

# Módulos Python

quinta-feira, 8 de setembro de 2022 16:44

Módulos são bibliotecas que ajudam no desenvolvimento do código, sendo eles módulos internos ou externos, internos usando importação, externos instalando e importando.

Sintaxe para importar um módulo:

```
import sys
```

Quando importamos o módulo assim precisamos sempre no início das funções o nome do módulo.

Exemplo:

```
import sys

print(sys.platform)
```

Quando o desenvolvedor quer utilizar apenas uma função do módulo usamos outra sintaxe:

Exemplo:

```
from sys import platform

print(platform)
```

Podemos também adicionar um apelido para a função solicitada do módulo:

Exemplo:

```
from sys import platform as apelido

print(apelido)
```

As funções dentro das bibliotecas python podem ser reescritas e funcionam dessa forma:

Exemplo:

```
from random import randint

def randint():
    return 'hahaha'

for i in range(10):
    print(randint())
```

A função randint tem como função principal sortear números inteiros, porém foi reescrita para escrever os valores hahaha na tela.

quando isso acontece podemos apelidar a função original.

Para importar tudo de um módulo podemos usar o \*

```
from random import *

print(randint(0,10))
```

Para fazer a instalação de um módulo podemos ir no terminal no python e usar o comando pip

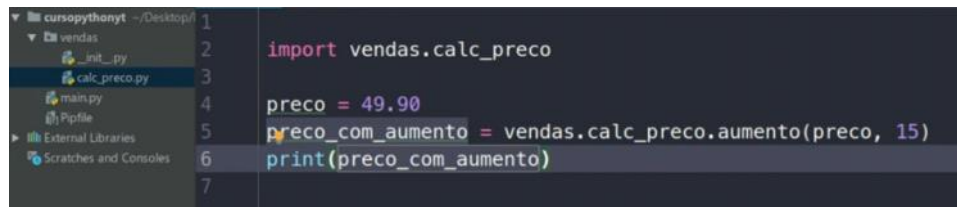
```
PS C:\Users\geron\PycharmProjects\Cursoudemy> pip install pymysql
```

dessa forma vai instalar um módulo externo.

## Criando um Pacote

Para a criação de um pacote deve-se ter dentro desse pacote um modulo chamado `__init__` e dentro do pacote você pode criar um modulo para receber funções:

Exemplo:

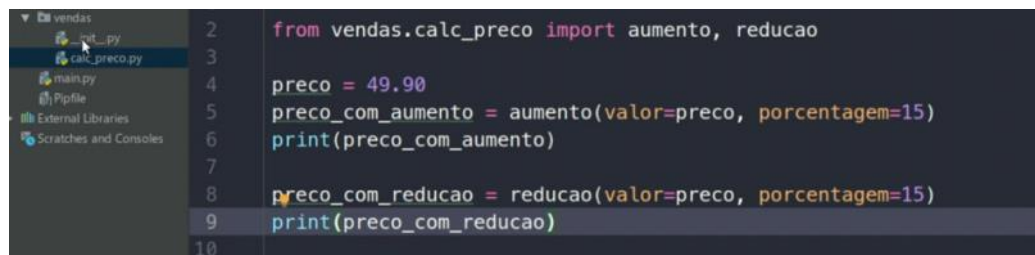


```
1 import vendas.calc_preco
2
3
4 preco = 49.90
5 preco_com_aumento = vendas.calc_preco.aumento(preco, 15)
6 print(preco_com_aumento)
7
```

Pacote Vendas e módulo `calc_preco`

para importar o pacote e modulo funciona da forma acima.

Podendo ser importado dessa forma

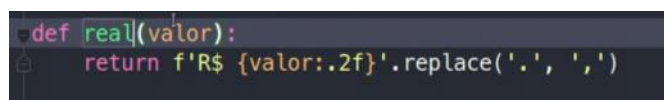


```
1 from vendas.calc_preco import aumento, reducao
2
3
4 preco = 49.90
5 preco_com_aumento = aumento(valor=preco, porcentagem=15)
6 print(preco_com_aumento)
7
8 preco_com_reducao = reducao(valor=preco, porcentagem=15)
9 print(preco_com_reducao)
10
```

Para fazer uma formatação do valor pra ficar com , ao inves de ponto criamos uma função de formata e usamos como parâmetro False e se tiver True formata o valor



```
1 from vendas.calc_preco import aumento, reducao
2 from vendas.formata import preco
3
4
5 preco = 49.90
6 preco_com_aumento = aumento(valor=preco, porcentagem=15, formata=True)
7 print(preco_com_aumento)
8
9 preco_com_reducao = reducao(valor=preco, porcentagem=15, formata=True)
10 print(preco_com_reducao)
```



```
1 def real(valor):
2     return f'R$ {valor:.2f}'.replace('.', ',')
```

```

from vendas.formata import preco

def aumento(valor, porcentagem, formata=False):
    r = valor + (valor * (porcentagem / 100))

    if formata:
        return preco.real(r)
    else:
        return r

def reducao(valor, porcentagem, formata=False):
    r = valor - (valor * (porcentagem / 100))

    if formata:
        return preco.real(r)
    else:
        return r

```

um modulo com pacote ficando com nome mt grande podemos apelida-lo para ficar mais facil

```

import vendas.formata.preco as formata

```