

Listas em Python

Listas em Python

Uma lista é uma coleção mutável, o que significa que você pode adicionar, remover ou modificar elementos após a criação da lista. Os elementos em uma lista são mantidos em uma ordem específica e podem ser acessados por meio de índices.

Para criar uma lista, você pode usar colchetes [] e separar os elementos com vírgulas. Exemplo: minha_lista = [1, 2, 3, 4, 5]

Listas em Python

Os elementos de uma lista não precisam ser do mesmo tipo. Você pode ter uma lista que contém números, strings, booleanos e até mesmo outras listas.

Você pode acessar elementos individuais de uma lista usando índices. Os índices em Python começam em 0.

Principais métodos de Listas

```
minha_lista = []  
  
minha_lista.append(10)  
print(minha_lista)  
#saída: [10]
```

`append(item)`: Adiciona um item ao final da lista.

```
lista = [1,2,3,4,5]  
lista.insert(5,6)  
print(lista)  
#saída: [1, 2, 3, 4, 5, 6]
```

`insert(index, item)`: Insere um item em uma posição específica da lista, deslocando os demais elementos.

Principais métodos de Listas

```
lista1 = [1,3,5,7,9]
lista1.remove(9)
print(lista1)
#saída:[1, 3, 5, 7]
```

`remove(item)`: Remove o primeiro elemento encontrado na lista que seja igual ao item especificado.

```
lista = [1,2,3,4,20,50,60,100]
print(lista.index(100))
#saída: 7
```

`index(item)`: Retorna o índice da primeira ocorrência do item na lista.

Principais métodos de Listas

```
1 lista = [0,1,2,3,4,5]
2 lista.pop(1)#pedindo pra retirar o item que está na posição 1
3 print(lista)
```

`pop(index)`: Remove e retorna o elemento da lista que está na posição `index`. Se o índice não for fornecido, o último elemento é removido.

```
1 lista = [0,1,2,3,4,5]
2 lista.pop()#sem posição informada, retira o ultimo elemento da lista
3 print(lista)
```

Principais métodos de Listas

```
lista = ["Pedra", "Papel", "Tesoura", "Cola"]
lista.extend(["caderno", "lápis", "Borracha"])
print(lista)
#saída: ['Pedra', 'Papel', 'Tesoura', 'Cola', 'caderno', 'Lápis', 'Borracha']
```

`extend(iterável)`: Adiciona todos os elementos de um iterável (como outra lista) ao final da lista atual.

```
lista = [0,1,1,1,2,3,4,5]
print(lista.count(1))
#saída:3
```

`count(item)`: Retorna o número de ocorrências do item na lista.

Principais métodos de Listas

```
lista = [9,1,5,4,8,6,5,3,2]
lista.sort()
print(lista)
#saída:[1, 2, 3, 4, 5, 5, 6, 8, 9]
```

`sort()`: Ordena os elementos da lista em ordem crescente.

```
lista = ["Maria","Joana","Paulo","João"]
lista.reverse()
print(lista)
#saída:['João', 'Paulo', 'Joana', 'Maria']
```

`reverse()`: Inverte a ordem dos elementos na lista.

Principais métodos de Listas

`len(lista)`: Retorna o número de elementos na lista.

```
lista = ["Maria", "Joana", "Paulo", "João"]
print(len(lista))
#saída:4
```

`sum(lista)`: Retorna a soma de todos os elementos da lista.

```
lista = [1,2,3,4,5,6,7,8,9,10]
print(sum(lista))
#saída:55
```

Principais métodos de Listas

```
lista = [1,2,3,4,5,6,7,8,9,10]
print(min(lista))
print(max(lista))
#saída:
#1
#10
```

`min(lista)`: Retorna o menor elemento na lista.

`max(lista)`: Retorna o maior elemento da lista.

```
lista = [1,2,3,4,5,6,7,8,9,10]
lista2 = lista[2:8]
print(lista2)
#saída:[3, 4, 5, 6, 7, 8]
```

`lista[a:b]`: fatiamento de listas. Retorna o intervalo de elementos que inicia na posição de `a`, até `b`, mas sem incluir `b`.

