Linguagem de Programação

Introdução à Linguagem Python

Ma. Vanessa Matias Leite

1

- Unidade de Ensino: 01
- Competência da Unidade: conhecer sobre os conceitos da Linguagem Python
- Resumo: compreensão sobre os elementos da linguagem Python e introdução aos comandos da linguagem
- Palavras-chave: Python, variável, função, parâmetros, if, for.
- Título da Teleaula: Introdução à Linguagem Python
- Teleaula nº: 01

2

Python

- Linguagem Simples;
- Desenvolvimento Web;
- Inteligência Artificial;
- Computação Gráfica;
- Big Data;

Fonte: https://bit.ly/2DyT7UD

3

Conceitos

A linguagem Python

4

Python

- Linguagem Interpretada;
- Tipagem dinâmica e forte;
- Uso da indentação como forma de definição de blocos de código;
- Muitas bibliotecas;

5

Python

Fonte: https://bit.ly/2DyT7UD

6

Variáveis em Python

<nome variável> = <valor>

- Sinal de "=" para atribuir valor
- Não é necessário declarar o tipo da variável;
- Podem possuir caracteres alfanuméricos e o caractere underscore (_);
- Não podem começar com números;
- Em Python, tudo é objeto.

7

Operadores em Python

Operador	Descrição
+	Soma
-	Subtração
*	Multiplicação
/	Divisão
pow(x,y)	Exponenciação

Fonte: Autor

8

Tipos de Dados em Python

Tipo	Valores
int	6, 89000, -67000, 0 ...
float	5.6, 3.1415, -21.55, 6.0 ...
bool	True, False
complex	3+4i
str	'bom dia', '9.0', '5'...

Fonte: Autor

9

Funções Básicas do Python

- print(): imprimir os argumentos passados;
- input(): entrada de dados;
- type(): mostra o tipo de dados;

10

Resolução da SP

Teste I

11

O primeiro teste consiste em mostrar que você sabe utilizar a linguagem Python. Apresente comandos simples como o famoso "Olá mundo" e contas simples de matemática.

12

Conceitos

Estruturas lógicas, condicionais e de repetição em Python

13

Operadores de comparação

Operador	Descrição
<	Menor
<=	Menor ou igual
>	Maior
>=	Maior ou igual
==	Igual
!=	Diferente

Fonte: Autor

14

Operadores lógicos

P	Q	P and Q	P or Q	not A
True	True	True	True	False
True	False	False	True	False
False	True	False	True	True
False	False	False	False	True

Fonte: Autor

True = Verdadeiro
False = Falso

15

Estruturas condicionais em Python

```
if <condição>:
    <bloco de código endentado>
<instrução não endentada>
```

```
if <condição>:
    <bloco de código endentado 1>
else:
    <bloco de código endentado 2>
<instrução não endentada>
```

16

Estruturas condicionais em Python

```
if <condição1>:
    <bloco de código endentado 1>
elif <condição2>:
    <bloco de código endentado 2>
elif <condição3>:
    <bloco de código endentado 3>
else:
    <último bloco de código endentado>
<bloco de código não endentado>
```

17

Estruturas de repetição em Python

```
for <variável> in <sequência>:
    <bloco de código endentado>
<instrução não endentada>
```

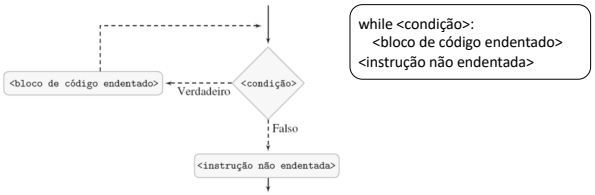
```
>>> for i in range(10):
        print(i, end=' ')

0 1 2 3 4 5 6 7 8 9
```

Fonte: Perkovic (2016)

18

Estruturas de repetição em Python



Fonte: Perkovic (2016)

```
while <condição>:
    <bloco de código endentado>
<instrução não endentada>
```

19

Resolução da SP

Teste II

20

No teste II foi pedido que você elabore um script para o cálculo do salário, sendo requisitado ao usuário a porcentagem de imposto para o cálculo.

21

Interação

Exercício de Fixação

22

Marque V Para verdadeiro e F para falso:

( ) O Python só pode ser utilizado para aplicações de inteligência artificial;

( ) Para declarar variáveis em Python não é necessário declarar o seu tipo;

( ) Python é uma linguagem que não utiliza delimitadores e por este motivo, seu código é confuso.

