



# SEMANA DO PYTHON DA HASHTAG

## Apostila Completa Aula 1

Aprenda a automatizar qualquer processo ou  
sistema com o Python  
Impressionador do absoluto zero!



# Parte 1

# Introdução

# O que vamos aprender

Nas primeira aula da Semana do Python você vai aprender a criar um código de automação de análise de dados e elaboração de relatórios do **absoluto zero**. Para isso, vamos passar por conceitos como:

Jupyter Notebook

Variáveis, métodos

Importação de bibliotecas

Uso de bibliotecas  
(pyautogui, time, pandas e  
pyperclip)

Enviar e-mails  
automaticamente

Após todos esses conhecimentos, seremos capazes de transformar uma tabela cheia de informações, nem um pouco fáceis de serem interpretadas ...

... em uma ferramenta automatizada de geração e envio automático de relatórios para um destinatário pré-definido

	A	B	C	D	E	F	G
1	Código Vend	Data	ID Loja	Produto	Quantid	Valor Unit	Valor Fin
2	65014	01/12/2019	Shopping Morumbi	Sunga Listrado	5	R\$ 114,00	R\$ 570,00
3	65014	01/12/2019	Shopping Morumbi	Casaco Listrado	1	R\$ 269,00	R\$ 269,00
4	65016	01/12/2019	Iguatemi Campinas	Sapato Listrado	2	R\$ 363,00	R\$ 726,00
5	65016	01/12/2019	Iguatemi Campinas	Casaco	1	R\$ 250,00	R\$ 250,00
6	65017	01/12/2019	Shopping SP Market	Gorro Liso	3	R\$ 92,00	R\$ 276,00
7	65018	01/12/2019	Rio Mar Shopping Fortaleza	Cueca Estampa	1	R\$ 66,00	R\$ 66,00
8	65018	01/12/2019	Rio Mar Shopping Fortaleza	Sunga Xadrez	1	R\$ 116,00	R\$ 116,00
9	65018	01/12/2019	Rio Mar Shopping Fortaleza	Casaco Listrado	1	R\$ 269,00	R\$ 269,00
10	65019	01/12/2019	Shopping União de Osasco	Polo Xadrez	2	R\$ 142,00	R\$ 284,00
11	65019	01/12/2019	Shopping União de Osasco	Tênis Linho	1	R\$ 294,00	R\$ 294,00
12	65020	01/12/2019	Shopping Morumbi	Pulseira Listrado	4	R\$ 79,00	R\$ 316,00
13	65020	01/12/2019	Shopping Morumbi	Camiseta Xadrez	5	R\$ 200,00	R\$ 1.000,00
14	65020	01/12/2019	Shopping Morumbi	Camisa Listrado	2	R\$ 108,00	R\$ 216,00
15	65024	01/12/2019	Passei das Águas Shopping	Camiseta Estampa	2	R\$ 196,00	R\$ 392,00
16	65025	01/12/2019	Shopping Ibirapuera	Calça Xadrez	5	R\$ 185,00	R\$ 925,00
17	65025	01/12/2019	Shopping Ibirapuera	Short Listrado	1	R\$ 102,00	R\$ 102,00
18	65025	01/12/2019	Shopping Ibirapuera	Meia Listrado	1	R\$ 37,00	R\$ 37,00
19	65027	01/12/2019	Norte Shopping	Relógio Listrado	1	R\$ 218,00	R\$ 218,00
20	65028	01/12/2019	Iguatemi Campinas	Sapato	5	R\$ 350,00	R\$ 1.750,00
21	65028	01/12/2019	Iguatemi Campinas	Pulseira Xadrez	1	R\$ 87,00	R\$ 87,00
22	65030	01/12/2019	Shopping Iguatemi Fortaleza	Relógio Liso	1	R\$ 216,00	R\$ 216,00



<input type="checkbox"/>	pythonimpressionador	Caixa de entrada	Loja: Todas as Lojas - Coe Lira, Valor Final Ticket Médio ID Loja Iguatemi Campin...	28 de jan.
<input type="checkbox"/>	pythonimpressionador	Caixa de entrada	Loja: Salvador Shopping - Coe Lira, Quantidade Valor Final Ticket Médio ID Loja Salvador Shop...	28 de jan.
<input type="checkbox"/>	pythonimpressionador	Caixa de entrada	Loja: Shopping Morumbi - Coe Lira, Quantidade Valor Final Ticket Médio ID Loja Shopping Mor...	28 de jan.
<input type="checkbox"/>	pythonimpressionador	Caixa de entrada	Loja: Shopping Iguatemi Fortaleza - Coe Lira, Quantidade Valor Final Ticket Médio ID Loja Sho...	28 de jan.
<input type="checkbox"/>	pythonimpressionador	Caixa de entrada	Loja: Ribeirão Shopping - Coe Lira, Quantidade Valor Final Ticket Médio ID Loja Ribeirão Shop...	28 de jan.
<input type="checkbox"/>	pythonimpressionador	Caixa de entrada	Loja: Shopping SP Market - Coe Lira, Quantidade Valor Final Ticket Médio ID Loja Shopping SP...	28 de jan.
<input type="checkbox"/>	pythonimpressionador	Caixa de entrada	Loja: Palladium Shopping Curitiba - Coe Lira, Quantidade Valor Final Ticket Médio ID Loja...	28 de jan.
<input type="checkbox"/>	pythonimpressionador	Caixa de entrada	Loja: Rio Mar Recife - Coe Lira, Quantidade Valor Final Ticket Médio ID Loja Rio Mar Recife B8...	28 de jan.
<input type="checkbox"/>	pythonimpressionador	Caixa de entrada	Loja: Novo Shopping Ribeirão Preto - Coe Lira, Quantidade Valor Final Ticket Médio ID Loja No...	28 de jan.
<input type="checkbox"/>	pythonimpressionador	Caixa de entrada	Loja: Shopping Vila Velha - Coe Lira, Quantidade Valor Final Ticket Médio ID Loja Shopping Vil...	28 de jan.
<input type="checkbox"/>	pythonimpressionador	Caixa de entrada	Loja: Shopping Eldorado - Coe Lira, Quantidade Valor Final Ticket Médio ID Loja Shopping Eld...	28 de jan.
<input type="checkbox"/>	pythonimpressionador	Caixa de entrada	Loja: Shopping União de Osasco - Coe Lira, Quantidade Valor Final Ticket Médio ID Loja Shop...	28 de jan.
<input type="checkbox"/>	pythonimpressionador	Caixa de entrada	Loja: Center Shopping Uberlândia - Coe Lira, Quantidade Valor Final Ticket Médio ID Loja Cent...	28 de jan.
<input type="checkbox"/>	pythonimpressionador	Caixa de entrada	Loja: Parque Dom Pedro Shopping - Coe Lira, Quantidade Valor Final Ticket Médio ID Loja Par...	28 de jan.
<input type="checkbox"/>	pythonimpressionador	Caixa de entrada	Loja: Shopping Center Interlagos - Coe Lira, Quantidade Valor Final Ticket Médio ID Loja Shop...	28 de jan.
<input type="checkbox"/>	pythonimpressionador	Caixa de entrada	Loja: Bourbon Shopping SP - Coe Lira, Quantidade Valor Final Ticket Médio ID Loja Bourbon S...	28 de jan.

# Introdução

## Entendendo a base de dados

As informações que vão alimentar o nosso código, serão dados referentes a vendas de vários produtos de diferentes lojas.

A imagem ao lado, mostra as primeiras linhas da tabela. As informações que temos em cada uma das colunas são:

- ☒ Código venda
- ☒ Data
- ☒ ID Loja
- ☒ Produto
- ☒ Quantidade
- ☒ Valor Unitário
- ☒ Valor Final

Portanto, para um melhor entendimento, na linha 2 temos o seguinte:

Uma venda com código **65014** no dia **01/01/2019**, na loja **Shopping Morumbi** do Produto **Sunga Lustrado** em uma quantidade igual a **5**, a um preço de **R\$ 114,00** por produto, totalizando uma venda de **R\$ 570,00**.

Como você pode ver, temos muitas informações de vendas nessa tabela, e qualquer interpretação desses dados não é uma tarefa fácil, o que é um problema.

	A	B	C	D	E	F	G
1	Código Vend	Data	ID Loja	Produto	Quantid	Valor Unit	Valor Final
2	65014	01/12/2019	Shopping Morumbi	Sunga Lustrado	5	R\$ 114,00	R\$ 570,00
3	65014	01/12/2019	Shopping Morumbi	Casaco Lustrado	1	R\$ 269,00	R\$ 269,00
4	65016	01/12/2019	Iguatemi Campinas	Sapato Lustrado	2	R\$ 363,00	R\$ 726,00
5	65016	01/12/2019	Iguatemi Campinas	Casaco	1	R\$ 250,00	R\$ 250,00
6	65017	01/12/2019	Shopping SP Market	Gorro Liso	3	R\$ 92,00	R\$ 276,00
7	65018	01/12/2019	Rio Mar Shopping Fortaleza	Cueca Estampa	1	R\$ 66,00	R\$ 66,00
8	65018	01/12/2019	Rio Mar Shopping Fortaleza	Sunga Xadrez	1	R\$ 116,00	R\$ 116,00
9	65018	01/12/2019	Rio Mar Shopping Fortaleza	Casaco Lustrado	1	R\$ 269,00	R\$ 269,00
10	65019	01/12/2019	Shopping União de Osasco	Polo Xadrez	2	R\$ 142,00	R\$ 284,00
11	65019	01/12/2019	Shopping União de Osasco	Tênis Linho	1	R\$ 294,00	R\$ 294,00
12	65020	01/12/2019	Shopping Morumbi	Pulseira Lustrado	4	R\$ 79,00	R\$ 316,00
13	65020	01/12/2019	Shopping Morumbi	Camiseta Xadrez	5	R\$ 200,00	R\$ 1.000,00
14	65020	01/12/2019	Shopping Morumbi	Camisa Lustrado	2	R\$ 108,00	R\$ 216,00
15	65024	01/12/2019	Passei das Águas Shopping	Camiseta Estampa	2	R\$ 196,00	R\$ 392,00
16	65025	01/12/2019	Shopping Ibirapuera	Calça Xadrez	5	R\$ 185,00	R\$ 925,00
17	65025	01/12/2019	Shopping Ibirapuera	Short Lustrado	1	R\$ 102,00	R\$ 102,00
18	65025	01/12/2019	Shopping Ibirapuera	Meia Lustrado	1	R\$ 37,00	R\$ 37,00
19	65027	01/12/2019	Norte Shopping	Relógio Lustrado	1	R\$ 218,00	R\$ 218,00
20	65028	01/12/2019	Iguatemi Campinas	Sapato	5	R\$ 350,00	R\$ 1.750,00
21	65028	01/12/2019	Iguatemi Campinas	Pulseira Xadrez	1	R\$ 87,00	R\$ 87,00
22	65030	01/12/2019	Shopping Iguatemi Fortaleza	Relógio Liso	1	R\$ 216,00	R\$ 216,00
23	65030	01/12/2019	Shopping Iguatemi Fortaleza	Camisa Gola V Xadrez	1	R\$ 107,00	R\$ 107,00
24	65031	01/12/2019	Shopping Center Interlagos	Meia Liso	4	R\$ 38,00	R\$ 152,00
25	65032	01/12/2019	Shopping Iguatemi Fortaleza	Sunga	2	R\$ 100,00	R\$ 200,00
26	65032	01/12/2019	Shopping Iguatemi Fortaleza	Pulseira Estampa	3	R\$ 87,00	R\$ 261,00
27	65033	01/12/2019	Iguatemi Campinas	Casaco Estampa	1	R\$ 256,00	R\$ 256,00
28	65034	01/12/2019	Shopping Vila Velha	Camisa Gola V Liso	4	R\$ 118,00	R\$ 472,00
29	65034	01/12/2019	Shopping Vila Velha	Sunga Lustrado	5	R\$ 384,00	R\$ 1.920,00



# Entendendo a solução final

Nosso exemplo se trata de um relatório diário que precisa ser enviado para a diretoria diariamente. Essa tarefa é repetitiva e não agrega valor ao processo.

Portanto, uma solução para o problema, é construir um *código* que reduza o nosso trabalho operacional, nos tornando mais eficientes.

Mas o que é um código? Vamos dizer que um código é como uma receita de cozinha para computadores. Nesse código, temos o passo a passo que indica ao computador o que fazer, e como fazer.

O principal objetivo desse código é permitir uma melhor interpretação dos dados, em termos de qualidade e velocidade de análise.

O nosso código disparará um e-mail de análise de indicadores para a diretoria automaticamente!

Então, vamos começar!

### Desafio:

Todos os dias, o nosso sistema atualiza as vendas do dia anterior. O seu trabalho diário, como analista, é enviar um e-mail para a diretoria, assim que começar a trabalhar, com o faturamento e a quantidade de produtos vendidos no dia anterior

E-mail da diretoria: [seugmail+diretoria@gmail.com](mailto:seugmail+diretoria@gmail.com)

Local onde o sistema disponibiliza as vendas do dia anterior: [https://drive.google.com/drive/folders/1mhXZ3JPAnekXP\\_4vX7Z\\_sJj35VWqayaR?usp=sharing](https://drive.google.com/drive/folders/1mhXZ3JPAnekXP_4vX7Z_sJj35VWqayaR?usp=sharing)

Para resolver isso, vamos usar o pyautogui, uma biblioteca de automação de comandos do mouse e do teclado

Relatório de Vendas de Ontem - Prezados, bom dia O faturamento de ontem foi de: R\$2917311.00 A quantidade de produtos foi de: 15227 Abs LiraPython –

Daniel Candiottto <daniel. [REDACTED]>  
para mim ▾

Prezados, bom dia

O faturamento de ontem foi de: R\$2,917,311.00  
A quantidade de produtos foi de: 15,227

Abs  
LiraPython

Parte 2

# O que é o Python

**INTENSIVÃO DE {#}**  
**PYTHON{#}**  
100% ONLINE & GRATUITO

# O que é o Python

O Python, é uma linguagem de programação.

