

# Clase 01

## Fundamentos de programación funcional

### Programa funcional

La programación funcional consiste en definir funciones y aplicarlas para procesar información.

Las “funciones” son verdaderamente funciones (parciales):

- Aplicar una función no tiene efectos secundarios.
- A una misma entrada corresponde siempre la misma salida
- Las estructuras de datos son inmutables.

Las **funciones son datos** como cualquier otro

- Se pueden pasar como parametros
- Se pueden devolver como resultados
- Pueden formar parte de estructuras de datos

### Expresiones

Las expresiones son secuencias de símbolos que sirven para representar datos, funciones y funciones aplicadas a los datos.

Una expresion puede ser:

1. Un **constructor**
2. Una **variable**
3. La **aplicación** de una expresión a otra

### Tipos

Un tipo es una especificación del invariante de un dato o de una función. El tipo de una función expresa un contrato.

#### Condiciones de tipado:

1. Todas las **expresiones** deben tener tipo.
2. Cada variable se debe usar siempre con un mismo tipo.
3. Los dos lados de una ecuación deben tener el mismo tipo.
4. El argumento de una función debe tener el tipo del dominio.
5. El resultado de una función debe tener el tipo del codominio.

(Sólo tienen sentido los programas bien tipados)

### Modelo de cómputo

Dada una **expresión**, se computa su valor usando **ecuaciones**.

Las expresiones bien tipadas que no tienen valor se indefinen (tienen valor  $\perp$ )

Un **programa funcional** está dado por un conjunto de ecuaciones. Más precisamente, por un conjunto de ecuaciones orientadas.

Una ecuación **e1 = e2** se interpreta desde dos puntos de vista:

1. Punto de vista **detaccional**.  
Declara que **e1** y **e2** tienen el mismo significado.
2. Punto de vista **operacional**.  
Computar el valor de **e1** se reduce a computar el valor de **e2**.

El **lado izquierdo** de una ecuación no es una **expresión** arbitraria. Debe ser una función aplicada a patrones.

Un **patrón** puede ser:

- Una **variable**.
- Un **comodín** `_`.
- Un **constructor** aplicado a patrones.

El **lado izquierdo** no debe contener variables repetidas.

Evaluar una **expresión** consiste en:

1. Buscar la subexpresión más externa que coincida con el **lado izquierdo** de una ecuación.
2. Reemplazar la subexpresión que coincide con el **lado izquierdo** de la ecuación por la expresión correspondiente al **lado derecho**.
3. Continuar evaluando la **expresión** resultante.

La evaluación se detiene cuando se da uno de los siguientes casos:

1. La **expresión** es un constructor o un constructor aplicado.
2. La **expresión** es una función parcialmente aplicada.
3. Se alcanza un estado de error.

(Un estado de error es una **expresión** que no coincide con las ecuaciones que definen a la función aplicada)

## **Funciones de orden superior**