Programação de Aplicativos

Professor: Euclides Paim
euclidespaim@gmail.com

Lógica de Programação

Sintaxe, Variáveis e Tipos de Dados

Professor: Euclides Paim

euclidespaim@gmail.com

Programação de Aplicativos

Sintaxe, Variáveis e Tipos de Dados

Sumário

- Sintaxe do Python
- Indentação em Python
- Variáveis em Python
- Comentários em Python
- Tipos de Dados

Programação de Aplicativos

Sintaxe, Variáveis e Tipos de Dados

Sintaxe

- Como vimos o Python pode ser executado direto na linha de comando:

```
>>> print("Hello, World!")
Hello, World!
```

Sintaxe

- Ou criando um arquivo Python no servidor, usando a extensão **.py**, e executando-o na linha de comando:

```
C:\Users\Your Name>python myfile.py
```

Programação de Aplicativos

Sintaxe, Variáveis e Tipos de Dados

Identação

Identação se refere aos **espaços** no início de uma linha de código.

- Enquanto em outras linguagens de programação a identação serve apenas para melhorar a legibilidade, no Python ela é **muito importante**.

- O Python usa a identação para indicar um **bloco de código.** Ex.:

```
if 5 > 2:
  print("Cinco é maior que dois!!")
```

Programação de Aplicativos

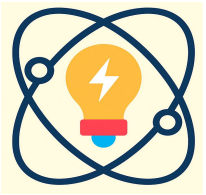
Sintaxe, Variáveis e Tipos de Dados

Variáveis em Python

Em Python as variáveis são criadas quando atribuímos valores a elas.

```
x = 5
y = "Hello, World!"
```

O Python não possui um comando específico para declarar uma variável.



Programação de Aplicativos

Sintaxe, Variáveis e Tipos de Dados

Variáveis

Variáveis são contêineres para armazenar valores de dados.

As variáveis **não** precisam ser declaradas com nenhum tipo específico e podem até alterar o tipo depois de terem sido definidas.

```
x = 5
y = "John"
print(x)
print(y)
```

Casting

Se quisermos especificar o **tipo de dado** de uma variável, podemos usar **casting** (conversão de tipo).

Podemos obter o tipo de dados de uma variável com a função `type()`.

```
x = str(3)    # x será '3'
y = int(3)    # y será 3
z = float(3)  # z será 3.0
print(type(x))
```

Programação de Aplicativos

Sintaxe, Variáveis e Tipos de Dados

Case-Sensitive

Os nomes de variáveis em Python diferenciam maiúsculas de minúsculas (são *case-sensitive*).

Por exemplo, `idade`, `Idade` e `IDADE` são três variáveis diferentes.

```
a = 4
A = "John"

#A não irá sobrescrever a
```

Programação de Aplicativos

Sintaxe, Variáveis e Tipos de Dados

Nomes de Variáveis

Uma variável pode ter um nome curto (como `x` e `y`) ou um nome mais descritivo (como `idade`, `nome_do_carro`, `volume_total`).

Regras para nomes de variáveis em Python:

- Um nome de variável deve começar com uma letra ou com o caractere sublinhado (`_`)
- Um nome de variável **não pode** começar com um número
- Um nome de variável pode conter apenas caracteres alfanuméricos e sublinhados (`A-Z`, `a-z`, `0-9`, e `_`)
- Nomes de variáveis diferenciam maiúsculas de minúsculas (`idade`, `Idade` e `IDADE` são variáveis diferentes)
- Um nome de variável **não pode ser** uma palavra reservada do Python (como `if`, `while`, `def`, etc.)

Programação de Aplicativos

Sintaxe, Variáveis e Tipos de Dados

Nomes de Variáveis com Múltiplas Palavras

Nomes de variáveis com mais de uma palavra podem ser difíceis de ler. Existem várias técnicas que você pode usar para torná-los mais legíveis:

- **Camel Case**

  Cada palavra, exceto a primeira, começa com letra maiúscula:

  ```
  myVariableName = "John"
  ```

- **Pascal Case**

  Cada palavra começa com letra maiúscula:

  ```
  MyVariableName = "John"
  ```

- **Snake Case**

  Cada palavra é separada por um caractere sublinhado (_):

  ```
  my_variable_name = "John"
  ```

Programação de Aplicativos

Sintaxe, Variáveis e Tipos de Dados

Tipos de Dados no Python

Em programação, **tipo de dado** é um conceito importante. As variáveis podem armazenar dados de diferentes tipos, e cada tipo permite fazer coisas diferentes. O Python possui os seguintes tipos de dados incorporados, organizados nas categorias abaixo:

- **Tipo de Texto:** `str`
- **Tipos Numéricos:** `int`, `float`, `complex`
- **Tipos de Sequência:** `list`, `tuple`, `range`
- **Tipo de Mapeamento:** `dict`
- **Tipos de Conjunto:** `set`, `frozenset`
- **Tipo Booleano:** `bool`
- **Tipos Binários:** `bytes`, `bytearray`, `memoryview`
- **Tipo Nulo:** `NoneType`

Programação de Aplicativos

Sintaxe, Variáveis e Tipos de Dados

Números em Python

Existem três tipos numéricos em Python:

- `int` – números inteiros, como `1`, `100`, `-5`
- `float` – números de ponto flutuante (decimais), como `3.14`, `-0.5`, `2.0`
- `complex` – números complexos, como `2 + 3j`

As variáveis com esses tipos são criadas automaticamente ao se atribuir um valor a elas.

