



DEPARTAMENTO  
DE COMPUTACION

Facultad de Ciencias Exactas y Naturales - UBA

# Trabajo Práctico

## Programación Funcional

2 de Abril de 2025

Paradigmas de Lenguajes de Programación

Integrante	LU	Correo electrónico
Solana Navarro	906/22	<a href="mailto:solanan3@gmail.com">solanan3@gmail.com</a>
Melli, Tomás Felipe	371/22	<a href="mailto:tomas.melli1@gmail.com">tomas.melli1@gmail.com</a>
Lourdes Wittmund Montero	1103/22	<a href="mailto:lourdesmonterochiara@gmail.com">lourdesmonterochiara@gmail.com</a>
Marco Romano Fina	1712/21	<a href="mailto:marcoromanofinaa@gmail.com">marcoromanofinaa@gmail.com</a>



**Facultad de Ciencias Exactas y Naturales**  
Universidad de Buenos Aires

Ciudad Universitaria - (Pabellón I/Planta Baja)

Intendente Güiraldes 2610 - C1428EGA

Ciudad Autónoma de Buenos Aires - Rep. Argentina

Tel/Fax: (+54 +11) 4576-3300

<http://www.exactas.uba.ar>

# Índice

<b>1</b>	<b>Introducción</b>	<b>2</b>
<b>2</b>	<b>Módulo Documento</b>	<b>2</b>
2.1	Ejercicio 2	2
2.1.1	Justificación de validez del invariante	2
2.2	Ejercicio 3	3
2.2.1	Justificación de validez del invariante	3
<b>3</b>	<b>Módulo PPON</b>	<b>3</b>
3.1	Ejercicio 9	3
3.1.1	Justificación del tipo de recursión	3
<b>4</b>	<b>Demostración</b>	<b>4</b>
4.1	Ejercicio 10	4
4.1.1	Probamos Lema 1	4
4.1.2	Probamos Lema 2	4
4.1.3	Probamos Lema 3	5
4.1.4	Demostración Ejercicio	5

# 1 Introducción

Para este trabajo práctico vamos a implementar dos módulos.

1. **Documento** : donde definiremos un tipo de dato Doc y funciones para trabajar con documentos. Un documento es una estructura cuyo objetivo es mostrarse por pantalla en forma amigable para el usuario.
2. **PPON** : donde definiremos un tipo de dato PPON y funciones para trabajar con este formato de datos compuesto. Además, definiremos cómo convertir un PPON a un Doc para mostrarlo en forma amigable.

## 2 Módulo Documento

Un documento está representado por la estructura recursiva Doc con la siguiente definición:

```
1 data Doc = Vacio | Texto String Doc | Linea Int Doc
```

- Vacio: representa un documento vacío.
- Texto: representa un documento que contiene como primer componente un texto.
- Linea: representa un documento que contiene como primer componente un salto de línea. La siguiente línea comienza con una cantidad de espacios indicada por un entero.

Un valor del tipo Doc debe cumplir con los siguientes invariantes: Sea  $Texto\ s\ d$  entonces:

- $s$  no debe ser el string vacío.
- $s$  no debe contener saltos de línea
- $d$  debe ser Vacio o Linea  $i\ d'$

Sea Linea  $i\ d$  entonces:

- $i \geq 0$

Ya tenemos definidas las funciones : *vacío*, *línea* y *texto*.

### 2.1 Ejercicio 2

Usamos **foldDoc** para implementar el operador:

```
1 (<+>) :: Doc -> Doc -> Doc
2 d1 <+> d2 = foldDoc
3     d2
4     (\s rec -> if primerComandoText rec
5         then Texto (s++(consigoString rec)) (consigoProxDoc rec)
6         else Texto s rec)
7     (\i rec -> Linea i rec)
8     d1
```

