Programação Python - Nível Intermediário

Listas e Como Iterar Sobre Elas

Listas em Python são estruturas de dados versáteis que armazenam coleções ordenadas de itens. Elas podem conter elementos de diferentes tipos, incluindo números, strings, e até mesmo outras listas. As listas são mutáveis, o que significa que seus elementos podem ser modificados após a criação.

Para criar uma lista, utilizamos colchetes “[ ]”:

```
frutas = ['maçã', 'banana', 'laranja']
numeros = [1, 2, 3, 4, 5]
```

A iteração sobre listas é uma operação fundamental em Python, permitindo processar cada elemento individualmente. O método mais comum para iterar sobre uma lista é usando o loop “for”:

```
frutas = ['maçã', 'banana', 'laranja']
for fruta in frutas:
    print(fruta)
```

Além do loop “for” básico, Python oferece outras formas de iteração:

1. **Usando enumerate()** - para obter o índice junto com o valor: `python for indice, fruta in enumerate(frutas): print(f"Índice {indice}: {fruta}")`
2. **Usando range()** - para iterar por índices: `python for i in range(len(frutas)): print(frutas[i])`
3. **List comprehensions** - para criar novas listas baseadas em listas existentes: `python frutas_maiusculas = [fruta.upper() for fruta in frutas]`

Criando e Usando Funções

Funções em Python são blocos de código reutilizáveis que executam uma tarefa específica. Elas ajudam a organizar o código, evitar repetição e tornar o programa mais

modular e fácil de manter. Uma função é definida usando a palavra-chave def, seguida pelo nome da função e parênteses que podem conter parâmetros.

Sintaxe básica de uma função:

```
def nome_da_funcao(parametro1, parametro2):
    # Corpo da função
    # Código a ser executado
    return resultado  # Opcional
```

Exemplos de funções:

1. **Função simples sem parâmetros:** *python def saudacao(): print("Olá, mundo!")*
2. **Função com parâmetros:** *python def somar(a, b): return a + b*
3. **Função com parâmetros padrão:** *python def saudacao_personalizada(nome="usuário"): print(f"Olá, {nome}!")*
4. **Função com número variável de argumentos:** *python def soma_varios(*numeros): return sum(numeros)*

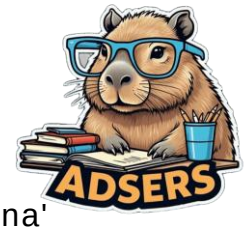
As funções em Python também podem ter escopo de variáveis (locais e globais) e podem ser aninhadas (funções dentro de funções).

Dicionários: Pares Chave-Valor

Dicionários em Python são estruturas de dados que armazenam coleções de pares chave-valor, onde cada chave é única. Diferentemente das listas que são indexadas por números inteiros, os dicionários são indexados por chaves, que podem ser de qualquer tipo imutável (como strings, números ou tuplas).

Para criar um dicionário, utilizamos chaves `{}` com pares chave-valor separados por dois pontos:

```
pessoa = {
    'nome':'Ana',
    'idade': 30,
    'profissao': 'Engenheira'
}
```



Operações comuns com dicionários:

1. **Acessar valores:** usando a chave: python nome = pessoa['nome'] #'Ana'
2. **Adicionar ou modificar valores:** python pessoa['email'] = 'ana@exemplo.com' # Adiciona nova chave-valor pessoa['idade'] = 31 # Modifica valor existente
3. **Verificar se uma chave existe:** python if 'nome' in pessoa: print("A chave 'nome' existe!")
4. **Iterar sobre um dicionário**: ```python # Iterar sobre chaves for chave in pessoa: print(chave)

Iterar sobre valores for valor in pessoa.values(): print(valor)

Iterar sobre pares chave-valor for chave, valor in pessoa.items(): print(f"{chave}: {valor}") ```

Métodos úteis:

1. `keys()`: retorna todas as chaves
2. `values()`: retorna todos os valores
3. `items()`: retorna todos os pares chave-valor
4. `get(chave, padrao)`: retorna o valor da chave ou um valor padrão se a chave não existir
5. `pop(chave)`: remove e retorna o valor da chave especificada

Referências:

Andrade, E. (2023). *3 Formas de Iterar uma Lista em Python*. DIO. Disponível em: https://www.dio.me/articles/3-formas-de-iterar-uma-lista-em-python

Soares, A. (2024). *For Python: aprenda a utilizar a loop for com exemplos*. Asimov Academy. Disponível em: https://hub.asimov.academy/blog/for-python/

Holanda, D. (2024). *Dicionários em Python: Chave e valor(es)*. DIO. Disponível em: https://www.dio.me/articles/dicionarios-em-python-chave-e-valores

Goldstein, B. F. (2023). *Python Aula 10 - Teórica: Estrutura de Dados Dicionário*. UFRJ. Disponível em: https://www.cos.ufrj.br/~bfgoldstein/python/compI/slides/aula10_teorica.pdf

Python Software Foundation. (2024). *5. Estruturas de dados — Documentação Python 3.13.3*. Disponível em: https://docs.python.org/pt-br/3.13/ tutorial/datastructures.html

Brandão, L. O. (2017). Introdução ao uso de funções em Python: variáveis locais, globais e aninhamento de funções. IME-USP. Disponível em:

https://www.ime.usp.br/~leo/mac2166/2017-1/line_py_introducao_funcoes.html - DIO. (2024). Explorando Funções em Python: O Poder de Definir e Utilizar Funções. Disponível em: https://www.dio.me/articles/explorando-funcoes-em-python-o-poder-de-definir-eutilizar-funcoes