Python - Aula 2

Variáveis e Operações básicas

Python - Variáveis

1) Declaração e atribuição

Declarar uma variável é criar um nome simbólico para um endereço de memória, que armazenará um valor.

Python - Variáveis

1) Declaração e atribuição

Declarar uma variável é criar um nome simbólico para um endereço de memória, que armazenará um valor.

Atribuição é definir um valor para a variável declarada.

Python - Variáveis

Em Python, a declaração e a atribuição acontecem simultaneamente:

```
>>>
>>> a = 10
>>> frase = "Vovó foi a feira"
>>> valor = 12.99
>>> c, d = 15, 20
>>>
```

Python - Variáveis

Uma variável não pode ser utilizada em uma expressão sem ter sido inicializada:

```
>>> liquido = bruto - imposto
```

Python - Variáveis

Uma variável não pode ser utilizada em uma expressão sem ter sido inicializada:

```
>>> liquido = bruto - imposto
```

Ambas as variáveis precisam ser previamente declaradas, caso contrário o seguinte erro acontecerá:

```
>>> liquido = bruto - imposto
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'bruto' is not defined
>>>
```

Python - Tipos de variáveis

2) O comando **type**

O comando **type** pode ser utilizado para visualizar o tipo da variável:

```
>>> type(a)
<class 'int'>
>>> type(frase)
<class 'str'>
>>> type(valor)
<class 'float'>
```

Python - Tipos de variáveis

3) Alterando o tipo de uma variável

Em Python não é preciso especificar o tipo da variável na declaração, pois a linguagem possui **inferência de tipos**.

Python - Tipos de variáveis

3) Alterando o tipo de uma variável

Em Python não é preciso especificar o tipo da variável na declaração, pois a linguagem possui **inferência de tipos**

A linguagem possui **tipagem automática**, assim podemos alterar o tipo de uma variável apenas alterando o valor dela:

Python - Tipos de variáveis

3) Alterando o tipo de uma variável

Em Python não é preciso especificar o tipo da variável na declaração, pois a linguagem possui **inferência de tipos**

A linguagem possui **tipagem automática**, assim podemos alterar o tipo de uma variável apenas alterando o valor dela:

```
>>>
>>> a = 10                          #Tipo inteiro
>>> a = "Vovó foi a feira"          #Alterado de inteiro para String
>>> a = 12.99                       #Alterado de String para Float
>>>
```

Python - Tipos de variáveis

Obs.: O **#** em Python é utilizado para acrescentar comentários no código

3) Alterando o tipo de uma variável

Em Python não é preciso especificar o tipo da variável na declaração, pois a linguagem possui **inferência de tipos**

A linguagem possui **tipagem automática**, assim podemos alterar o tipo de uma variável apenas alterando o valor dela:

```
>>>
>>> a = 10                          #Tipo inteiro
>>> a = "Vovó foi a feira"          #Alterado de inteiro para String
>>> a = 12.99                       #Alterado de String para Float
>>>
```

Python - Tipos de variáveis

3) Alterando o tipo de uma variável

O Python possui **tipagem forte**, evitado assim várias operações que no geral não faria sentido:

```
>>> a = "10"          #String
>>> b = 10            #Inteiro
>>> a + b             #Soma de String com Inteiro
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: must be str, not int
```

Python - Tipos de variáveis

3) Alterando o tipo de uma variável

Nesse caso é necessário converter **explicitamente** a variável antes da operação:

```
>>> a + str(b)
'1010'
>>> int(a) + b
20
>>>
```

Python - Tipos de variáveis

3) Alterando o tipo de uma variável

É importante observar que conversões explícitas nem sempre são possíveis:

```
>>> a = "Vovó foi a feira"
>>> int(a)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ValueError: invalid literal for int() with base 10: 'Vovó foi a feira'
>>>
```

Python - Tipos de variáveis

4) Tipos de dados

x = “ ” → tipo **str** → **str()**

x = 1 → tipo **int** → **int()**

x = 1.0 → tipo **float** → **float()**

x = True → tipo **bool** → **bool()**

x = None → indefinido (**null**)

Python - Tipos de variáveis

4) Tipos de dados

x = [] ⇒ tipo **list** ⇒ **list()**

x = () ⇒ tipo **tuple** ⇒ **tuple()**

x = {} ⇒ tipo **dict** ⇒ **dict()**

Python - Operações básicas

5) Operadores

- **Relacionais**
- **Lógicos**
- **Aritméticos**
- **Bit a Bit**

Python - Operações básicas

5.1) Operadores Relacionais

Operador	Descrição	Exemplo
>	Maior que	3 > 2
<	Menor que	2 < 3
>=	Maior ou igual que	2 >= 2
<=	Menor ou igual que	3 <= 3
!=	Diferente de	True != False
==	Igual a	4 == 4

Python - Operações básicas

A resposta de uma operação relacional é **SEMPRE** um Booleano (True ou False)

5.1) Operadores Relacionais

Operador	Descrição	Exemplo
>	Maior que	3 > 2
<	Menor que	2 < 3
>=	Maior ou igual que	2 >= 2
<=	Menor ou igual que	3 <= 3
!=	Diferente de	True != False
==	Igual a	4 == 4

Python - Operações básicas

5.2) Operadores Lógicos

Operador	Descrição	Exemplo
not	Não	not True
and	E	True and False
or	Ou	True or False

Python - Operações básicas

A resposta de uma operação lógica é **SEMPRE** um Booleano (True ou False)

5.2) Operadores Lógicos

Operador	Descrição	Exemplo
not	Não	not True
and	E	True and False
or	Ou	True or False

Python - Operações básicas

5.3) Operadores Aritméticos

Operador	Descrição	Exemplo
+	Soma	2 + 3
-	Subtração	5 - 4
*	Multiplicação	3 * 4
/	Divisão	6 / 3
//	Divisão inteira	5 // 4
**	Exponenciação	10 ** 2
%	Resto da Divisão	10 % 2

Python - Operações básicas

5.4) Operadores Bit a Bit

Operador	Descrição	Exemplo
~	Complemento	~b
<<	Deslocar bits a esquerda	b << 1
>>	Deslocar bits a direita	b >> 1
&	E	b & 0x01
^	Ou exclusivo	b ^ 0x01
|	Ou	b | 0x01

Python - Operações básicas

5.5) Precedência em operações matemáticas

- **Parêntesis**
- **Exponenciação**
- **Multiplicação e Divisão**
- **Adição e Subtração**

De acordo com o padrão de regras matemáticas.

Python - Operações básicas

5.5) Precedência em operações matemáticas

Atenção. Em:

```
>>>
>>> 2 ** 2 ** 3
256
>>>
```

A exponenciação mais a direita acontece **primeiro**. Para alterar a ordem é necessário o uso de parêntesis:

```
>>>
>>> (2 ** 2) ** 3
64
>>>
```