Conversão de Tipos (Type Conversion)

Você pode converter de um tipo numérico para outro usando os métodos `int()`, `float()` e `complex()`:

```
x = 1    # int
y = 2.8  # float

#convertendo de int para float:
a = float(x)

#convertendo de float para int:
b = int(y)

print(type(a))
print(type(b))
```

Programação de Aplicativos

Sintaxe, Variáveis e Tipos de Dados

Exercícios

1. Leitura e Exibição de Dados

Crie um programa que solicite ao usuário seu nome e idade, armazene cada informação em uma variável adequada e, em seguida, exiba a mensagem:

```
"Olá, [nome]! Você tem [idade] anos."
```

Programação de Aplicativos

Sintaxe, Variáveis e Tipos de Dados

Exercícios

2. Crie um programa que solicite dois números ao usuário:

o número de produtos e
o preço do produto.

O programa deve calcular o valor total da compra e exibir:

"O total da sua compra é R$ [total]."

2. Conversão de Dados e Cálculo de IMC

Peça ao usuário para informar sua *altura* e *peso*. Em seguida, calcule o Índice de Massa Corporal (IMC) utilizando a fórmula:

IMC = peso / (altura * altura)

Exiba o valor do IMC com duas casas decimais e uma mensagem indicando a classificação do IMC:

- **Abaixo do peso:** IMC abaixo de 18.5
- **Peso normal:** IMC entre 18.5 e 24.9
- **Sobrepeso:** IMC entre 25 e 29.9
- **Obesidade:** IMC acima de 30

Programação de Aplicativos

Sintaxe, Variáveis e Tipos de Dados

3. Cálculo da Média de Notas

Solicite ao usuário as notas de 4 disciplinas e calcule a média. Se a média for maior ou igual a 7, informe "Aprovado". Caso contrário, informe "Reprovado".

Condições e Instruções *If* em Python

O Python suporta as condições lógicas usuais da matemática:

- Igual a: `a == b`
- Diferente de: `a != b`
- Menor que: `a < b`
- Menor ou igual a: `a <= b`
- Maior que: `a > b`
- Maior ou igual a: `a >= b`

Essas condições podem ser usadas de várias formas, sendo mais comuns nas instruções **"if"** e em **laços (loops)**.

Uma instrução "if" é escrita usando a palavra-chave `if`.

Programação de Aplicativos

Sintaxe, Variáveis e Tipos de Dados

O que é o `if` em Python?

O `if` é usado para **executar um bloco de código apenas se uma determinada condição for verdadeira**.

A sintaxe básica é:

```
if condição:
  #bloco de código a ser executado se a condição for verdadeira
```

Programação de Aplicativos

Sintaxe, Variáveis e Tipos de Dados

Exemplo `if` em Python?

```
a = 33
b = 200
if b > a:
  print("b é maior que a")
```

Neste exemplo, usamos duas variáveis, `a` e `b`, que são utilizadas como parte da instrução `if` para testar se `b` é maior que `a`. Como `a` é 33 e `b` é 200, sabemos que 200 é maior que 33, e por isso exibimos na tela a mensagem: **"b é maior que a"**.

Identação O Python conta usa identação (espaço em branco no início de uma linha) para definir o escopo do código. Outras linguagens de programação geralmente usam *chaves* para esse fim.

elif

A palavra-chave `elif` é a forma que o Python utiliza para dizer:

"se as condições anteriores não forem verdadeiras, então tente esta condição."

```
a = 33
b = 33
if b > a:
  print("b é maior que a")
elif a == b:
  print("a e b são iguais")
```

Neste exemplo, **a é igual a b**, portanto a primeira condição não é verdadeira, mas a condição `elif` é verdadeira, então **imprimimos na tela que "a e b são iguais"**.

`else`

A palavra-chave `else` captura **qualquer coisa que não tenha sido atendida pelas condições anteriores**.

```
a = 200
b = 33
if b > a:
  print("b é maior que a")
elif a == b:
  print("a e b são iguai")
else:
  print("a é maior que b")
```

Neste exemplo, **a é maior que b**, portanto a primeira condição não é verdadeira, e a condição `elif` também não é verdadeira, então vamos para a condição `else` e **imprimimos na tela que "a é maior que b"**.

Obs.: Também podemos usar um `else` **sem o** `elif`.

Exercício: Verificação de Senha

Enunciado:

Crie um programa que simule um sistema de login simples. Peça ao usuário que digite uma senha. Se a senha for igual a `"ifc2025"`, exiba a mensagem **"Acesso permitido"**. Caso contrário, exiba **"Senha incorreta. Tente novamente."**

Referências

Referências Básicas

CORMEN, Thomas H.; LEISERSON, Charles E.; RIVEST, Ronald L.; STEIN, Clifford. ***Algoritmos: teoria e prática*****.** 3. ed. Rio de Janeiro: Elsevier, 2013.

SEBESTA, Robert W. ***Conceitos de linguagens de programação***. 10. ed. São Paulo: Pearson, 2018.

MANZANO, José Augusto N. G.; OLIVEIRA, Jayr Figueiredo de. ***Algoritmos: lógica para desenvolvimento de programação de computadores*****.** 27. ed. São Paulo: Érica, 2016.

DOWNEY, Allen B. ***Pense em Python: como pensar como um cientista da computação***. Tradução de Cássio de Souza Costa. 2. ed. São Paulo: Novatec, 2016.

Referências Complementares

IEPSEN, Edécio Fernando. ***Lógica de programação e algoritmos com JavaScript***. 2. ed. São Paulo: Novatec, 2022.

Referências na Internet

https://www.w3schools.com/python/