

Introdução ao Python

Prof. : José Luiz Vilas Boas

Introdução

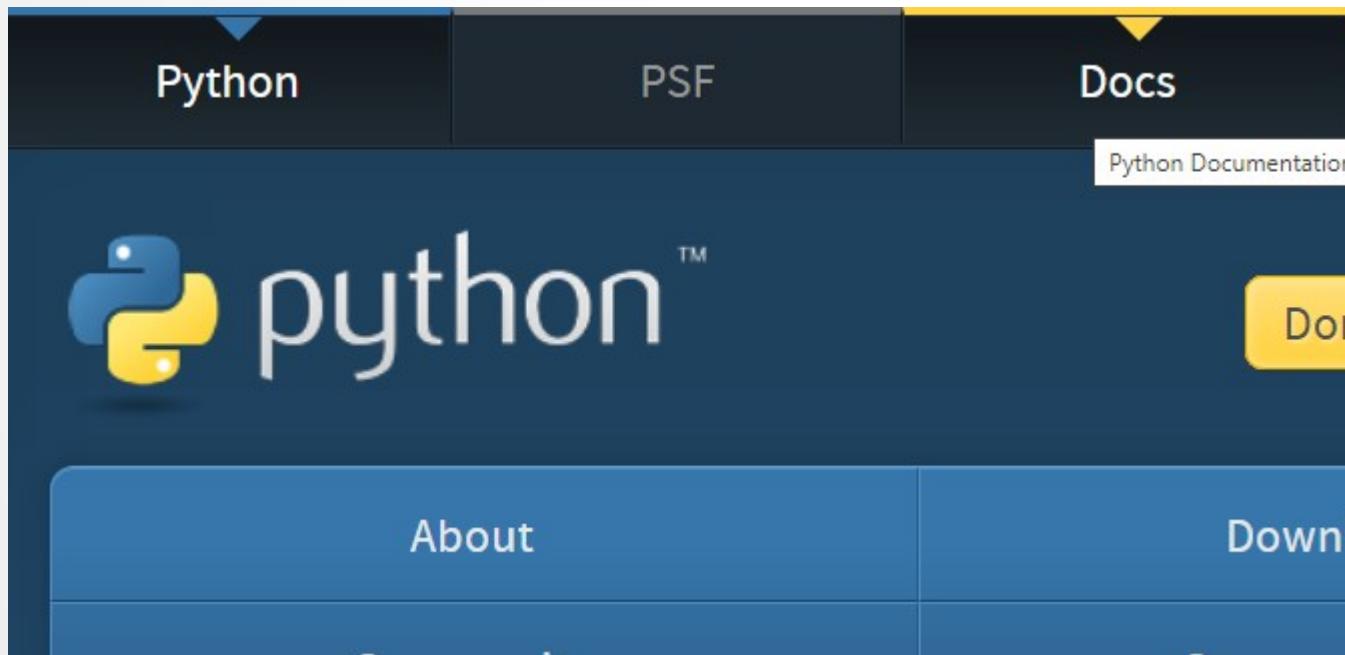
- Python é uma linguagem de programação generalistas, ou seja, pode ser utilizada para qualquer propósito.
- Em Python é possível programar para qualquer SO, web, mobile, etc..;
- Python pode-se criar parcial ou total aplicações YouTube, Google, Instagram, Dropbox, Pinterest, Spotify, Blender 3D, BitTorrent, etc....

Características

- Criado em 1991 por Guido Van Rossum;
- Simples e de fácil aprendizado;
- É Portátil;
- Pode-se desenvolver aplicações web;
- Muito utilizada para Inteligência Artificial;
- Big Data;
- Ciência de dados;
- Muito adequado para processamento de imagens.

Instalação

- URL [python.org](https://www.python.org);
- Inúmeras informações como documentação:



Instalação

- Versão para instalação no windows:

Help us raise \$60,000 USD by December 31!

Get Started

Whether you're new to programming or an experienced developer, it's easy to learn and use Python.

[Start with our Beginner's Guide](#)

Download

Python source code and installers are available for download for all versions!

[Latest: Python 3.9.0](#)

Aqui!!!