#### 2.1.1 Justificación de validez del invariante

- Sea  $Texto\ s\ d$  :
  1. Sea  $d1 < + > d2$ , sabiendo que  $d1$  y  $d2$  cumplen con el invariante, por tanto, no tiene un string vacío, por tanto, en la unión de ambos no contendrán vacío ya que concatenamos dos cadenas de texto que no son vacíos (string no vacíos).
  2. Sea  $d1 < + > d2$ , sabiendo que  $d1$  y  $d2$  cumplen con el invariante, por tanto, no tiene un salto de línea, la unión de ambos implica concatenar dos String que ya cumplen el invariante.
  3. Sea  $d1 < + > d2$ , que cumplen el invariante, al concatenarlos, el único momento donde podemos encontrarnos frente a una situación conflictiva es al momento de concatenar un documento  $d1$  finalizado en una forma  $Texto\ s\ Vacio$  y donde  $d2$  comience con  $Texto\ s'\ d'$ . Para dar solución y mantener el invariante, nos aseguramos de concatenarlos de forma tal que no quede un documento del formato  $Texto\ s\ d$  donde  $d$  es  $Texto\ s'\ d'$ .
- Sea Linea  $i\ d$  : dados  $d1, d2$  que cumplen con el invariante, en la implementación utilizamos el constructor (Linea  $i\ d$ ), por tanto queda tal cual como estaba.

## 2.2 Ejercicio 3

```
1 indentar :: Int -> Doc -> Doc
2 indentar i d = foldDoc
3     vacio
4     (\txt rec -> Texto txt rec)
5     (\espacios rec -> Linea (espacios + i) rec) d
```

### 2.2.1 Justificación de validez del invariante

- Sea  $\text{Texto } s \ d$  :
  1. No se modifica, por tanto, no contendrá un  $s$  vacío
  2. No se modifica, por tanto, no contendrá un  $s$  de salto de línea.
  3. No se modifica, por tanto, se mantiene el invariante.
- Sea  $\text{Linea } i \ d$  : como el documento que indentamos cumple el invariante y por precondition donde  $i$  es parámetro de indentar  $i > 0$ , entonces, se cumple que  $i \geq 0$ .

## 3 Módulo PPON

```
1 data PPON = TextoPP String | IntPP Int | ObjetoPP [(String, PPON)] deriving (Eq, Show)
```

### 3.1 Ejercicio 9

```
1 pponADoc :: PPON -> Doc
2 pponADoc (TextoPP s) = texto (show s)
3 pponADoc (IntPP i) = texto (show i)
4 pponADoc (ObjetoPP pares) =
5     if pponObjetoSimple (ObjetoPP pares)
6     then aplanar (entreLlaves (map (\(x,y) -> texto (show x) <+> texto ":" <+> pponADoc y) pares))
7     else entreLlaves (map (\(x,y) -> texto (show x) <+> texto ":" <+> pponADoc y) pares)
```

#### 3.1.1 Justificación del tipo de recursión

EL tipo de recursión es **primitiva**. Esto se debe a que la función hace uso de la estructura al momento de hacer la evaluación de la guarda del if en *if pponObjetoSimple (ObjetoPP pares)* sin que esté dentro del llamado recursivo. Por otra parte, la recursión utilizada es *explícita* ya que es visible el llamado de la función a sí misma en el código.

## 4 Demostración

### 4.1 Ejercicio 10

Demostrar, utilizando razonamiento ecuacional e inducción estructural, que para todo  $n, m :: \text{Int}$  positivos y  $x :: \text{Doc}$  se cumple:

$$\text{indentar } n(\text{indentar } m x) = \text{indentar } (n + m) x$$

Se sugiere demostrar los siguientes lemas :

1.  $\text{indentar } k \text{ Vacio} = \text{Vacio} \forall k :: \text{Int positivo}$
2.  $\text{indentar } k (\text{Texto } s d) = \text{Texto } s (\text{indentar } k d) \forall k :: \text{Int positivo}, s :: \text{String y } d :: \text{Doc}$
3.  $\text{indentar } k (\text{Linea } k d) = \text{Linea } (m + k) (\text{indentar } m d) \forall m, k :: \text{Int positivo y } d :: \text{Doc}$

Tenemos las siguientes funciones con sus etiquetas para realizar la correcta demostración de los lemas y, consecuentemente, el ejercicio.

