2013

Decode Team

Instituto Politécnico de tomar

Portugol

Equivalências de estruturas entre Portugol e Lua

Índice

[Nota Geral: 3](#_Toc358165556)

[Algumas notas sobre Lua: 3](#_Toc358165557)

[Estrutura Início 3](#_Toc358165558)

[Início: 3](#_Toc358165559)

[Estrutura Fim 3](#_Toc358165560)

[Fim: 3](#_Toc358165561)

[Variáveis 4](#_Toc358165562)

[Equivalência entre TIPOS de variáveis 4](#_Toc358165563)

[Definição e atribuição de variáveis 4](#_Toc358165564)

[Estruturas input/output 4](#_Toc358165565)

[Input – Ler 4](#_Toc358165566)

[Output – Escrever 5](#_Toc358165567)

[Estruturas de Decisão 5](#_Toc358165568)

[Condição *“if”* e *“if else”* 5](#_Toc358165569)

[Condição *“while”* 7](#_Toc358165570)

[Condição *“do while”* 8](#_Toc358165571)

[Exemplos práticos 8](#_Toc358165572)

[Estrutura Conector 9](#_Toc358165573)

[Conector 9](#_Toc358165574)

[Funções 9](#_Toc358165575)

[Definir funções 9](#_Toc358165576)

[Definir função 9](#_Toc358165577)

[Exemplos do uso de funções 10](#_Toc358165578)

[Estrutura de retorno 11](#_Toc358165579)

[Return 11](#_Toc358165580)

[Operadores 12](#_Toc358165581)

[Aritméticos 12](#_Toc358165582)

[Lógicos 12](#_Toc358165583)

[Relacionais 12](#_Toc358165584)

[ANEXO 13](#_Toc358165585)

[Algoritmo com o uso da condição “if” 13](#_Toc358165586)

[Fluxograma 13](#_Toc358165587)

[Código 13](#_Toc358165588)

[Esquema detalhado 14](#_Toc358165589)

[Algoritmo com o uso da condição “if else” 15](#_Toc358165590)

[Fluxograma 15](#_Toc358165591)

[Código: 15](#_Toc358165592)

[Algoritmo com o uso da condição “while” 16](#_Toc358165593)

[Fluxograma 16](#_Toc358165594)

[Código 16](#_Toc358165595)

[Algoritmo com o uso da condição “ do while” 17](#_Toc358165596)

[Fluxograma 17](#_Toc358165597)

[Código 17](#_Toc358165598)

[Algoritmo com o uso de uma função 18](#_Toc358165599)

[Fluxogramas 18](#_Toc358165600)

[Código 18](#_Toc358165601)

# Nota Geral:

Devido à especificação da linguagem, a tradução só é possível depois de ser executado o fluxograma.

# Algumas notas sobre Lua:

* É case sensitive.
* Necessita sempre do código indentado pois é a única maneira de o Python saber o que faz está dentro e fora dos ciclos porque não possui um ‘End’.

# Estrutura Início

## Início:

Início

O Python não tem equivalência ao Inicio.

# Estrutura Fim

## Fim:

Fim

O Python não tem equivalência ao Fim.

# Variáveis

## Equivalência entre TIPOS de variáveis

|  |  |  |
| --- | --- | --- |
| TIPO | Portugol | Python |
| Inteiro | Inteiro | Number |
| Real | Real | Number |
| Texto | Texto | String |
| Lógico | Logico | boolean |

Tabela - Tipos de variáveis

Nota: Lua não diferencia valores inteiros de valores reais.

## Definição e atribuição de variáveis

Nota: Ao utilizar variáveis em Lua não é necessário informar o tipo desta variável como fazemos em outras linguagens.

# Estruturas input/output

## Input – Ler

variavel

io.write'<texto> '

variavel = io.read()

## Output – Escrever

expressao

Para escrever no ecrã:

Print '<texto>'

Ou

Print ( '<texto>',<variável>)

# Estruturas de Decisão

## Condição *“if”* e *“if else”*

condição

FALSE

TRUE

Instruções 2

Instruções 1

Para TRUE, escrever:

**if** <condição> **then**  
  
<instruções>  
  
**end**

Para FALSE:

Se Instruções 2 for igual a (conector) não fazer nada.