IDEs

Existem diversas IDEs disponíveis, contudo, irei apresentar apenas as três que eu creio que são as principais:

- Pycharm: IDE desenvolvida e mantida pela JetBrains. É uma excelente opção no sentido de que possui versão completamente gratuita e muito intuitiva. Disponível em : <<https://www.jetbrains.com/pt-br/pycharm/>>.
- Anaconda: É uma suite, ou seja, possui uma série de ferramentas(IDEs) dentre as principais o Spyder. Disponível em : <<https://www.anaconda.com/products/individual>>.

IDEs

- O Google Colaboratory: é um ambiente de notebooks Jupyter que não requer configuração e é executado na nuvem. Você pode programar em python usando todos os recursos de um servidor Google. Disponível em:
[<https://colab.research.google.com/notebooks/intro.ipynb>](https://colab.research.google.com/notebooks/intro.ipynb).

Variáveis

- Variáveis em um programa Python estão associadas com a posições de memória que armazenam informações;
- Nomes de variáveis podem ter vários caracteres;
- O primeiro caracter ter que ser uma letra;
- O restante pode ter letras, dígitos e subliquados.

Variáveis

- Exemplos de nomes de variáveis:

Corretos

Contador

teste23

Alto_alegre

teste

_teste

Incorretos

1Contador

oi!teste

Alto...alegre

-teste

Variáveis

- Palavras-chave em Python não podem ser utilizadas como nome de variáveis : int, if, do ...;
- Python é case sensitive:
- cont <> Cont <> CONT;

Palavras-chave em Python

and, as, assert, break, class, continue,
def, del, elif, else, except, exec,
finally, for, from, global, if, import,
in, is, lambda, nonlocal, not, or,
pass, raise, return, try, while, with,
yield, True, False, None.

Variáveis - Exercício

- Escolha uma das alternativas que possua nome válidos para variável:
 - a) if, a_b, J3, _teste
 - b) i, j , int, obs
 - c) 9xy, a36, x, --j
 - d) 4_t, /fim, h
 - e) Nenhuma das opções

Tipos de Variáveis

- O tipo de uma variável define os valores que ela pode assumir e as operações que podem ser realizadas com ela;
- **Inteiros**: valores positivos ou negativos, que não possuem uma parte fracionária. Exemplos: 1, 30, 40, 12, -50;
- **Float (real)**: valores positivos ou negativos, que podem possuir uma parte fracionária (também podem ser inteiros). Exemplos: 1.4, 6.7, 10.3, 100, -47;
- **Texto (string)**: qualquer elemento presente no teclado. Exemplos: "José", "Maria", "M", 'F';
- **Lógico (boolano, verdadeiro ou falso)**: True, False.

Declaração de Variáveis

- No Python não tem a necessidade de declarar o tipo de variável, basta apenas atribuir um valor a uma que ele mesmo se encarrega de identificar o tipo;

Exemplo: Declarando Variaveis.ipynb

Contantes

- É um tipo de variável, cujo o valor não se altera durante a execução do programa;
- Contudo, no Python esse conceito não existe, pois as variáveis são tratadas como objetos e sempre serão dinâmicas.

Operadores Aritméticos

- Exemplos:

Operador	Significado	Exemplo
+	Adição de dois valores	$z = x + y$
-	Subtração de dois valores	$z = x - y$
*	Multiplicação de dois valores	$z = x * y$
/	Quociente de dois valores	$z = x / y$

Exemplo : Operadores Aritmeticos.ipynb

Exibir valores no console

- print

A função print() é uma das funções de saída de dados.

Exemplo: Comandos de saída e entrada.ipynb

Lendo suas variáveis do teclado

- input;
- A função input() é uma função de entrada de dados da linguagem python;
- Aguarda o usuário entrar o dado;
- Criar a variável x como string e armazena o valor;
- Independente do tipo de dado informado, a variável será sempre string

Exemplo: Comandos de saída e entrada.ipynb

Conversão de valores

- `x = int(z);`
- `w = str(m);`
- `t = float(l) .`

Exemplo: Convertendo valores.ipynb

Comentários

- Comentário é um trecho do programa utilizado para descrever uma observação ou mesmo desabilitar uma linha ou parte do código;
- Pode ser usado como lembretes e documentação do código;
- Existem duas maneiras de se comentar:
“ tudo o que vier na
linha depois do “, não será considerado e posso comentar
múltiplas linhas;
Este texto é um comentário.