Ok.... Mas o que é uma linguagem de programação??

Assim como temos diferentes línguas para falarmos, existem diversas línguas que nos permitem “falar” com os computadores.

Entre as línguas de produção, o Python é uma das mais fáceis de aprender e uma das que mais cresce no mundo em termos de utilização.

Pode ser utilizado em diversas áreas:

- Data Science;
- Automação de processos;
- Desenvolvimento de sites;
- Inteligência artificial;
- Vários outros

**Curiosidade:** Seu nome apesar de geralmente ser vinculado a cobra, não tem essa origem.... Na verdade, ele é uma homenagem a um grupo de comédia inglês chamado Monty Python 😊



Parte 3

# Jupyter Notebooks



# O que é? Como acesso?

Os códigos em Python precisam de uma plataforma para serem escritos.

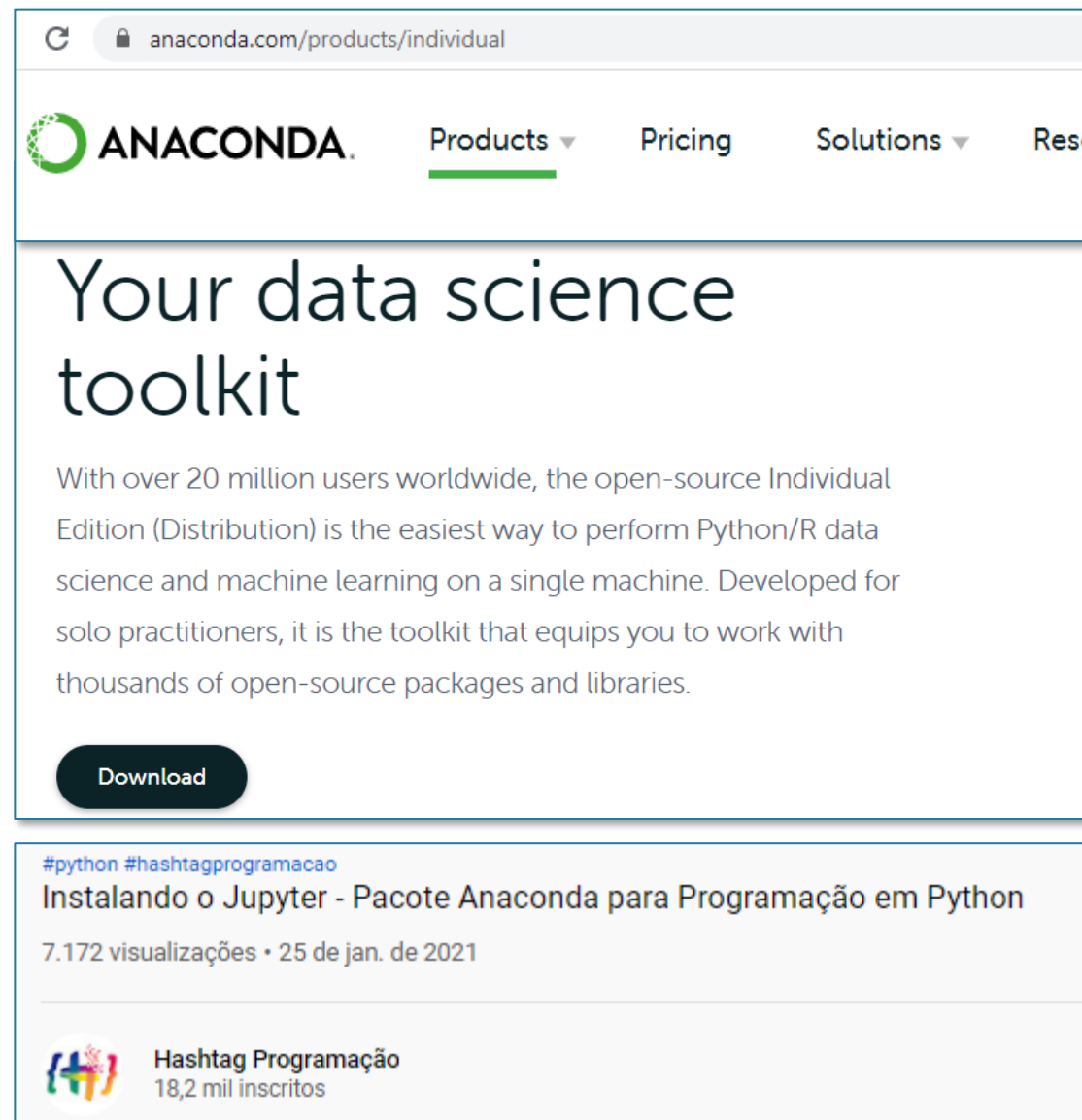
Essas plataformas na programação são chamadas de IDEs. Existem várias: Visual Studio, PyCharm, Atom, Google Colab, etc...

Todas podem ser utilizadas pra programação e execução dos códigos, usaremos o Jupyter Notebook que é uma ferramenta gratuita que existe dentro do Anaconda (uma espécie de grande caixa de ferramentas do Python).

Para usarmos o Jupyter Notebook, iremos instalar o Anaconda [\(link\)](#).

aAs próximas páginas desse capítulo são todos os passos para instalação correta do Jupyter mas caso você prefira, pode acessar nosso vídeo no Youtube explicando esse mesmo passo a passo:

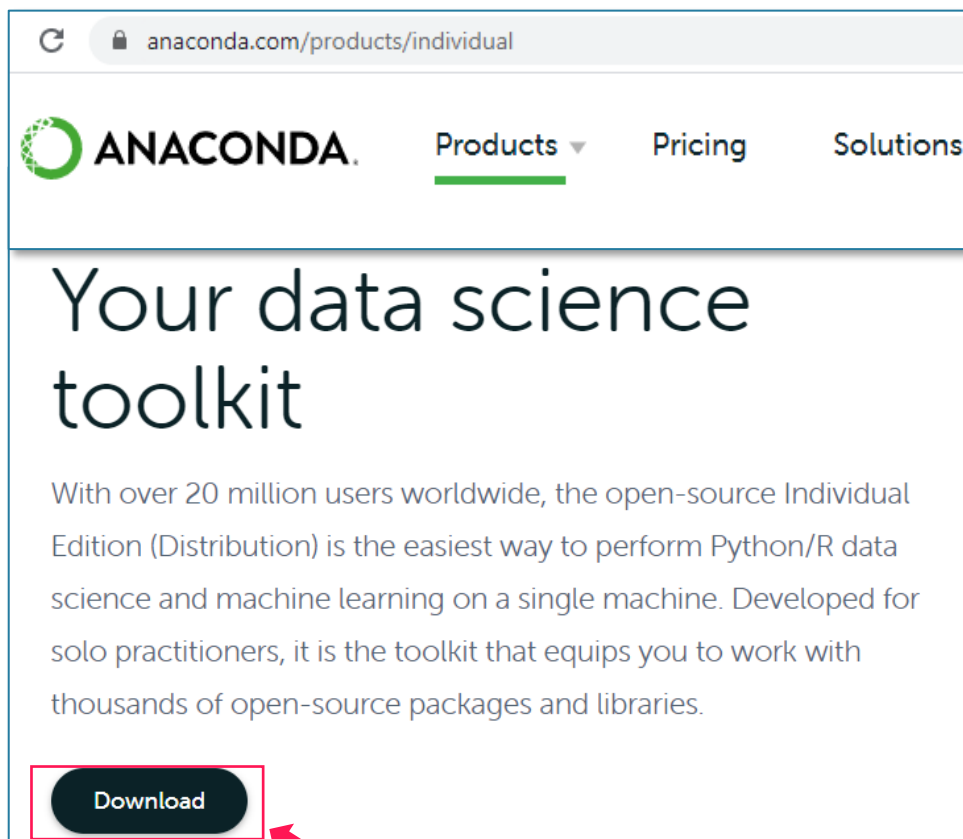
[Instalando o Jupyter - Pacote Anaconda para Programação em Python](#)



The screenshot shows the Anaconda website's product page for the Individual Edition. The URL in the browser is [anaconda.com/products/individual](https://anaconda.com/products/individual). The page features the Anaconda logo, navigation links for Products, Pricing, Solutions, and Resources. The main heading is "Your data science toolkit". Below this, a paragraph states: "With over 20 million users worldwide, the open-source Individual Edition (Distribution) is the easiest way to perform Python/R data science and machine learning on a single machine. Developed for solo practitioners, it is the toolkit that equips you to work with thousands of open-source packages and libraries." A prominent "Download" button is visible. At the bottom, there is a social media section for the hashtag #python #hashtagprogramacao, featuring a video titled "Instalando o Jupyter - Pacote Anaconda para Programação em Python" with 7,172 views and a date of 25 de jan. de 2021. The hashtag section also includes a logo and mentions "Hashtag Programação" with 18,2 mil inscritos.

# Jupyter Notebook Instalação

1) No link indicado, clique em **Download**



anaconda.com/products/individual

ANACONDA. Products Pricing Solutions

## Your data science toolkit

With over 20 million users worldwide, the open-source Individual Edition (Distribution) is the easiest way to perform Python/R data science and machine learning on a single machine. Developed for solo practitioners, it is the toolkit that equips you to work with thousands of open-source packages and libraries.

**Download**

2) **Escolha** a opção adequada para seu computador

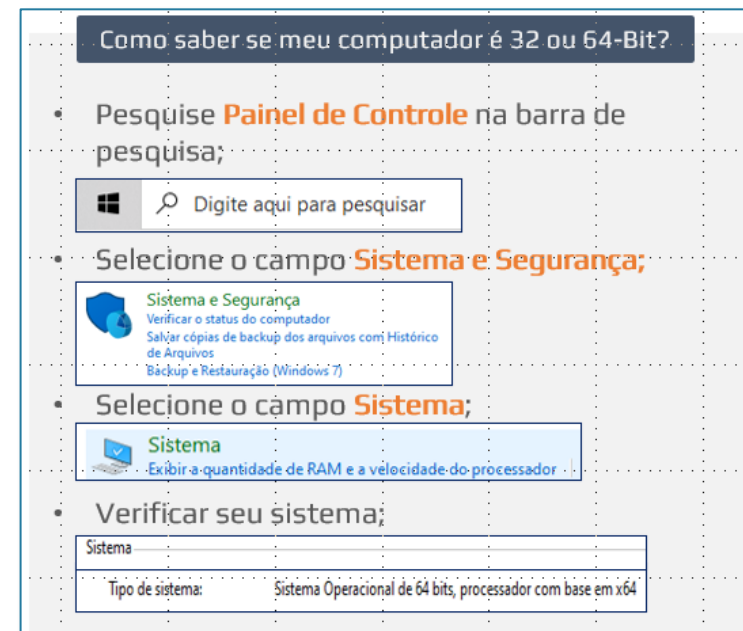


Windows

Python 3.8

64-Bit Graphical Installer (457 MB)

32-Bit Graphical Installer (403 MB)

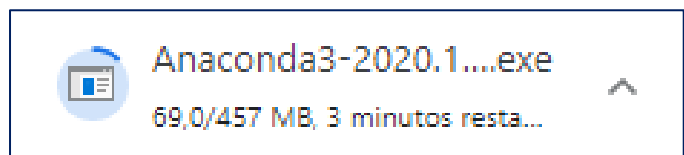


Como saber se meu computador é 32 ou 64-Bit?

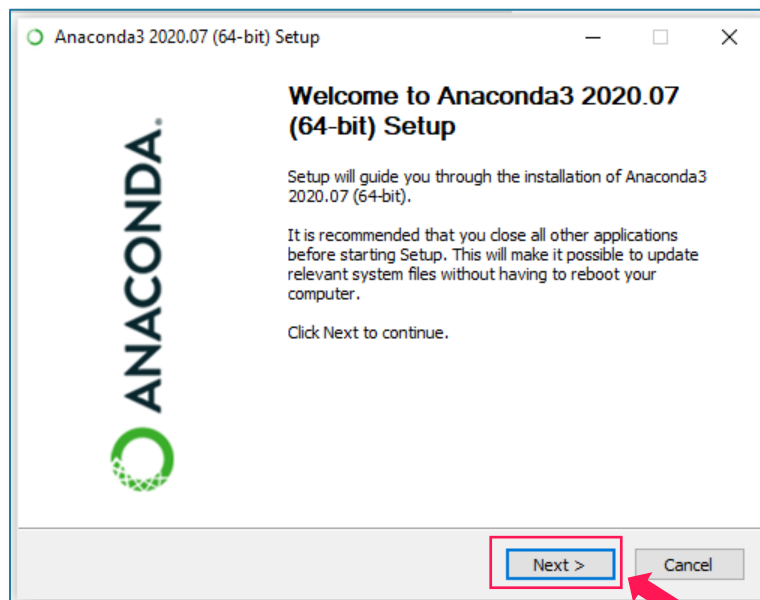
- Pesquise **Painel de Controle** na barra de pesquisa;
- Selecione o campo **Sistema e Segurança**;
- Selecione o campo **Sistema**;
- Verificar seu sistema;

