



Linguagem de Programação

Professor Anderson I. S. Abreu

Estruturas de dados em Python - Parte II



Tópicos da nossa aula:

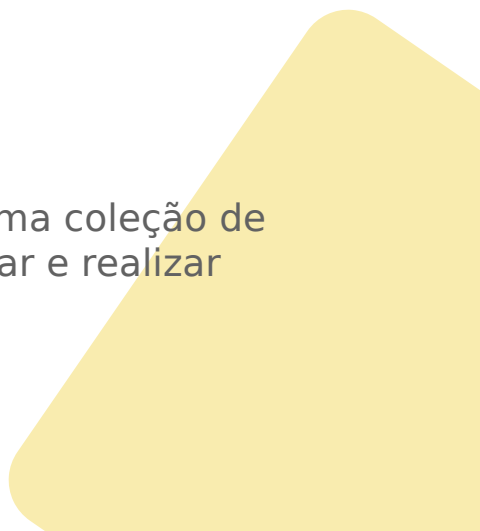
1. Objetos do tipo set;
2. Objetos do tipo Mapping;
3. Objetos do tipo Array NumPy.



1

Objetos do tipo set

Um conjunto, ou set, é uma estrutura de dados que representa uma coleção de elementos únicos, sem repetição. Veremos como criar, modificar e realizar operações com conjuntos.



Semelhança com conjuntos da matemática

Os objetos do tipo "set" habilitam operações de conjuntos, como união, interseção, diferença e muito mais;

Essa estrutura é especialmente útil para realizar testes de associação e eliminar valores duplicados em uma sequência;

Podemos adicionar um novo elemento a um conjunto usando `add(valor)` e remover elementos com `remove(valor)`.

Criar objeto do tipo set:

1. Usando um par de chaves e elementos separados por vírgulas, por exemplo: `set1 = {'a', 'b', 'c'}`.

2. Usando o construtor de tipo `set(iterable)` com um objeto iterável, como uma lista, uma tupla ou mesmo uma sequência de caracteres (string).

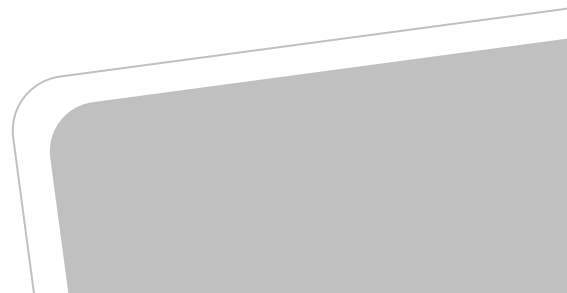
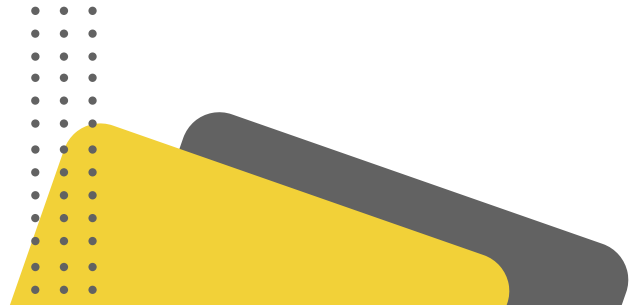
Criando a partir do vazio!

Vamos criar nosso conjunto agora.

Criar um conjunto vazio.

Adicionar elementos.

Verificar se existe um elemento dentro do conjunto.





2

Objetos do tipo mapping

Foco no dicionário (dict) em Python. Dicionários são estruturas que associam chaves a valores, permitindo o armazenamento e recuperação eficiente de informações. Veremos como criar dicionários, adicionar itens e realizar operações de busca.

