RELATÓRIO MÓDULO 2

Prática: Linguagem de Programação Python (I)

Willian Augusto Soder de Souza

O módulo 2 contém dois vídeos bem explicativos e abrangentes que têm por objetivo introduzir a linguagem de programação Python. Por ser a principal linguagem utilizada em algoritmos de aprendizado de máquina, essa introdução é extremamente importante, pois é fundamental ter uma boa base na linguagem para escrever algoritmos mais avançados.

Primeiramente, são explicados os conceitos básicos de criação de variáveis e seus tipos (inteiros, float, etc.). Em seguida, são apresentados conceitos importantes dentro de atribuições:

- **Listas:** São coleções ordenadas e mutáveis de itens, definidas usando colchetes ([]), podendo conter elementos de qualquer tipo, como números e strings.
- **Tuplas:** São coleções ordenadas e imutáveis de itens, definidas usando parênteses (()), podendo conter elementos de qualquer tipo, como números e strings.
- **Conjuntos:** São coleções não ordenadas e mutáveis de itens únicos, definidos usando chaves ({}), sem elementos duplicados.
- **Dicionários:** São coleções desordenadas e mutáveis de pares chave-valor, definidos usando chaves ({}), onde cada par é separado por dois pontos (:).

Após isso, são explicados de maneira exemplificada os operadores lógicos de comparação, adição, etc. Em qualquer linguagem, os operadores são cruciais, pois são responsáveis pela lógica principal do código. O próximo conteúdo abordado no vídeo são os laços de repetição:

- **For:** É usado para iterar sobre uma sequência (como listas, tuplas, dicionários, conjuntos ou strings), executando um bloco de código uma vez para cada item na sequência.
- **While:** Similar ao laço ‘for’, porém usado para repetir um bloco de código enquanto uma condição especificada for verdadeira.

Após isso, é apresentado um conceito de extrema importância não só em Python, mas em todas as linguagens de programação: funções e argumentos (parâmetros). Uma função é um bloco de código reutilizável que realiza uma tarefa específica, ajudando a organizar e reutilizar o código de forma modular e legível. Uma função é definida usando a palavra-chave def, seguida pelo nome da função, parênteses e dois pontos. O código da função é indentado abaixo da definição. Também é explicada a possibilidade de colocar uma função dentro da outra, o que possibilita poupar tempo de execução em alguns casos. Já os argumentos de função são valores que você pode passar para uma função quando a chama, permitindo que a função use esses valores em suas operações. Os argumentos são especificados dentro dos parênteses na definição da função.

Os próximos conceitos ensinados no vídeo são:

- **Map:** A função map aplica uma função a todos os itens em uma lista (ou qualquer outro iterável) e retorna um mapa objeto (um iterador). A função passada como argumento é aplicada a cada item do iterável.
- **Reduce:** A função reduce pertence ao módulo functools e é usada para aplicar uma função de duas entradas cumulativamente aos itens de um iterável, reduzindo-o a um único valor.
- **Comprehension:** É uma forma curta e rápida de criar listas, conjuntos ou dicionários a partir de outros iteráveis como listas, strings. Em vez de usar loops tradicionais (for), é possível

escrever tudo em uma linha usando uma sintaxe especial entre colchetes ([]), chaves ({}) ou chaves e colchetes ({}).
- **Lambdas:** São funções anônimas pequenas e de uma linha que podem ser definidas rapidamente usando a palavra-chave lambda. Elas são úteis quando é necessário uma função simples para operações rápidas sem precisar definir uma função tradicional usando def.

Por fim, o último assunto comentado no vídeo é a respeito de classes. Uma classe é um tipo de estrutura que define como um objeto deve ser criado, servindo como um modelo para criar objetos que possuem atributos (variáveis) e métodos (funções) associados. Classes são fundamentais para a programação orientada a objetos (POO), permitindo encapsular dados e funcionalidades relacionadas em um único objeto. Os principais pontos destacados dentro desse assunto são:

- **Property:** É uma função embutida que permite definir métodos para acessar e modificar atributos de uma classe de maneira controlada. Ela é usada para implementar o conceito de "getter" (permite definir um método para retornar o valor de um atributo) e "setter" (permite definir um método para modificar o valor de um atributo), oferecendo mais controle sobre como os atributos são acessados e alterados.
- **Classmethod:** São métodos que operam na classe em si, em vez de operar em instâncias individuais da classe. Eles têm acesso aos atributos da classe, mas não aos atributos específicos de cada instância.
- **Staticmethod:** É um método que não recebe implicitamente uma referência para a instância (self) ou para a classe (cls). Eles são métodos que operam independentemente de qualquer instância específica ou da própria classe.

Este relatório inclui os principais tópicos abordados nas aulas, oferecendo uma visão geral dos conceitos ensinados. Para exemplos específicos de uso e outros pontos, consulte o código em anexo.