Tipo de sistema: Sistema Operacional de 64 bits, processador com base em x64

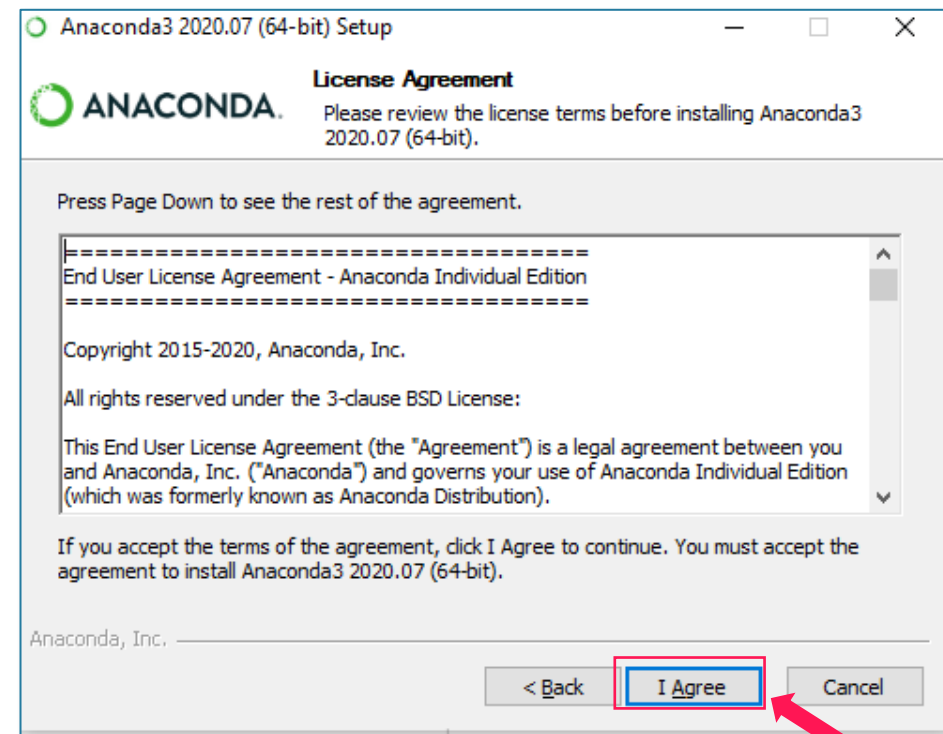
### 3) Fazer Download



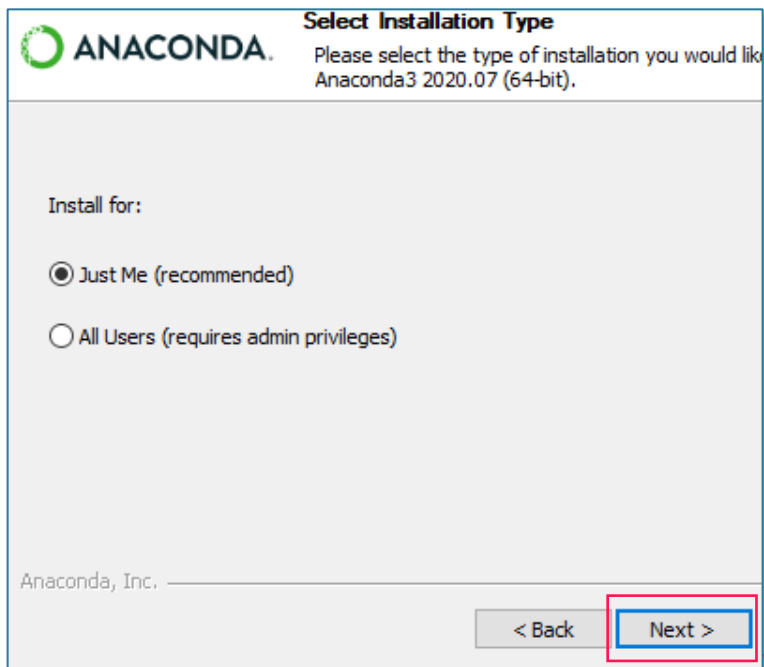
### 3) Abra o instalador do Anaconda



### 4) Aceite os termos de uso



5) No link indicado, clique em **Download**

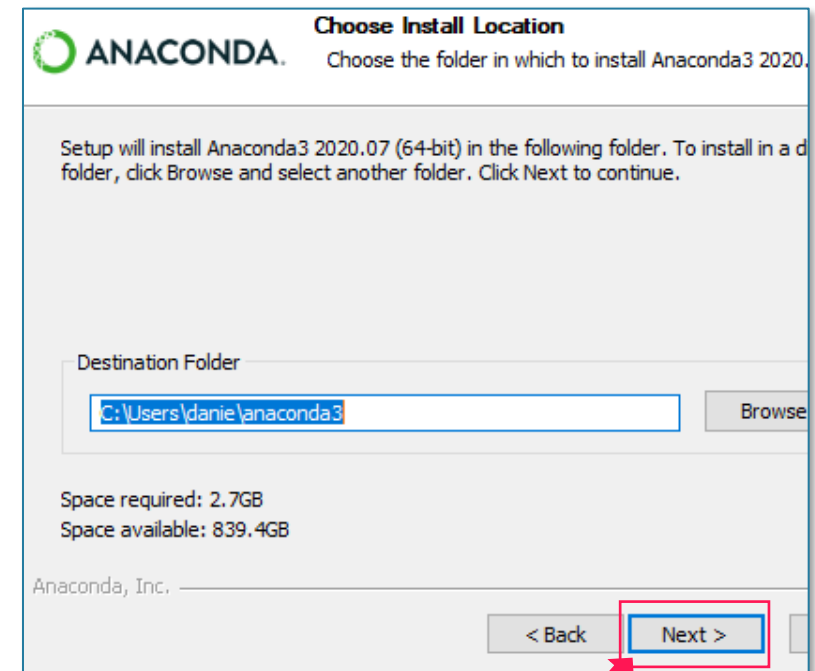


Nessa parte da instalação indicamos a opção **JUST ME** pois em teoria apenas o seu usuário precisa ter o Anaconda instalado.

PORÉM, em alguns casos, a instalação **JUST ME**, gera algumas falhas na inicialização do **JUPYTER NOTEBOOK** que vamos utilizar durante o curso.

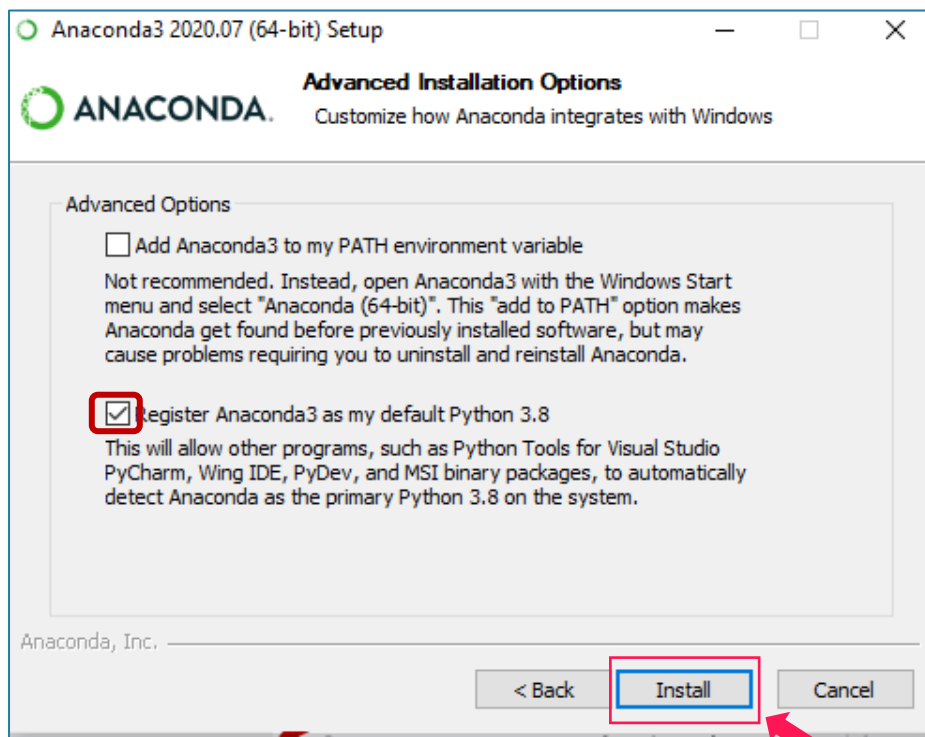
Caso aconteça com você, reinstale e utilize a opção **ALL USERS**.

6) Aperte **Next** e siga a pasta padrão definida pelo Anaconda

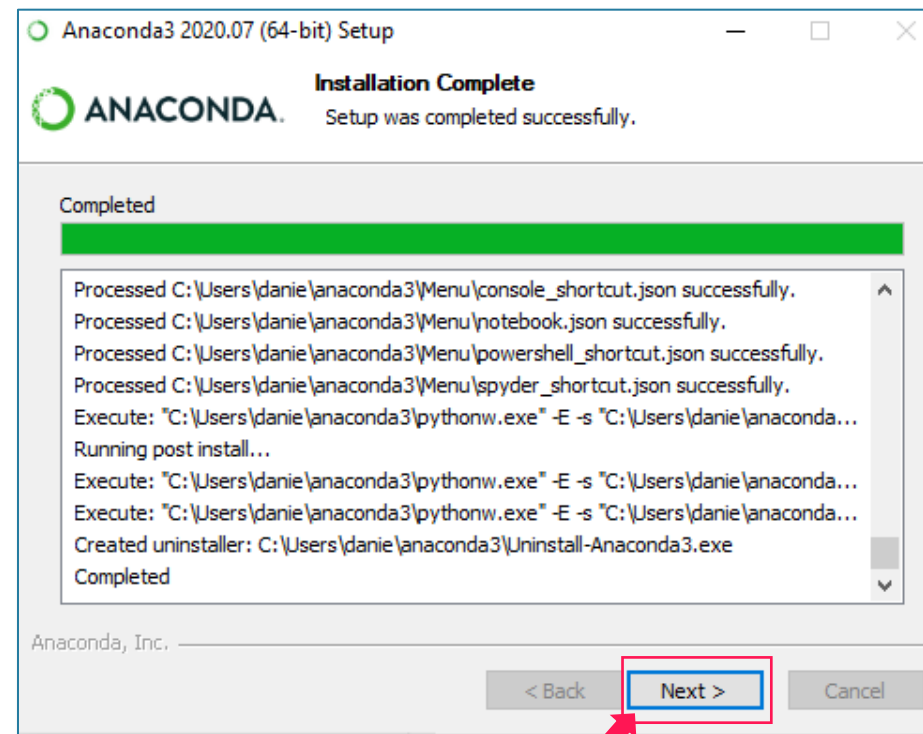


# Jupyter Notebook Instalação

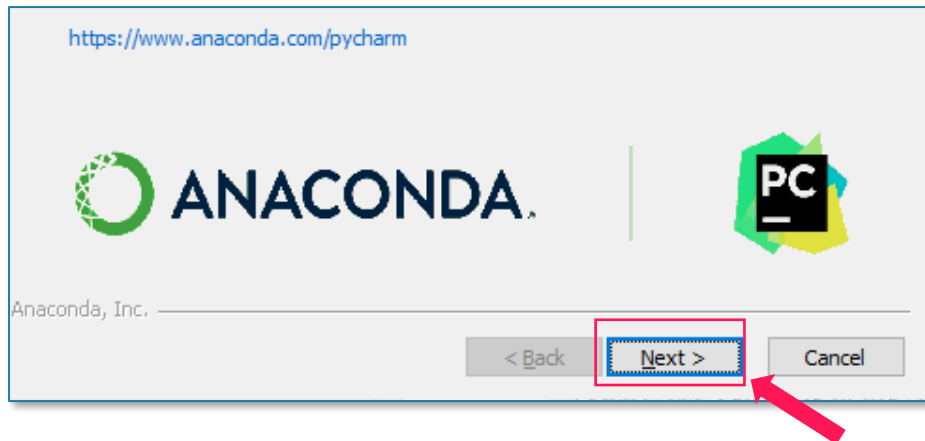
7) Defina o Anaconda como seu Python padrão e siga com a instalação clicando em **Install**



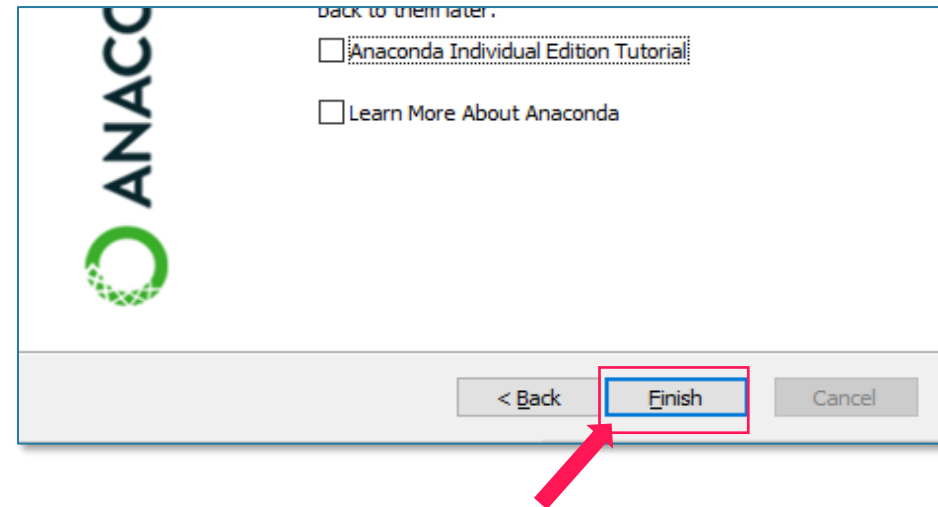
8) Ao fim da instalação clique em Next



9) Mais um **Next**



10) Clique em **Finish** para finalizar a instalação



Pronto! Anaconda instalado. Agora vamos ver se está tudo OK para começarmos!