Chaves e Valores

Dados que estabelecem uma relação entre chaves e valores são conhecidas como objetos do tipo mapping;

Em Python, o principal objeto que atende a essa propriedade é o dicionário, representado pelo tipo dict;

Dicionários são mutáveis, podemos modificar o valor associado a uma chave existente ou criar novas chaves;

Podemos criar dicionários em Python de várias maneiras:

1. Usando um par de chaves para denotar um dicionário vazio: `dicionario1 = {}`

2. Usando pares de elementos na forma "chave: valor" separados por vírgulas: `dicionario2 = {'um': 1, 'dois': 2, 'três': 3}`

3. Usando o construtor de tipo `dict()`.

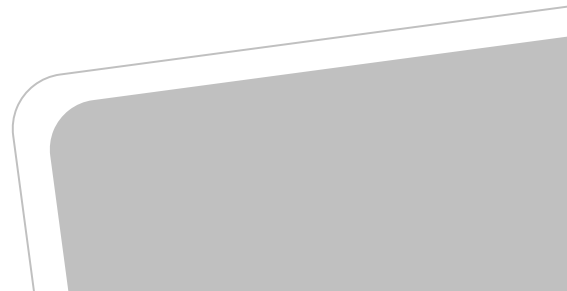
Criando dicionário a partir do vazio!

Vamos criar nosso dicionário agora.

Criar um dicionário vazio.

Adicionar elementos.

Verificar chaves e valores!





3

Objetos do tipo array NumPy

O NumPy é uma biblioteca essencial para computação científica em Python, fornecendo recursos avançados para manipular arrays multidimensionais. Veremos como criar, realizar operações e acessar elementos em arrays NumPy.

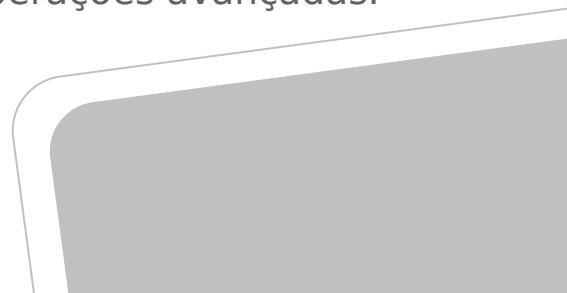
NumPy

Desenvolvida para suportar a computação científica com Python;

Oferece uma ampla gama de funcionalidades, incluindo arrays multidimensionais e funções sofisticadas;

Fornece ferramentas para integração com código em C/C++ e Fortran, bem como recursos essenciais de álgebra linear, transformada de Fourier e geração de números aleatórios;

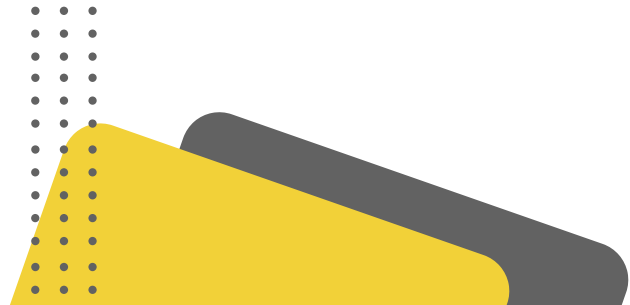
A biblioteca NumPy é particularmente valiosa para cientistas de dados e desenvolvedores de soluções de inteligência artificial, permitindo lidar eficientemente com matrizes de dados complexas e realizar operações avançadas.



Nada como a prática!

Vamos ver as vantagens de usar o NumPy !


Se você estiver interessado em explorar mais sobre o NumPy, pode encontrar a documentação completa em <https://numpy.org/>.





Aplicando a aula!

Suponha que estamos gerenciando um evento científico em que participantes de diferentes regiões do mundo se inscreveram. Cada participante forneceu informações sobre sua localização, afiliação a instituições de pesquisa e áreas de interesse. O objetivo é realizar análises sobre a distribuição geográfica dos participantes, suas afiliações e as áreas de interesse predominantes. Como podemos utilizar os conhecimentos dessa aula para resolver?



Obrigado!

