# Introdução ao Python

## Algoritmos e Programação
## Introdução ao Python
## Variáveis – Expressões – Funções

Prof. Fabio Fernando Kobs, Dr.

# Agenda:

- **Introdução ao Python;**
- **Variáveis;**
- **Expressões;**
- **Funções;**
- **Exercícios.**

# Python

- Python é uma linguagem de programação criada por Guido van Rossum em 1991.
- Objetivos do projeto da linguagem: produtividade e legibilidade. Em outras palavras, é uma linguagem criada para produzir código bom e fácil de manter de maneira rápida.
- Características da linguagem que ressaltam esses objetivos:
  - baixo uso de caracteres especiais, o que torna a linguagem muito parecida com pseudocódigo;
  - uso de identação para marcar blocos;
  - quase nenhum uso de palavras-chave voltadas para a compilação;
  - coletor de lixo para gerenciar automaticamente o uso da memória.
- Fácil traduzir o raciocínio em um algoritmo;
- Python Brasil (https://python.org.br/).

# Python – Instalação (gratuito)

- Pacote principal (baixar e instalar)

  https://python.org.br/ menu Inicie-se

  *Depois de instalado o pacote acima...*

- IDE (Ambiente Integrado de Desenvolvimento)

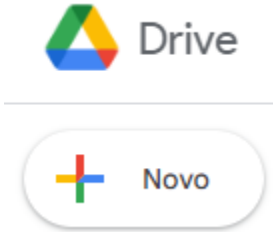
  PyCharm (https://www.jetbrains.com/pycharm/)

  (e instalar a versão Community)

- **Python *online*** *(sem instalação em seu computador)*:
  - Google colab
    https://colab.research.google.com/

# Python – Google Colaboratory

## Acessando / Acrescentando o Colab




- A partir de uma conta Google, acessar o Drive;
- Clicar em “Novo”;
- Clicar em “Mais” (*seguir ao passo seguinte caso não apareça o Google Colaboratory*);
- Clicar em “Conectar mais apps”;
- Busque por *Colaboratory* e clique em “instalar” (concordar e fazer o *login*, autenticar via SMS, ...).

# Conceitos Iniciais

## Objetos e Tipos de Dados:

- Um objeto em linguagem de programação representa a posição na memória onde será armazenado. Os objetos em Python apresentam os seguintes atributos:
  - Tipo: O tipo de um objeto determina os *valores que o objeto pode receber e as operações que podem ser executadas nesse objeto*, por exemplo, número inteiro.
  - Valor: O valor de um objeto é o *índice de memória ocupada por essa variável*, determinado pelo tipo da variável (que é onde o dado estará armazenado).
  - Tempo de vida: A vida de um objeto é o *intervalo de tempo de execução de um programa* em Python.

# Conceitos Iniciais

Nomes:

- A fim de utilizar um objeto em um programa em Python, esse objeto deve ter um nome.
- O nome de um objeto é uma **variável** usada para identificar esse objeto em um programa.

  Exemplo: i = 57

  Esta indicação **cria um objeto com nome *i*** e liga vários atributos com esse objeto, como o índice (endereço) de memória, o seu tipo e o dado armazenado. Neste exemplo, o tipo do objeto é ***int*** (inteiro) e o valor alocado na memória na variável *i* é **57**.

# Variável

- Ao nomear as variáveis, se faz necessário seguir algumas regras (podem ser diferentes para cada linguagem).

- Regras:
  - O primeiro caractere deve ser obrigatoriamente uma **letra**;
  - Se houver mais de um caractere, só poderá usar: **letra**, **algarismo** ou o **símbolo sublinha** ( **_** );
  - Nomes de variáveis escritas com letras maiúsculas serão diferentes de letras minúsculas, por exemplo, média é diferente de MÉDIA.
  - Nenhuma palavra reservada poderá ser nome de uma variável;
  - Não é permitido espaço no nome da variável.

# Variável

Exemplos de declaração:

| Nomes Válidos | Nomes Inválidos |
|---|---|
| Media, alt, a2, PESO, salario_medio, Nota1, Nota2, nota1, nota_1, PesoAluno, peso_do_aluno | 2w<br>Media*aluno<br>peso do aluno<br>salario médio<br>Nota 1<br>Nota-1 |

# Variável

Tipos:

***Numérica***

- Variáveis numéricas são aquelas que armazenam dados numéricos, podendo ser divididas em dois tipos:
- ***int***
  - ✓ Os números inteiros são aqueles que **não possuem componentes decimais ou fracionários**, podendo ser positivos ou negativos.
  - ✓ As variáveis compostas com esses números são chamadas de **VARIÁVEIS INTEIRAS**.
  - ✓ Exemplos: -12; 2021; -25000; 32767.