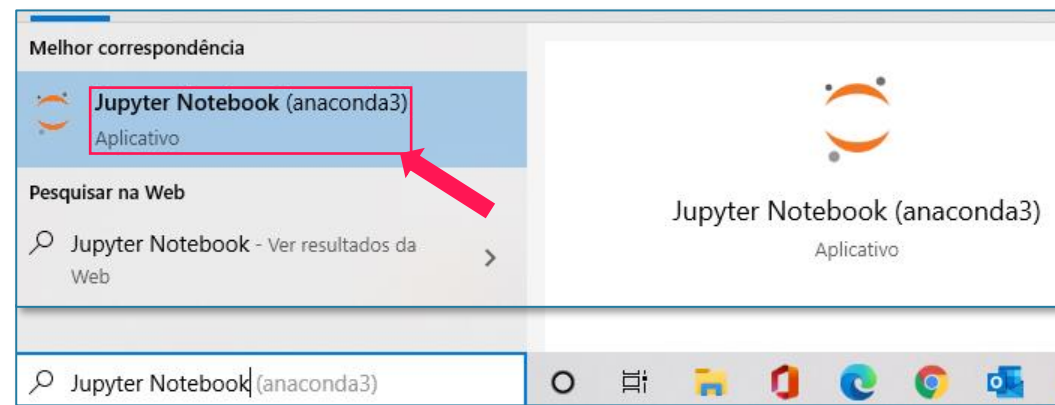


# Jupyter Notebook

## Iniciando o Jupyter

Para inicializarmos o Jupyter, basta digitar **Jupyter Notebook** no menu iniciar do seu computador.

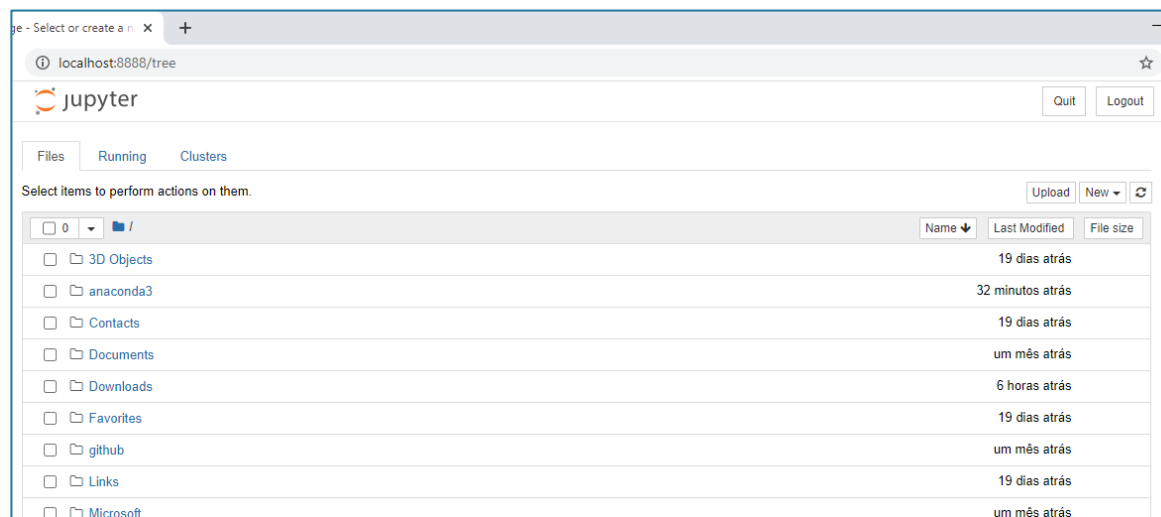
Uma nova janela será aberta no seu navegador padrão de internet.



### ATENÇÃO !

Ao clicar no ícone do Jupyter seu navegador padrão deverá abrir o Jupyter Notebook como no print 2.

Além disso, uma janela preta com o símbolo do Jupyter irá abrir. **Não feche esta janela!** Ela é o Jupyter Notebook sendo rodado pelo seu computador.



# Jupyter Notebook

## Inicializando o Jupyter

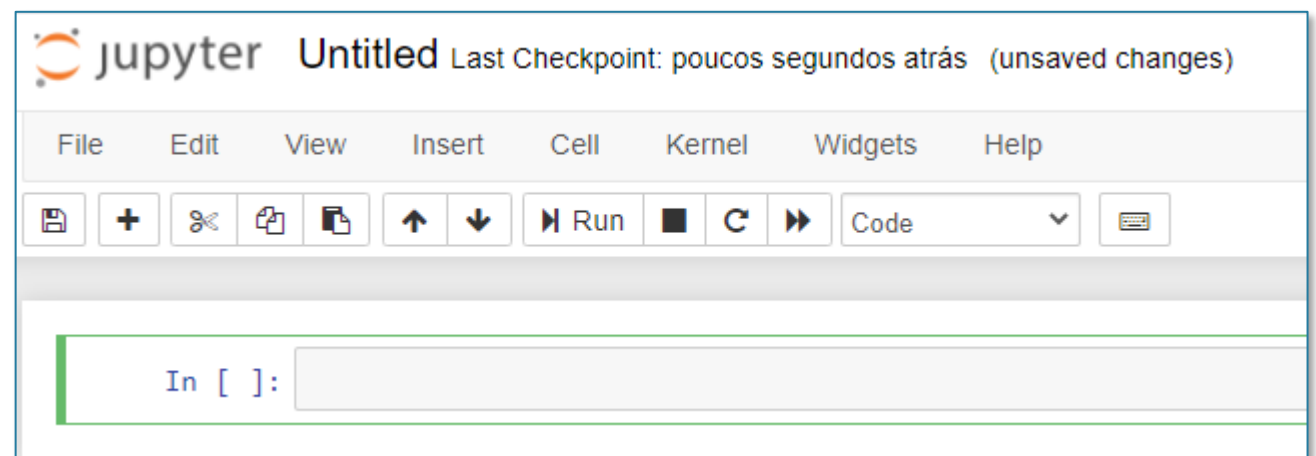
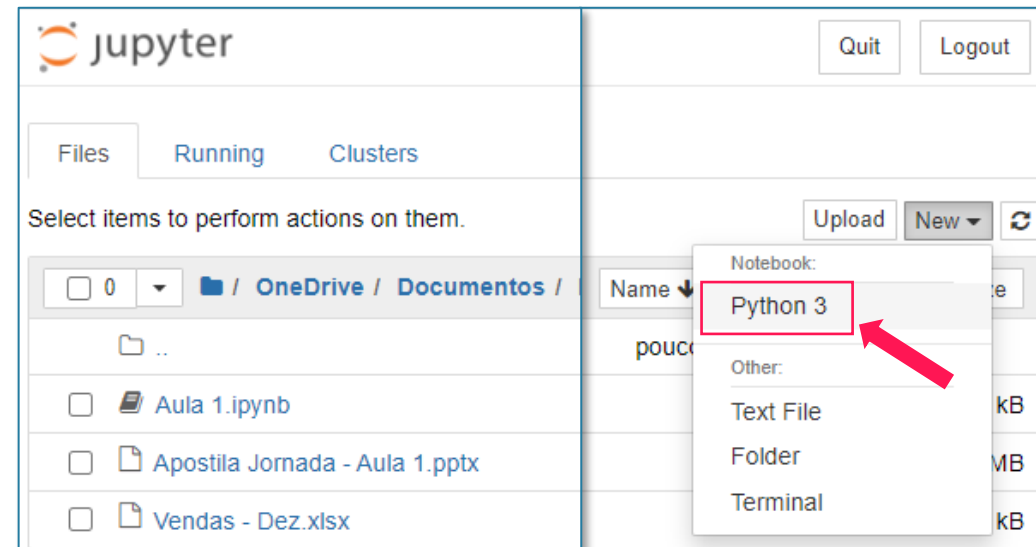
Após os preparativos, vamos acessar o nosso primeiro arquivo para criarmos nosso código em Python.

Esse arquivo se chama Notebook e possui o formato *.ipynb*. Ele só será aberto dentro de plataformas como o Google Colab ou Jupyter Notebook.

Para **criarmos um novo Notebook**, devemos clicar em **New > Python3**, assim como apresentado na figura ao lado.

Ao criar o novo Notebook, o Jupyter Notebook abrirá a janela ao lado. Aqui, é onde escreveremos nosso programa.

Mas antes de tudo, vamos entender a interface dessa plataforma.



# Entendendo a interface

The image shows the Jupyter Notebook interface with several annotations in Portuguese:

- Nome do arquivo. Formato ipynb.**: Points to the **Untitled** text in the top bar.
- File Edit View Insert Cell Kernel Widgets Help**: The menu bar.
- +**: A red box around the plus icon in the toolbar, with an arrow pointing to the label **Cria nova célula**.
- Run**: A red box around the Run button in the toolbar, with an arrow pointing to the label **Executa o código**.
- In [ ]:**: The prompt for the code cell.
- Célula onde deve ser escrito o código**: A red box around the code input area.
- Região de Output**: A red box around the output area below the code cell.

Parte 5

# Importando bibliotecas

# Importando bibliotecas

O Python por si só já possui uma série de funcionalidades que nos permitem ser mais ágeis e eficientes na programação.

No entanto, por se tratar de um código aberto, diversos pacotes de código foram criados para ajudar ainda mais a elaboração dos códigos.

Esse pacotes são chamados de **bibliotecas**.

Aqui vamos importar três bibliotecas que nos ajudarão no nosso processo de automação.

- **pyautogui** ([documentação](#))
- **time** ([documentação](#))
- **pyperclip** ([documentação](#))

Essas bibliotecas não estão “instaladas” no Python, por isso, precisamos antes de tudo instalá-las. Algumas bibliotecas são nativas do Python, ou seja, “já vem no pacote” outras precisam ser instaladas.

## Welcome to PyAutoGUI's documentation!

PyAutoGUI lets your Python scripts control the mouse and keyboard to automate interactions with other applications. The API is designed to be as simple. PyAutoGUI works on Windows, macOS, and Linux, and runs on Python 2 and 3.

## Welcome to Pyperclip's documentation!

Pyperclip provides a cross-platform Python module for copying and pasting text to the clipboard.

To copy text to the clipboard, pass a string to `pyperclip.copy()`. To paste the text from the clipboard, call `pyperclip.paste()` and the text will be returned as a string value.

## time — Time access and conversions

This module provides various time-related functions. For related functionality, see also the `datetime` and `calendar` modules.

# Importando bibliotecas

Antes de importarmos essas bibliotecas no Jupyter, precisamos garantir que esses programas estão instalados no Python.

O Python possui um instalador padrão que ajuda muito nesses casos.

Esse instalador se chama **PIP**. Para acessá-lo vamos acessar o **prompt de comando** do Anaconda.

**Sabemos que são MUITOS nomes possivelmente novos, mas não se preocupe. Assusta mais do que é de fato complicado 😊.**

Para acessar o prompt basta escrever no menu iniciar conforme escrito abaixo e apresentado ao lado:

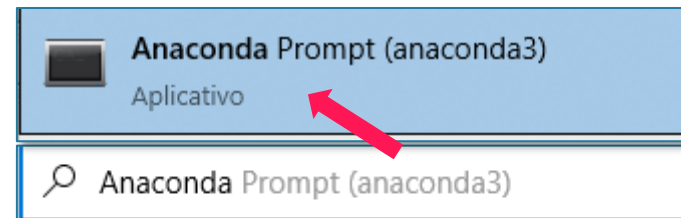
**Anaconda Prompt**

Basta agora usar o comandos abaixo:

**pip install pyautogui**

**pip install time**

**pip install pyperclip**



```
(base) C:\Users\danie>pip install pyautogui
```

```
(base) C:\Users\danie>pip install time
```

```
(base) C:\Users\danie>pip install pyperclip
```



## ATENÇÃO !

- 1) Ao usarmos o pip a biblioteca será instalada. Esse processo pode demorar alguns segundos/minutos;
- 2) Aqui estamos instalando todas as bibliotecas, mas algumas bibliotecas são nativas do Python e não precisam ser instaladas.



# Importando e visualizando os dados

## Importando bibliotecas

Com as bibliotecas instaladas, podemos voltar para nosso Notebook e desenvolver as nossas primeiras linhas de código.

Quando criamos um código, precisamos **importar** as bibliotecas que iremos usar. É como se fossemos cozinhar e pegássemos no armário, antes de tudo, a panela, a frigideira, etc...

A estrutura de importação está apresentada ao lado.

Vamos entender de forma resumida o que significa cada uma das bibliotecas que importaremos:

- **pyautogui**: Ferramenta de automação (Python assume controle do seu teclado por exemplo);
- **time**: Facilita no desenvolvimento de códigos que envolvem tempo, tempo de espera etc...;
- **pyperclip**: Permite copiar e colar via Python.

