Estruturas de Dados em Python para Ciência de Dados

Estruturas de dados são fundamentais em ciência de dados, pois permitem a organização, manipulação e análise eficiente de grandes volumes de dados. Compreender como usar e manipular diferentes tipos de coleções é crucial para realizar operações de processamento de dados, limpeza, análise estatística e aprendizado de máquina de forma eficaz.

---

1. Strings

O que é uma String

Uma string é uma sequência de caracteres delimitada por aspas simples ou duplas.

Indexação de Strings

Strings em Python são indexadas, o que significa que cada caractere na string tem uma posição associada.

Exemplo:

```
texto = "Python"
primeira_letra = texto[0]  # P
ultima_letra = texto[-1]   # n
```

Fatiamento de Strings

Fatiamento permite acessar substrings dentro de uma string.

Exemplo:

```
sub_texto = texto[1:4]  # yth
```

Propriedade Imutável de uma String

Strings são imutáveis, o que significa que seus caracteres não podem ser alterados após a criação.

Formatação de Strings

Formatação permite a inserção de valores em strings.

Exemplo:

```
nome = "Maria"
idade = 30
mensagem = f"Meu nome é {nome} e eu tenho {idade} anos."
```

Interação sobre Strings

Você pode iterar sobre cada caractere de uma string.

Exemplo:

```
for letra in "Python":
    print(letra)
```

Métodos Comuns de Strings

Exemplos:

```
texto = " ciência de dados "

# Remover espaços em branco
print(texto.strip())

# Converter para maiúsculas
print(texto.upper())

# Substituir substrings
print(texto.replace("dados", "informação"))

# Encontrar substrings
print(texto.find("ciência"))
```

Mais Alguns Métodos do Tipo `str`

Exemplos:

```
texto = "python é incrível"

# Dividir string em lista
print(texto.split())

# Verificar prefixo
print(texto.startswith("python"))

# Verificar sufixo
print(texto.endswith("incrível"))
```

---

3. Listas

O que é uma Lista

Uma lista é uma coleção ordenada e mutável de elementos.

Operações Básicas com Listas

Exemplo:

```
numeros = [1, 2, 3, 4, 5]
print(numeros[0])  # Primeiro elemento
print(numeros[-1]) # Último elemento
```

Métodos Comuns de Listas

Exemplos:

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

# Adicionar elemento
numeros.append(6)

# Remover elemento
numeros.remove(2)

# Ordenar lista
numeros.sort()

# Inverter lista
numeros.reverse()
```

Mais Métodos de uma Lista

Exemplos:

```
# Tamanho da lista
print(len(numeros))

# Contar ocorrências
print(numeros.count(3))

# Obter índice de elemento
print(numeros.index(4))
```

Compreensão em Listas

Compreensão de listas é uma forma concisa de criar listas.

Exemplo:

```
quadrados = [x**2 for x in range(10)]
```

Iteração sobre Listas

Exemplo:

```
for numero in numeros:
    print(numero)
```

---

4. Tuplas

O que é uma Tupla

Uma tupla é uma coleção ordenada e imutável de elementos.

Operações Básicas com Tuplas

Exemplo:

```
coordenadas = (10, 20)
print(coordenadas[0])  # Primeiro elemento
```

Vantagens das Tuplas

Tuplas são mais rápidas e ocupam menos espaço em comparação com listas.

Iteração do Tipo Tupla

Exemplo:

```
for coord in coordenadas:
    print(coord)
```

Principais Métodos do Tipo `tuple`

Exemplos:

```
# Tamanho da tupla
print(len(coordenadas))

# Contar ocorrências
print(coordenadas.count(10))

# Obter índice de elemento
print(coordenadas.index(20))
```

---

5. Dicionários

O que são Dicionários

Dicionários são coleções desordenadas de pares chave-valor.

Operações Básicas com Dicionários

Exemplo:

```
aluno = {"nome": "João", "idade": 20}
print(aluno["nome"])  # Acessar valor pela chave
```

Métodos Comuns de Dicionários

Exemplos:

```
aluno = {"nome": "João", "idade": 20}

# Adicionar par chave-valor
aluno["curso"] = "Ciência de Dados"

# Remover par chave-valor
del aluno["idade"]
```

```
# Obter todas as chaves
print(aluno.keys())

# Obter todos os valores
print(aluno.values())
```

Outros Métodos do Tipo `dict`

Exemplos:

```
# Obter itens
print(aluno.items())

# Limpar dicionário
aluno.clear()

# Copiar dicionário
novo_aluno = aluno.copy()
```

Iteração sobre Dicionários

Exemplo:

```
for chave, valor in aluno.items():
    print(f"{chave}: {valor}")
```

---

6. Exemplos

Exemplo 1: Manipulação de Strings

```
frase = "Python para Ciência de Dados"
palavras = frase.split()
for palavra in palavras:
    print(palavra.upper())
```

Exemplo 2: Operações com Listas

```
numeros = [2, 4, 6, 8, 10]
quadrados = [num**2 for num in numeros]
print(quadrados)
```

Exemplo 3: Trabalhando com Tuplas

```
pessoa = ("Ana", 28, "Engenheira")
nome, idade, profissao = pessoa
print(f"Nome: {nome}, Idade: {idade}, Profissão: {profissao}")
```

Exemplo 4: Uso de Dicionários

```
produto = {"nome": "Laptop", "preço": 1500, "estoque": 30}
for chave, valor in produto.items():
    print(f"{chave}: {valor}")
```

---

7. Exercícios

1. **Manipulação de Strings:**
   - Dada a string "Ciência de Dados", crie uma nova string onde todas as letras são maiúsculas e substitua "Dados" por "Informação".
2. **Operações com Listas:**
   - Crie uma lista de números ímpares de 1 a 20 e imprima a soma de todos os elementos.
3. **Trabalhando com Tuplas:**
   - Crie uma tupla com os números de 1 a 5 e verifique se o número 3 está presente na tupla.
4. **Uso de Dicionários:**
   - Crie um dicionário para armazenar as notas de um aluno em três disciplinas e calcule a média das notas.

---

8. Referências

- Documentação Oficial do Python
- Guia de Estilo para Código Python (PEP 8)