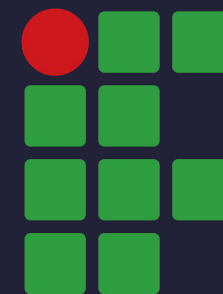




MINICURSO

# INTRODUÇÃO AO PYTHON E SUAS APLICAÇÕES NA ENGENHARIA

IV SEMANA ACADÊMICA DE ENG. ELÉTRICA



**INSTITUTO  
FEDERAL**  
Tocantins





## Sobre o minicurso:

Esse minicurso trata sobre as tecnologias disponíveis da linguagem Python e algumas de suas aplicações na Engenharia Elétrica.

Não é preciso conhecimento prévio em programação, mas caso possua, será de grande valia.

Para acessar os arquivos utilizados neste minicurso basta acessar o link a seguir:  
<link>

A ferramenta utilizada para desenvolvimento das atividades será o Google Colab, uma ferramenta online que permite escrita e execução de códigos direto pelo navegador





# Estrutura

Alguns dos tópicos abordados durante o minicurso serão:

- Cálculos integrais diferenciais;
- Cálculo matricial;
- Conversão entre sistemas de coordenadas;
- Além de aprender a realizar de forma simplificada diversas outras tarefas repetitivas.

Além de desafios propostos ao longo da nossa jornada.

# Estrutura

O curso está organizado da seguinte maneira:

- Porque Python?
- Operadores elementares e lógicos.
- Tipos de dados
- Controle de fluxo [if, else, for, while]
- Funções
- Gráficos [Bibliotecas Matplotlib e Seaborn]
- A biblioteca Numpy [Numerical Python]
- Cálculo matricial



## Crescente uso no mercado, mas porque?

- Qualidade de software
- Produtividade
- Amplo suporte da comunidade
- Fácil integração de componentes

# Operadores Elementares e Lógicos

## Operadores lógicos:

and, or, not

## Operadores algébricos:

+ adição

- subtração

\* multiplicação

/ divisão

// divisão (arrendondada para menos)