Exemplo: Comentarios em python.ipynb

Operadores de Atribuição Simplificada

Operador	Significado		Exemplo	
<code>+=</code>	soma e atribui	<code>x +=y</code>	igual	<code>x = x + y</code>
<code>-=</code>	subtrai e atribui	<code>x -=y</code>	igual	<code>x = x - y</code>
<code>*=</code>	multiplica e atribui	<code>x *=y</code>	igual	<code>x = x * y</code>
<code>/=</code>	divide e atribui o quociente	<code>x /=y</code>	igual	<code>x = x / y</code>
<code>%=</code>	Divide e atribui o resto	<code>x %=y</code>	igual	<code>x = x % y</code>

Exemplo: Operadores Atribuicao Simpl.ipynb

Note: existem outros operadores, mas por hora serão vistos apenas estes.

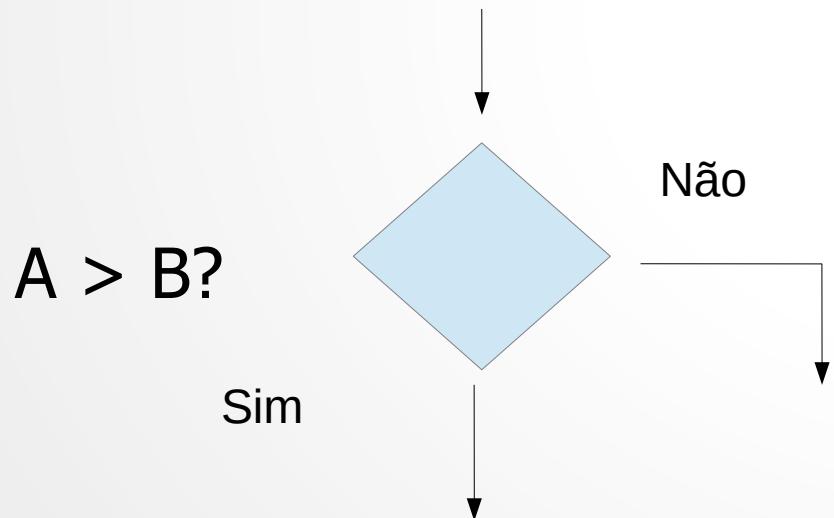
Operadores Relacionais

- A linguagem C possui um total de seis operadores relacionais, como mostra a tabela.

Operador	Significado	Exemplo
>	Maior que	$x > 5$
\geq	Maior ou igual a	$x \geq 10$
<	Menor do que	$x < 5$
\leq	Menor ou igual a	$x \leq 10$
\equiv	Igual a	$X \equiv 10$ ou ' $X \equiv Y$ '
\neq	Diferente de	$x \neq 10$

Comandos de controle condicional

- Na vida real tomamos decisões a todo o momento baseadas em uma situação existente. Em um algoritmo, chamamos esta situação de condição. Associada a uma condição, existirá uma alternativa possível de ações.



Comandos de controle condicional

- Comando IF

Em python, o comando **if** é utilizado sempre que é necessário escolher entre dois caminhos dentro do programa ou quando se deseja executar um ou mais que estejam sujeitos ao resultado de um teste. A forma geral é:

```
if (condição):  
    sequência de comandos
```

Comandos de controle condicional

- Se a condição for verdadeira, a sequência de comandos será executada.
- Se a condição for falsa, a sequência de comandos não será executada, e o programa continuará a ser executado.

Comandos de controle condicional

- Comando ELSE
- O comando **else** pode ser entendido como um complemento do comando **if**. Ele auxilia o comando **if** na tarefa de escolher entre vários caminhos a serem seguidos dentro do programa. A forma geral é:

Exemplo: Comandos de decisao.ipynb

Comandos de controle condicional

- Comando ELSE

```
if (condicao):  
    primeira sequência de comandos  
else:  
    segunda sequência de comandos
```

Comandos de controle condicional

- Aninhamento de IF
- Um **if** aninhado é simplesmente um comando **if** utilizado dentro do bloco de comandos **if** (ou **else**) mais externo. Basicamente, é um comando **if** dentro de outro.

