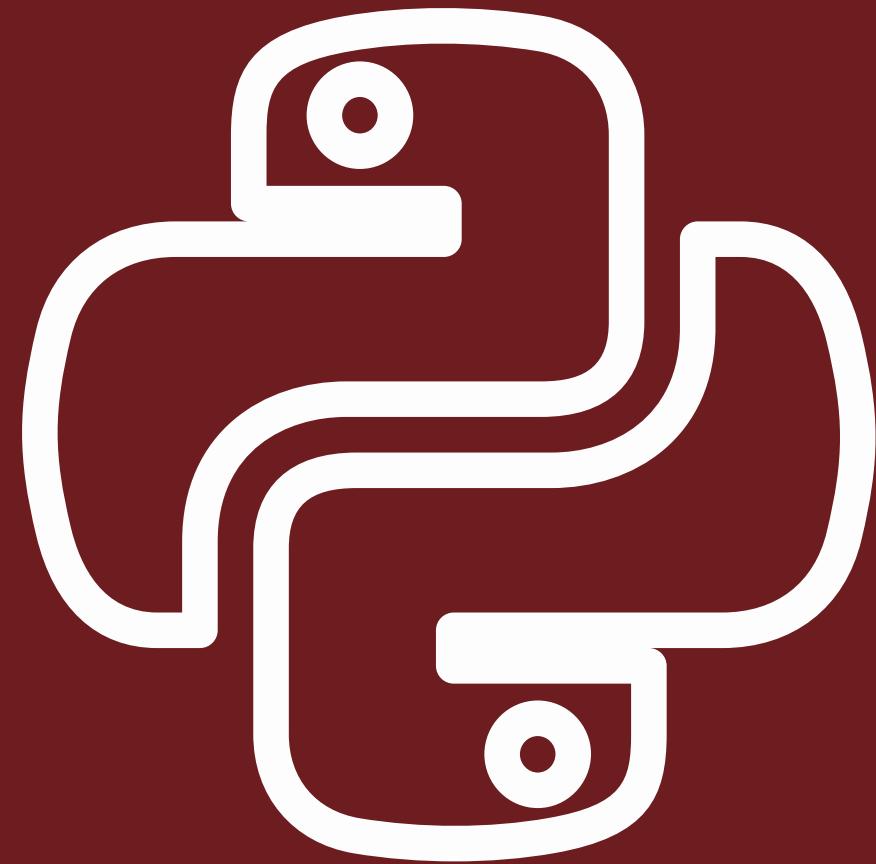
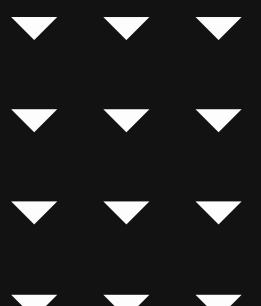
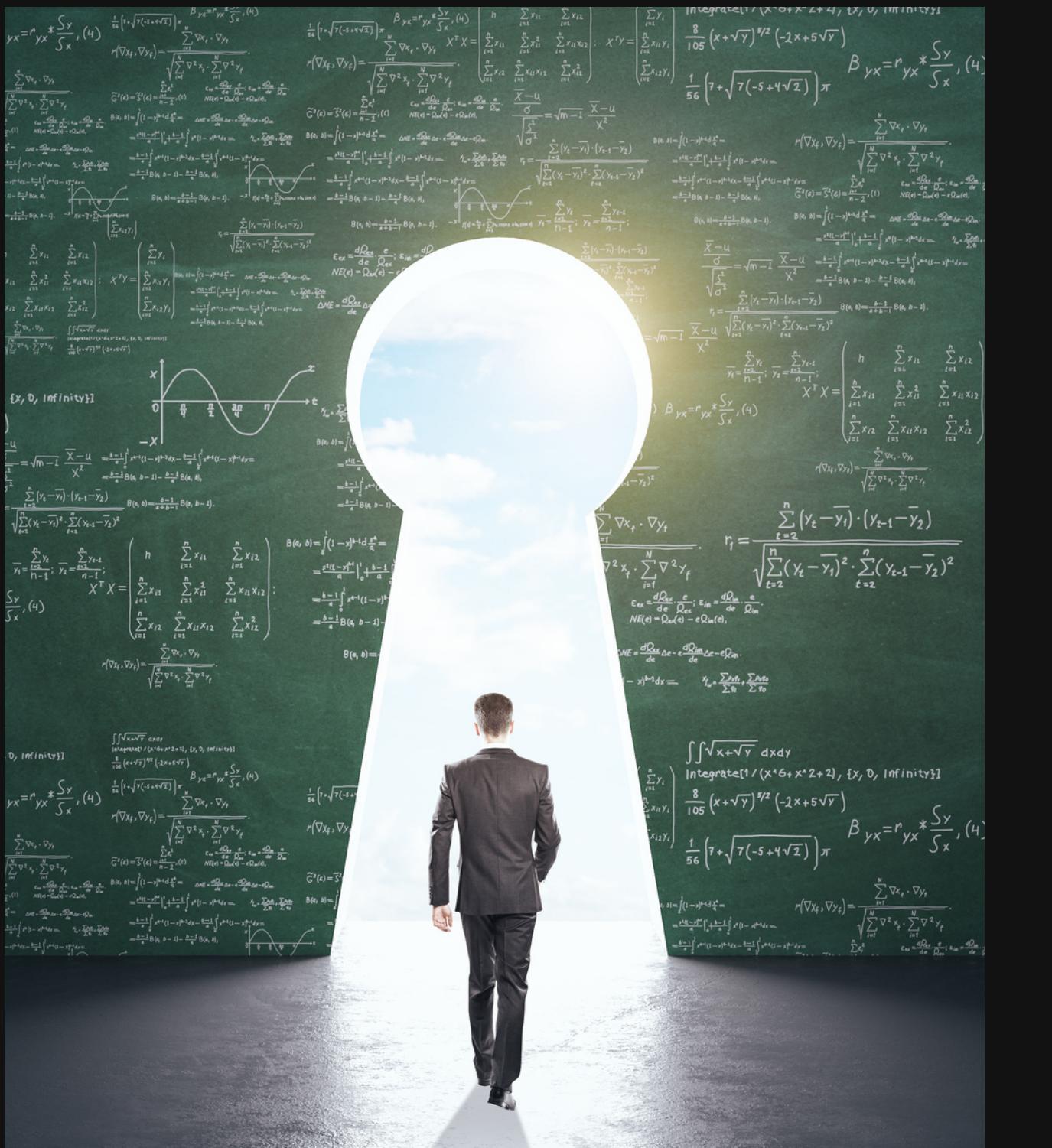