Nome da biblioteca. Existem centenas de bibliotecas disponíveis no Python. Só depende da sua curiosidade ☺.

```
In [1]: import pyautogui
import time
import pyperclip
```

Indica ao Python que precisamos importar uma biblioteca

Parte 6

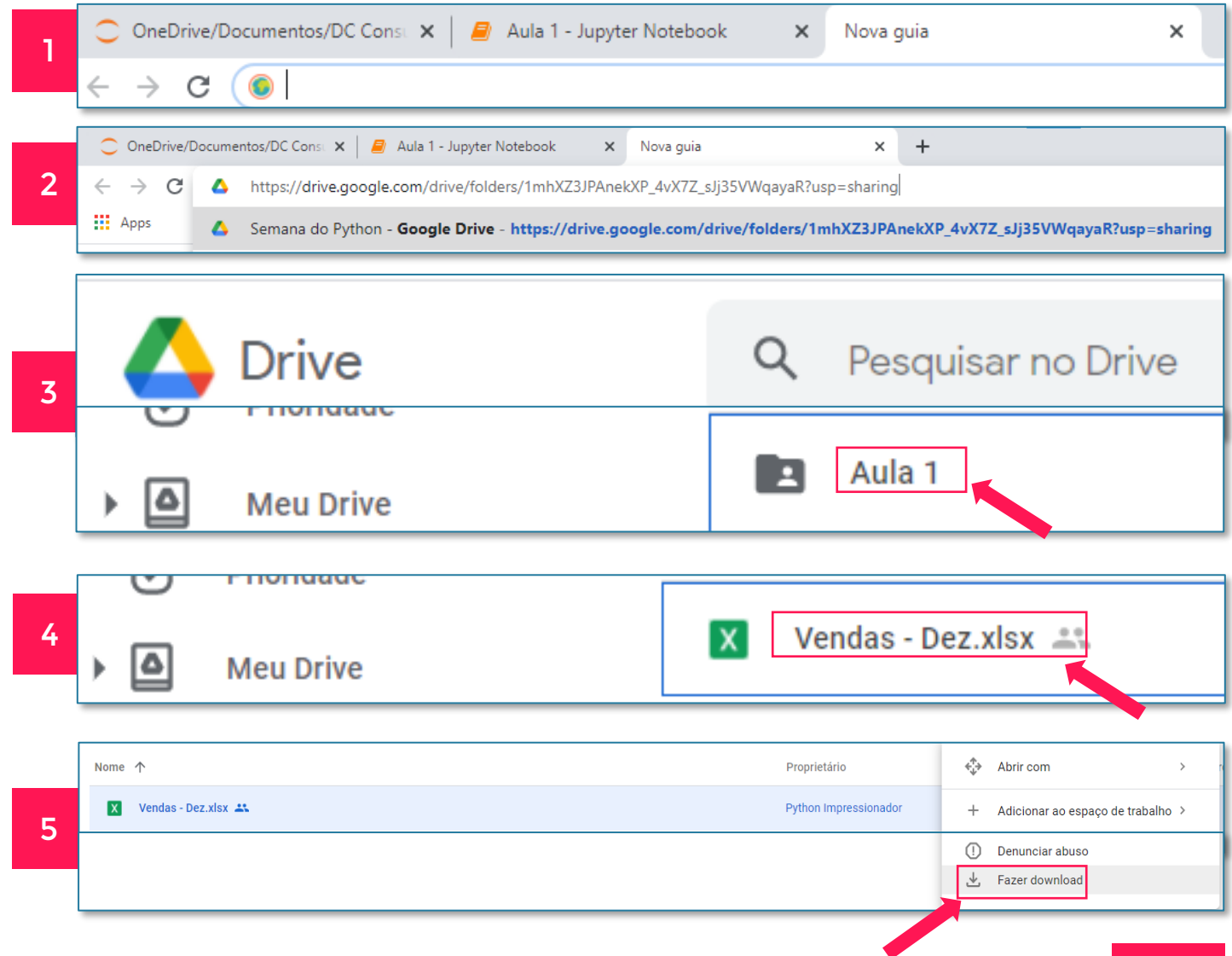
# Baixando a base de dados

# Entendendo o passo a passo

Como dito anteriormente, a biblioteca pyautogui nos permite criar um código que simula que estamos usando o computador.

Nessa etapa, nosso objetivo é sem tocar no mouse ou no teclado realizar as tarefas a seguir:

- 1) Abrir um navegador de internet; (**usando teclado**)
- 2) Copiar o link do Google Drive onde está nossa base de dados; (**usando teclado**)
- 3) Entrar no Google Drive onde está nossa base de dados; (**usando teclado**)
- 4) Abrir pasta Aula 1 do Google Drive; (**usando mouse**)
- 5) Baixar planilha **Vendas - Dez.xlsx**; (**usando mouse**)



# Baixando a base de dados (1/8)

O pyautogui funciona como se você estivesse executando as tarefas mas na verdade é o Python que está “clcando” nas coisas.

A primeira etapa, é abrir um navegador. Aqui, temos 2 opções:

- 1) Abrir um novo navegador ( primeiras 3 linhas de código );
- 2) Abrir apenas uma guia do navegador já aberto por estarmos usando o Jupyter Notebook.

Se optarmos pela primeira opção, usaremos o primeiro código ao lado.

Caso opte pela segunda opção se utilize do segundo código ao lado.

Perceba que no segundo caso, usamos o caractere **#**. Sua função é comentar o código. Logo, as linhas que estão com esse símbolo no início não serão executadas pelo Python, passam a ser apenas um texto.

```
import pyautogui
import time
import pyperclip

pyautogui.PAUSE = 1
#abrir navegador
pyautogui.press("winleft")
pyautogui.write("chrome")
pyautogui.press("enter")
pyautogui.alert("Vai começar, aperte OK e não mexa em nada")
pyautogui.hotkey('ctrl', 't')
```

```
import pyautogui
import time
import pyperclip

pyautogui.PAUSE = 1
#abrir navegador
#pyautogui.press("winleft")
#pyautogui.write("chrome")
#pyautogui.press("enter")
pyautogui.alert("Vai começar, aperte OK e não mexa em nada")
pyautogui.hotkey('ctrl', 't')
```

# Baixando a base de dados (2/8)

Vamos entender um pouco mais o código em si e como o Python funciona. Para esse exemplo, usaremos o caso de número 2.

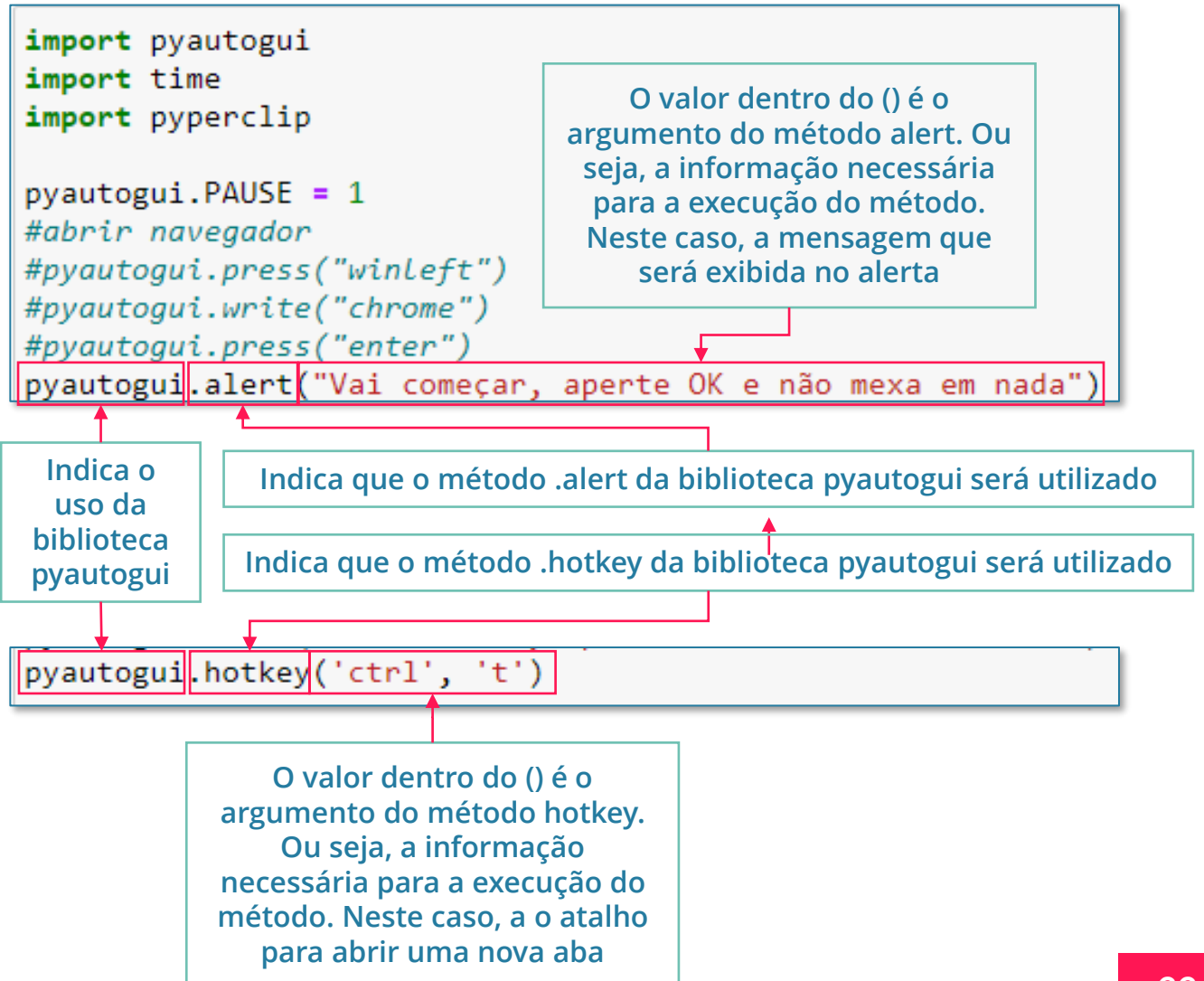
Perceba que a estrutura ao usarmos o pyautogui sempre é a mesma:

**pyautogui.algumcomando**

A estrutura é sempre a mesma pois estamos “chamando” a biblioteca sempre que queremos usá-la. A primeira parte **pyautogui**, indica que usaremos alguma função dessa biblioteca específica. Já a segunda parte (**.alert()**, **.hotkey()**) são as funções\* dessa biblioteca que desempenham uma função específica, por exemplo:

- **.alert**-> Cria um alerta na tela do computador
- **.hotkey** -> Executa um atalho

\*oficialmente são métodos, mas não se preocupe tanto com nomes por enquanto 😊



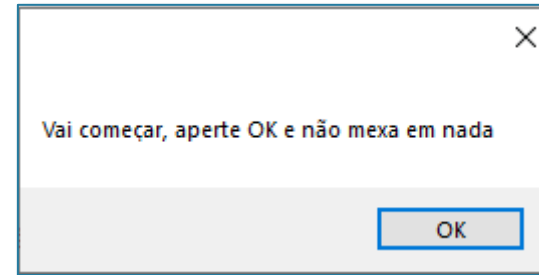
# Baixando a base de dados (3/8)

Se rodarmos apenas esse bloco de código, teremos 2 atividades ocorrendo.

Importante frisar que enquanto o pyautogui está rodando, ele não trava seu computador. Ou seja, caso você use o mouse ou mude de aba, o computador irá executar.

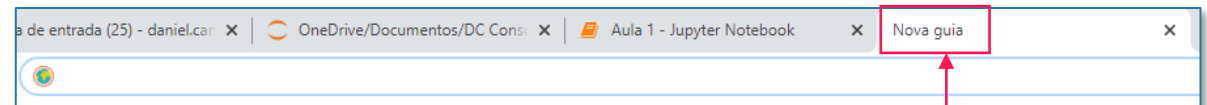
Esse é o motivo do alerta criado. Ele em teoria, não precisaria existir. O mesmo só existe para comunicar ao usuário que uma automação estará ocorrendo e que usar o PC enquanto isso acontece pode afetar o resultado final.

```
pyautogui.alert("Vai começar, aperte OK e não mexa em nada")
```



.alert() nos permite criar uma *dialog box* para avisar ao usuário que o processo de automação começará.

```
pyautogui.hotkey('ctrl', 't')
```



.hotkey() permite que o Python simule que estamos apertando ctrl+t no nosso teclado. O que é um atalho para criação de nova guia no Chrome



# Baixando a base de dados (4/8)

Vamos para o nosso próximo bloco de código. Seguindo as etapas indicadas anteriormente, agora que abrimos a nova guia, precisamos acessar o Google Drive onde está a nossa base.

Perceba que para acessarmos o drive, precisamos de um endereço ou um *link* e ao invés de simplesmente usá-lo, criaremos uma variável que armazenará essa informação. Essencialmente, uma variável no Python é como uma caixinha que armazena informação.

Sempre que nos referirmos à aquela caixinha, estamos nos referindo ao seu conteúdo. No código abaixo podemos ver que o endereço ( url grande em vermelho) foi armazenada em uma variável **link**.

```
# abrir drive
# ensinar aqui o write
link = "https://drive.google.com/drive/folders/1mhXZ3JPAnekXP_4vX7Z_sJj35VWqayaR?usp=sharing"
pyperclip.copy(link)
pyautogui.hotkey("ctrl", "v")
pyautogui.press("enter")
time.sleep(15)
```

Criamos uma variável chamada link que receberá o endereço do drive que queremos acessar.

Aqui utilizamos a biblioteca pyperclip e seu método .copy(). O uso de .copy(link) significa que estamos armazenando na memória(CTRL+C) o valor da variável link. Ou seja, o endereço do google drive

Assim como fizemos anteriormente, usamos o método .hotkey() para colar o valor armazenado na memória.

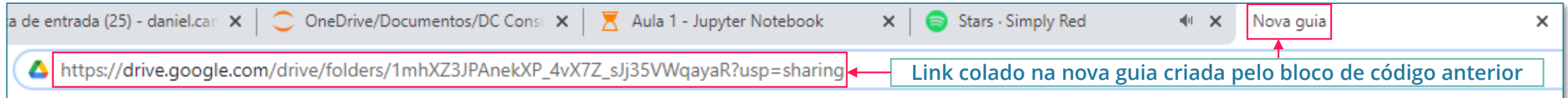
Aqui, usamos a biblioteca time e seu método .sleep().  
O valor '15' dentro do método sleep indica que o computador deve aguardar 15 segundos antes de ir para a próxima linha de código. Possibilitando o carregamento da página.

O método .press() permite o Python de "apertar" uma tecla no teclado. Nesse caso, ENTER. Essa linha de código tem por objetivo acessar o link que foi copiado na barra de endereço.

