}$
  - Upla o tupla: $\texttt{T}_0 \times \texttt{T}_1 \times \cdots \times \texttt{T}_k$.
    $\begin{cases} (7,5)_0 = 7 \\ ('a', \texttt{DOM}, 78)_2 = 78 \end{cases}$
- especificación
  - nombre
  - parámetros
  - tipo de dato del resultado
  - etiquetas: opcionales en los `requiere`y `asegura`
- Funciones sobre secuencias:
  1. Longitud, `length`(a), $|a|$, a.`length`
  2. Indexación: $\texttt{seq}[i : \mathbb{Z}] : \texttt{T}$. Se nota: $a[i]$
  3. Pertenece: $\texttt{pertenece}(x : \texttt{T},\ s : \texttt{seq}\langle\langle\rangle\rangle\texttt{T}) : Bool$. Se nota: $\texttt{pertenece}(x, s)$ o $x \in s$
  4. Igualdad: $\texttt{seq}\langle\texttt{T}\rangle = \texttt{seq}\langle\texttt{T}\rangle$
  5. Cabeza: $\texttt{head}(a : \texttt{seq}\langle\texttt{T}\rangle) : \texttt{T}$
  6. Cola: $\texttt{tail}(a : \texttt{seq}\langle\texttt{T}\rangle) : \texttt{seq}\langle\texttt{T}\rangle$
  7. Agregar cabeza: $\texttt{addFirst}(t : \texttt{T}, a : \texttt{seq}\langle\texttt{T}\rangle) : \texttt{seq}\langle\texttt{T}\rangle$
  8. Concatenación: $\texttt{concat}(a : \texttt{seq}\langle\texttt{T}\rangle, b : \texttt{seq}\langle\texttt{T}\rangle) : \texttt{seq}\langle\texttt{T}\rangle$ (notación: $a + +b$)
  9. Subsecuencia: $\texttt{subseq}(a : \texttt{seq}\langle\texttt{T}\rangle, d, h : \mathbb{Z}) : \texttt{seq}\langle\texttt{T}\rangle$
  10. Cambiar una posición: $\texttt{setAt}(a : \texttt{seq}\langle\texttt{T}\rangle, i : \mathbb{Z}, val : \texttt{T}) : \texttt{seq}\langle\texttt{T}\rangle$
- Modularización:
  - Descomponer un problema grande en otros más chicos.
  - Componerlos y obtener la solución al problema original.

  Esto favorece muchos aspectos de calidad como:
  - Reutilización, una función auxiliar puede ser utilizada en muchos contextos)
  - Es más fácil probar algo chino que algo grande.
  - La declaratividad, es más fácil de entender.
- Programa Funcional

- ecuaciones orientadas
- evaluación de una expresión
- Transparencia referencial. El contexto no afecta a la función. Permite demostrar correctitud de forma modular.
- Formación de expresiones:
  - atómicas o formas normales. Irreducibles. $\sim$ valores. Ej: 2, false, (3, true)
  - compuestas. Se forman con expresiones atómicas y operaciones. Ej: 1+1, (4-1, true || false)