# ESTRUTURA DE DADOS





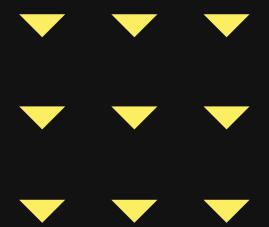
# INTRODUÇÃO AO PYTHON



# INTRODUÇÃO AO PYTHON



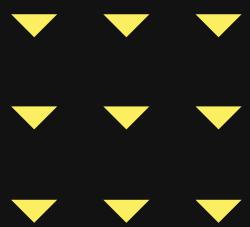
Criada em 1991 por Guido Van Rossum



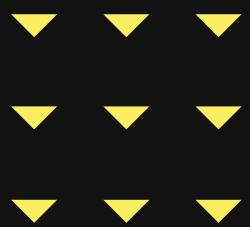
# INTRODUÇÃO AO PYTHON



Simples e de fácil  
aprendizado!

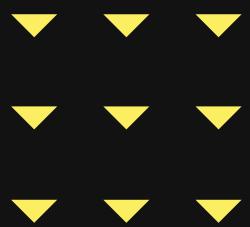


# INTRODUÇÃO AO PYTHON

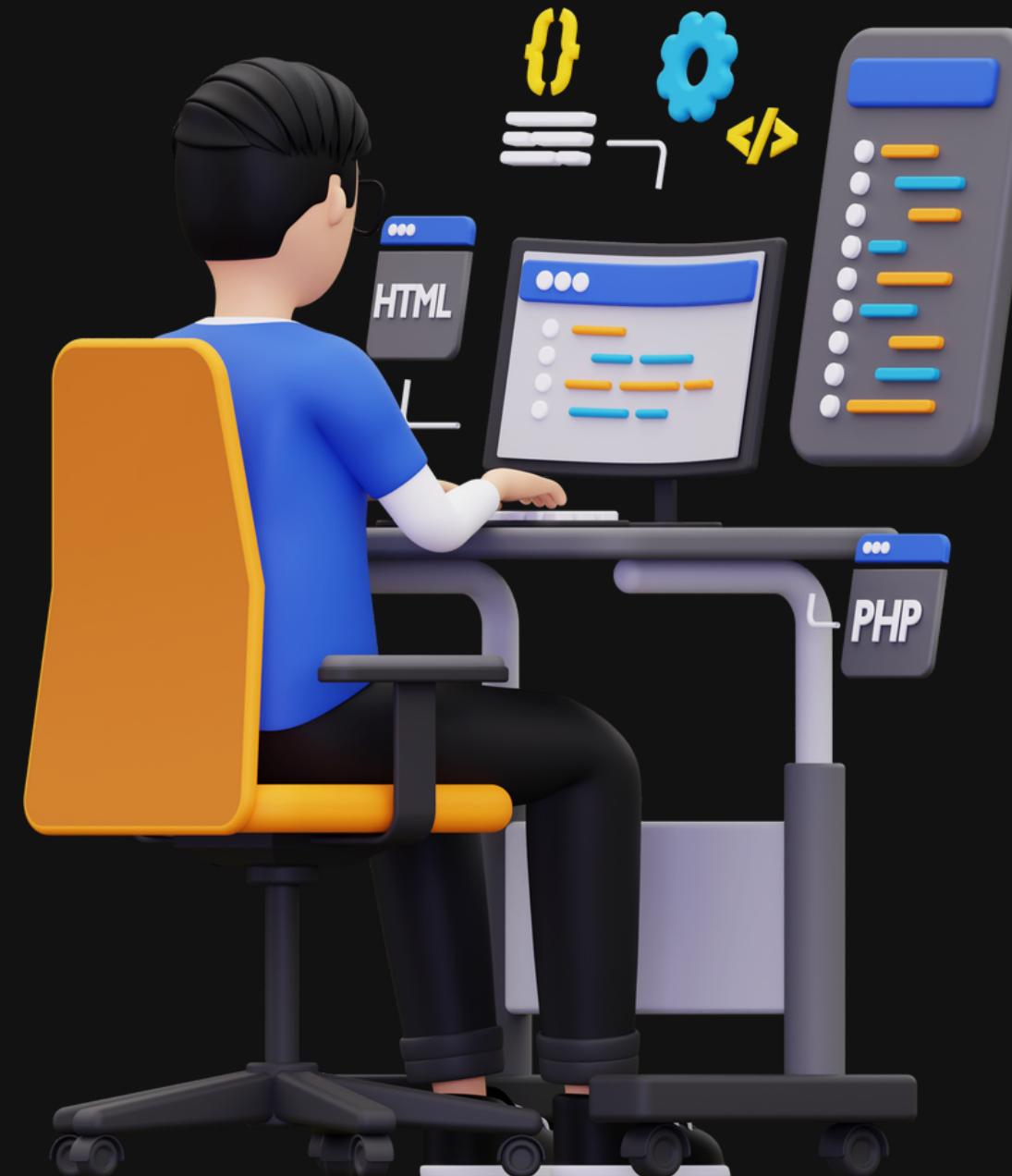
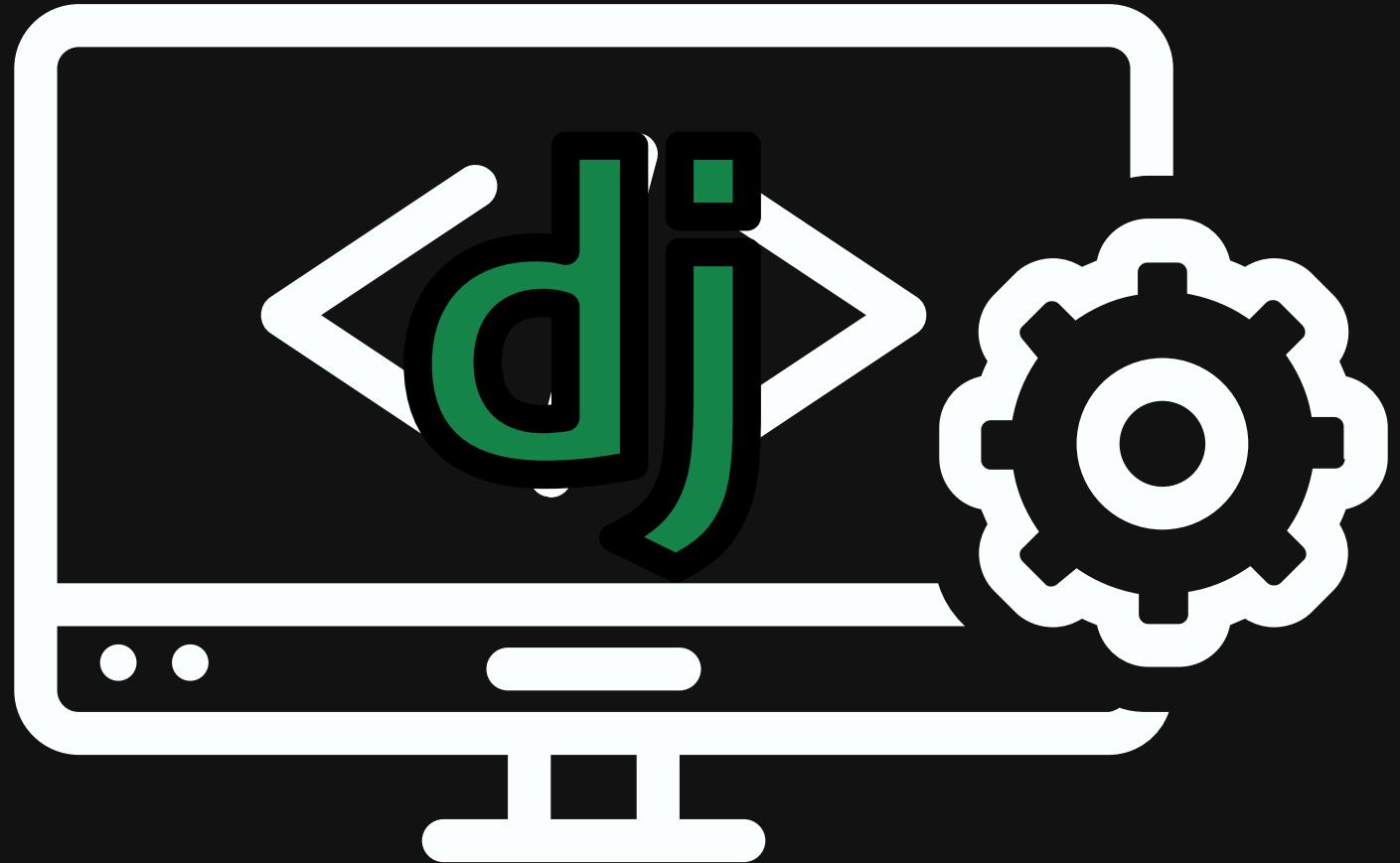


