Curso: Programación Lógica y Funcional

Unidad 2: Programación funcional

Semana 08 - Sesión 15: Listas y Cadenas

Docente: Carlos R. P. Tovar

UTP Universidad Tecnológica del Perú

Dudas de la anterior sesión

INICIO
Objetivo de la sesión

Al final de la sesión, el estudiante será capaz de manipular listas y cadenas eficientemente en Haskell, aplicando funciones de alto orden y comprensión de listas.

Recordatorio y motivación

- **Recordatorio:** Las cadenas en Haskell son listas de caracteres: String = [Char]
- **Motivación:** La manipulación eficiente de texto es crucial en aplicaciones reales
- **Pregunta detonante:**
  "¿Cómo podemos procesar y transformar textos de manera funcional y elegante?"

TRANSFORMACIÓN
Relación entre cadenas y listas

```
-- String es un alias de [Char]
type String = [Char]

-- Ejemplos de equivalencia
"Hola" == ['H','o','l','a’]      -- True
length "Haskell" == 7      -- True
```

Funciones básicas de cadenas

```
-- Conversiones
show 123    -- "123"
read "456"  -- 456

-- Manipulación básica
length "texto"          -- 5
reverse "Haskell"       -- "lleksaH"
"Ha" ++ "skell"         -- "Haskell"
```

```
-- Extracción
head "Hola"     -- 'H'
tail "Hola"     -- "ola"
init "Hola"     -- "Hol"
last "Hola"     -- 'a'
```

Funciones de texto útiles

```
import Data.Char (toUpper, toLower, isDigit, isAlpha)

-- Transformación de caso
map toUpper "hola"    -- "HOLA"
map toLower "HASKELL" -- "haskell"

-- Verificación
all isDigit "12345"   -- True
any isAlpha "123a"    -- True

-- División y unión
words "hola mundo haskell"  -- ["hola","mundo","haskell"]
unwords ["hola","mundo"]    -- "hola mundo"
```

Comprensión de listas con cadenas

```
-- Filtrar dígitos de una cadena
[ c | c <- "a1b2c3", isDigit c ]  -- "123"

-- Convertir a mayúsculas solo vocales
[ if c `elem` "aeiou" then toUpper c else c | c <- "haskell" ]
-- "hAskEll"

-- Pares de caracteres y sus posiciones
[ (i, c) | (i, c) <- zip [0..] "Haskell" ]
-- [(0,'H'),(1,'a'),(2,'s'),(3,'k'),(4,'e'),(5,'l'),(6,'l')]
```

PRACTICA
Ejercicio guiado - Contador de palabras

```
contarPalabras :: String -> Int
contarPalabras texto = length (words texto)

-- Versión con filtrado de palabras vacías
contarPalabrasSeguro :: String -> Int
contarPalabrasSeguro = length . filter (not . null) . words
```

Ejercicios en clase

- **Palíndromos:** Verificar si una cadena es palíndromo
- **Iniciales:** Extraer las iniciales de un nombre completo
- **Codificación simple:** Rot13 sobre una cadena
- **Estadísticas de texto:** Contar líneas, palabras y caracteres

Solución ejemplo - Palíndromos

```
esPalindromo :: String -> Bool
esPalindromo texto = texto == reverse texto

-- Ignorando espacios y mayúsculas
esPalindromoAvanzado :: String -> Bool
esPalindromoAvanzado texto =
    let textoLimpio = filter (/= ' ') (map toLower texto)
    in textoLimpio == reverse textoLimpio
```

Funciones útiles para explorar:

- lines / unlines - Trabajar con líneas de texto
- intercalate - Unir con separador
- isPrefixOf / isSuffixOf - Verificar inicio/fin
- takeWhile / dropWhile - Extraer según condición

Ejercicios propuestos para práctica:

```
-- 1. Contar ocurrencias de un carácter
contarCaracter :: Char -> String -> Int

-- 2. Eliminar espacios duplicados
eliminarEspaciosDuplicados :: String -> String

-- 3. Formatear nombre: "juan perez" -> "Juan Perez"
formatearNombre :: String -> String

-- 4. Extraer dominios de emails
extraerDominios :: [String] -> [String]
```

Tareas

- Implementar un analizador de texto básico
- Crear funciones para estadísticas de documentos
- Procesar un archivo de texto real

CIERRE
Conclusiones

- Las cadenas son listas, por lo que todas las funciones de listas aplican
- Haskell proporciona herramientas poderosas para procesamiento de texto
- La inmutabilidad garantiza seguridad en manipulación de cadenas
- Las comprensiones de listas son ideales para transformaciones complejas

Aplicaciones prácticas

- **Procesamiento de logs:** Análisis de archivos de texto
- **Limpieza de datos:** Filtrado y normalización de texto
- **Análisis léxico:** Procesadores simples de lenguaje natural
- **Generación de reportes:** Formateo de salidas de texto

Universidad
Tecnológica
del Perú