# Variável

***Numérica***

- ***float***
  - ✓ Os números de ponto flutuante são aqueles que **podem possuir componentes decimais ou fracionários**, podendo também ser positivos ou negativos.
  - ✓ As variáveis compostas com estes números pertencentes aos conjuntos dos números reais são chamadas de **VARIÁVEIS FLOAT** (ponto flutuante).
  - ✓ Exemplos: -23.01 número *float* negativo com duas casas decimais

    0.0 número *float* com uma casa decimal
  - ✓ **Utilizar ponto como separador entre a parte inteira e fracionária**.
  - ✓ Não utilizar nada como separador de milhar. Exemplo: ~~1.000.000~~ (um milhão) é escrito 1000000.

# Variável

Exercício: Complete marcando inteiro (*int*) ou ponto flutuante (*float*):

| Número | Tipo numérico | |
|---|---|---|
| 5 | ( ) int | ( ) float |
| 5.0 | ( ) int | ( ) float |
| 4.3 | ( ) int | ( ) float |
| -2 | ( ) int | ( ) float |
| 100 | ( ) int | ( ) float |
| 1.333 | ( ) int | ( ) float |

# Variável

## Tipos de Variáveis:

***Lógico***

- Também conhecido como booleano. É representado no algoritmo pelos dois únicos valores lógicos possíveis: verdadeiro ou 1 (***True***) ou falso ou 0 (***False***).

  Exemplo em Python:

  resultado = **True**

  aprovado = **False**

# Variável

Tipos de Variáveis:

***str (string)***

- Armazena cadeias de caracteres (sequência de símbolos como **letras, números, sinais de pontuação, espaços** etc.) como nomes e textos.

  Exemplo: ~~João e Maria~~.

- Para possibilitar a separação entre o texto do programa e o conteúdo de uma *string*, usar aspas (“) para delimitar o início e o fim da sequência de caracteres, ou apóstrofos (‘).

  Exemplo: “João e Maria” ou ‘João e Maria’.

# Expressões

- **Conjunto de variáveis e constantes numéricas que relacionam-se por meio de operadores**, compondo uma fórmula, que uma vez avaliada, resulta num valor.

*Exemplo*:

Media = (Nota1 + Nota2) / 2

onde:

*Media é a variável que receberá o resultado da expressão;*

*Nota1 e Nota2 são as variáveis com as notas;*

*+ e / são os operadores;*

*2 é a constante numérica.*

# Expressões

As expressões dividem-se em:

**1. ARITMÉTICAS**

- Expressões aritméticas são aquelas cujo **resultado da avaliação é do tipo numérico**, seja ele inteiro ou ponto flutuante.
- Somente o uso de operadores aritméticos e **variáveis numéricas** é permitido em expressões deste tipo.

# Expressões

*Soma*

- Na matemática, representada pelo sinal + e, em expressões em termos computacionais, pelo mesmo sinal. Exemplo:

| | |
|---|---|
| A + B | Expressão que simboliza a soma do valor de duas variáveis. |
| 2 + 3 | Nessa expressão, o valor retornado é a soma dos valores dados, isto é, 5. |

# Expressões

*Subtração*

- Na matemática, representada pelo sinal – e, em expressões em termos computacionais, pelo mesmo sinal. Exemplo:

| | |
|---|---|
| A – B | Expressão que simboliza a subtração do valor de duas variáveis. |
| 3 – 2 | Nessa expressão, o valor retornado é o resto, isto é, 1. |

# Expressões

*Multiplicação*

- Na matemática, representada pelos sinais x ou . e, em expressões em termos computacionais, pelo sinal *****. Exemplo:

| | |
|---|---|
| B * D | Expressão que simboliza a multiplicação do valor de duas variáveis. |
| 3 * 2 | Nessa expressão, o valor retornado é o produto dos valores dados, isto é, 6. |