Portátil

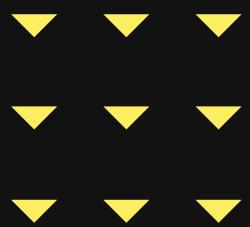
# INTRODUÇÃO AO PYTHON



## Desenvolvimento Web

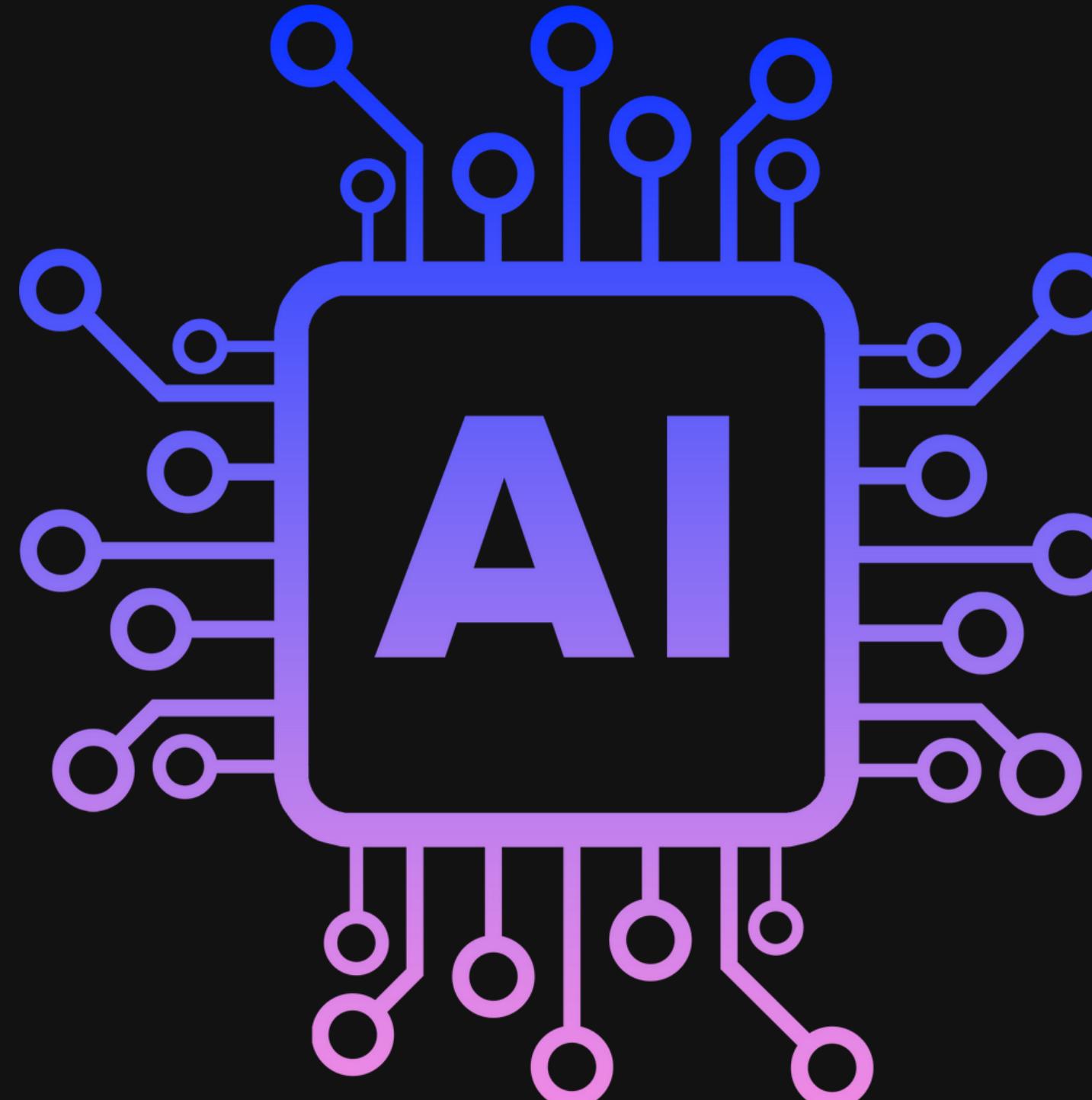


# INTRODUÇÃO AO PYTHON

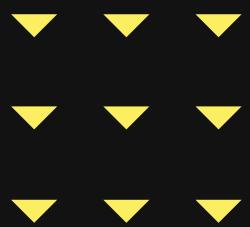


- Simples e de fácil aprendizado!
- Portátil
- Desenvolvimento Web
- Inteligência Artificial

# INTRODUÇÃO AO PYTHON



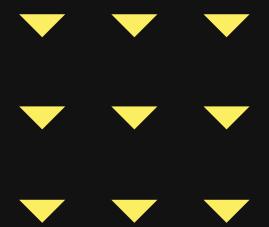
Inteligência Artificial



# INTRODUÇÃO AO PYTHON



Big Data

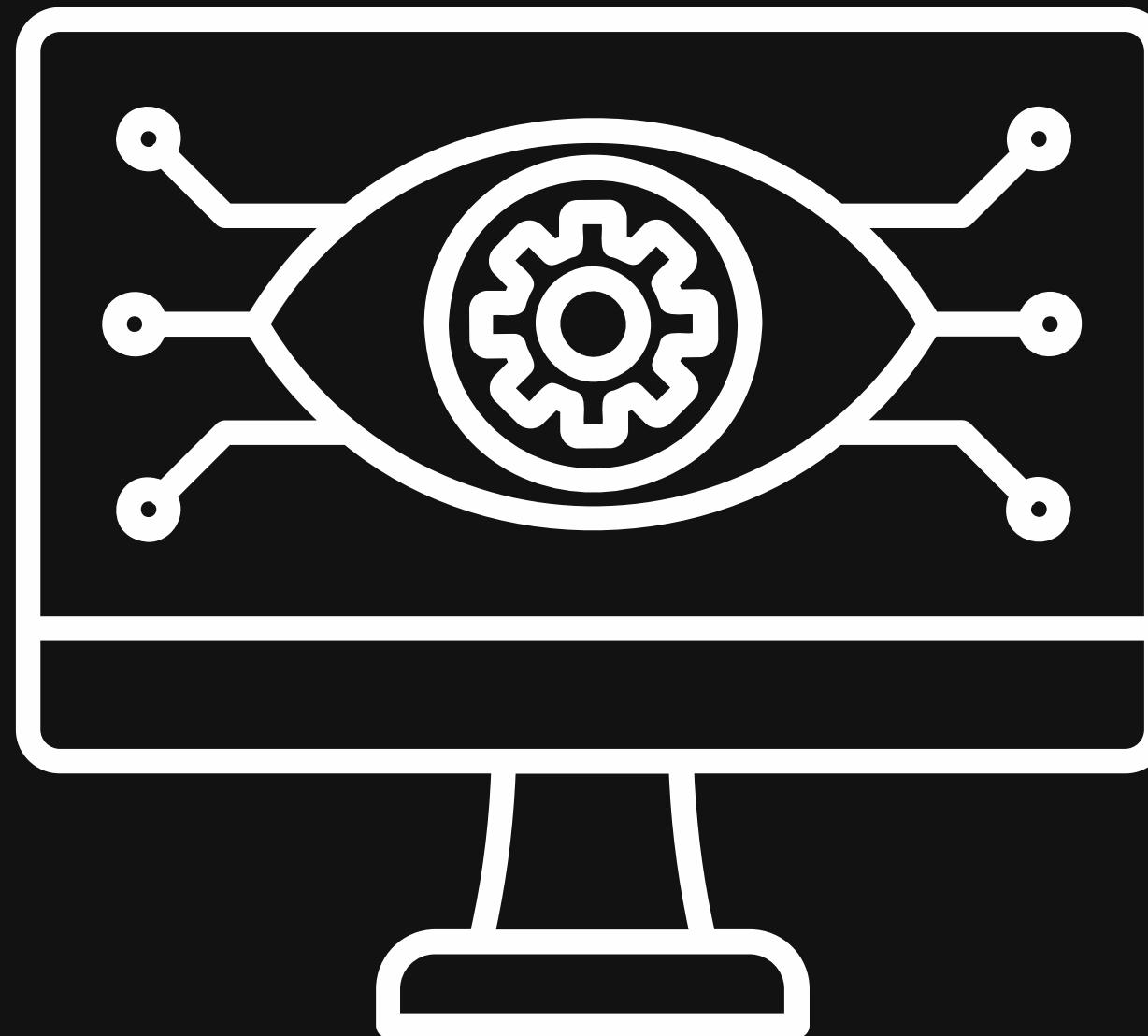
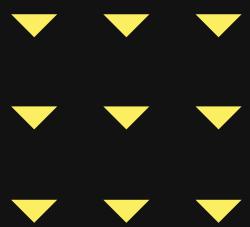