Comandos de controle condicional

```
if (condicao):  
    sequencia de comandos  
  
else:  
  
    if(condicao):  
        sequencia de comandos  
  
    else:  
  
        sequencia de comandos
```

Comandos de controle condicional

- elif. Sua forma geral é:

```
if (condicao):
```

```
    sequencia de comandos
```

```
elif (condicao):
```

```
    sequencia de comandos
```

```
else:
```

```
    sequencia de comandos
```

Exemplo: Estrutura de decisao com and e or.ipynb

Operadores Lógicos

- A linguagem C possui três operadores lógicos:

Operador	Significado	Exemplo
and	Operador E	(x >= 0 and x <= 9)
or	Operador OU	(a == 'F' or b!= 32)
not	Operador NEGAÇÃO	not(x == 10)

Tabela Verdade

a	b	not a	not b	a and b	a or b
False	False	True	True	False	False
False	True	True	False	False	True
True	False	False	True	False	True
True	True	False	False	True	True

As sequências de escape

- As sequências de escape permitem o envio de caracteres de controle não gráficos para dispositivos de saída.

\n	nova linha (new line)
\r	retorno de carro
\t	tabulação horizontal
\'	apóstrofe
\“	aspas
\f	Alimentação de folha

Exemplo: Caracteres de escape.ipynb

Comandos SWITCH

- Em python nativamente não temos comando SWITCH;
- Seria necessário criar artifícios para simular, portanto, não iremos nos aprofundar nesse tema.

Comandos condicionais com and e or

- Também é possível combinar o uso de operadores **and** e **or**;
- Isso permite trabalhar com condicionais mais complexas.

Exemplo:Estrutura de decisao com and e or.ipynb

Comandos de repetição

- São comandos que possibilitam que um bloco de comando seja executado mais de uma vez. A forma geral é:
enquanto condição faça
 sequência de comandos;
fim enquanto.

Comandos de repetição

- Em Python temos dois tipos de comandos: while e o for;
- A sequência de comandos a ser repetida está subordinada a uma condição;
- Que pode ser uma condição lógica ou de acordo com um número de vezes;
- Também podemos inserir o comando “**break**” para interromper um determinado bloco de código.

Exemplo: Comandos de repeticao.ipynb

Comandos de repetição – Laço Infinito

- Um laço infinito (ou loop infinito) é uma sequência de comandos em um programa de computador que sempre se repete, ou seja, infinitamente;
- Ocorre por algum laço erro de programação, quando:
 - Não definimos uma condição de parada.
 - A condição de parada existe, mas nunca é atingida.

Comandos de repetição – Laço Infinito

- Exemplo: algoritmo que exibe o valor de um número até que o mesmo continua sendo menor que 10.

a = 1

while a < 10:

 print(a)

o que falta aqui? → (a = a + 1 ou a+=1)

Comandos de repetição

- Exemplo: algoritmo que exibe o valor de um número até que o mesmo continua sendo verdadeiro.

a = True;

While a == True:

 print(a)

o que falta aqui? —————> if (a==True):

 a = False

Exemplo: Comandos de repeticao.ipynb

Break e Continue

- O comando **break** interrompe, quebra ou pausa uma laço de repetição, ou seja, para o looping;
- O comando continue faz com que a execução do laço pule para o início e, tudo que vier abaixo não pe executado.

Exemplo:Comandos de repeticao.ipynb

Listas

- Listas é uma estrutura que permite guardar diversos dados dentro;
- Os dados são inseridos de forma organizada e indexada;
- É como se pegasse várias variáveis de vários tipos e colocasse tudo dentro de uma lista, de forma “indexada”;

Importante: Diferentemente de outras linguagens como C ou Java, Python não trabalha com array ou vetores. Ademais, nessas linguagens cada array precisa ser criado de acordo com o tipo que será armazenada, ou seja, não é permitido inserir valores inteiros, reais ou textos em um mesmo vetor.

Exemplo: Listas.ipynb

Manipulação de Listas

- Nas listas pode-se: inserir elementos, remover elementos um a um ou todos de uma vez.

Exemplo: Manipulacao de lista.ipynb

Tuplas

- Tuplas é uma estrutura semelhante a uma lista;
- Entretanto, a lista é dinâmica e a tupla estática, portanto, não pode-se manipulá-las com os comandos `append`, `remove` ;
- Deletar todos os elementos de uma tupla com o comando `del`, como se faz em uma lista é possível;
- Na lista os elementos são inseridos com uma colchete, já a tupla, usa-se parênteses.