% resto da divisão

\*\* exponencial

## Operadores relacionais

< menor que

> maior que

<= menor ou igual

>= maior ou igual

== igualdade

!= desigualdade



# Operadores Elementares e Lógicos

## Operadores de atribuição


`=` atribuição

`+=` atribuição com soma

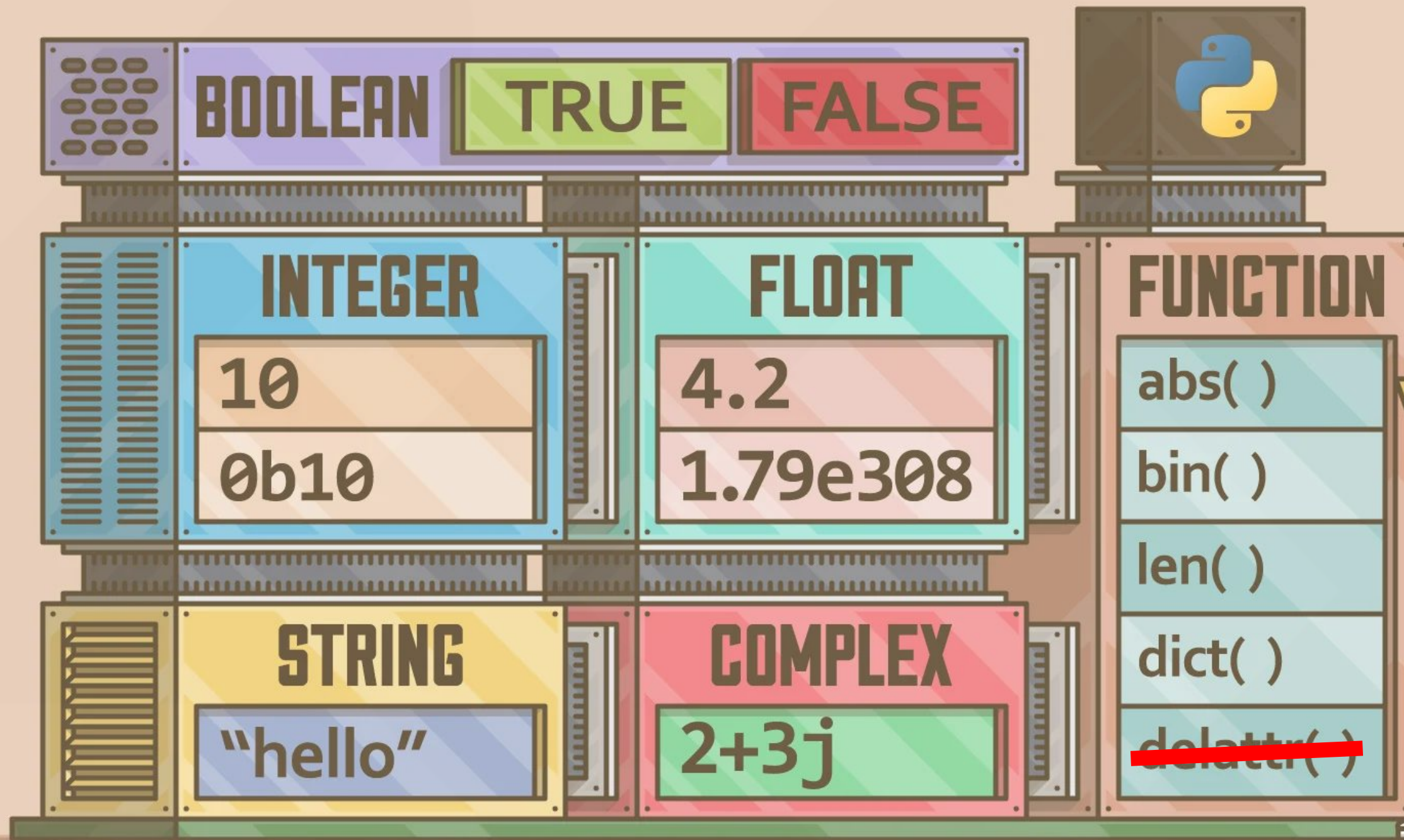
`-=` atribuição com subtração

`* =` atribuição com multiplicação

`/=` atribuição com divisão









# TIPOS DE DADOS

## TIPOS BÁSICOS



**4 tipos de variáveis principais:**

- **Valores quantitativos discretos (int)**
- **Valores quantitativos contínuos (float)**
- **Valor qualitativos nominais (string, object, bool)**

**\*existem formatos específicos de variáveis, como data e tempo**

# TIPOS DE DADOS

## COLEÇÕES



### LISTAS

É uma coleção de valores indexados que inicia em que o primeiro elemento tem índice 0, o segundo tem índice 1 e assim por diante.

### TUPLAS

É uma coleção de valores indexados que segue a mesma lógica da lista, porém seus valores são constantes, ou seja, os valores das tuplas não podem ser alterados,

### DICIONÁRIOS

É uma coleção de dados que funciona baseado no princípio de *chave:valor* no qual cada chave adicionada possui um respectivo valor.

# CONTROLE DE FLUXO

## LAÇOS E CONDICIONAIS

### if

- Provavelmente o mais conhecido comando de controle de fluxo;
- É possível aninhar if's para criar condicionais mais complexas.
- Uma sequência  
*if... elif... elif...else*  
pode facilmente substituir os comandos *switch : case*



### elif

- A palavra-chave 'elif' é uma abreviação para 'else if', e é útil para evitar indentação excessiva.
- Não é obrigatória

### else

- A parte *else* é responsável por conter as ações em caso de nenhuma das condições ser atendida.
- Não é obrigatória

# CONTROLE DE FLUXO

## LAÇOS E CONDICIONAIS



### FOR

O loop *for* em Python é utilizado para iterar uma sequência. A sequência por sua vez é um conjunto de elementos.

### RANGE

A função embutida `range()` gera progressões aritméticas que são ideais para iterar sobre sequências

### WHILE

O laço *while* executa determinado bloco de código enquanto determinada expressão for verdadeira.

Em python não há Do/While

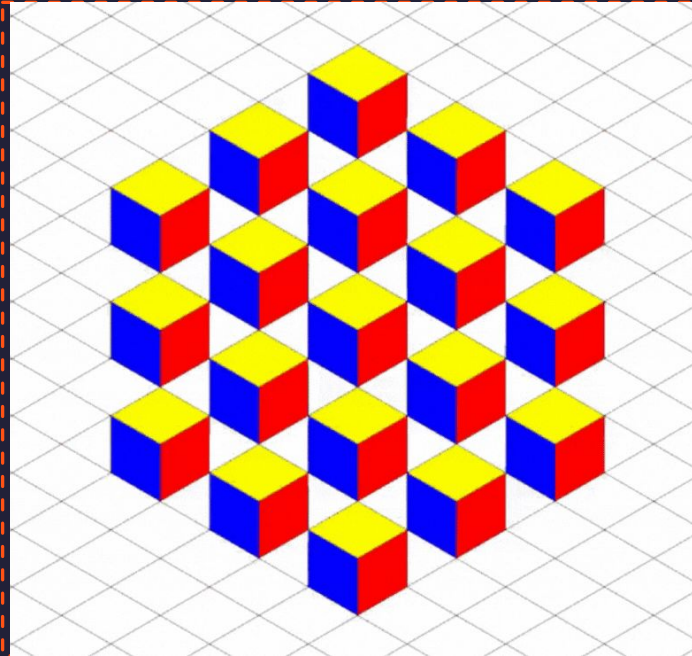
- Possuem tarefas específicas.
- Funções são blocos de códigos que podem ser reutilizados
- São usadas, geralmente, para evitar redundâncias no código
- Podem retornar valores dos mais variados tipos de dados
- são definidas utilizando a palavra reservada *def*

```
def hello(meu_nome,idade):  
    print('Olá',meu_nome,'\nSua idade é:',idade)
```

# A BIBLIOTECA NUMPY

## CÁLCULO DE MATRIZES

O Python tem um módulo especialmente desenvolvido para criar e tratar matrizes, o Numpy é uma biblioteca robusta que tem muitos métodos para se trabalhar com matrizes.