# INTRODUÇÃO AO PYTHON



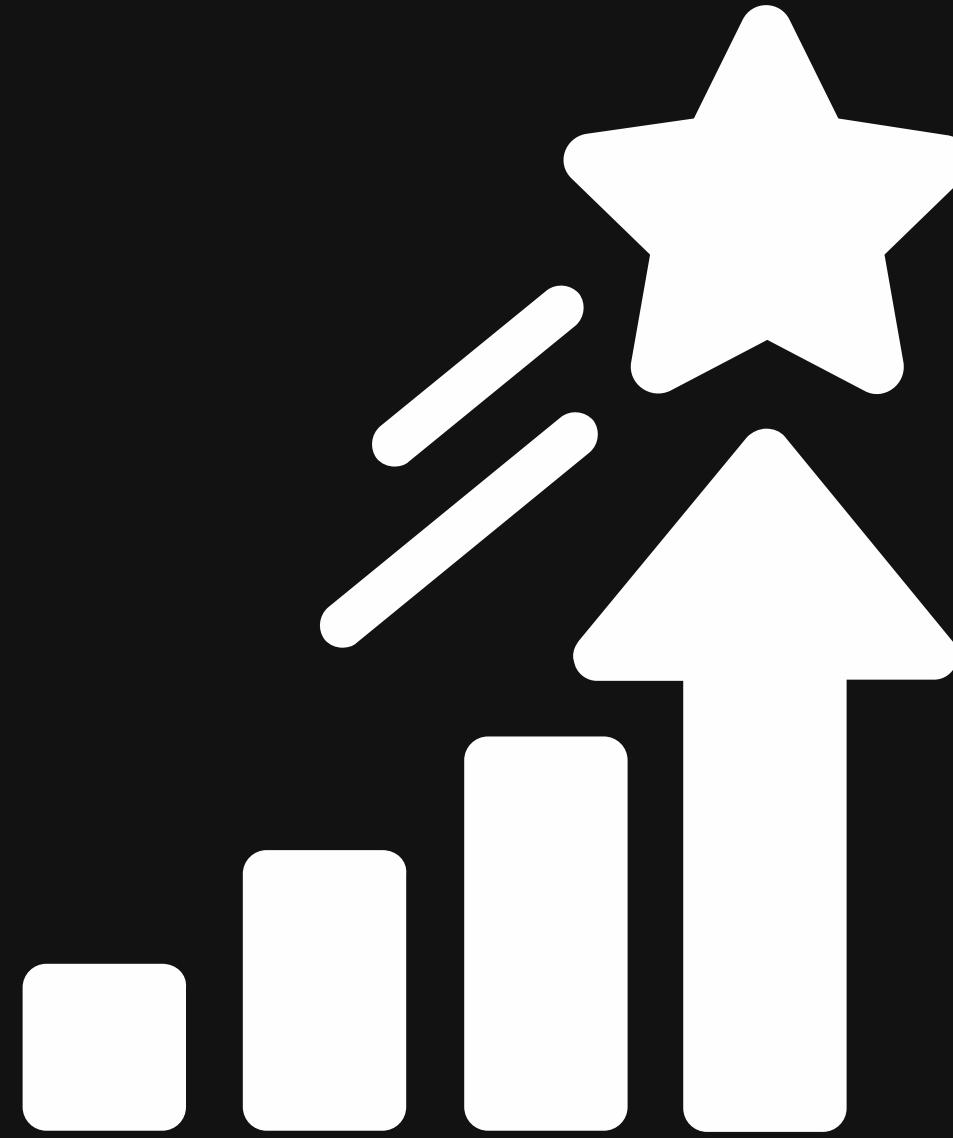
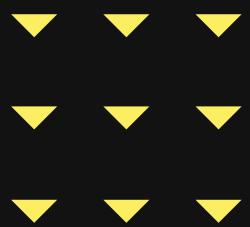
Ciência de Dados

# INTRODUÇÃO AO PYTHON



Computação Gráfica

# INTRODUÇÃO AO PYTHON



Popularidade

# INTRODUÇÃO AO PYTHON

## Rank das linguagens mais populares

Feb 2024	Feb 2023	Change	Programming Language	Ratings	Change
1	1		 Python	15.16%	-0.32%
2	2		 C	10.97%	-4.41%
3	3		 C++	10.53%	-3.40%

# INTRODUÇÃO AO PYTHON

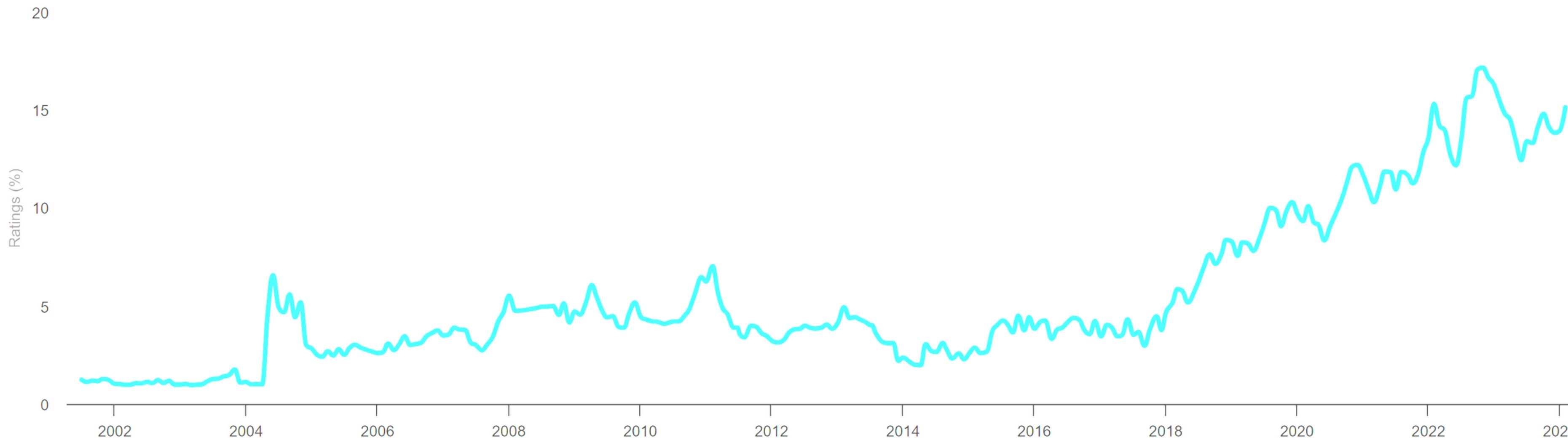
⬆️ Highest Position (since 2001): #1 in Feb 2024