# Baixando a base de dados (5/8)

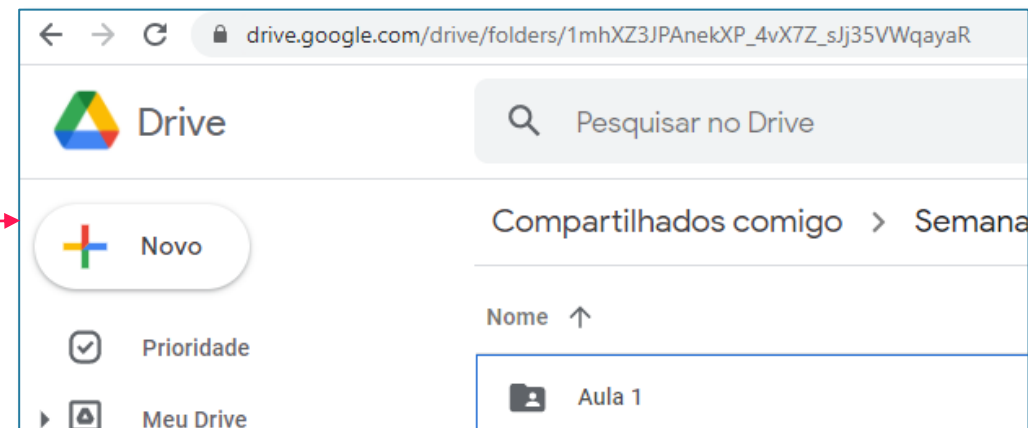
Vamos rodar esse novo bloco de código e ver o que acontece na prática no nosso computador. Lembrando que já possuímos uma nova guia do Google Chrome criada.

```
link = "https://drive.google.com/drive/folders/1mhXZ3JPAnekXP_4vX7Z_sJj35VWqayaR?usp=sharing"
pyperclip.copy(link)
pyautogui.hotkey("ctrl", "v")
```



```
pyautogui.press("enter")
time.sleep(15)
```

"Enter" na barra de endereços do Google Chrome redireciona para ao Drive.



# Baixando a base de dados (6/8)

Nosso objetivo agora é conseguir acessar a pasta Aula 1 do nosso drive. Até agora só utilizamos o pyautogui para cessar nosso teclado, mas é possível fazê-lo acessar também o nosso mouse para clicarmos na pasta.

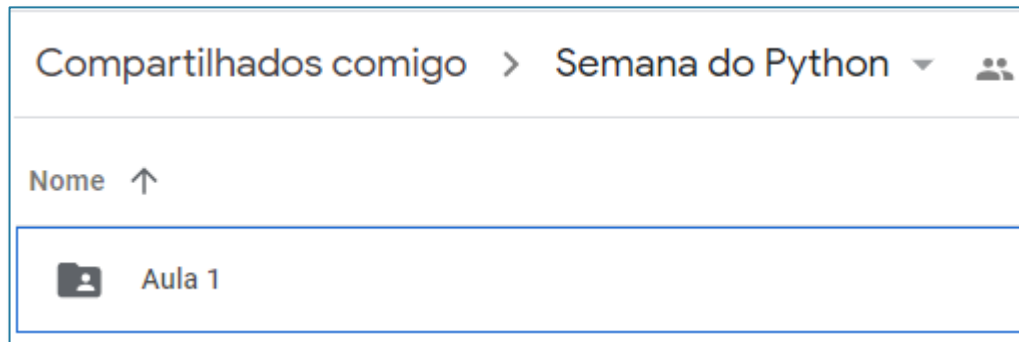
Mas antes disso, precisamos indicar **ONDE** clicar.

Para isso, usaremos o método **position()** da biblioteca pyautogui.

Essencialmente o que é feito por esse método é indicar a posição da “setinha” do mouse em um dado momento.

Ou seja, se posicionarmos o mouse sobre Aula 1, poderemos saber em que pixel da tela clicar para abri-la.

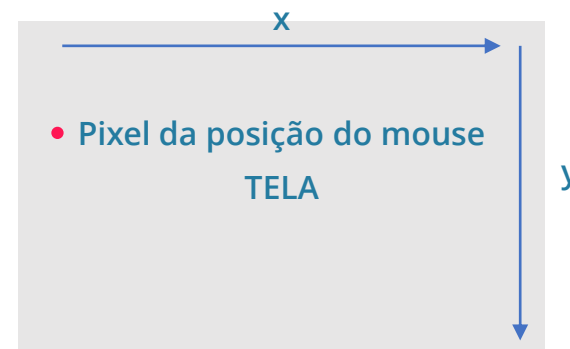
Sabendo a posição, podemos usar o método **.click()** do pyautogui para clicar na pasta e seguirmos com nossa automação.



```
pyautogui.position()
```

Esse método fornece a posição (x,y) do mouse no momento da execução do código.

Importante frisar que essa posição é o pixel relativo a tela do computador que está executando o código.



# Baixando a base de dados (7/8)

No computador usado para criar essa apostila, os pontos ao lado indicam os locais de **click**.

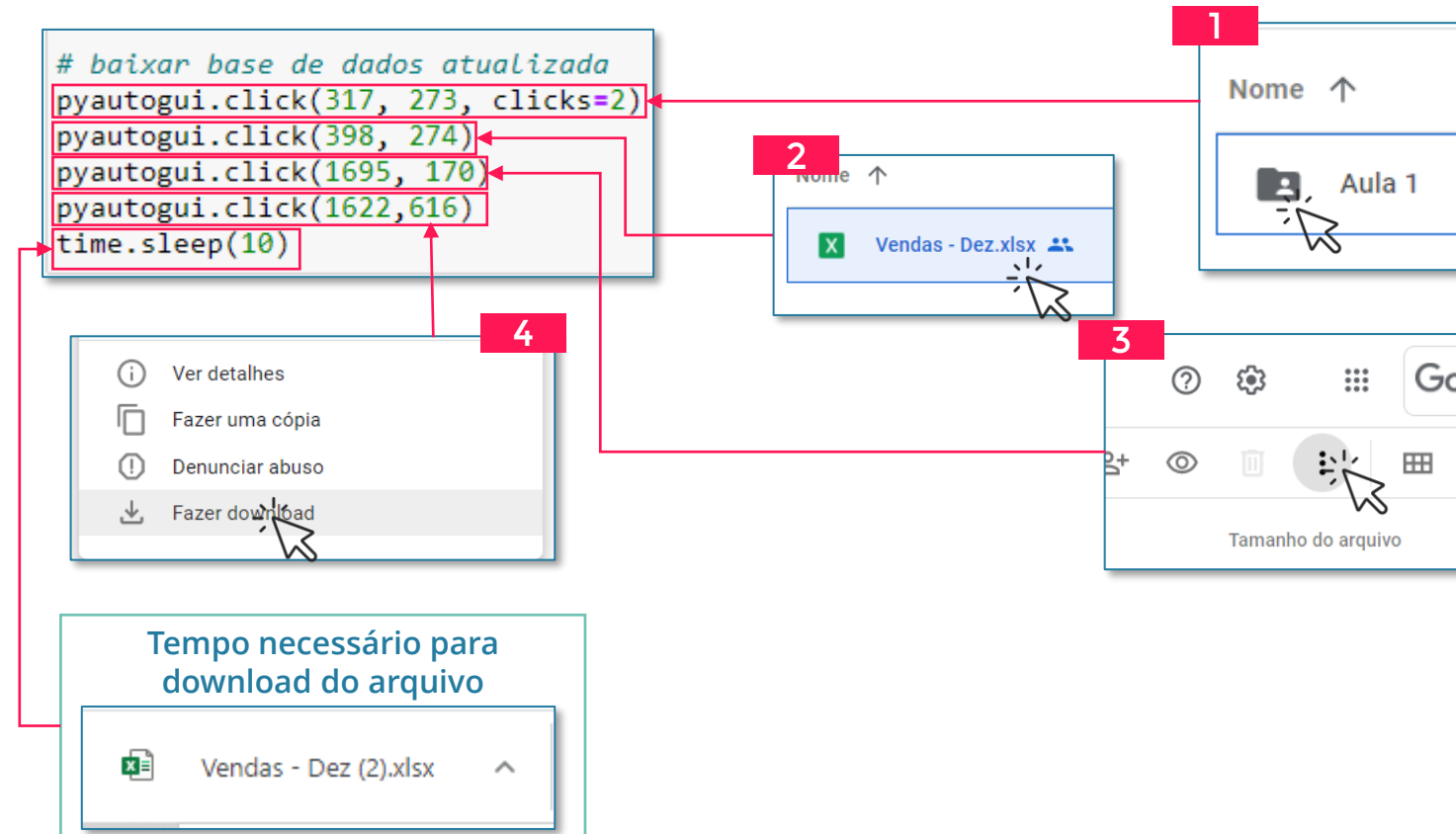
Importante frisar que esses pontos podem e possivelmente variar no caso do seu computador.

Para descobrir quais são esses pontos de uma forma simples em um primeiro momento é rodar o código linha a linha posicionando o mouse no local desejado e utilizando o código abaixo para exibir o ponto desejado:

```
print(pyautogui.position())
```

Ao lado você poderá verificar todos as linhas de código e seus respectivos “clicks”.

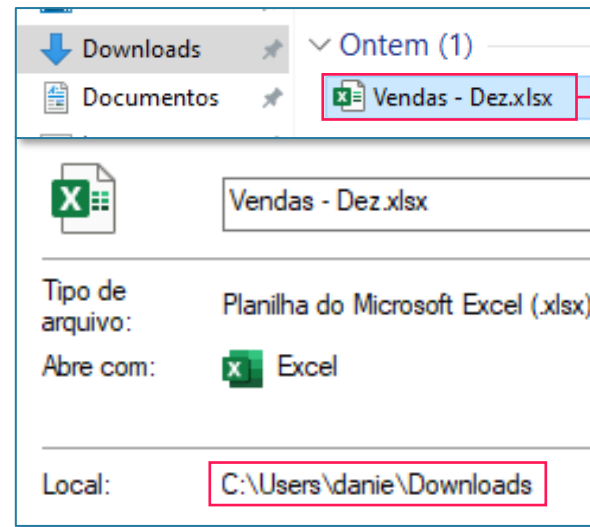
Perceba que que na primeira linha de código conseguimos dar um “duplo click” usando o argumento **clicks=2**



# Baixando a base de dados (8/8)

Agora temos nossa base de dados baixada. Por padrão, essa pasta irá para sua pasta de Downloads. No entanto, isso pode ser diferente caso exista alguma configuração distinta no seu computador.

Além disso, vamos usar na próxima etapa o caminho deste arquivo. O caminho que apresentamos ao lado **É DIFERENTE** do caminho do seu computador.



	A	B	C	D	E	F	G
1	Código Vend	Data	ID Loja	Produto	Quantid	Valor Unit	Valor Final
2	65014	01/12/2019	Shopping Morumbi	Sunga Listr	5	R\$ 114,00	R\$ 570,00
3	65014	01/12/2019	Shopping Morumbi	Casaco List	1	R\$ 269,00	R\$ 269,00
4	65016	01/12/2019	Iguatemi Campinas	Sapato List	2	R\$ 363,00	R\$ 726,00
5	65016	01/12/2019	Iguatemi Campinas	Casaco	1	R\$ 250,00	R\$ 250,00
6	65017	01/12/2019	Shopping SP Market	Gorro Liso	3	R\$ 92,00	R\$ 276,00
7	65018	01/12/2019	Rio Mar Shopping Fortaleza	Cueca Esta	1	R\$ 66,00	R\$ 66,00
8	65018	01/12/2019	Rio Mar Shopping Fortaleza	Sunga Xadr	1	R\$ 116,00	R\$ 116,00
9	65018	01/12/2019	Rio Mar Shopping Fortaleza	Casaco List	1	R\$ 269,00	R\$ 269,00
10	65019	01/12/2019	Shopping União de Osasco	Polo Xadre	2	R\$ 142,00	R\$ 284,00
11	65019	01/12/2019	Shopping União de Osasco	Tênis Linhc	1	R\$ 294,00	R\$ 294,00
12	65020	01/12/2019	Shopping Morumbi	Bolsa Linh	1	R\$ 216,00	R\$ 216,00

Parte 7

# Importando a base de dados



# Importando a base de dados (parte 1/3)

Temos agora a base de dados baixada do Drive. Agora precisamos acessá-la via nossa automação para fazermos as análises necessárias.

Para isso, vamos utilizar uma nova biblioteca que é uma das mais úteis e famosas no que se refere a análise de dados:

**PANDAS** ([link](#)).

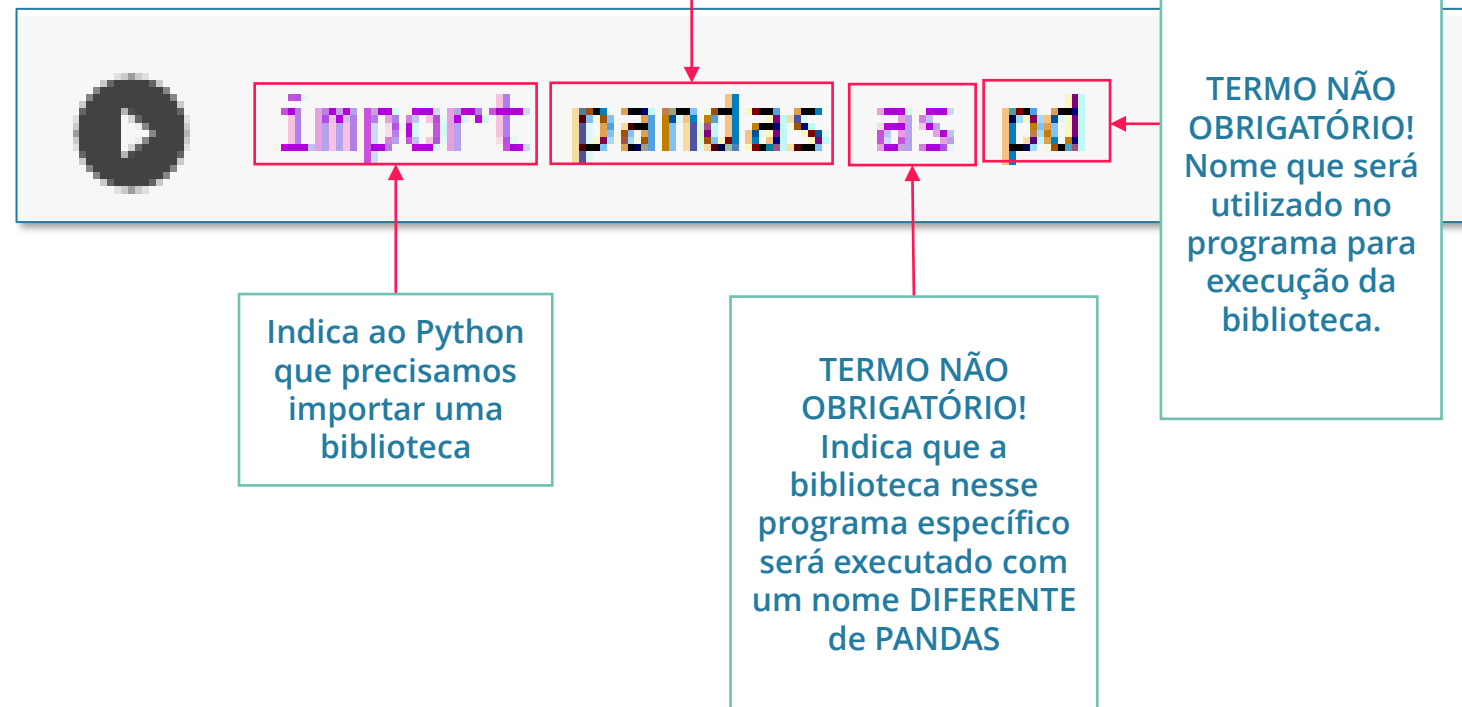
Assim como fizemos com as demais bibliotecas precisamos garantir importar essa biblioteca antes de continuarmos usando o **IMPORT** como apresentado na figura ao lado.

