2013

Decode Team

Instituto Politécnico de tomar

Portugol

Equivalências de estruturas entre Portugol e Python

Índice

[Nota Geral: 3](#_Toc358163780)

[Algumas notas sobre Python: 3](#_Toc358163781)

[Estrutura Início 3](#_Toc358163782)

[Início: 3](#_Toc358163783)

[Estrutura Fim 3](#_Toc358163784)

[Fim: 3](#_Toc358163785)

[Variáveis 3](#_Toc358163786)

[Equivalência entre TIPOS de variáveis 3](#_Toc358163787)

[Definição e atribuição de variáveis 4](#_Toc358163788)

[Estruturas input/output 4](#_Toc358163789)

[Input – Ler 4](#_Toc358163790)

[Output – Escrever 5](#_Toc358163791)

[Estruturas de Decisão 5](#_Toc358163792)

[Condição *“if”* e *“if else”* 5](#_Toc358163793)

[Exemplos práticos 6](#_Toc358163794)

[Condição *“while”* 6](#_Toc358163795)

[Condição *“do while”* 7](#_Toc358163796)

[Exemplos práticos 7](#_Toc358163797)

[Estrutura Conector 7](#_Toc358163798)

[Conector 7](#_Toc358163799)

[Funções 8](#_Toc358163800)

[Definir funções 8](#_Toc358163801)

[Definir função 8](#_Toc358163802)

[Exemplos do uso de funções 9](#_Toc358163803)

[Estrutura de retorno 9](#_Toc358163804)

[Return 9](#_Toc358163805)

[Operadores 10](#_Toc358163806)

[Aritméticos 10](#_Toc358163807)

[Lógicos 10](#_Toc358163808)

[Relacionais 10](#_Toc358163809)

[ANEXO 11](#_Toc358163810)

[Algoritmo com o uso da condição “if” 11](#_Toc358163811)

[Fluxograma 11](#_Toc358163812)

[Código 11](#_Toc358163813)

[Esquema detalhado 12](#_Toc358163814)

[Algoritmo com o uso da condição “if else” 13](#_Toc358163815)

[Fluxograma 13](#_Toc358163816)

[Código: 13](#_Toc358163817)

[Algoritmo com o uso da condição “while” 14](#_Toc358163818)

[Fluxograma 14](#_Toc358163819)

[Código 14](#_Toc358163820)

[Algoritmo com o uso da condição “ do while” 15](#_Toc358163821)

[Fluxograma 15](#_Toc358163822)

[Código 15](#_Toc358163823)

[Algoritmo com o uso de uma função 16](#_Toc358163824)

[Fluxogramas 16](#_Toc358163825)

[Código 16](#_Toc358163826)

# Nota Geral:

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

# Algumas notas sobre Python:

* 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 | Int |
| Real | Real | Float |
| Texto | Texto | String |
| Lógico | Logico | boolean |

Tabela - Tipos de variáveis

## Definição e atribuição de variáveis

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

Mas por vezes e para evitar que o Python confunda o nome de uma variável que tem um valor inteiro com uma string, basta antes do colocar antes do nome da variável Int ou se for outro tipo colocar o respectivo tipo antes( real, boolean ou float)

Int(n)

Exemplo:

n= input('Digite um numero: ')

if **int**(n)%2==0:

print('Numero Par')

# Estruturas input/output

## Input – Ler

variavel

<variável>= input('<texto>')

## 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>**:**

<instruções>

Para FALSE:

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

Senão, escrever:

**else:**

<instruções>

### Exemplos práticos

#### Condição “if”

n= input('Digite um numero: ')

if int(n)%2==0:

print('Par')

#### Condição “if else”

n= input('Digite um numero: ')

if int(n)%2==0:

print('Numero Par')

else:

print('Numero Impar')

## Condição *“while”*

condição

true

false

instruções

**while** <condição>**:**

   <instruções>

## Condição *“do while”*

instruções

TRUE

condição

FALSE

Instruções

Nota: Não tem equivalência

### Exemplos práticos

#### Condição “while”

i=1

while int(i)<=10:

print(i)

i=i+1

#### Condição “do while”

Nota: Não tem equivalência

# Estrutura Conector

## Conector

# Funções

## Definir funções

Exemplo( a , b, . . .)

### Definir função

**def**<nome\_função>:

<instruções>Chamada de funções

Fact(Nome)

NOME(PARAMETRO);

### Exemplos do uso de funções

k=input('Digite um numero: ')

def fact(k):

if k == 0:

return 1

else:

return (int(k) \* fact( int(k) - 1 ))

j=fact(k)

print(fact(k))

# Estrutura de retorno

## Return

expressao

return expressao;

# Operadores

### Aritméticos

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

Tabela - Equivalência de operadores lógicos

### Relacionais

|  |  |  |
| --- | --- | --- |
| Nome | Portugol | Python |
| 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

n= input('Digite um numero: ')

if int(n)%2==0:

print('Par')

### Esquema detalhado

|  |  |
| --- | --- |
| Inicio | import java.util.Scanner;  public class VerificaPAR {  public static void main(String[] args) { |
| n | n= input('Digite um numero: ') |
| FALSE  “Par”  n%2=0  TRUE | if int(n)%2==0:  True:  print('Par') |
|  | } |
| Inicio | fact(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:

n= input('Digite um numero: ')

if int(n)%2==0:

print('Numero Par')

else:

print('Numero Impar')

## 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 int(i)<=10:

print(i)

i=i+1

## 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

k=input('Digite um numero: ')

def fact(k):

if k == 0:

return 1

else:

return (int(k) \* fact( int(k) - 1 ))

j=fact(k)

print(fact(k))