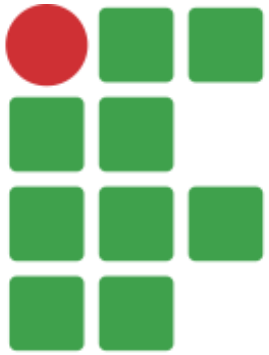

INSTITUTO FEDERAL
Paraíba

Campus Cajazeiras

# ALGORITMOS E LÓGICA DE PROGRAMAÇÃO

PROFESSORES LEANDRO, MICHEL
CAJAZEIRAS / IFPB

# Tuplas

# OBJETIVOS

- Compreender os tipos de dados em coleções;
- Compreender as coleções de dados existentes e suas operações;
- Desenvolver algoritmos utilizando Tuplas.

# ROTEIRO

- coleção ordenada de valores;
- tuplas

# Estruturas em sequência

- Sequências!
  - A **sequência** numérica nada mais é do que uma **sequência** de números.
  - Podem ser **indexados** por algum valor **ordinal posicional**.
  - Uma sequência pode ser finita ou infinita.
  - Uma sequência pode ser crescente, descrente, constante ou oscilante.
- Lista: lista1 = [1, -2, 3, "IFPB", "ifpb", "ALGORITMOS"]
- Tupla: tupla1 = (1, "IFPB', 3.14156, (2,3), "algoritmos")
- String: oracao1 = "vamos para a aula de algoritmos!"


https://youtu.be/dX5SL2YbjWw

# Tuplas

- Uma tupla é uma coleção ordenada de valores.
- **As tuplas estão ordenadas** - as tuplas mantêm uma ordem posicional da esquerda para a direita entre os itens que contêm.
- **Acessado por índice** - os itens em uma tupla podem ser acessados por meio de um índice.
- **As tuplas podem conter qualquer tipo de objeto** - podem ser números, strings, listas e até mesmo outras tuplas.

As tuplas são parecidas com listas

**As tuplas são imutáveis** - você não pode adicionar, excluir ou alterar itens após a tupla ser definida.

# Tuplas

## Sintaxe

**Tupla = (a, b, ..., z)**

podemos criar uma **tupla** colocando uma sequência de itens separados por vírgulas entre parênteses ()

main.py

```
1   # Uma tupla de inteiros
2   T1 = (1, 2, 3)
3
4   # Uma tupla de strings
5   T2 = ("vermelho", "verde", "azul")
6
7   print(T1)
8   print(T2)
9
10  # Imprime (1, 2, 3)
11  # Imprime ('vermelho', 'verde', 'azul')
```

```
(1, 2, 3)
('vermelho', 'verde', 'azul')
>
```

# Tuplas

## Exemplos

Os itens de uma tupla não precisam ser do mesmo tipo. A tupla a seguir contém um **inteiro**, uma **string**, um **float** e um **booleano**.

main.py

```
1  # Uma tupla com tipos de dados mistos
2  T = (1, "abc", 1.23, True)
3  print(T)
4  # Imprime (1, 'abc', 1.23, True)
```

```
(1, 'abc', 1.23, True)
```

Uma tupla contendo **zero itens** é chamada de **tupla vazia** e você pode criar uma com colchetes vazios()

main.py

```
1  # Uma tupla vazia
2  T = ()
3  print(T)
4  # Imprime ()
```

```
()
```

# Tuplas

Se a tupla possuir apenas **um elemento dentro de si**, é necessário indicar que ela é uma tupla ou dentro de si colocar uma vírgula como separador mesmo que não haja um segundo elemento.

Já quando temos **2 ou mais elementos dentro da tupla**, não é mais necessário indicar que ela é uma tupla, o interpretador identificará automaticamente.

## Exemplos

main.py

```python
# criando uma tupla
T1 = tuple("Davi")
T2 = ("Daniele",)
T3 = ("Davi", "Daniele")
print(T1)
print(T2)
print(T3)
# Imprime ('D', 'a', 'v', 'i')
# Imprime ('Daniele',)
# Imprime ('Davi', 'Daniele')
```

```
('D', 'a', 'v', 'i')
('Daniele',)
('Davi', 'Daniele')
```

main.py

```python
# criando uma tupla
T1 = ("Davi")
T2 = ("Daniele",)
T3 = ("Davi", "Daniele")
```

```
Davi
('Daniele',)
('Davi', 'Daniele')
```

# Tuplas

Podemos acessar itens individuais em uma tupla usando um índice entre colchetes. Observe que a indexação da tupla começa em 0.

