

LISTAS

IFBA – Campus Ilhéus
Curso: Back End com Python

Objetivo da Aula

- Manipular coleções de dados em listas.

Criação de Listas

- Listas são estruturas que armazenam vários valores em uma única variável, e os elementos podem ser de qualquer tipo (inteiros, *strings*, etc).

- Sintaxe básica:

```
nomes = ["Ana", "Carlos", "Beatriz"]
```

```
numeros = [10, 20, 30, 40]
```

```
mistura = [1, "texto", True, 3.5]
```

Acesso aos Elementos da Lista

- Cada item da lista tem um **índice**, que começa do 0.

- Exemplo:

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

```
print(frutas[0]) # maçã
```

```
print(frutas[1]) # banana
```

```
print(frutas[2]) # uva
```

Acesso aos Elementos da Lista

- Também é possível acessar de trás para frente com índices negativos:

```
print(frutas[-1]) # uva
```

```
print(frutas[-2]) # banana
```

Métodos importantes de listas

⦿ append()

- Adiciona um item no final da lista.

```
nomes = ["João", "Maria"]  
nomes.append("Pedro")  
print(nomes) # ['João', 'Maria', 'Pedro']
```

Métodos importantes de listas

⦿ insert()

- Insere um item em uma posição específica.

```
nomes = ["João", "Maria"]
```

```
nomes.insert(1, "Ana")
```

```
print(nomes) # ['João', 'Ana', 'Maria']
```

Métodos importantes de listas

⦿ remove()

- Remove o primeiro item com o valor indicado.

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

```
frutas.remove("maçã")
```

```
print(frutas) # ['banana', 'maçã']
```


Métodos importantes de listas

⦿ pop()

- Remove um item da lista e retorna esse item.
- Por padrão, ele remove o último elemento da lista, mas também é possível indicar a posição (índice) do item que deseja remover.

```
frutas = ["maçã", "banana", "uva"]  
removido = frutas.pop()  
print(frutas)      # ['maçã', 'banana']  
print(removido)    # uva
```

Métodos importantes de listas

● pop()

- Remover item de uma posição específica:

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

```
removido = frutas.pop(0)
```

```
print(frutas)      # ['banana', 'uva']
```

```
print(removido)    # maçã
```

- *Se usar pop() com um índice que não existe, o Python mostra um erro!*

Fatiamento de Listas (Slicing)

- É possível **pegar partes** da lista.

- Sintaxe:

`lista[início:fim:passo]`

- O início é o índice de onde começa (inclusivo)
- O fim é onde termina (exclusivo)
- O passo indica de quantos em quantos elementos pula (opcional)

Fatiamento de Listas (Slicing)

- Exemplos:

```
numeros = [10, 20, 30, 40, 50, 60]
```

```
print(numeros[1:4]) # [20, 30, 40]
```

```
print(numeros[:3]) # [10, 20, 30]
```

```
print(numeros[::-2]) # [10, 30, 50]
```

```
print(numeros[::-1]) # [60, 50, 40, 30, 20, 10]  
(lista invertida)
```

Exercícios

1. Crie uma lista com os nomes de 4 colegas da sala. Adicione um novo nome no final com `append()` e outro no início com `insert()`.
2. Remova um nome da lista usando `remove()`.
3. Mostre apenas os 3 primeiros nomes da lista usando fatiamento.
4. Inverta a lista usando fatiamento.