Para facilitar vamos usar a abreviação **pd**. Ou seja, sempre que quisermos usar o pandas usaremos **pd.COISA QUE QUEREMOS FAZER**.

Vamos abordar isso com um pouco mais de detalhe nas próximas páginas.

Nome da biblioteca. Existem centenas de bibliotecas disponíveis no Python. Só depende da sua curiosidade 😊.

Nesse caso estamos importando a biblioteca PANDAS. Ela é muito usada no tratamento e análise de grandes quantidades de dados.



# Importando a base de dados (parte 2/3)

Agora que temos nossas bibliotecas importadas, podemos usar uma das suas funções para nos auxiliar a importar a nossa base de dados.

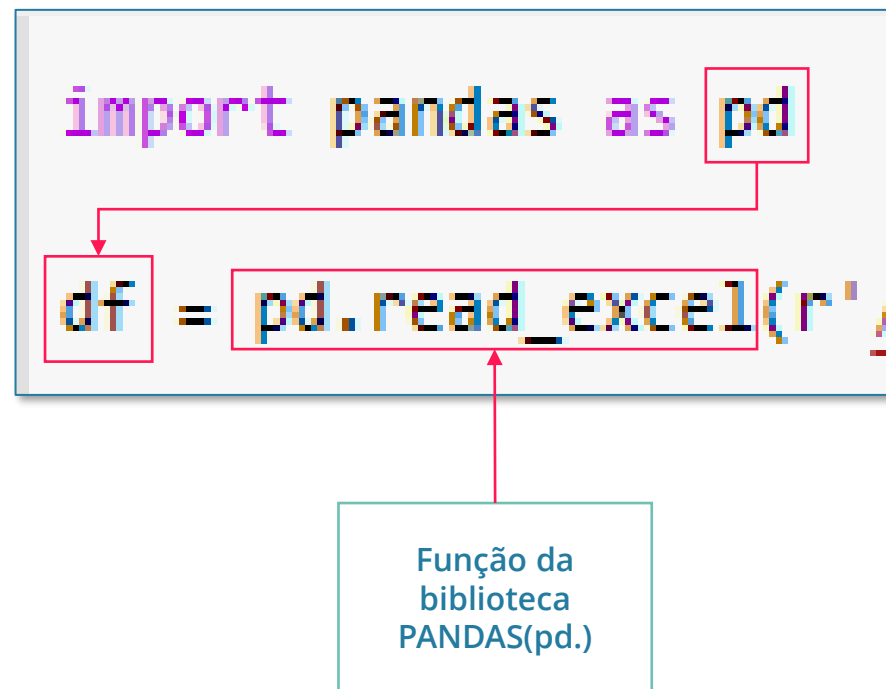
Nesse exemplo, nossa base está no arquivo **Vendas.xls**.

Para acessarmos as fórmulas dentro do **PANDAS** vamos usar a sintaxe:

**pd.AÇÃO**

Como vimos anteriormente, “**pd**” é uma abreviação para facilitar a programação. Ele representa pandas.

Já ação, nesse caso, será de **Leitura de um arquivo Excel específico**. O nome dessa função no pandas é o “**read\_excel**”



# Importando base de dados (parte 3/3)

A função `pd.read_excel` necessita de alguns parâmetros para que possa fazer a importação dos dados. São eles:

- Local do arquivo;
- Nome do arquivo;

```
import pandas as pd

df = pd.read_excel(r'C:/Users/danie/Downloads/Vendas - Dez.xlsx')
```

Caminho no drive onde está salvo o arquivo Vendas.xlsx

r' indica que o texto é uma *raw string*. Ou seja, sem caracteres especiais.

Nome do arquivo

Além disso, você percebeu que bem no início da linha de código temos “`df =`” ?

No Python, é necessário criarmos um lugar onde seja possível **armazenar os dados** lidos do nosso arquivo excel.

Esse local, chamaremos de **variável**. Existem diversos tipos de variáveis com propósitos distintos.

Por enquanto, o que precisamos entender é que nela, se pode armazenar informação.

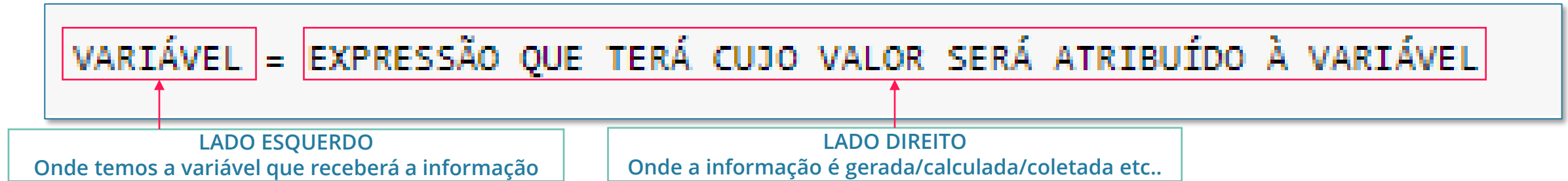
Toda variável possui um nome, no nosso caso, vamos chamá-la de **df** (esse nome é uma abreviação de **dataframe**).

A estrutura ‘**df =**’ deverá ser lida como “**A variável df recebe**”.

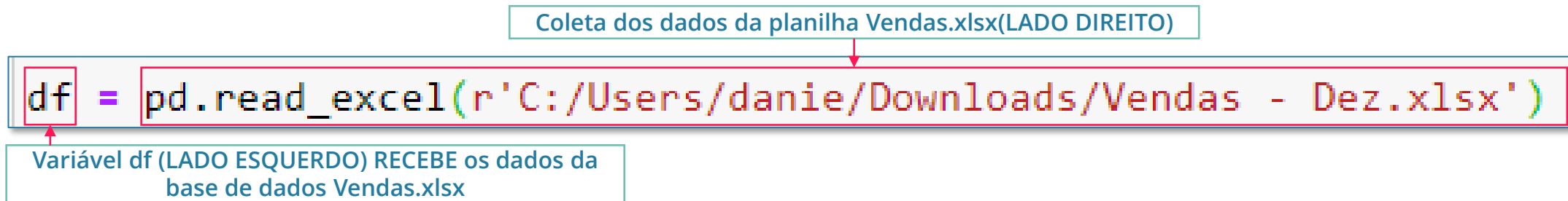
# Atribuindo valor a uma variável

Vamos entender um pouco melhor a estrutura que atribui valor a uma variável.

Sempre o que estiver no lado esquerdo do “=” estará recebendo um valor (numérico ou não) da expressão do lado direito.



Agora, vamos analisar novamente nossa linha de código:



# Visualizar a base de dados importada

Bem, já importamos nossa base de dados...

Agora vamos tentar visualizá-la !

Usamos a função **DISPLAY()** para exibir nossos dados coletados.

A função display necessita de uma informação: O que será apresentado. Vamos usar nossa variável **df** que está armazenando nossos dados. Assim, temos:

**display(df)**

Perceba, os dados já foram formatados. Esse é uma das vantagens do **PANDAS**. Ao ler o arquivo excel, os dados já são compilados em uma tabela, o que nos ajuda na visualização.

Além disso, temos informações interessantes como o número de linhas e colunas dessa tabela apresentados na parte inferior da tabela.

```
import pandas as pd
```

```
df = pd.read_excel(r'C:/Users/danie/Downloads/Vendas - Dez.xlsx')
```

```
display(df)
```

Função display que apresenta os dados armazenados na variável df

	Código Venda	Data	ID Loja	Produto	Quantidade	Valor Unitário	Valor Final
0	65014	2019-12-01	Shopping Morumbi	Sunga Listrado	5	114	570
1	65014	2019-12-01	Shopping Morumbi	Casaco Listrado	1	269	269
2	65016	2019-12-01	Iguatemi Campinas	Sapato Listrado	2	363	726
3	65016	2019-12-01	Iguatemi Campinas	Casaco	1	250	250
4	65017	2019-12-01	Shopping SP Market	Gorro Liso	3	92	276
...	...	...	...	...	...	...	...
7084	69996	2019-12-26	Center Shopping Uberlândia	Short Listrado	2	102	204
7085	69996	2019-12-26	Center Shopping Uberlândia	Mochila	4	270	1080
7086	69996	2019-12-26	Center Shopping Uberlândia	Pulseira Estampa	1	87	87
7087	69997	2019-12-26	Ribeirão Shopping	Camisa Listrado	1	108	108
7088	69997	2019-12-26	Ribeirão Shopping	Short Linho	2	133	266

7089 rows x 7 columns

7089 linhas  
7 colunas

Parte 8

# Calculando os indicadores

# Calculando Faturamento e Qtde Produtos

Agora que importamos nossa base de dados, vamos iniciar nossa análise.

Como nosso objetivo é criar um relatório que permita dizer o resultado das vendas, vamos criar variáveis que armazenarão os valores que queremos:

- **Faturamento** : Soma todo o faturamento das vendas. Se utiliza da coluna **['Valor Final']**;
- **Quantidade vendida** : Soma todo a quantidade de produtos vendidos. Se utiliza da coluna **['Quantidade']**;

Para a criação desse indicador usaremos um novo método, o **.sum()**.

```
faturamento = df['Valor Final'].sum()  
qtde_produtos = df['Quantidade'].sum()
```

Variável  
**qtde\_produtos**  
receberá o valor  
total de vendas

Dataframe  
**df**

Indica que apenas a  
coluna **'Quantidade'**  
do dataframe **df** será  
considerada

Soma a coluna  
escolhida

	Código Venda	Data	ID Loja	Produto	Quantidade	Valor Unitário	Valor Final
0	65014	2019-12-01	Shopping Morumbi	Sunga Listrado	5	114	570
1	65014	2019-12-01	Shopping Morumbi	Casaco Listrado	1	269	269
2	65016	2019-12-01	Iguatemi Campinas	Sapato Listrado	2	363	726
3	65016	2019-12-01	Iguatemi Campinas	Casaco	1	250	250
4	65017	2019-12-01	Shopping SP Market	Gorro Liso	3	92	276
...	...	...	...	...	...	...	...
7084	69996	2019-12-26	Center Shopping Uberlândia	Short Listrado	2	102	204
7085	69996	2019-12-26	Center Shopping Uberlândia	Mochila	4	270	1080
7086	69996	2019-12-26	Center Shopping Uberlândia	Pulseira Estampa	1	87	87
7087	69997	2019-12-26	Ribeirão Shopping	Camisa Listrado	1	108	108
7088	69997	2019-12-26	Ribeirão Shopping	Short Linho	2	133	266

Parte 9

# Criando um relatório via e-mail



# Criando um relatório via e-mail

## Passo a passo

Temos nossos indicadores calculados.

Usando novamente o pyautogui, vamos criar códigos que nos permitam enviar esses indicadores via e-mail.

Os passos que precisamos seguir são:

- 1) **ABRIR** o gmail;
- 2) Clicar em **ESCREVER**( novo e-mail);
- 3) Escrever o **DESTINATÁRIO**;
- 4) Selecionar o campo **ASSUNTO**;
- 5) Selecionar o campo **CORPO DO E-MAIL** e escrever o e-mail usando os indicadores calculados;
- 6) **ENVIAR** E-mail;



## Criando um relatório via e-mail

# Abrindo o e-mail

Começando pelo passo 1, vamos entender quais são as linhas de código necessárias para executar a tarefa de abrir o Gmail.

```
# abrir aba gmail  
pyautogui.hotkey('ctrl', 't')  
pyautogui.write("mail.google.com")  
pyautogui.press('enter')  
time.sleep(5)
```

Assim como feito anteriormente, usaremos o método `hotkey()` para abrir uma nova aba

Ao abrirmos a página, a barra de endereço está automaticamente selecionada. Assim, usaremos o método `.write()` para escrever o endereço do gmail.

Ao abrirmos a página, a barra de endereço está automaticamente selecionada. Assim, usaremos o método `.write()` para escrever o endereço do gmail.

Aguarda a abertura da página antes de prosseguir para a próxima linha de código.

# Criar um novo e-mail

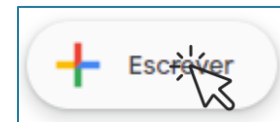
Com o GMAIL aberto, vamos agora criar um novo e-mail.

