PROGRAMACIÓN LÓGICA Y FUNCIONAL

**SESIÓN 05**

**CICLO: AGOSTO 2020**

**Contenido** 

1. Introducción

2. Funciones internas de listas 

3. Functores

4. Bibliografía



**Pautas de trabajo**

Pautas de trabajo

• Los días que tengamos clases debemos conectarnos a través de Zoom.

• La participación de los estudiantes se dará través del **chat de Zoom.**

• En Canvas encontrarán la clase de hoy, el ppt de la sesión 05, Laboratorio 05

**Recordando la sesión anterior**

**¿ Qué se entiende por funciones recursivas?** Levantemos la mano para participar

**Logro del Aprendizaje **

Al finalizar la presente sesión el estudiante: 

1. Conoce los métodos de las listas y la definición

de functores.

2. Entiende la utilidad de las listas y functores en la

programación funcional.





**1. Introducción**

¿Consideras que las 

listas son útiles? Si /

No? ¿Por qué?



**1. Introducción**

**INDICE**

[0] [1] [2] [3] [4]

**15 17 10 20 15**

**ELEMENTO**

****

**2. Funciones internas de Listas**

**x:ys** es la lista obtenida añadiendo x al principio de ys.

*Ejms*:

prelude> 2:[5,3] (enter) → [2,5,3]

prelude> 2:[] (enter) → [2]

prelude > 'u':"nion" (enter) → "union"



**2. Funciones internas de Listas**

**xs ++ ys** es la concatenación de xs e ys.

*Ejms*:

prelude > [2,5] ++ [3,7,6] (enter) → [2,5,3,7,6] prelude > "Poli" ++ "zonte" (enter) → Polizonte



**2. Funciones internas de Listasxs !! n** es el elemento n-ésimo de xs.

*Ejms*:

prelude > [7,9,6,5] !! 2 → 6

prelude > "Sevilla" !! 3 → 'i'



**2. Funciones internas de Listas**

**drop n xs** elimina los n primeros elementos de xs.

*Ejms*:

prelude > drop 2 [7,5,9,6,8] → [9,6,8]

prelude > drop 9 [7,5,9,6,8] → []

prelude > drop (-4) [7,5,9,6,8] → [7,5,9,6,8] 

**2. Funciones internas de Listas**

**elem x ys** verifica si x pertenece a ys. Devuelve un valor booleano.

*Ejms*:

prelude > elem 3 [5,3,7] → True

prelude > 3 `elem` [5,3,7] → True

prelude > 4 `elem` [5,3,7] → False



**2. Funciones internas de Listas**

**head xs** es el primer elemento de la lista xs.

*Ejms*:

prelude > head [3,2,5] → 3

prelude > head "Berilio" → 'B'



**2. Funciones internas de Listas**

**init xs** es la lista obtenida eliminando el último elemento de xs.

*Ejms*:

prelude > init [3,7,2] → [3,7]

prelude > init "cintas" → "cinta"



**2. Funciones internas de Listas**

**length xs** es el número de elementos de la lista xs.

*Ejms*:

prelude > length [4,2,5] → 3

prelude > length "Peru" → 4



**2. Funciones internas de Listas**

**reverse xs** es la inversa de la lista xs.

*Ejms*:

prelude > reverse [3,5,2,4] → [4,2,5,3]

prelude > reverse ['p','t','u'] → "utp"



**3. Functores **

La clase Functor puede ser 

implementada por las estructuras de

datos a las cuales se les pueda **aplicar**

**una función a todos sus elementos**.

La especificación de la clase Functor es: 

class Functor f where

fmap :: (a -> b) -> f a -> f b



**3. Functores**

fmap es una función interna de Haskell que mapea una función a todos los elementos del tipo que implemente la clase.

En el tipo de fmap ((a -> b) -> f a -> f b ), se puede ver que el primer parámetro es una función a -> b, esa es la que se aplica a todos los elementos dentro de f y el segundo parámetro es el functor, f a.



**Video**

****https://www.youtube.com/watch?v=w0oWpYuCDig



**Prueba de control**

Relacione las columnas:

1. head 2. Init

3. Length 4. xs !! n

( ) muestra el tamaño de la lista ( ) eliminan el último elemento de la lista

( ) devuelve el elemento “n” de la lista

( ) devuelve el primer elemento de la lista



**Utilidad**

Qué hemos aprendido el día de hoy? 

Utiliza el chat para participar

**Actividad grupal**

LABORATORIO 05



