## Funções

* Uma função é um bloco de código construído para execução futura.
  + Esse código fica pronto para execução, mas depende de uma chamada para realmente executar.
  + Ou seja, ao encontrar a declaração de uma função, o programa **NÃO** executa aquele código.
  + Apenas o deixa na memória.
* Em algum momento subsequente, pode-se CHAMAR a função.
* Isso funciona como um desvio de fluxo:
  + O código para de executar no ponto em que é feita a chamada da função.
  + A execução é desviada para o código da função, que então é executada.
  + Por fim, a execução retorna para o ponto em que houve a chamada da função, continuando dali.
* Sintaxe.
  + Declaração da função:

*def nome\_da\_funcao(parametros):*

*bloco\_de\_codigo*

*return informação*

* Chamada da função:

*nome\_da\_função(parametros)*

* Detalhando os elementos de uma função.
  + nome\_da\_funcao:
    - Obedece às mesmas regras da criação de uma variável.
  + parâmetros:
    - Podem aparecer em qualquer quantidade, inclusive podem não haver parâmetros.
    - Um parâmetro é uma variável local da função, que recebe de fora, no momento da chamada, as informações de que a função necessita para operar.
    - Essas variáveis só existirão (só serão acessíveis) durante a execução da função.
    - Podem ser passadas como parâmetros informações de quaisquer tipos, desde valores simples como inteiros, booleanos, floats, até strings e listas (vetores, matrizes).
    - Cópias dos valores fornecidos são feitas nas variáveis dos parâmetros.
  + Retorno da função:
    - É opcional.
    - Comando *return*.
    - Assim que o comando return é executado, a função é IMEDIATAMENTE encerrada.
      * Mesmo que estejamos no meio de um laço, ou que haja mais linhas de código após o comando return.
    - O valor retornado é enviado para o local onde a função foi chamada, e pode ser usado de diversas formas:
      * Atribuído a alguma variável;
      * Em uma operação matemática;
      * Passado como parâmetro para outra função;
      * Etc.