Principais métodos de Listas

elemento in lista: Retorna True caso o elemento esteja dentro da lista.

```
lista = [1,2,3,4,5,6,7,8,9,10]
if 3 in lista:
    print("O 3 está na lista, retornou true")
#saída:O 3 está na lista, retornou true
```

elemento not in lista: Retorna True caso o elemento não esteja dentro da lista e False caso esteja.

```
lista = [1,2,3,4,5,6,7,8,9,10]
if 30 not in lista:
    print("O 30 não está na lista, retornou true")
#saída:O 30 não está na lista, retornou true
```

Principais métodos de Listas

```
lista = ["Casa", "Rua", "Logradouro", "Numero", "Apto"]
print(lista[3])
#saída: Numero
```

`lista[posição]`: Retorna o valor que está nessa posição na lista.

```
lista = ["Casa", "Rua", "Logradouro", "Numero", "Apto"]
lista[2] = "Rua Alto Do Bom Viver"
print(lista)
#saída: ['Casa', 'Rua', 'Rua Alto Do Bom Viver', 'Numero', 'Apto']
```

`lista[posição] = valor`: Substitui o elemento que estava na posição pelo valor estipulado após o sinal de igual

Principais métodos de Listas

```
lista1 = [2,4,6,8,10]
lista2 = lista1[0:3]
print(lista2)
#saída:[2, 4, 6]
```

lista2 = lista[a:b]: Retorna uma nova sequência com os elementos no intervalo do fatiamento

```
lista = [1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16]
del lista[2:7]
print(lista)
#saída:[1, 2, 8, 9, 10, 11, 12, 13, 14, 15, 16]
```

del lista[a:b]: Remove os elementos no intervalo do fatiamento (sem incluir o último).

Principais métodos de Listas

```
lista = [1,3,5,7,9,11]
lista2 = lista.copy()
print(lista2)
#saída: [1, 3, 5, 7, 9, 11]
```

`copy()`: Retorna uma cópia rasa
de uma lista.

Agora que já vimos todos os comandos de listas, antes de
continuarmos com tuplas, iremos exercitar!

Principais métodos de Listas

Existe também um `for` para percorrer listas:

```
lista = ["Aluno(a)", "Do", "Curso", "De", "Back", "End"]
#para cada item dentro da lista:
for i in lista:
    #mostre esse elemento
    print(i)
#Aluno(a)
#Do
#Curso
#De
#Back
#End
```

Principais métodos de Listas

O python e algumas outras linuagens consideram palavras como se fosse listas, mas é importante destacar que nem todo método de listas serve para uma string:

```
nome = "ALUNA"
for letra in nome:
    print(letra)
#A
#L
#U
#N
#A
```

E se por um acaso quisermos formar essa palavra, com as letras lado a lado?

Principais métodos de Listas

Existe um parâmetro que vai dentro do print chamado end. Dentro dele, ao ser usado com laço for, você determina como será feito no final. O for geralmente quebra a linha, por isso as coisas saem abaixo uma da outra!

```
nome = "ALUNA"
for letra in nome:
    #lebre-se de definir aquilo que vai ficar no final
    #dentro do parâetro end.
    #como nesse caso queremos um espaço,
    #colocamos uma string vazia
    print(letra, end=" ")
#A L U N A
```

Exercícios de Fixação

Escreva um programa que recebe uma lista de números e retorna uma nova lista apenas com os números pares.

Escreva uma função que recebe uma lista de palavras e retorna uma nova lista contendo apenas as palavras que possuem mais de 5 caracteres.

Escreva uma função que recebe duas listas e retorna uma nova lista contendo apenas os elementos comuns entre as duas listas.

Escreva um programa que recebe uma lista de números e retorna a soma dos elementos em posições ímpares.

Exercícios de Fixação

Escreva um programa que recebe uma lista de strings e retorna uma nova lista contendo as strings invertidas.

Escreva uma função que recebe uma lista de números e retorna o segundo maior número da lista.

Escreva um programa que recebe uma lista de nomes e retorna uma nova lista contendo apenas os nomes que começam com a letra "A".

Escreva uma função que recebe uma lista de números e retorna uma nova lista contendo apenas os números positivos.

Exercícios de Fixação

Escreva um programa que recebe duas listas e retorna uma nova lista contendo os elementos que aparecem em apenas uma das listas.

Escreva um programa que recebe uma lista de números e retorna o número que aparece com mais frequência.

Escreva um programa que recebe uma lista de strings e retorna uma nova lista contendo apenas as strings que são palíndromos.

Exercícios de Fixação

Escreva um programa que recebe uma lista de nomes e retorna uma nova lista contendo apenas os nomes que terminam com a letra "s".