# Expressões

*Divisão*

- Na matemática, representada pelo sinal ÷ e, em expressões computacionais, pelo sinal /. Exemplo:

| | |
|---|---|
| A / B | Expressão que simboliza a divisão do valor de duas variáveis. |
| 6 / 2 | Nessa expressão, o valor retornado é a divisão dos valores dados, que, no caso, será equivalente a 3. |

# Expressões

*Exponenciação*

- Na matemática, representada pela base e por um expoente e em expressões em termos computacionais pelo sinal **** e o número que se quer elevar. Exemplo:

| | |
|---|---|
| A ** 2 | Expressão que simboliza o valor da variável ao quadrado. |
| 2 ** 3 | Nessa expressão, o valor retornado é o resultado da exponenciação do valor 2 ao cubo, que no caso será equivalente a 8. |

# Expressões

*Resto*

- É usado em expressões em termos computacionais quando se deseja encontrar o **resto da divisão de dois *números inteiros***. Exemplo:

| | |
|---|---|
| K % Y | Expressão que simboliza a intenção de achar o resto da divisão do valor da variável K pelo valor da variável Y. |
| 5 % 2 | Nessa expressão, o valor retornado é o resto da divisão do primeiro pelo segundo número, que, no caso, será equivalente a 1. |

# Expressões

As expressões dividem-se em:

**2. RELACIONAIS**

- Uma expressão relacional, ou simplesmente relação, é uma **comparação realizada entre dois valores de mesmo tipo básico** (por ex.: *int*, *float*, *str*).

- Estes valores são representados na relação por meio de constantes, variáveis ou expressões aritméticas.

# Expressões

## Operadores Relacionais:

- Como exemplos de operadores relacionais matematicamente conhecidos tem-se:

| **Operação** | **Matemática** | **Operador em Python** |
|---|---|---|
| **Igualdade** | = | == ou ***is*** |
| **Diferente** | ≠ | **!=** ou ***is not*** |
| **Maior que** | > | > |
| **Menor que** | < | < |
| **Maior ou igual a** | ≥ | >= |
| **Menor ou igual a** | ≤ | <= |

# Expressões

## Exemplo do uso de operadores relacionais:

```
>>> a = 1        # a recebe 1
>>> b = 5        # b recebe 5
>>> a == b       # a é igual a b?
False
>>> b > a        # b é maior que a?
True
>>> a < b        # a é menor que b?
True
>>> a >= b       # a é maior ou igual a b?
False
>>> a != b       # a é diferente de b?
True
```

# Expressões

As expressões dividem-se em:

**3. LÓGICAS**

- Denomina-se expressão lógica a expressão cujos operadores são lógicos.
- Python suporta sobretudo três operadores básicos: ***not*** (não), ***and*** (e), ***or*** (ou).

# Expressões

## Operadores Lógicos:

- Tem-se:

| Operador | Matemática | Notação em Python | Ordem de hierarquia |
|---|---|---|---|
| **Negação** | **não** | ***not*** | **1º** |
| **Conjunção** | **e** | ***and*** | **2º** |
| **Disjunção** | **ou** | ***or*** | **3º** |

# Expressões

## Operador ***not***

- A operação de negação também é chamada de inversão, pois um valor verdadeiro negado se torna falso e vice-versa.
- Exemplo de tabela verdade do operador ***not***:

| X | not X |
|---|---|
| True | False |
| False | True |

# Expressões

## Operador ***and***

- Resulta verdadeiro apenas quando seus dois operadores forem verdadeiros.
- Exemplo da tabela verdade do operador ***and***:

| X | Y | X and Y |
|---|---|---|
| V | V | V |
| V | F | F |
| F | V | F |
| F | F | F |

# Expressões

## Operador ***or***

- Resulta falso apenas quando seus dois operadores forem falsos.
- Exemplo da tabela verdade do operador ***or***:

| X | Y | X or Y |
|---|---|---|
| V | V | V |
| V | F | V |
| F | V | V |
| F | F | F |

# Exercício: Complete a tabela utilizando a = True, b = False e c = True.

| Expressão | Resultado | |
|---|---|---|
| a and a | ( ) True | ( ) False |
| b and b | ( ) True | ( ) False |
| not c | ( ) True | ( ) False |
| not b | ( ) True | ( ) False |
| not a | ( ) True | ( ) False |
| a and b | ( ) True | ( ) False |
| b and c | ( ) True | ( ) False |
| a or c | ( ) True | ( ) False |
| b or c | ( ) True | ( ) False |
| c or a | ( ) True | ( ) False |
| c or b | ( ) True | ( ) False |
| c or c | ( ) True | ( ) False |
| b or b | ( ) True | ( ) False |

