



MANIPULAÇÃO DE LISTAS NO PYTHON

Vamos aprender como manipular Listas no Python de forma pythônica com esse Guia COMPLETO!

Crie Ebooks técnicos incríveis em minutos com IA

Conheça a 1ª IA Especializada na criação de Ebooks **com código!**



Chega de formatar código no Google Docs



Deixe que nossa IA faça o trabalho pesado

 Syntax Highlight

 Adicione Banners Promocionais

 Edite em Markdown em Tempo Real

 Infográficos feitos por IA

TESTE AGORA 

Olá **Pythonista**!

Nesse post vamos continuar falando sobre **LISTAS**!

Se você ainda não acessou nosso post introdutório sobre Listas, pode parar por aqui, [clicar nesse link e dar uma lida nesse post COMPLETÃO!](#)

Agora que você já fez isso, podemos continuar! 😊

Métodos para manipulação de Listas

O Python tem vários métodos disponíveis em listas que nos permite manipulá-las.

Veremos agora quais são os métodos e como utilizá-los!

Método `index`

Utilizado encontrar a posição de um valor especificado. Exemplo:

```
lista = ['Carro', 'Casa', 'Hotel', 'Casa']

pos = lista.index('Casa')

print(f'O item desejado está na posição: {pos}')
```

Saída:

```
O item desejado está na posição: 1
```

Método count

O método `count(elemento)` retorna o número de vezes que o valor especificado aparece na lista. Exemplo:

```
lista = ['Carro', 'Casa', 'Hotel', 'Casa']

pos = lista.count('Casa')

print(f'O item desejado aparece: {pos}')
```

Saída:

```
O item desejado aparece: 2
```

Método append

Para adicionar um elemento ao final da lista, use o método `append(elemento)` :

```
lista = ['Python', 'Academy']

lista.append('adicionado')

print(lista)
```

Saída:

```
['Python', 'Academy', 'adicionado']
```

Método `insert`

Para adicionar um item em um índice especificado, use o método `insert(indice, elemento)`:

```
lista = ['Python', 'Academy']

lista.insert(0, 'Blog')

print(lista)
```

Saída:

```
['Blog', 'Python', 'Academy']
```

Método `extend`

O método `extend(iterável)` adiciona os elementos de uma lista especificada (ou qualquer outro iterável) ao final da lista:

```
sacola = ['Laranja', 'Banana']
legumes = ['Xuxu', 'Batata']

sacola.extend(legumes)

print(sacola)
```

Saída:

```
['Laranja', 'Banana', 'Xuxu', 'Batata']
```

É possível concatenar as listas para obter o mesmo resultado em uma nova variável. Exemplo:

```
sacola = ['Laranja', 'Banana']
legumes = ['Xuxu', 'Batata']

juntos = sacola + legumes

print(juntos)
```

Saída:

```
['Laranja', 'Banana', 'Xuxu', 'Batata']
```

E também podemos percorrer uma das listas, adicionando elementos à outra com o método `append()`, assim:

```
sacola = ['Laranja', 'Banana']
legumes = ['Xuxu', 'Batata']

for legume in legumes:
    sacola.append(legume)

print(sacola)
```

Saída:

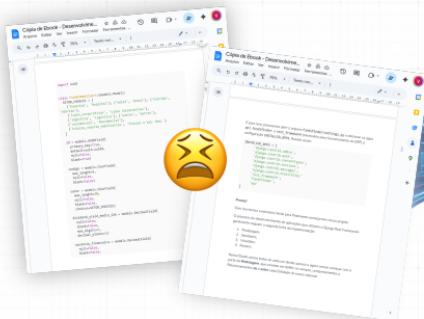
```
['Laranja', 'Banana', 'Xuxu', 'Batata']
```



*Estou desenvolvendo o **DevBook**, uma plataforma que usa IA para gerar ebooks técnicos profissionais. Não deixe de conferir!*

Crie Ebooks técnicos incríveis em minutos com IA

Conheça a 1ª IA Especializada na criação de Ebooks **com código!**



Chega de formatar código no Google Docs



Deixe que nossa IA faça o trabalho pesado

 Syntax Highlight

 Adicione Banners Promocionais

 Edite em Markdown em Tempo Real

 Infográficos feitos por IA

TESTE AGORA! PRIMEIRO CAPÍTULO 100% GRÁTIS 

Método `remove`

Para remover um item com valor especificado, use o método

`remove(elemento):`

```
lista = ['Blog', 'Python', 'Academy']
lista.remove('Blog')
print(lista)
```

Saída:

```
['Python', 'Academy']
```

Outra forma de remover elementos de uma lista é utilizando a função `del` do próprio Python, assim:

```
lista = [10, 20, 30, 40, 50]
del lista[2]
print(lista)
```

Saída:

```
[10, 20, 40, 50]
```

Com a `del` também é possível excluir uma lista completamente:

```
lista = [10, 20, 40, 50]
del lista
print(lista)
```

Aqui o Python irá retornar um erro, já que a variável `lista` não está mais definida no momento da chamada do `print`:

```
Traceback (most recent call last):
  File "listas_testes.py", line 3, in <module>
    print(lista)
NameError: name 'lista' is not defined
```

Método `pop`

Para remover um item do índice especificado e ainda retorná-lo, use o método `pop(indice)`, dessa forma:

```
lista = ['Banana', 'limão', 'Carro', 'Laranja']

item = lista.pop(2)

print('Item:', item)
print('Lista: ', lista)
```