⬇️ Lowest Position (since 2001): #13 in Feb 2003

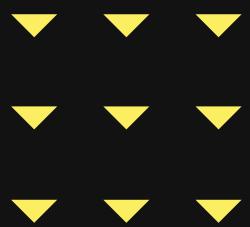
🏆 Language of the Year: 2007, 2010, 2018, 2020, 2021

TIOBE Index for Python

Source: [www.tiobe.com](http://www.tiobe.com)



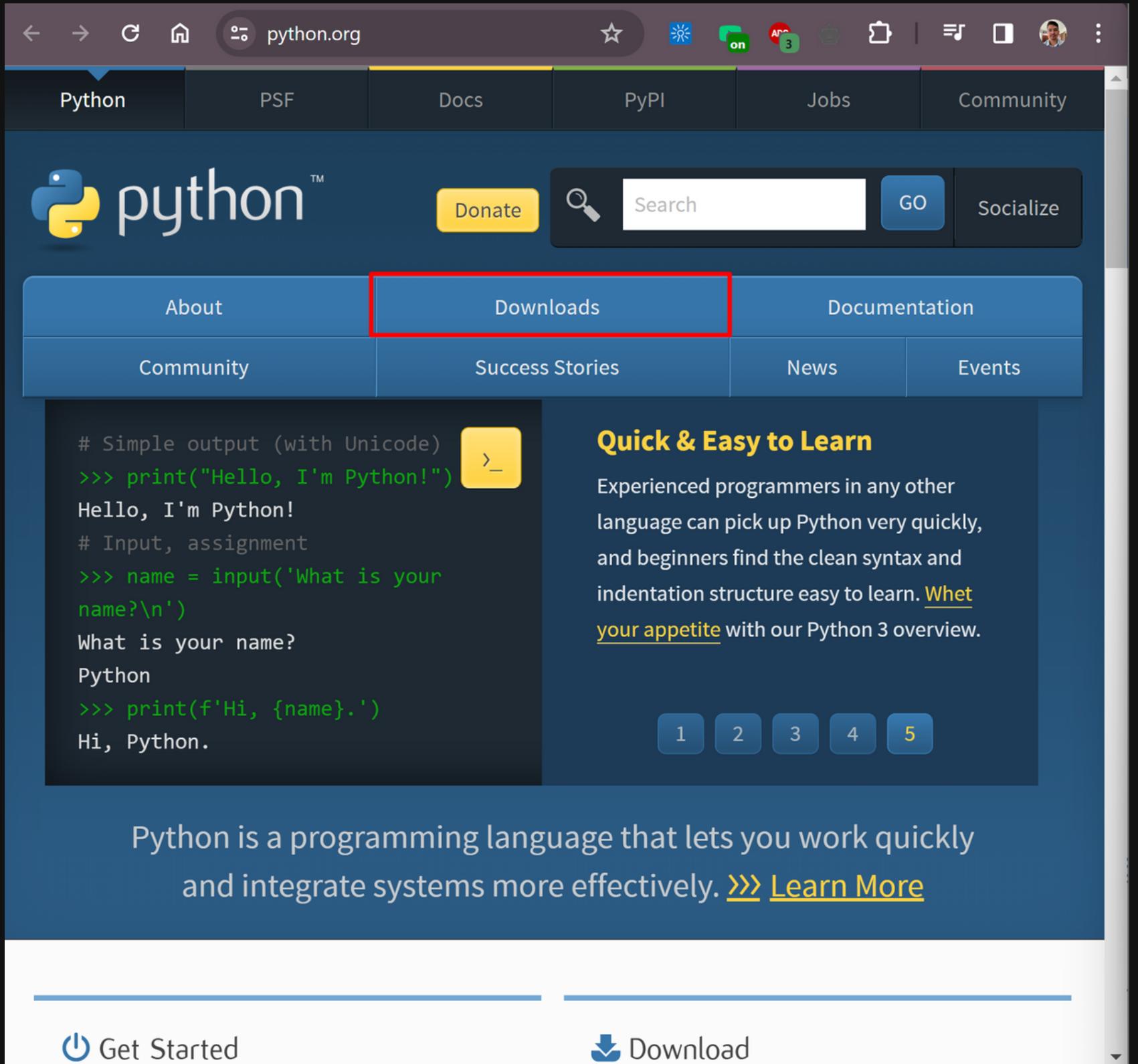
# INTRODUÇÃO AO PYTHON



Verifique por você mesmo como anda o rank atual

# INSTALAÇÃO DO PYTHON

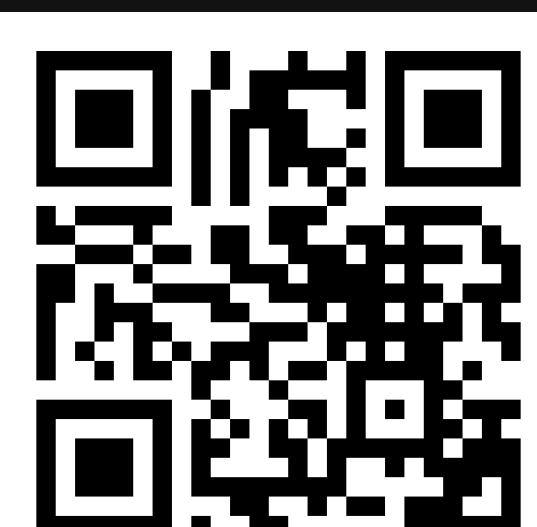
## Instalando o Python



A screenshot of a web browser displaying the Python.org homepage. The URL 'python.org' is visible in the address bar. The page features a dark blue header with various navigation links like 'Python', 'PSF', 'Docs', 'PyPI', 'Jobs', and 'Community'. Below the header, there's a large Python logo and a search bar. A red box highlights the 'Downloads' button in the main navigation menu, which is currently active. To the right of the menu, there's a section titled 'Quick & Easy to Learn' with some introductory text and a 'What your appetite' link. On the left, there's a code editor window showing a simple Python script. At the bottom, there's a call-to-action button labeled 'Get Started' and another labeled 'Download'.

