Trabalhando com Lista, Funções e 'List Comprehension'

Calculando a Média de uma Lista

Exemplo:

```
notas = [7, 8.5, 9.5, 10]
media = sum(notas) / len(notas)
print(media)
```

Explicação: Este código calcula a média das notas armazenadas em uma lista. A função `sum(notas)` soma todos os elementos da lista `notas`, e `len(notas)` retorna o número de elementos na lista. Dividindo a soma pelo número de elementos, obtemos a média.

Contando Vogais em uma Frase

Exemplo:

```
frase = "Quero saber quantas vogais tem nessa frase"
vogais = "aeiouAEIOU"
contador_vogais = sum(1 for char in frase if char in vogais)
print(contador_vogais)
```

Explicação: Aqui, contamos quantas vogais existem em uma determinada frase. Usamos uma expressão geradora `(1 for char in frase if char in vogais)` para gerar um 1 para cada caractere na frase que também esteja no conjunto de caracteres `vogais`. A função `sum()` soma esses valores, resultando na contagem total de vogais.

Filtrando Números Pares de uma Lista

Exemplo:

```
numeros = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
pares = [num for num in numeros if num % 2 == 0]
print(pares)
```

Explicação: Este trecho de código utiliza list comprehension para filtrar e obter apenas os números pares de uma lista. `num % 2 == 0` verifica se o número é par. Se for, o número é incluído na nova lista `pares`.

Convertendo Strings para Maiúsculas

Exemplo:

```
palavras = ["python", "lista", "funções"]
palavras_maiusculas = [palavra.upper() for palavra in palavras]
print(palavras_maiusculas)
```

Explicação: Neste exemplo, cada string na lista `palavras` é convertida para maiúscula usando o método `.upper()`. A list comprehension cria uma nova lista com as palavras modificadas.

Criando uma Lista de Quadrados dos Números

Exemplo:

```
numeros = range(1, 6)
quadrados = [num ** 2 for num in numeros]
print(quadrados)
```

Explicação: Aqui, geramos os quadrados de números de 1 a 5 usando list comprehension. `num ** 2` eleva cada número ao quadrado.

Filtrando Itens Únicos de uma Lista

Exemplo:

```
itens = ["maçã", "banana", "maçã", "pera", "banana", "laranja"]
itens_unicos = list(set(itens))
print(itens_unicos)
```

Explicação: Este código remove itens duplicados da lista `itens` convertendo-a primeiro em um conjunto com `set(itens)`, que automaticamente descarta duplicatas, e então de volta para uma lista com `list(set(itens))`.

Encontrando o Menor e o Maior Valor

Exemplo:

```
valores = [45, 32, 89, 77, 12]
menor_valor = min(valores)
maior_valor = max(valores)
print(f"Menor: {menor_valor}, Maior: {maior_valor}")
```

Explicação: Utilizamos as funções `min()` e `max()` para encontrar, respectivamente, o menor e o maior valor dentro da lista `valores`.

Concatenando Listas

Exemplo:

```
lista1 = [1, 2, 3]
lista2 = [4, 5, 6]
lista3 = lista1 + lista2
print(lista3)
```

Explicação: Este trecho demonstra como concatenar duas listas usando o operador `+`. O resultado é uma nova lista que combina os elementos de ambas.

Multiplicando Todos os Elementos por um Número

Exemplo:

```
numeros = [1, 2, 3, 4, 5]
multiplicados = [num * 2 for num in numeros]
print(multiplicados)
```

Explicação: Multiplicamos cada elemento da lista `numeros` por 2. A list comprehension facilita a aplicação desta operação a todos os elementos.

Separando Strings em Listas

Exemplo:

```
frase = "Python é incrível"
palavras = frase.split()
print(palavras)
```

Explicação: O método `.split()` divide a string `frase` em uma lista de palavras, usando espaços como separador padrão.

Unindo Elementos de uma Lista em uma String

Exemplo:

```
palavras = ["Python", "é", "incrível"]
frase = " ".join(palavras)
print(frase)
```

Explicação: O método `.join()` une os elementos da lista `palavras` em uma única string, inserindo um espaço entre cada elemento.

Transformando Strings em Listas de Caracteres

Exemplo:

```
palavra = "Python"
caracteres = list(palavra)
print(caracteres)
```

Explicação: Ao passar a string `palavra` para a função `list()`, obtemos uma lista contendo cada caractere da string como um elemento separado.

Invertendo Strings

Exemplo:

```
s = "Python"
invertido = s[::-1]
print(invertido) # Saída nohtyP
```

Explicação: A sintaxe `s[::-1]` é um exemplo de fatiamento de strings que inverte a ordem dos caracteres.

Enumerando Itens de uma Lista

Exemplo:

```
frutas = ["maçã", "banana", "pera"]
for indice, fruta in enumerate(frutas, start=1):
    print(f"{indice}: {fruta}")
```

Explicação: `enumerate()` adiciona um contador aos itens da lista `frutas` e os retorna como pares de `indice` e `fruta`, facilitando a enumeração dos itens.

Filtrando Listas com `filter()`

Exemplo:

```
numeros = range(-5, 6)
positivos = list(filter(lambda x: x > 0, numeros))
print(positivos)
```

Explicação: Usamos `filter()` com uma função lambda para criar uma lista apenas com os números positivos. A função `lambda x: x > 0` retorna `True` para valores maiores que 0, e `filter()` aplica essa função a cada elemento de `numeros`, incluindo apenas os valores para os quais a função retorna `True`.

Conclusão

Cada um desses exemplos mostra diferentes maneiras de utilizar funções para manipular listas e strings, ilustrando a flexibilidade e eficiência de Python para processamento de dados e manipulação de strings.