Senão, escrever:

**else**  
  
<instruções>

Exemplos práticos

#### Condição “if”

**if** <condição> **then**  
  
<instruções>  
  
**end**

#### Condição “if else”

io.write'Digite um número:'

n = io.read()

if n%2==0 then

print 'Numero Par'

else

print 'Numero Impar'

end

## Condição *“while”*

condição

true

false

instruções

**while** <condição> **do**  
  
<instruções>  
  
**end**

Nota:Em Lua o ciclo While é composto sempre pelo ‘do’.

## Condição *“do while”*

instruções

TRUE

condição

FALSE

Instruções

Nota: Não tem equivalência

### Exemplos práticos

#### Condição “while”

i=1

while i<=10 do

print(i)

i=i+1

end

#### Condição “do while”

Nota: Não tem equivalência

# Estrutura Conector

## Conector

# Funções

## Definir funções

Exemplo( a , b, . . .)

### Definir função

**Function**<nome\_função>

<Instruções>

**end**

Chamada de funções

Function(Nome)

NOME(PARAMETRO);

### Exemplos do uso de funções

io.write'Digite um numero: '

k=io.read()

function fact(k)

if tonumber(k) == 0 then

return 1

else

return k \* fact(k - 1)

end

end

j=fact(k)

print(j)

# Estrutura de retorno

## Return

expressao

return expressao;

# Operadores

### Aritméticos

|  |  |  |
| --- | --- | --- |
| Nome | Portugol | Lua |
| Adição | a + b | a + b |
| Subtração | a – b | a - b |
| Divisão | a / b | a / b |
| Multiplicação | a \* b | a \* b |
| Resto da divisão inteira |  | a % b |
| Potenciação |  | ^ |

Tabela - Equivalência de operadores aritméticos

### Lógicos

|  |  |  |
| --- | --- | --- |
| Nome | Portugol | Lua |
| Disjunção | a && b | and |
| Conjunção | a || b | or |
| Negação |  | not |

Tabela - Equivalência de operadores lógicos

### Relacionais

|  |  |  |
| --- | --- | --- |
| Nome | Portugol | Lua |
| Igual | a == b | a == b |
| Diferente | a != b | a ~= b |
| Maior | a > b | a > b |
| Maior ou igual | a >= b | a >= b |
| Menor | a < b | a < b |
| Menor ou igual | a <= b | a <= b |

Tabela - Equivalência de operadores relacionais

# ***ANEXO***

Para uma compreensão mais abrangente do uso das estruturas, ficam alguns exemplos mais extensivos, com o uso de várias estruturas em algoritmos completos.

## Algoritmo com o uso da condição “if”

**Problema:** Verificar se um número introduzido pelo utilizador é par.

### Fluxograma

Inicio

n

n%2=0

TRUE

FALSE

“Par”

Fim

### Código

io.write'Digite um número:'

n = io.read()

if n%2==0 then

print 'Par'

end

### Esquema detalhado

|  |  |
| --- | --- |
| Inicio | Não tem Equivalência! |
| n | io.write'Digite um número:' |
| FALSE  “Par”  n%2=0  TRUE | if n%2==0 then  True:  print('Par') |
|  | } |
| Inicio | function(k);  chama a função fact, com o parâmetro k |

## Algoritmo com o uso da condição “if else”

**Problema:** Verificar se um número introduzido pelo utilizador é par ou ímpar.

### Fluxograma

### 

TRUE

Fim

“Par”

n%2=0

n

Inicio

FALSE

“Impar”

### Código:

io.write'Digite um número:'

n = io.read()

if n%2==0 then

print 'Numero Par'

else

print 'Numero Impar'

end

## Algoritmo com o uso da condição “while”

**Problema:** Escrever um número de 1 a 10.

### Fluxograma

Inicio

i=1

i <= 10

FALSE

TRUE

i

i=i + 1

Fim

### Código

i=1

while i<=10 do

print(i)

i=i+1

end

## Algoritmo com o uso da condição “ do while”

**Problema:** Pedir um número positivo.

### Fluxograma

Inicio

i = 0

i

TRUE

i < 0

FALSE

Fim

### Código

Sem Equivalência!

## Algoritmo com o uso de uma função

**Problema:** Factorial de um número.

### Fluxogramas

**Código principal**

**Função fact(k)**

fact(k)

Inicio

TRUE

FALSE

k > 2

i = 5

j = fact ( i )

k

k \* fact(k - 1)

j

Fim

Fim

### Código

io.write'Digite um numero: '

k=io.read()

function fact(k)

if tonumber(k) == 0 then

return 1

else

return k \* fact(k - 1)

end

end

j=fact(k)

print(j)