Antes de clicarmos precisamos descobrir ONDE clicar. Para isso, vamos usar novamente o método `.position()`.

**DICA:** Em uma célula separada, use o `sleep` para retardar a execução do `pyautogui` e usando o mouse posicione-o sobre o botão Escrever e aguarde.

Como nos utilizamos do **PRINT**, ele irá nos fornecer as coordenadas 😊. Agora fica fácil de seguir, basta colocar as coordenadas dentro do nosso `pyautogui.click()` conforme apresentado ao lado.

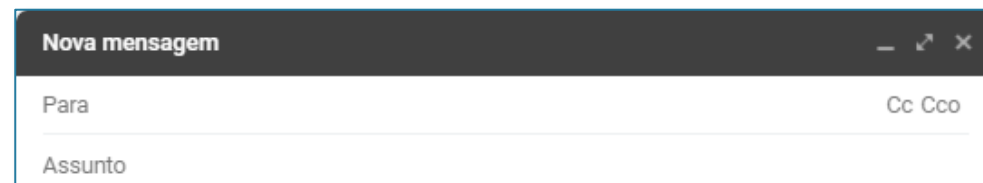
```
time.sleep(5)
print(pyautogui.position())
```



```
Point(x=94, y=157)
```

Uso das coordenadas no método `click()`

```
pyautogui.click(94, 157)
```

A imagem mostra a interface de uma nova mensagem no Gmail, com campos para "Para", "Assunto" e "Cc Cco".

# Criando um relatório via e-mail

## Inserindo cabeçalho

Perceba que ao criarmos um novo e-mail, o campo **DESTINATÁRIO**, já está selecionado, logo, só nos resta escrever a informação que queremos.

Feito essa etapa, vamos para a próxima informação necessária, o cabeçalho.

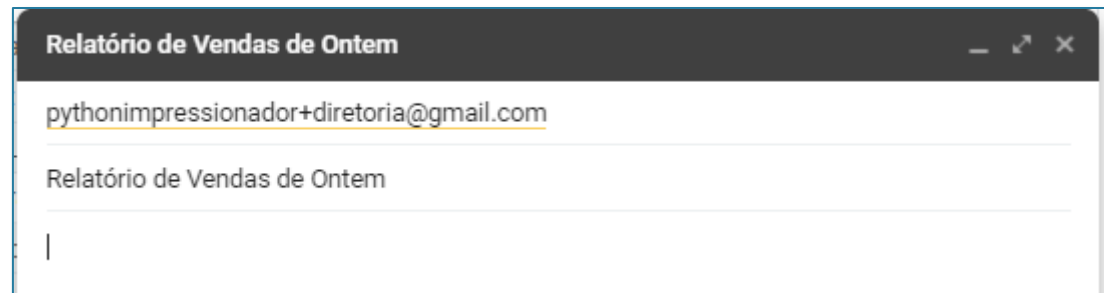
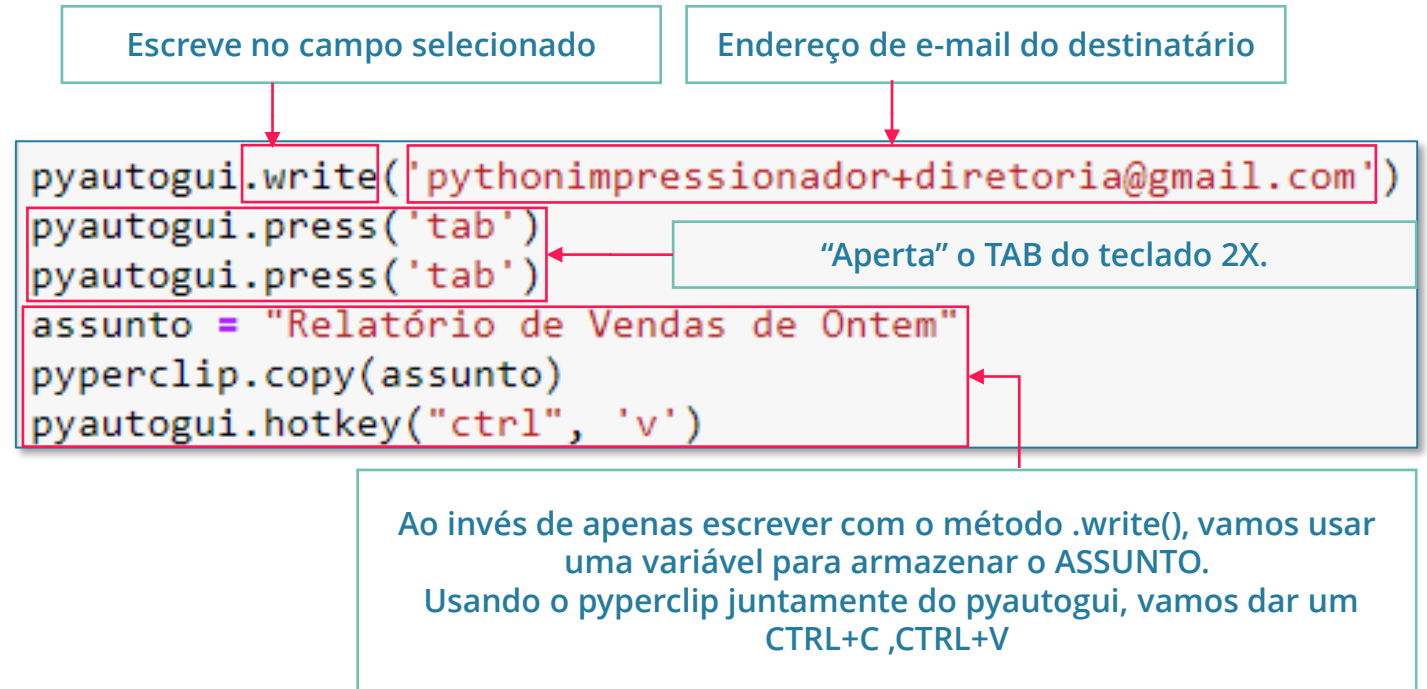
Temos algumas opções aqui, mas uma das mais simples é ao invés de usarmos o mouse para selecionar, é utilizar o **TAB**.

Perceba que nesse caso, usamos o **TAB 2X** seguidas:

**Primeira:** “Finaliza” o preenchimento do e-mail;

**Segunda:** Muda do campo **DESTINATÁRIO** para o campo o campo **ASSUNTO**;

**DICA:** Antes de programar, teste você mesmo executando as operações para ter mais clareza do que o Python está executando 😊.



# Escrevendo o corpo do E-mail (1/2)

Estamos no campo assunto e precisamos iniciar a escrever nosso corpo do e-mail.

Primeiro passo é acessar esse campo. Assim como fizemos no passo anterior, vamos utilizar o `pyautogui.press('tab')`.

Aqui temos um fator novo. Nosso texto, não é um texto fixo. Como estamos automatizando esse relatório, é importante que nosso texto consiga coletar as informações calculadas na etapa de cálculo dos indicadores **FATURAMENTO** e **QTDE\_PRODUTOS**.

Para isso, usaremos novamente o recurso da criação de uma variável que armazenará todo o texto do corpo do e-mail. Essa variável será chamada **texto**.

Perceba na imagem ao lado que além de apenas texto, usamos a estrutura `{ ... }`.

Essa estrutura nos permite criar um texto variável que se utiliza das variáveis `faturamento` e `qtde_produtos` calculadas anteriormente.

**f** indica que haverá um texto com variáveis INPUTADAS a ele

Texto fixo, sempre estará no relatório.

```
pyautogui.press("tab")
texto = f"""
Prezados, bom dia

O faturamento de ontem foi de: R${faturamento:,.2f}
A quantidade de produtos foi de: {qtde_produtos:,}

Abs
LiraPython"""
```

**faturamento:** será substituído pelo valor calculado na etapa de cálculo;

**:.2f:** Indica a formatação que esse valor será apresentado.

**(:)** indica que haverá uma formatação

**(,)** indica que o separador de milhar será VÍRGULA

**(.2f)** indica que o separador de casa decimais será PONTO e serão 2 casas decimais

**qtde\_produtos:** será substituído pelo valor calculado na etapa de cálculo;

**(:)** indica que haverá uma formatação

**(,)** indica que o separador de milhar será VÍRGULA

## Escrevendo o corpo do E-mail (2/2)

Cursor posicionado no campo de corpo de e-mail.

Texto armazenado na nossa variável.

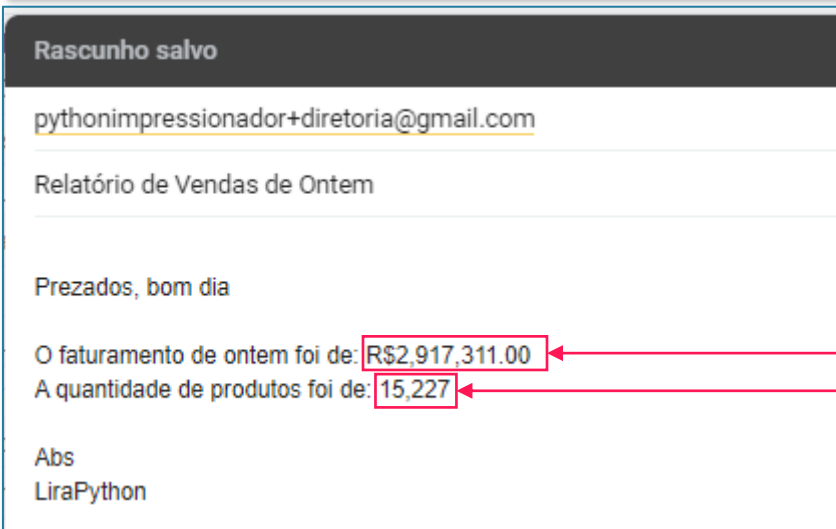
Agora utilizando novamente o pyperclip e o pyautogui, basta copiar e colarmos o valor no campo desejado.

Feito isso, basta **ENVIAR** o e-mail usando o atalho **CTRL+ ENTER**.

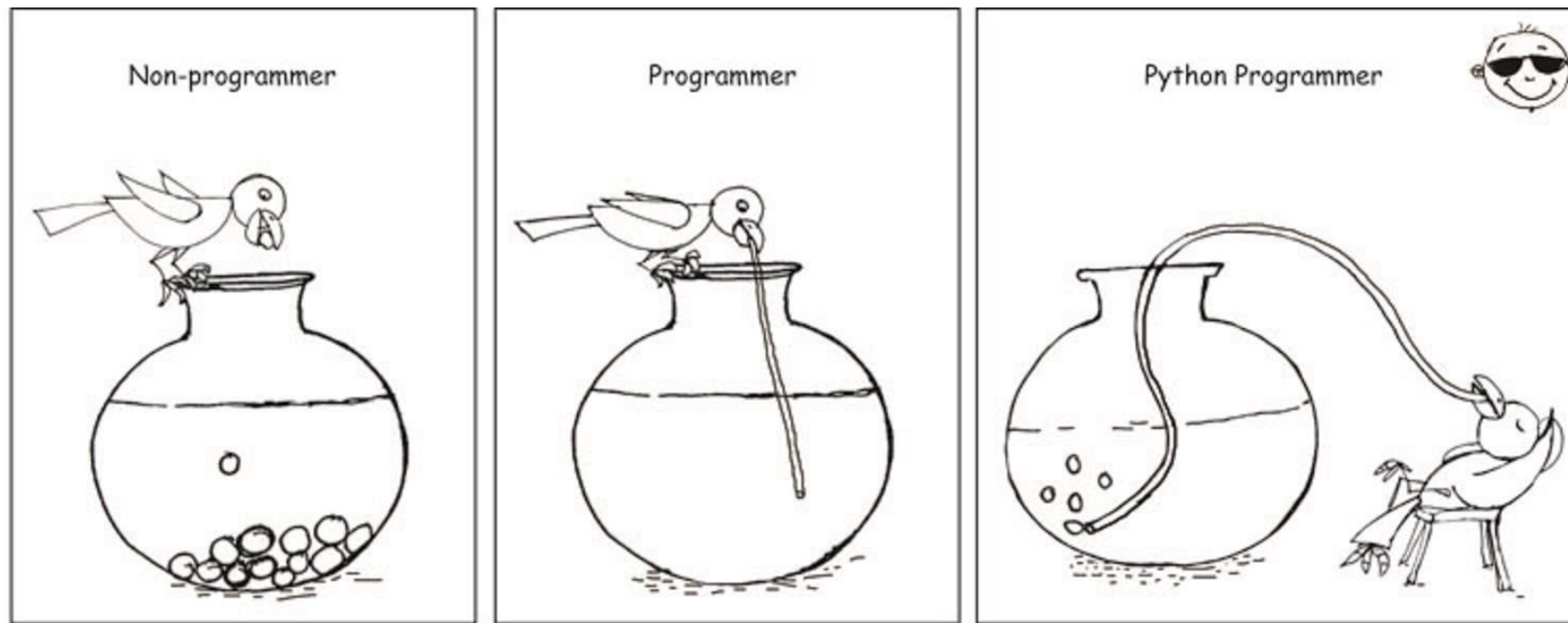
```
pyautogui.press("tab")
texto = f"""
Prezados, bom dia

O faturamento de ontem foi de: R${faturamento:,.2f}
A quantidade de produtos foi de: {qtde_produtos:,}

Abs
LiraPython"""
pyperclip.copy(texto)
pyautogui.hotkey("ctrl", 'v')
pyautogui.hotkey('ctrl', 'enter')
```



# PRONTO! AGORA É SÓ IMPRESSIONAR O CHEFE



# INTENSIVÃO DE PYTHON {#}

100% ONLINE & GRATUITO

Ainda não segue a gente no Instagram e nem é inscrito no nosso canal do Youtube? Então corre lá!



@hashtagprogramacao



[youtube.com/hashtag-programacao](https://youtube.com/hashtag-programacao)

