**Para saber mais: PIP e PyPI**

O PIP é um **gerenciador de bibliotecas** para Python. Nele temos acesso às bibliotecas já instaladas em nossa aplicação junto a versão de cada uma delas. A partir das linhas de comando, podemos instalar, atualizar e remover pacotes Python. Com o pip, podemos facilmente instalar pacotes de terceiros em projetos Python.

Para conseguirmos acessar todos os pacotes instalados em nosso Jupyter Notebook, no Colab, podemos escrever o seguinte código:

# Imprimindo todos os pacotes instalados no ambiente e suas versões

**!pip** list

O pip funciona conectando-se ao [Python Package Index (PyPI)](https://pypi.org/) que é o maior repositório centralizado para pacotes Python com milhares de bibliotecas disponíveis para instalação. Podemos pesquisar no PyPI para encontrar pacotes que atendam às necessidades e, em seguida, usar o pip para instalá-los em nossos projetos.

O PyPI é mantido pela [Python Software Foundation](https://www.python.org/psf-landing/) e contém uma ampla variedade de pacotes Python. Nele são publicados pacotes Python para que outras pessoas desenvolvedoras possam usá-los. Logo, o pip e o PyPI são duas ferramentas importantes em Python.

**Para saber mais: outras formas de importação**

Já trabalhamos com duas formas de importação de pacotes: o import nome\_biblioteca para todo o pacote; e o from nome\_biblioteca import metodo para apenas um método de uma dada biblioteca.

A importação de métodos específicos de uma biblioteca pode trazer algumas vantagens para o nosso projeto, como:

* **Maior clareza no código**: importar apenas os métodos que vamos usar torna o código mais claro e fácil de entender.
* **Redução de conflitos de nome**: quando importamos uma biblioteca inteira, podemos acabar tendo conflitos de nome com outras variáveis ou funções em nosso código.

Além das formas vistas anteriormente, podemos citar mais dois exemplos que podemos encontrar ao longo de suas práticas e estudos da linguagem Python:

**from nome\_biblioteca import met\_1, met\_2**

Este código resulta na importação de 2 ou mais métodos de uma biblioteca, não necessitando repetir a importação desta a cada método desejado. Podemos, por exemplo, importar 2 métodos da biblioteca *random* para colher uma amostra de 5 valores de uma lista de 20 valores gerada aleatoriamente com números de 0 a 99.

**from** random **import** randrange, sample

lista = []

**for** i **in** range(0, 20):

lista.append(randrange(100))

sample(lista, 5)

Copiar código

**Saída:** [28, 66, 53, 81, 85]

**from nome\_biblioteca import \***

Esta forma é utilizada para importar todos os métodos de uma dada biblioteca. A diferença desta para o import nome\_biblioteca é que, neste caso, não precisamos usar o nome da biblioteca para chamar um método. Podemos passar apenas o nome dele. Por exemplo, se formos calcular a raiz quadrada de certo número poderíamos seguir por uma das duas formas:

**Usando import nome\_biblioteca**

**import** math

n = int(input("Digite um número positivo para calcular sua raiz quadrada:"))

print(f"\nA raiz quadrada de {n} é igual a {math.sqrt(n)}")

Copiar código

**Usando from nome\_biblioteca import \***

**from** math **import** \*

n = int(input("Digite um número positivo para calcular sua raiz quadrada:"))

print(f"\nA raiz quadrada de {n} é igual a {sqrt(n)}")

Copiar código

Note que, no segundo exemplo, suprimimos o nome math utilizando o método desejado e escrevendo o código com menos caracteres.

**Atenção**: A importação nesse sentido precisa de alguns cuidados:

* Podemos ter choque de nomes entre as variáveis. Por exemplo: no caso de termos uma função chamada sqrt antes de importar a da biblioteca math.
* Não fica explícito de onde aquela variável, método ou classe veio.