# Expressões

## Critérios de precedência dos operadores

A seguir, relacionam-se os critérios de precedência dos operadores. *Se precisar alterar esta hierarquia, deve-se usar os parênteses*.

| Hierarquia | |
|---|---|
| **Primeiro** | **Parênteses e funções** |
| **Segundo** | **Potência e resto** |
| **Terceiro** | **Multiplicação e divisão** |
| **Quarto** | **Adição e subtração** |
| **Quinto** | **Operadores relacionais** |
| **Sexto** | **Operadores lógicos** |

# Expressões

Exemplos de expressões Relacionais:

- Considere **a**, **b** e **c** variáveis numéricas.
- Como exemplos de expressões lógicas tem-se:
  - a + b == 0 **and** c **!=** 1
  - Essa expressão verifica se o resultado da soma dos valores das variáveis **a** e **b** é igual a **0** e (and) se o valor da variável **c** é diferente de **1**. O resultado será considerado **True** se as duas expressões relacionais forem **verdadeiras**.

# Exercício:

**1)** Escreva o resultado das expressões:

**a)** *True* ***or*** *False* ***and not*** *True*

**b)** Para salário = 100 e idade = 20:

*salario > 1000* ***and*** *idade > 18*

**c)** Para salário = 2000 e idade = 30:

*salario > 1000* ***and*** *idade > 18*

**2)** Escreva uma expressão lógica para determinar se uma pessoa deve ou não pagar imposto. Considere que pagam impostos pessoas cujo salário é maior ou igual que R$ 10.200,00.

**3)** Calcule o resultado da expressão **A > B and C or D**, utilizando os valores da tabela a seguir.

| A | B | C | D | Resultado |
|---|---|---|---|---|
| 1 | 2 | True | False | |
| 10 | 3 | False | False | |
| 5 | 1 | True | True | |

**4)** Escreva uma expressão lógica que será utilizada para decidir se um aluno foi ou não aprovado. Para ser aprovado, a média do aluno deve ser maior ou igual a 7.

# Funções – ***print*()**

- A função ***print*()** informa que será exibido algo na tela, ou seja, a função exibe uma mensagem na tela do computador.
- Sempre que desejar mostrar algo para o usuário, como uma mensagem ou o resultado de uma operação de cálculo, usa-se a função ***print*()**.
- Exemplo:

  print(“Oi”)  # ou  print(‘Oi’)

- Os parênteses são utilizados para separar os parâmetros de uma função, e as aspas (ou apóstrofos) para indicar o início e o fim do texto da mensagem.

# Funções – **Tamanho** de uma *string*

- Pode ser obtido utilizando-se a função ***len*()**, retornando um valor do tipo inteiro.
- Se a *string* é vazia (representada por “”, ou seja, duas aspas sem nada entre elas), seu tamanho é igual a zero.
- Exemplos:

```
>>> print (len(“A”))
1
>>> print (len(“AB”))
2
>>> print (len(“”))
0
>>> print (len(“O rato roeu a roupa”))
____
```

# Funções – **Concatenação**

- O conteúdo de variáveis *string* podem ser somados, ou melhor, concatenados (juntados), utilizando o operador de adição (+). Assim, “AB” + “C” é igual a “ABC”.
- Exemplos:

```
>>> s = “ABC”
>>> print (s + “C”)
ABCC
>>> print (s + “D” * 4)
ABCDDDD
>>> print (“X” + “-”*10 + “X”)


>>> print (s + “x4 = ” + s*4)
```

# Funções – **Composição**

- Indica a composição da *string* anterior com o conteúdo da variável X, e depois novamente com a *string*.
- O valor da variável X é buscado na memória e inserido na expressão.
- O método ***format*()** em Python formata uma *string* que contém campos entre chaves ‘{}’ que são substituídos pelos argumentos de *format*.
- **Atenção**, nomes de variáveis nunca possuem aspas.
- O **\n** dentro das aspas indica nova linha.
- Por exemplo, exibir que “João tem X anos”, onde X é uma variável numérica:

```
print('João tem', X, '\nanos’)
# ou
print('João tem {} \nanos'.format(X))
```