$\text{vacio} :: \text{Doc}$   
 $\text{vacio} = \text{Vacio } \{V_0\}$

$\text{linea} :: \text{Doc}$   
 $\text{linea} = \text{Linea } 0 \text{ Vacio } \{L_0\}$

$\text{texto} :: \text{String} \rightarrow \text{Doc}$   
 $\text{textot} | "'elem't = error'" \{T_1\}$   
 $\text{texto } [] = \text{Vacio } \{T_2\}$   
 $\text{texto } t = \text{Texto } t \text{ Vacio } \{T_3\}$

$\text{foldDoc} :: b \rightarrow (\text{String} \rightarrow b \rightarrow b) \rightarrow (\text{Int} \rightarrow b \rightarrow b) \rightarrow \text{Doc} \rightarrow b$   
 $\text{foldDoc } f\text{Vacio } f\text{Texto } f\text{Linea } \text{doc} = \text{case doc of}$   
 $\text{Vacio} \rightarrow f\text{Vacio } \{FD_1\}$   
 $\text{Texto } s d \rightarrow f\text{Texto } s (\text{rec } d) \{FD_2\}$   
 $\text{Linea } i d \rightarrow f\text{Linea } i (\text{rec } d) \{FD_3\}$   
 $\text{where rec} = \text{foldDoc } f\text{Vacio } f\text{Texto } f\text{Linea } \{FD_4\}$

$\text{indentar} :: \text{Int} \rightarrow \text{Doc} \rightarrow \text{Doc}$   
 $\text{indentar } i = \text{foldDoc } \text{vacio } (\text{text rec} \rightarrow \text{Texto text rec}) (\text{line rec} \rightarrow \text{Linea } (\text{line} + i) \text{ rec}) \{I_0\}$

#### 4.1.1 Probamos Lema 1

$$\text{indentar } k \text{ Vacio} = \text{Vacio} \forall k :: \text{Int positivo}$$

$\text{ppio. de reemplazo} = \text{foldDoc vacio } (\text{text rec} \rightarrow \text{Texto text rec}) (\text{line rec} \rightarrow \text{Linea } (\text{line} + k) \text{ rec}) \text{ Vacio} = \text{Vacio}$   
 $\stackrel{FD_1}{=} \text{vacio} = \text{Vacio}$   
 $\stackrel{V_0}{=} \text{Vacio} = \text{Vacio}$   
Como se quería probar.

#### 4.1.2 Probamos Lema 2

$$\text{indentar } k (\text{Texto } s d) = \text{Texto } s (\text{indentar } k d) \forall k :: \text{Int positivo}, s :: \text{String y } d :: \text{Doc}$$

$\text{ppio de reemplazo} = \text{foldDoc vacio } (\text{text rec} \rightarrow \text{Texto text rec}) (\text{line rec} \rightarrow \text{Linea } (\text{line} + k) \text{ rec}) (\text{Texto } s d)$   
 $\stackrel{FD_2}{=} (\text{text rec} \rightarrow \text{Texto text rec}) s (\text{foldDoc vacio } (\text{text rec} \rightarrow \text{Texto text rec}) (\text{line rec} \rightarrow \text{Linea } (\text{line} + k) \text{ rec}) d)$   
 $\stackrel{\beta \text{ y ppio. de reemplazo}}{=} (\text{rec} \rightarrow \text{Texto } s \text{ rec}) (\text{indentar } k d)$   
 $\stackrel{\beta}{=} \text{Texto } s (\text{indentar } k d)$   
Como se quería probar.

### 4.1.3 Probamos Lema 3

$$\text{indentar } m (\text{Linea } k \ d) = \text{Linea } (m + k) (\text{indentar } m \ d) \ \forall m, k :: \text{Int positivo y } d :: \text{Doc}$$