## Passo 1

Vá para o site  
<https://www.python.org/>



# INSTALAÇÃO DO PYTHON

## Instalando o Python

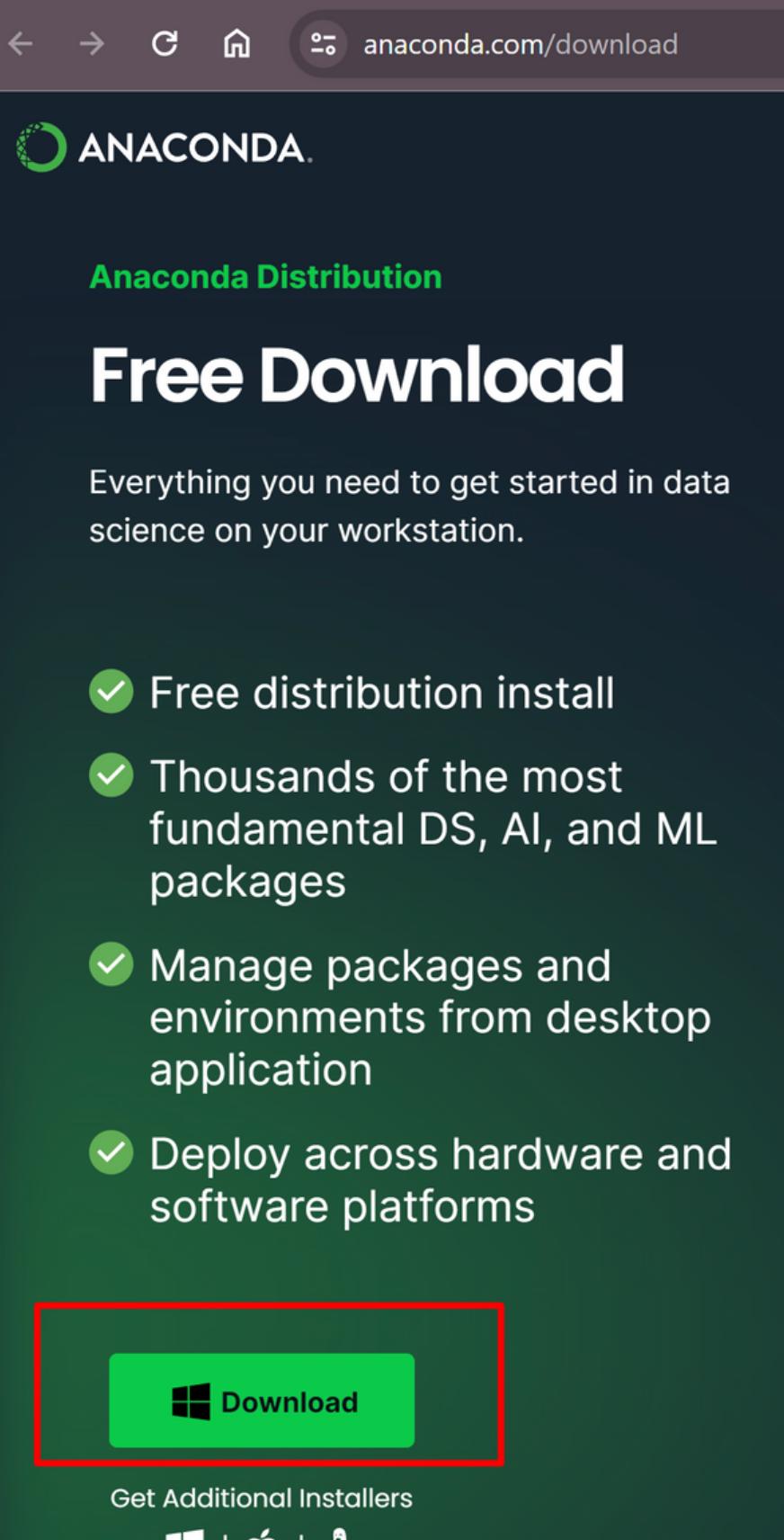


The screenshot shows the Python.org Downloads page. At the top, there's a navigation bar with links for Python, PSF, Docs, PyPI, Jobs, and Community. Below the navigation is a search bar with a magnifying glass icon and a 'GO' button. To the right of the search bar are 'Socialize' and 'Donate' buttons. The main content area features a large blue banner with the Python logo and the text 'python™'. Below the banner, there are tabs for About, Downloads, Documentation, Community, Success Stories, News, and Events. A prominent yellow button labeled 'Download Python 3.12.2' is highlighted with a red border. To the left of the button, the text 'Download the latest version of Python' is displayed. In the center of the page is a cartoon illustration of two boxes hanging from parachutes against a blue sky with clouds. Below the illustration, there's text about different OS versions and development prereleases. At the bottom, there's a section titled 'Active Python Releases' with a link to the Python Developer's Guide.

### Passo 1

Faça o **Download** da última versão e instale-a.

# INSTALAÇÃO DO PYTHON



A screenshot of a web browser showing the Anaconda Distribution download page. The URL in the address bar is [anaconda.com/download](https://anaconda.com/download). The page features a dark green header with the Anaconda logo and the text "Anaconda Distribution". Below this, a large white button labeled "Free Download" is prominently displayed. To its right, the text "Everything you need to get started in data science on your workstation." is visible. A bulleted list of benefits follows, each preceded by a green checkmark:

- ✓ Free distribution install
- ✓ Thousands of the most fundamental DS, AI, and ML packages
- ✓ Manage packages and environments from desktop application
- ✓ Deploy across hardware and software platforms

At the bottom of the page, there is a red-bordered green button with the Windows logo and the word "Download". Below this button, the text "Get Additional Installers" is followed by icons for Windows, Mac, and Linux.

