✨ Listas e Tuplas ✨

> O que são, Como Usar e Diferenças Entre Listas e Tuplas

Nesta aula, vamos aprender sobre listas e tuplas em Python. Se você não tem ideia do que são listas ou tuplas, não se preocupe — começaremos do básico. Elas são tipos de estruturas de dados usadas para armazenar coleções de itens. Você pode pensar neles como caixas que guardam várias coisas.

💡 O que é uma Lista? 💡

Uma lista em Python é uma coleção de itens. Imagine que você quer armazenar o nome de várias frutas. Em vez de criar uma variável para cada fruta, você pode colocar todas em uma lista.

```
frutas = ["maçã", "banana", "laranja"]
```

Aqui criamos uma lista chamada frutas que contém três elementos: "maçã", "banana", e "laranja". Listas são definidas usando colchetes [] e os itens são separados por vírgulas.

Por que usar listas?

- Para armazenar múltiplos itens em uma única variável.
- Para facilitar o acesso e a modificação desses itens.

⚡ Como Acessar Itens em uma Lista ⚡

Podemos acessar os itens de uma lista pelo índice (a posição do item). Em Python, a contagem começa no 0, então o primeiro item tem índice `0`, o segundo tem `1`, e assim por diante.

```
print(frutas[0])  # Exibe "maçã"
print(frutas[2])  # Exibe "laranja"
```

`frutas[0]` acessa o primeiro item, `"maçã"`, enquanto `frutas[2]` acessa o terceiro item, `"laranja"`.

⚡ Modificando Itens em uma Lista ⚡

Como as listas são mutáveis, podemos alterar os itens que estão dentro delas.

```
frutas[1] = "morango"
print(frutas)  # Exibe ['maçã', 'morango', 'laranja']
```

Mudamos o segundo item de `"banana"` para `"morango"`.

⚡ Adicionando e Removendo Itens ⚡

Podemos adicionar ou remover itens de uma lista facilmente.

Adicionar Itens:

Para adicionar um item ao final da lista, usamos o método `.append()`.

```
frutas.append("uva")
print(frutas)  # Exibe ['maçã', 'morango', 'laranja', 'uva']
```

Remover Itens:

Para remover um item específico, usamos o método `.remove()`.

```
frutas.remove("morango")
print(frutas)  # Exibe ['maçã', 'laranja', 'uva']
```

⚡ Métodos Mais Comuns de Listas ⚡

Python oferece vários métodos úteis para trabalhar com listas. Aqui estão alguns dos mais comuns:

1. `append(item)`:

Adiciona um item ao final da lista.

```
frutas.append("abacaxi")
print(frutas)  # Exibe ['maçã', 'laranja', 'uva', 'abacaxi']
```

2. `insert(posicao, item)`:

Insere um item na posição especificada.

```
frutas.insert(1, "morango")
print(frutas)  # Exibe ['maçã', 'morango', 'laranja', 'uva', 'abacaxi']
```

3. `remove(item)`:

Remove o primeiro item com o valor especificado.

```
frutas.remove("uva")
print(frutas)  # Exibe ['maçã', 'morango', 'laranja', 'abacaxi']
```

4. `pop(posicao)`:

Remove o item na posição especificada (ou o último item, se nenhuma posição for fornecida).

```
frutas.pop(2)
print(frutas)  # Exibe ['maçã', 'morango', 'abacaxi']
```

5. `len(lista)`:

Retorna o número de itens na lista.

```
print(len(frutas))  # Exibe 3
```

6. `sort()`:

Ordena a lista em ordem crescente (alfabética ou numérica).

```
numeros = [4, 2, 8, 1]
numeros.sort()
print(numeros)  # Exibe [1, 2, 4, 8]
```

7. `reverse()`:

Inverte a ordem dos itens na lista.

```
numeros.reverse()
print(numeros)  # Exibe [8, 4, 2, 1]
```

8. `index(item)`:

Retorna o índice do primeiro item com o valor especificado.

```
indice = frutas.index("morango")
print(indice)  # Exibe 1
```

💡 O que é uma Tupla? 💡

Uma tupla em Python é muito parecida com uma lista, mas existe uma diferença importante: tuplas são imutáveis. Isso significa que, uma vez criadas, não podemos mudar os itens dentro dela.

```
cores = ("vermelho", "verde", "azul")
```

Criamos uma tupla chamada cores que contém três elementos: `"vermelho"`, `"verde"`, e `"azul"`.

Tuplas são definidas usando parênteses `()` e os itens são separados por vírgulas.

Por que usar tuplas?

- Para armazenar dados que não devem ser alterados.
- Elas são mais rápidas do que listas, pois não podem ser modificadas.

⚡ Acessando Itens em uma Tupla ⚡

Assim como nas listas, podemos acessar os itens de uma tupla usando índices.

```
print(cores[1])  # Exibe "verde"
```

`cores[1]` acessa o segundo item, "verde".

Tentando Modificar uma Tupla:

```
cores[0] = "amarelo"  # Isso vai gerar um erro!
```

Como as tuplas são imutáveis, você não pode alterar o valor de `"vermelho"` para `"amarelo"`.

⚡ Diferenças Entre Listas e Tuplas ⚡

Característica	Lista	Tupla
Mutabilidade	Mutável (pode mudar)	Imutável (não pode mudar)
Definição	Colchetes `[]`	Parênteses `()`
Uso Comum	Armazenar itens que mudam	Dados que não mudam
Exemplo	`["maçã", "banana"]`	`("vermelho", "verde")`

Resumindo:

- Use **listas** quando precisar alterar os itens.
- Use **tuplas** quando precisar de uma coleção de itens **constantes**.

⚡ Conversão Entre Listas e Tuplas ⚡

Podemos converter uma lista em tupla e vice-versa usando as funções `list()` e `tuple()`.

Lista para Tupla:

```
frutas_tupla = tuple(frutas)
print(frutas_tupla)  # Exibe ('maçã', 'laranja', 'uva')
```

Tupla para Lista:

```
cores_lista = list(cores)
print(cores_lista)  # Exibe ['vermelho', 'verde', 'azul']
```

📋 Resumo da Aula 📋

- **Lista:** Coleção mutável de itens. Use colchetes [] e modifique seus itens quando necessário.
- **Tupla:** Coleção imutável de itens. Use parênteses () e mantenha seus itens constantes.
- **Acessando Itens:** Use índices para acessar elementos tanto em listas quanto em tuplas.
- **Modificando Itens:** Listas são mutáveis (podem ser modificadas), tuplas não.
- **Conversão:** Use list() e tuple() para converter entre listas e tuplas.