Saída:

```
Item: Carro
Lista: ['Banana', 'limão', 'Laranja']
```

Método `clear`

Esse método é utilizado para remover todos os elementos de uma lista, dessa forma:

```
lista = [10, 20, 40, 50]

lista.clear()

print(lista)
```

Saída:

```
[]
```

Método `copy`

Esse método retorna uma cópia da lista especificada.

```
lista = ['Python', 'Academy']

lista_copiada = lista.copy()

print(lista_copiada)
print(lista)
```

Saída:

```
['Python', 'Academy']
['Python', 'Academy']
```

Método `reverse`

O método `reverse` é utilizado para reverter a ordem dos elementos de uma lista:

```
lista = [1, 2, 3, 4, 5]

lista.reverse()

print(lista)
```

Saída:

```
[5, 4, 3, 2, 1]
```

Método `sort`

Esse método é utilizado para ordenar a lista.

Também é possível criar uma função para definir seus próprios critérios de ordenação com `sort(key=funcao)`.

Exemplo:

```
lista = [1, 4, 5, 2, 4]  
  
lista.sort()  
  
print(lista)
```

Saída:

```
[1, 2, 4, 4, 5]
```

Adicionando o parâmetro `reverse=True`, é possível ordenar a lista em ordem decrescente.

Para deixar do modo padrão basta colocar `reverse=False`:

```
lista = [1, 4, 5, 2, 4]  
  
lista.sort(reverse=True)  
  
print(lista)
```

Saída:

```
[5, 4, 4, 2, 1]
```

Keywords de manipulação de Listas

Nesta parte veremos algumas *Keyword* da própria linguagem que utilizam listas como parâmetro, executando alguma ação.

Keyword len()

Retorna a quantidade de itens em uma lista, utilizando o método `len(iterável)`:

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

print('Quantidade de itens:', len(lista))
```

Saída:

```
Quantidade de itens: 6
```

Keyword min()

A função `min(iterável)` devolve o item com menor valor da lista ou iterável de entrada:

```
lista = [2, 4, 8, 1]

print('Menor valor da lista:', min(lista))
```

Saída:

```
Menor valor da lista: 1
```

Keyword `max()`

Retorne o maior valor da lista ou iterável especificado `mix(iterável)`:

```
lista = [2, 4, 8, 1]
print('Maior valor da lista:', max(lista))
```

Saída:

```
Maior valor da lista: 8
```

Keyword `sorted()`

A função `sorted()` é utilizada para ordenar a lista de entrada:

```
lista = [2, 4, 8, 1]
lista_ord = sorted(lista)
print(lista_ord)
```

Saída:

```
[1, 2, 4, 8]
```

Keyword `reversed()`

A função `reversed()` reverte a ordem da lista de entrada. **Exemplo:**

```
lista = [1, 2, 3, 4, 7]

for item in reversed(lista):
    print(item)
```

Saída:

```
7
4
3
2
1
```

Conclusão

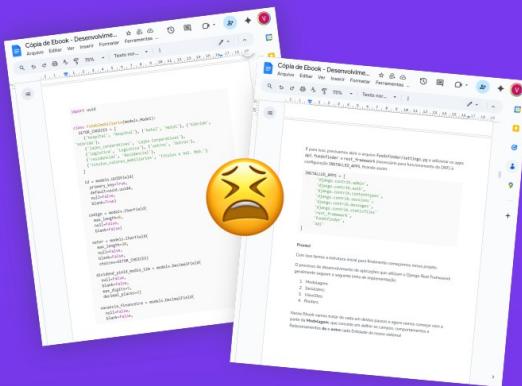
Esse post mostrou como o Python nos entrega tudo que precisamos para manipular Listas de uma maneira **rápida, prática e eficiente!**

Espero que esse post tenha sido útil pra você e nos vemos no próximo post!



Crie Ebooks técnicos em minutos com IA

Conheça a 1ª IA Especializada na criação de Ebooks **com código!**



Chega de formatar código no Google Docs



Arquitetura de Software Moderna

```
import python
import python

class Arquitetura_de_Software_Moderna:
    ...
    def share(self):
        pass
    ...
    return "Arquitetura de Mod", "arquitetura_mod"
}

def __init__(self):
    if user.username == self.username:
        self.username = self.username + self.username
        self.password = self.password + self.password
        self.name = self.name + self.name
    ...
    return self.username
}

resource saabell0
```

AI-generated system

A arquitetura com prolívia algoritmo software amadeirado de fusões modernas. Sesemtos tímicoscausus concretiza modulaçao estruturada externa. Chaveio e aonex dialektos AI-generated sistema si generated system oplemonia copiente enemot.

```
graph TD
    UserInput[User input] --> DataProcessor[Data processor]
    DataProcessor --> Agents[Agents]
    Agents --> Archestrator[Archestrator]
    Agents --> Cache[Cache]
    Agents --> Orchestrator[Orchestrator]
    SystemOutput[System output] --> DataProcessor
    Archestrator --> SystemOutput
```

Clean layout

Gentilmente Alia maticot en turbacit evicticos that alion ossibid to coenize Inugra with opegrath en oncees dibos. Net layout in gremarios formatacione exzma um dñivormour exzistem foa miltibid diginucleus, poiso ee dñor alour fumilat.



</> Syntax Highlight

Adicione Banners Promocionais

Deixe que nossa IA faça o trabalho pesado

Edite em Markdown em Tempo Real

TESTE AGORA

PRIMEIRO CAPÍTULO 100% GRÁTIS