Exemplo: `Tuplas.ipynb`

Dicionários

- Os dicionários são estruturas diferentes das listas e tuplas;
- Dicionário se resumo em **chave:valor**;
- A sintaxe de um dicionário é definida por chaves {};
- Deve segui a ordem **chave:valor** e vírgula como separador.

Exemplo: Dicionarios.ipynb

Conjuntos

- A declaração de conjuntos é semelhante a de tuplas;
- É uma lista com sintaxe de dicionário;
- Não são indexados;
- Podemos ter valores repetidos como os conjuntos numéricos;
- Podemos fazer as mesmas operações: interseção, união, etc... .

Exemplo: Conjuntos.ipynb

Funções

- São blocos de códigos que podem ser reutilizáveis;
- Podem ser chamadas quantas vezes forem necessárias;
- Podem ser chamada por outros programas;
- Existem funções pré definidas do Python com print, input, etc...;
- E funções podem ser criadas por usuários;
- Em Python usa-se o comando **DEF** para definir uma função.

Exemplo: Funcoes.ipynb

Módulos

- São conjuntos de funcionalidades organizadas em arquivos;
- Em alguns casos precisamos instalar e depois importar.

Exemplo: Modulos.ipynb

Funções Padrão

- São internas do Python;
- Não há a necessidade de fazer um import;
- abs(): retorna o valor absoluto;
- max(): retorna o maior valor;
- min(): retorna o menor valor;
- round(): arredonda um valor;
- sum (): soma valor(s).

Exemplo: Funcoes.ipynb

Criando um arquivo de Funções

Podemos criar um arquivo de funções, e depois fazer um import para utilizá-las:

Iremos ver o exemplo na prática, passo a passo.

Exemplo: Importando Funcoes User.ipynb e funcoesUser.py

Expressões Lambdas

- Também denominadas de funções inline, anônima ou expressão lambda;
- Permite criar funções ad-hoc, ou seja, em tempo de execução sem usar o **def**;
- Funcionam da mesma forma que as funções convencionais, ou seja, com **def**;

Características do Lambda

- O corpo do Lambda é uma único expressão e não um bloco;
- Funciona como uma instrução de retorno de uma função **def**;
- São muito úteis quando usadas com as intruções map(), filter() e reduce();
- A diferença entre o **def** e o lambda é que o primeiro cria um objeto e atribui um nome (nome da função) e o segundo cria um objeto que retorna um resultado em tempo de execução.

Exemplo: Expressoes lambda.ipynb

Manipulando arquivos

- Em python podemos manipular arquivos do tipo texto, excel, csv, e-mails, etc;
- Você pode instalar pacotes ou utilizar algumas funções internas de python (Funções Built-in).

Exemplo: Manipulando arquivos.ipynb

Operações com Datasets com Funções Built-in

Exemplo: Operacoes com Datasets.ipynb

Manipulando Arquivos TXT, CSV e JSON

Exemplo: Manipulando arquivos TXT_CCSV_JSON.ipynb

Funções built-in

- Outras funções built-in

Exemplo: Outras funcoes built-in.ipynb

List Comprehension

- List Comprehension nada mais é do que uma forma rápida e eficiente de criar uma lista a partir de outra lista.

Exemplo: Outras funcoes built-in.ipynb

Tratamento de Erros e Exceções

Exemplo: Tratamento de Erros e Excecoes.ipynb

Orientação a Objetos – Classe e Objetos

- Classe é um agrupamento de objetos que possuem características os mesmos atributos e métodos;
- Para saber os detalhes sobre OO vejam o vídeo na URL:

Exemplo: Orientacao a Objetos.ipynb

Extras

- Instalação de pacotes: <https://pypi.org/>
- Instalando um pacote: !pip install biopython ou pip install biopython.
- Verificando a lista de pacotes instalados: conda list (isso na máquina local).

Exemplo: Modulos e pacotes.ipynb

Python

Referências:

Python Doc. Disponível em:<<https://docs.python.org/>>. Acesso em 13/01/2021.

FELTRIN, Fernando Belomé. Python do ZERO à Programação Orientada a Objetos. Ebook Kindle: Amazon, 2020. 204 p.