23

Conceitos

Funções em Python

24

Funções Built-in

int()	**float()**	**range()**
sum()	**bool()**	**type()**
pow()	**max()**	**print()**
set()	**len()**	**list()**

Fonte: Autor

25

Função definida pelo usuário

Blocos de código que realizam tarefas que normalmente precisam ser executadas diversas vezes dentro da aplicação.

```
def NomeDaFuncao(arg1, arg2, argn):
        <codigo>
        return NomeDoObjetoARetornar
```

26

Função definida pelo usuário

```
def Soma(X, Y):
    R = X + Y
    return R

a = int(input("Digite um valor para a: "))
b = int(input("Digite um valor para b: "))
c = int(input("Digite um valor para c: "))

s = Soma(a, b)
print("a + b = {0}".format(s))
s = Soma(a, c)
print("a + c = {0}".format(s))
s = Soma(b, c)
print("b + c = {0}".format(s))
print("Fim do Programa")
```

Fonte: Banin (2018)

27

Escopo de funções

- Ambiente externo à função: será chamado de Global.
- Ambiente interno à função: será chamado de Local.

28

```
def EstudaEscopo():
    Y = X * 2
    print("X global existe dentro função: valor = {0}".format(X))
    print("Y local existe dentro função: valor = {0}".format(Y))

print("Início do Programa")
X = 10
print("X global existe fora da função: valor = {0}".format(X))
EstudaEscopo()
print("Fim do Programa")
```

Fonte: Banin (2018)

29

Conceitos

Funções com Parâmetros

30

Funções com parâmetros

Os parâmetros representam dados de entrada a serem utilizados pela função e são opcionais.

```
def LerInteiro():
    n = int(input("Digite um número inteiro: "))
    return n

x = LerInteiro()
print("Valor lido na função = {0}".format(x))
```

Fonte: Banin (2018)

31

Funções com parâmetros

Valores Padrão: Os parâmetros podem apresentar valores-padrão atribuídos na definição da função.

```
def Soma(X, Y = 1):
    R = X + Y
    return R

print("Início do Programa")
a = int(input("Digite um valor para a: "))
b = int(input("Digite um valor para b: "))
s = Soma(a, b)
```

Fonte: Banin (2018)

32

Funções com parâmetros

```
def Soma(X, Y):
...      ↑  ↑
s = Soma(a, b)
```

```
s = Soma(Y = b, X = a)
```

⇩

Parâmetros Nomeados

33

Empacotamento e desempacotamento de parâmetros

É preciso utilizar o operador "*" para informar ao interpretador que a lista deve ser desempacotada.

```
>>> def Soma(*valores):
        r = 0
          for i in valores:
           r += i
          return r

>>> Soma(3, 9)
12
>>> Soma(1, 2, 3, 4)
10
```

Fonte: Banin (2018)

34

Empacotamento e desempacotamento de parâmetros

```
>>> def ExibeFormatado(a, b, c):
        print("1° valor = {}".format(a))
        print("2° valor = {}".format(b))
        print("3° valor = {}".format(c))

>>> L = [31, 77, 193]
>>> ExibeFormatado(*L)
1° valor = 31
2° valor = 77
3° valor = 193
>>> L = [43, 22, 323, 31]
>>> ExibeFormatado(*L)
Traceback (most recent call last):
  File "<pyshell#32>", line 1, in <module>
    ExibeFormatado(*L)
```

Fonte: Banin (2018)

35

Funções Anônimas

- Expressão Lambda;
- Funções que não possuem nome;
- Retorno é implícito;

lambda <argumentos> **:** <expressão>

36

Funções Anônimas

```
media = lambda valores : sum(valores) / len(valores)
```

```
def media(valores):
    return sum(valores) / len(valores)
```

37

Resolução da SP

Teste III

38

Elabore uma função em Python que calcule para um mesmo salário diversos valores com imposto descontado, utilizando o comando while e função.

39

Resolução da SP

Exercício Proposto

40

Interação

Dúvidas?

41

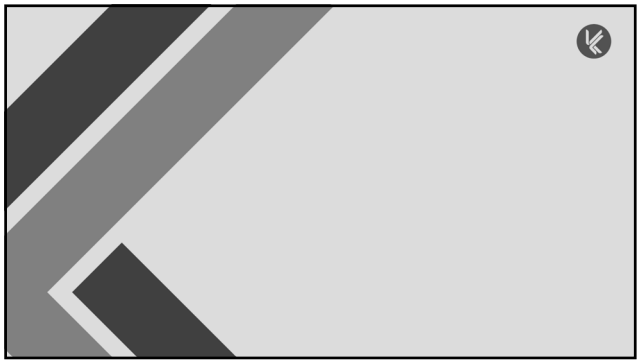
Conceitos

Recapitulando

42

- Características da Linguagem Python;
- Declaração de variáveis;
- Condicionais: if, elif, else;
- Repetição: for e while;
- Funções;
- Funções anônimas.

43



44