O objeto `numpy.array` é a estrutura que o Numpy trabalha, podendo ter  $n$  dimensões e é diferente das listas basicamente pela forma de indexação e pelos métodos suportados



# A BIBLIOTECA NUMPY

## CÁLCULO DE MATRIZES

- *ndarray.ndim*: o número de dimensões da matriz.
- *ndarray.shape*: as dimensões da matriz.
- *ndarray.size*: o número total de elementos da matriz.

### Atributos Importantes

- *ndarray.dtype*: descreve o tipo dos elementos na matriz.
- *ndarray.itemsize*: o tamanho (bytes) dos elementos da matriz.
- *ndarray.data*: *buffer* contendo os elementos da matriz.

# A BIBLIOTECA NUMPY

## CÁLCULO DE MATRIZES

### Métodos de Inicialização

- `numpy.arange(start, stop, step, dtype=None, *, like=None)`
- `numpy.zeros()`
- `numpy.ones()`
- `numpy.empty()`
- `numpy.linspace(start, stop, num=50, endpoint=True, dtype=None, axis=0)`

# A BIBLIOTECA SYMPY

## COMPUTAÇÃO SIMBÓLICA

A computação simbólica trata da computação de objetos matemáticos de forma simbólica.

Isto significa que os objectos matemáticos são representados de forma genérica e precisa.

Possui capacidade de manipular em forma simbólica expressões matemáticas e realizar cálculos.

# A BIBLIOTECA SYMPY

## COMPUTAÇÃO SIMBÓLICA

```
from sympy import *  
x, y, z = symbols('x y z')  
init_printing()
```

```
Integral(sqrt(1/x), x)
```

$$\int \sqrt{\frac{1}{x}} dx$$

# GRÁFICOS

## MATPLOTLIB E SEABORN

O pacote **matplotlib** é voltado para criação de gráficos de alta qualidade.

É muito usado hoje no meio científico, além de ser software livre.

Há maneiras diferentes de se produzir a mesma coisa com o **matplotlib**.

Com a *Matplotlib* os resultados não são tão agradáveis esteticamente.

Por esse motivo é comum usarmos juntamente com a *Matplotlib* a biblioteca *Seaborn*.

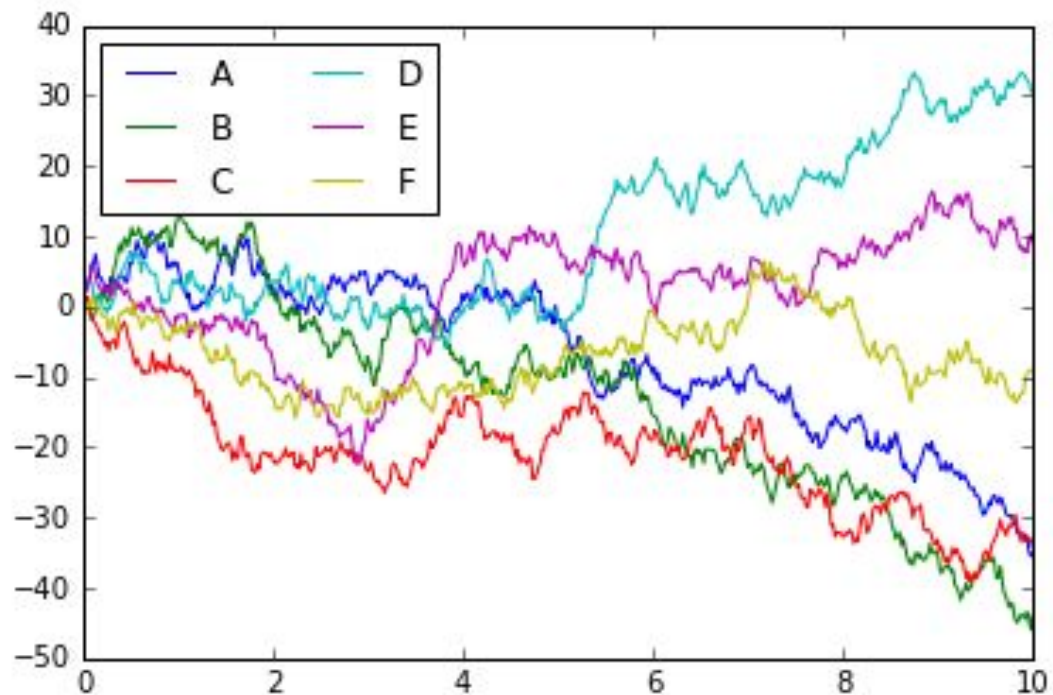
*Seaborn* também é uma biblioteca gráfica para Python.

É construída sobre a **matplotlib** e possui integração com **pandas**.

As suas funções de plotagem operam em *Dataframes* e matrizes.

# GRÁFICOS

## MATPLOTLIB E SEABORN



MATPLOTLIB



SEABORN



Obrigado pela Atenção !

Para dúvidas e materiais:

[augustoanb@gmail.com](mailto:augustoanb@gmail.com)

(63) 992306793

MUITO OBRIGADO !