## Exemplos

| -3 | -2 | -1 |
|---|---|---|
| Vermelho | Verde | Azul |
| 0 | 1 | 2 |

```
1   #         -3          -2         -1
2   T1 = ("Vermelho", "Verde", "Azul")
3   #          0           1          2
4
5   print(T1)
6   # Imprime ('Vermelho', 'Verde', 'Azul')
```

```
('Vermelho', 'Verde', 'Azul')
```

```
5   print(T1)
6   print(T1[2])
7   # Imprime ('Vermelho', 'Verde', 'Azul')
8   # Imprime Azul
```

```
('Vermelho', 'Verde', 'Azul')
Azul
```

# Tuplas

A imutabilidade da tupla é aplicável apenas ao nível superior da própria tupla, não ao seu conteúdo.
Por exemplo, uma **lista** dentro de uma **tupla** pode ser alterada normalmente.

## Exemplos

```
1    #
2    T1 = (1, [5, 3], 4)
3    T1[1][0] = 2
4    print(T1)
5    # Imprime (1, [2, 3], 4)
```

```
(1, [2, 3], 4)
>
```

# Tuplas

## Operações em Tuplas

A tabela abaixo mostra um resumo dos principais operadores para manipulação de listas em Python.

| Operador | Descrição | Exemplo |
|---|---|---|
| + | concatena duas tupla | (1, 2, 3) + (4, 5)<br>retorna (1, 2, 3, 4, 5) |
| * | repete (replica) uma tupla múltiplas vezes | (0, 1) * 2<br>retorna (0, 1, 0, 1) |
| [i] | retorna o i-ésimo elemento da tupla | (10, 20, 30)<br>print[1] retorna 20 |
| [i:j] | retorna a sub-tupla que vai dos índices i até j-1 | (1, 3, 5, 7, 9)<br>Print(1:3)<br>retorna (3, 5) |
| a = b[:] | 2 cópias independentes | Exemplos Usando Tuplas |
| a = b | os 2 referenciam o mesmo objeto | |

# Tuplas

## métodos em tuplas

A tabela abaixo mostra um resumo dos principais métodos para manipulação de listas em Python.

| método | Descrição | Exemplo |
|---|---|---|
| Index() | procura a primeira ocorrência do item fornecido e retorna seu índice. Se o item especificado não for encontrado, ele gerará a **exceção** 'ValueError'. Os argumentos opcionais **inicio** e **fim** limitam a pesquisa a uma subsequência particular da **tupla** . | tupla.index(item, inicio, fim) |
| count | contar quantas vezes um elemento aparece numa tupla | tupla.count('D') |

# Tuplas

## funções em tuplas

A tabela abaixo mostra um resumo das principais funções para manipulação de tuplas em Python.

| método | Descrição | Exemplo |
|---|---|---|
| sum | Soma todos os elementos | Soma = sum(tupla) |
| len | Descobrir o tamanho de uma tupla | tamanho = len(tupla) |
| sorted | Retornar uma **lista** cópia ordenada da tupla | listaOrdenada = sorted(tupla) |
| max | Retorna o maior item | Maior = max(tupla) |
| min | Retorna o menor item | Menor = min(tupla) |
| | | |

# Tuplas

## Excluir uma tupla

As tuplas não podem ser modificadas, portanto, obviamente, não podemos excluir nenhum item dela.
No entanto, podemos excluir a tupla completamente com a palavra-chave **del** .

```
# deletar a tupla
del T
print(T) # NameError: name 'T' is not defined
```

# Exercícios 1

1 – gere tupla dinamicamente de tamanho 5, de tamanho aleatório entre 1 e 20

2 – verifique se um determinado elemento esta contido na tupla ('a','b',1,2,'c',3,4,'d',5)

3 – inicie a tupla abaixo, e contar os itens, use o método **count**().

('a','b','b','a','c','c','c','d','d','e')

Obs: explicar a **Counter**() da **collection**.