3. Funções e Módulos – Organizando o Raciocínio

Funções são a maneira de **modularizar** e organizar o código, reunindo um conjunto de comandos sob um nome para possibilitar o **reuso de código**.

Declaração e Execução de Funções:

A função pode ser vista como uma "caixa preta": você envia valores (parâmetros) e ela devolve um resultado.

• **Sintaxe:** Uma função é declarada usando a palavra-chave **def**, seguida pelo nome da função, parênteses (para os parâmetros) e o corpo de comandos indentado.

• **Parâmetros:** São os valores passados para a função. Podem ser **nomeados** e obrigatórios, ou podem ter **valores padrão** definidos, permitindo que o programador omita a passagem desses valores na chamada.

• **Retorno:** A palavra-chave **return** é usada para devolver o resultado da função.

• **Ordem:** É essencial que a função seja declarada **antes** de ser chamada, pois Python é uma linguagem interpretada e a execução é sequencial.

• **Recursividade:** Uma função pode invocar a si mesma. Este recurso exige uma **condição de parada** bem definida para evitar o erro de *stack overflow* (estouro de pilha), que ocorre quando o espaço de memória LIFO (Last In First Out) da pilha esgota-se por excesso de chamadas.

• **Métodos:** Funções que pertencem a uma classe são chamadas de métodos.

Reutilização de Código com Módulos:

A modularização permite que funções e classes sejam organizadas em arquivos separados, chamados **módulos**.

• **Importação:** Utiliza-se a palavra-chave **import** para chamar um módulo (ex: import math). É possível importar elementos específicos usando a sintaxe from <módulo> import <elemento> (ex: from math import pi).