# Funções – **Composição**

- O método ***format*()** em Python formata uma *string* que contém campos entre chaves ‘{}’ que são substituídos pelos argumentos de *format*.
- Também se pode usar *strings* literais formatadas, comece uma *string* com **f** ou **F**, antes de abrir as aspas. Dentro dessa *string*, pode-se escrever uma expressão Python entre caracteres { e }, que podem se referir a variáveis, ou valores literais.
- Observe a diferença no exemplo para exibir que “João tem X anos”, onde X é uma variável numérica:

```
print('João tem {} anos'.format(X))
# ou
print(f'João tem {X} anos')
```

# Funções – **Composição**

- Exemplo com números decimais:

```
>>> print(‘{}’.format(5))          # ou   print(f‘{5}’)
5

>>> print('{:.2f}'.format(5))      # ou   print(f‘{5:.2f}’)
5.00
>>> print(f‘{‘João’} tem {22} anos e R$ {98.1:.2f} na carteira')
João tem 22 anos e R$ 98.10 na carteira

>>> nome = “João”
>>> idade = 22
>>> dinheiro = 98.1
>>> print(f‘{nome} tem {idade} anos e R$ {dinheiro:.2f} na carteira')
João tem 22 anos e R$ 98.10 na carteira
```

# Referências


- LOPES, Anita; GARCIA, Guto. **Introdução à programação**: 500 algoritmos resolvidos. Rio de Janeiro: Elsevier: Campus, 2002.
- MENEZES, Nilo Ney Coutinho. **Introdução à Programação com Python**: Algoritmos e lógica de programação para iniciantes. 2 ed. 5 reimp. São Paulo: Novatec, 2017.
- SCHILDT, Herbert. **C Completo e Total**. São Paulo: Editora Pearson Makron Books, 1997.

# Exercícios

1. Indique com um X quais dos dados seguintes são do tipo ***int***.

( ) 1000
( ) “0”
( ) “-900”
( ) True
( ) -234
( ) 23
( ) “sala de aula!”
( ) 0
( ) False
( ) -1.23

# Exercícios

2. Indique com um X quais dos dados seguintes são do tipo ***float***.

( ) -292
( ) “0.82”
( ) “-90.90”
( ) True
( ) -234
( ) -99.23
( ) “doze”
( ) 23.1242
( ) False
( ) -1.2233

# Exercícios

3. Indique com um X quais dos dados seguintes são do tipo ***string*** ou ***str***.

( ) “exercicio”
( ) “0.832”
( ) -90.90
( ) “True”
( ) True
( ) 1299.23
( ) ‘CINCO’
( ) 43.12123
( ) False
( ) -1.2233

# Exercícios

4. Indique com um X os nomes válidos para uma variável.

( ) endereco
( ) 21brasil
( ) fonem@
( ) nomeusuario
( ) nome*usuario
( ) ;nome
( ) 111
( ) END$A&*
( ) nome2
( ) e-mail
( ) e_mail

# Exercícios

5. Classifique os dados abaixo de acordo com seu tipo:

I = *int*  F = *float*  S = *str*  L = lógico  X = indefinido

| | | |
|---|---|---|
| ( ) 2 | ( ) -0.0001 | ( ) -0.0 |
| ( ) 3 | ( ) +0.05 | ( ) False |
| ( ) 0.0 | ( ) +3257 | ( ) 'True' |
| ( ) True | ( ) "a" | ( ) "abc" |
| ( ) -1 | ( ) "+3257" | ( ) Falso |
| ( ) -32 | ( ) "+3257.0" | ( ) 22 |
| ( ) "+36" | ( ) "-0.0" | ( ) "V" |
| ( ) +32.0 | ( ) false | ( ) 10.0 |

# Exercícios

6. Analise as expressões abaixo e diga qual a saída (***True*** ou ***False***):

| Expressão | Saída |
|---|---|
| **True or False** | |
| **False and False** | |
| **not False** | |
| **2.5 == 3.0** | |
| **3.0 == (6.0/2)** | |
| **2 != 4** | |
| **5 > 1** | |
| **7 <= 7** | |
| **6 != 5** | |
| **((True or False) or (2>3) or (2.5 == 3))** | |
| **(True and True) or (not True) and (False and True)** | |
| **((False or False) or (3>3) or (2.5 == 3.5))** | |
| **((True and True) or (2<3) or (2 >= 3))** | |
| **(False and True) or (not True) and (True and True)** | |
| **(True and False) or (not False) and (False or True)** | |