04 - Listas

André Rauber Du Bois
dubois@inf.ufpel.edu.br
Computação - CDTec - UFPel

Listas

- Listas são uma das principais estruturas de dados usadas em programação funcional
- Listas em Elixir são delimitadas por colchetes, com os elementos sendo separados por vírgula
- **Exemplos:**

[1,2,3,4,5]

[true, false, true]

[1, 2, 3, "asdf", true, false]

[[1, 2, 3], [1, 2], [1]]

[]

Concatenação de Listas

- Operador ++

```
iex(1)> [1,2,3] ++ [4,5]
[1, 2, 3, 4, 5]
iex(2)> [1] ++ [2]++[4,5]
[1, 2, 4, 5]
iex(3)> 5 ++ [2,3]
** (ArgumentError) argument error
    :erlang.++(5, [2, 3])
iex(3)> [[2,3,4]] ++ [2,3]
[[2, 3, 4], 2, 3]
```

Pattern Matching - Casamento de padrões

- O operador **=** não é uma atribuição e sim um operador de pattern matching
- Uma variável, sempre casa o padrão com qualquer valor

```
iex(1)> x = 1

1

iex(2)> x

1

iex(3)> y = [1,2,3,4]

[1, 2, 3, 4]
```

```
iex(4)> y
[1, 2, 3, 4]
iex(5)> z = [x,y]
[1, [1, 2, 3, 4]]
iex(6)> z
[1, [1, 2, 3, 4]]
iex(7)> [z,x,y]
[[1, [1, 2, 3, 4]], 1, [1, 2, 3, 4]]
```

Pattern Matching

- Podemos usar pattern matching para *“destruir”* estruturas complexas:

```
iex(2)> [x,y,z] = [1,2,3]

[1, 2, 3]

iex(3)> y

2

iex(4)> z

3

iex(5)> [x,y] = [1,2,3]

** (MatchError) no match of right hand side value: [1, 2, 3]
```

Pattern Matching

- Podemos usar o padrão **[cabeca|resto]** separar a cabeça do resto de uma lista:

```
iex(1)> [h|t] = [1,2,3,4]

[1, 2, 3, 4]

iex(2)> h

1

iex(3)> t

[2, 3, 4]

iex(4)> [h|t] = []

** (MatchError) no match of right hand side value: []
```

- Podemos usar **[h|l]** para inserir o elemento **h** na frente da lista **l**
- Em Elixir, as listas são encadeadas, onde **[h|t]** representa o nó ligando o **h** ao **t,** chamado de ***cons cell***

```
iex(1)> [1|[2,3,4]]

[1, 2, 3, 4]

iex(2)> [true|[false,3]]

[true, false, 3]

iex(3)> [[1,2,3]|[4,5,6]]

[[1, 2, 3], 4, 5, 6]

iex(4)> [1|[2|[3|[4|[5|[]]]]]]

[1, 2, 3, 4, 5]
```

Funções sobre listas

- Funções que percorrem listas são implementadas usando recursão
- A recursão em cima de listas geralmente possui 2 casos
  - O caso da lista vazia: **[]**
  - O caso da lista com pelo menos 1 elemento: **[h|t]**

```
defmodule Exemplo do
 def soma_lista([]) do
   0
 end
 def soma_lista([head|tail]) do
   head + soma_lista(tail)
 end
end
```

```
iex(1)> Exemplo.soma_lista([1,2,3,4])
10
```

```
defmodule Exemplo do
 def quadrado_lista([]) do
   []
 end
 def quadrado_lista([head|tail]) do
   [head*head | quadrado_lista(tail)]
 end
end
```

```
iex(1)> Exemplo.quadrado_lista([1,2,3,4,5])
[1, 4, 9, 16, 25]
```