Otimizando o Fibonacci

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

Fibonacci: Recursão

```
defmodule Fib do
    def fibr(0), do: 0
    def fibr(1), do: 1
    def fibr(n), do: fibr(n-1) + fibr(n-2)
 end
```

- Problema: Para cada chamada recursiva, recomputa valores computados anteriormente

Outra solução: Elegante

```
defmodule Fib do
 def fib_iterative(n) when n < 2 do
   fibr(n)
 end
 def fib_iterative(n) do
   Enum.at(Enum.reduce(2..n, [0, 1], fn i, acc -> acc ++ [Enum.at(acc, i-1) + Enum.at(acc, i-2)]
   end), n)
 end
end
```

Enum.reduce([1,2,3,4], 0, +) = 0 + 1 + 2 + 3 + 4

Outra solução: Tail call Optimization

```
defmodule Fib do
 def fib(0), do: 0
 def fib(1), do: 1
 def fib(n), do: fib_(2,n,0,1)
 defp fib_(cont, n, f1, f2) when cont == n, do: f1 + f2
 defp fib_(cont, n, f1, f2), do: fib_(cont+1, n, f2, f1+f2)
 end
```

- Solução parecida com a solução em C

Tempo de cada solução

iex(4)> elem(:timer.tc(&Fib.fib/1,[50]),0)

2

iex(5)> elem(:timer.tc(&Fib.fibr/1,[50]),0)

103252326

iex(6)> elem(:timer.tc(&Fib.fib_iterative/1,[50]),0)

135