

# *Linguagens de Alto Nível*

*(continuação...)*

## Estrutura de Linguagens

Francisco Sant'Anna

Sala 6020-B

francisco@ime.uerj.br

<http://github.com/fsantanna-uerj/EDL>



# *Trabalho 1*

## *Linguagem Lua*

**Francisco Sant'Anna**

# Introdução

- Multi-paradigma
  - imperativa, OO, funcional
- Multi-plataforma (ANSI-C)
  - PC (Windows, Mac, Linux), S
- Dinâmica
  - *eval*, tipagem dinâmica, tabelas
- Foco em *scripting*
  - configuração, macros, extensões
  - nicho em video games

```
// imperativa
for i=1, 10 do
  if i % 2 == 0 then
    local v = i*i
    print(i,v)
  end
end
```

```
// OO
function SET (self, v)
  self.v = v
end
o = { v=0, set=SET }
o.set(o,10)
o:set(10)
```

```
// funcional
t = { 10,1,5 }
table.sort(t,
  function (v1,v2)
    return v1 > v2
  end
) -- {10,5,1}
```







# Origens e Influências

- Desenvolvida na PUC-Rio
- Foi influenciada:
  - Modula/Pascal (sintaxe)
  - Scheme (semântica)
  - CLU (atribuição e retorno múltiplo)
  - SNOBOL e AWK (array associativo)
- Influenciou:
  - Io, GameMonkey, JavaScript, Julia, MiniD, Red, Ring, Ruby, Squirrel, MoonScript, C--

# Peculiaridades

- Tabela como único mecanismo estrutural
  - arrays, dicionários, objetos, módulos, etc
  - metatabelas: alteram a semântica de tabelas

## ■ Funções

- valores de primeira classe (como números, strings e tabelas)
- aninháveis, anônimas, *closures*

## ■ Co-rotinas

- iteradores/geradores
- programação assíncrona
- multithreading cooperativo

# Closures

- Exemplo 1: um contador
- Exemplo 2: dois contadores (cópias)
- Exemplo 3: dois contadores (closures)

# Closures

```
function criar ()  
    local v = 0  
    return function ()  
        v = v + 1  
        return v  
    end  
end
```

```
local c1 = criar()  
local c2 = criar()
```

```
print(c1()) -- 1  
print(c2()) -- 1  
print(c2()) -- 2  
print(c1()) -- 2
```



# Discussão - Expressividade

- Closure = Função + Ambiente
  - Objeto = Métodos + Propriedades
- Ambiente exige alocação dinâmica
  - Cada vez que uma closure é criada, um novo ambiente é criado, com espaço para as variáveis e seus valores.
- Como expressar aquele código em...
  - Java?
    - Classes e objetos. Praticamente transformar linha a linha.
  - C?
    - Exige uma nova estrutura de dados. Closure vai ser um ponteiro para a função e outro ponteiro para uma `struct` com as variáveis. (programa precisa ser *violentamente* reestruturado)