$$\begin{aligned} & \text{ppio. de reemplazo} \quad \equiv \quad \text{foldDoc vacio } (\text{text rec-} \rightarrow \text{Texto text rec}) (\text{line rec-} \rightarrow \text{Linea } (\text{line} + m) \text{ rec}) (\text{Linea } k \ d) \\ & \stackrel{FD_3}{=} (\text{line rec-} \rightarrow \text{Linea } (\text{line} + m) \text{ rec}) \ k \ (\text{foldDoc vacio } (\text{text rec-} \rightarrow \text{Texto text rec}) (\text{line rec-} \rightarrow \text{Linea } (\text{line} + m) \text{ rec}) \ d) \\ & \text{ppio. de reemplazo} \quad \equiv \quad (\text{line rec-} \rightarrow \text{Linea } (\text{line} + m)) \ k \ (\text{indentar } m \ d) \\ & \stackrel{\beta}{=} (\text{rec-} \rightarrow \text{Linea } (k + m) \text{ rec}) (\text{indentar } m \ d) \\ & \stackrel{\beta}{=} \text{Linea } (k + m) (\text{indentar } m \ d) \\ & \text{comutatividad de la suma} \quad \equiv \quad \text{Linea } (m + k) (\text{indentar } m \ d) \end{aligned}$$

Como se quería probar.

### 4.1.4 Demostración Ejercicio

$$\text{indentar } n (\text{indentar } m \ x) = \text{indentar } (n + m) \ x$$

Hacemos inducción estructural sobre el tipo Doc. Lo recordamos :

```
1 data Doc = Vacio | Texto String Doc | Linea Int Doc
```

$\forall n, m :: \mathbb{Z}^+. x :: \text{Doc}$

Definimos :

#### 1. Caso Base :

$$P(\text{Vacio})$$

$$\text{indentar } n (\text{indentar } \text{Vacio}) = \text{indentar } (n + m) \ \text{Vacio}$$

$$\stackrel{I_0}{=} \text{indentar } n (\text{Vacio}) = \text{indentar } (n + m) \ \text{Vacio}$$

$$\stackrel{\text{Lema}_1}{=} \text{Vacio} = \text{Vacio}$$

#### 2. Caso Inductivo : Vamos a presentar entonces la **Hipótesis Inductiva**.

$$\forall d :: \text{Doc. } \text{indentar } n (\text{indentar } m \ d) = \text{indentar } (n + m) \ d$$

Para el caso de este tipo de datos tenemos dos casos :

$$(a) \ \forall d :: \text{Doc.} \forall s :: \text{String. } P(d) \implies P(\text{Texto } s \ d)$$

$$(b) \ \forall d :: \text{Doc.} \forall i :: \mathbb{Z}^+. P(d) \implies P(\text{Linea } i \ d)$$

Vamos a comenzar.

(a)

$$\text{indentar } n (\text{indentar } m (\text{Texto } s \ d)) = \text{indentar } (n + m) (\text{Texto } s \ d)$$

$$\stackrel{\text{Lema}_2}{=} \text{indentar } n (\text{Texto } s (\text{indentar } m \ d)) = \text{indentar } (n + m) (\text{Texto } s \ d)$$

$$\stackrel{\text{Lema}_2}{=} \text{Texto } s (\text{indentar } n (\text{indentar } m \ d)) = \text{indentar } (n + m) (\text{Texto } s \ d)$$

$$\stackrel{HI}{=} \text{Texto } s (\text{indentar } (n + m) \ d) = \text{indentar } (n + m) (\text{Texto } s \ d)$$

$$\stackrel{HI}{=} \text{indentar } (n + m) (\text{Texto } s \ d) = \text{indentar } (n + m) (\text{Texto } s \ d)$$

Como se quería probar.

(b)

$$\text{indentar } n (\text{indentar } m (\text{Linea } i \ d)) = \text{indentar } (n + m) (\text{Linea } i \ d)$$

$$\stackrel{\text{Lema}_3}{=} \text{indentar } n (\text{Linea } (m + i) (\text{indentar } m \ d)) = \text{indentar } (n + m) (\text{Linea } i \ d)$$

$$\stackrel{\text{Lema}_3}{=} \text{Linea } (n + m + i) (\text{indentar } n (\text{indentar } m \ d)) = \text{indentar } (n + m) (\text{Linea } i \ d)$$

$$\stackrel{HI}{=} \text{Linea } (n + m + i) (\text{indentar } (n + m) \ d) = \text{indentar } (n + m) (\text{Linea } i \ d)$$

$$\stackrel{\text{Lema}_3}{=} \text{indentar } (n + m) (\text{Linea } i \ d) = \text{indentar } (n + m) (\text{Linea } i \ d)$$

Como se quería probar.