## Instalando o Python - forma alternativa

### Passo 1

No site

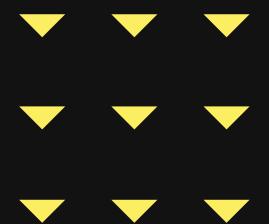
<https://www.anaconda.com/download>

Faça o **Download** da última versão e instale-a.



# INSTALAÇÃO DO PYTHON

## Instalando o Python - forma alternativa



ANACONDA.

### Thank you for downloading!

Didn't download? [Go here](#) to download your version. For installation assistance, refer to [Troubleshooting](#).

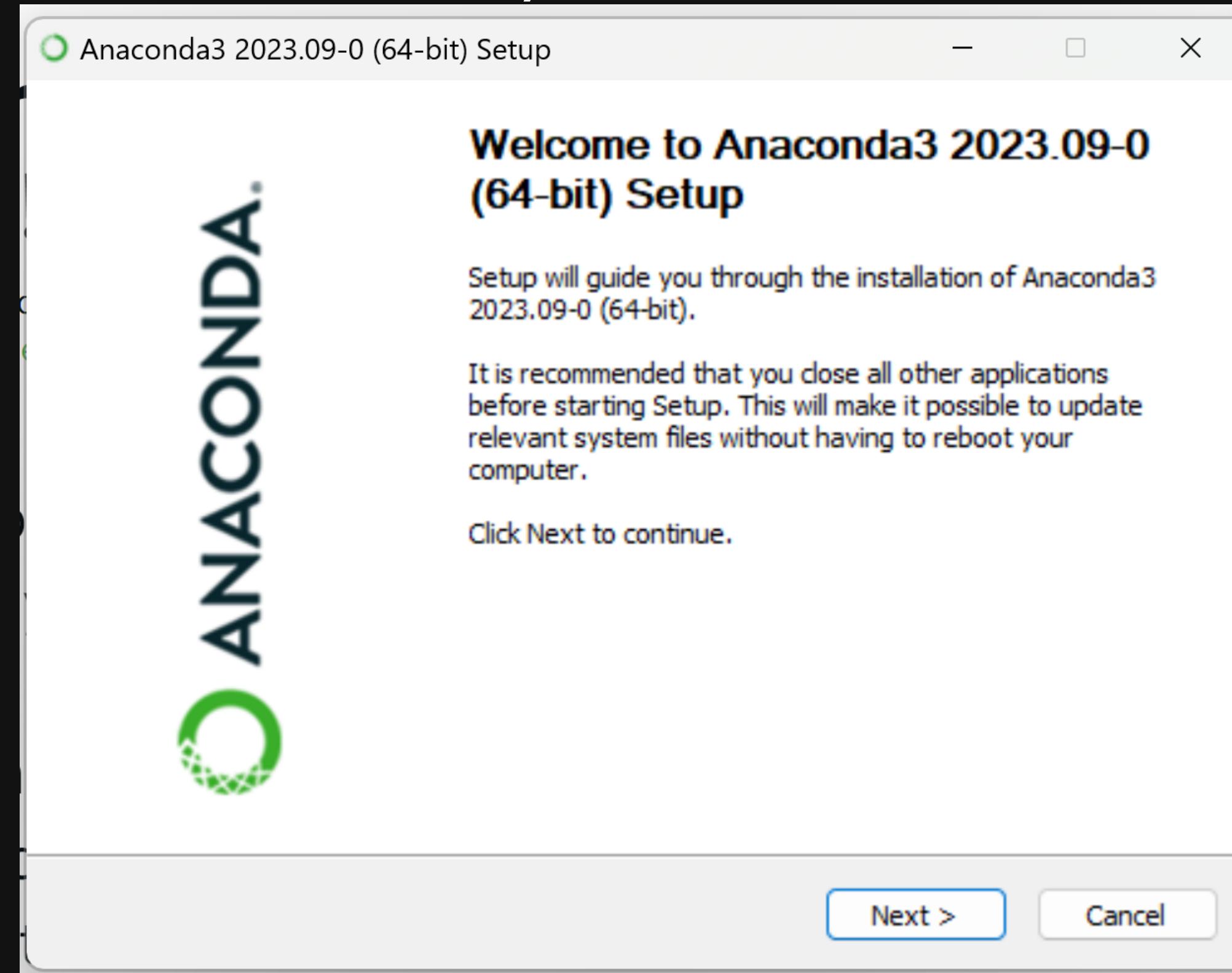


Skip the installation process with our cloud-based, JupyterLab notebook experience – fully-loaded with Anaconda packages, Panel app deployment, sample data catalogs, and notebook templates.

[Launch Cloud Notebooks >](#)

# INSTALAÇÃO DO PYTHON

## Instalando o Python - forma alternativa



# INSTALAÇÃO DO PYTHON

A vantagem de se utilizar o Anaconda é a questão da possibilidade de já instalar previamente bibliotecas necessárias para desenvolvimento web, análise de dados e IA... além da facilidade de se manipular ambientes virtuais.

# INSTALAÇÃO DO PYTHON



## Data Exploration and Transformation

- pandas

- Intake

- NumPy

- Dask

- Apache Airflow



## Visualization

- Matplotlib

- seaborn

- Plotly

- Bokeh

- HoloViews



## AI and Machine Learning

- scikit-learn

- TensorFlow

- Keras

- PyTorch

- XGBoost

# INSTALAÇÃO DO PYTHON



## Natural Language Processing

- NLTK

- Gensim

- Transformers

- PyTorch



## GUI and Front-End Development

- Flask

- Django

- Pyramid

- Tornado

- CherryPy



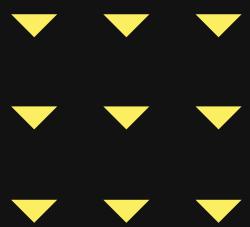
## R Packages

- ggplot2

- dplyr

- tidyr

- readr



Instale um editor de código ou uma IDE

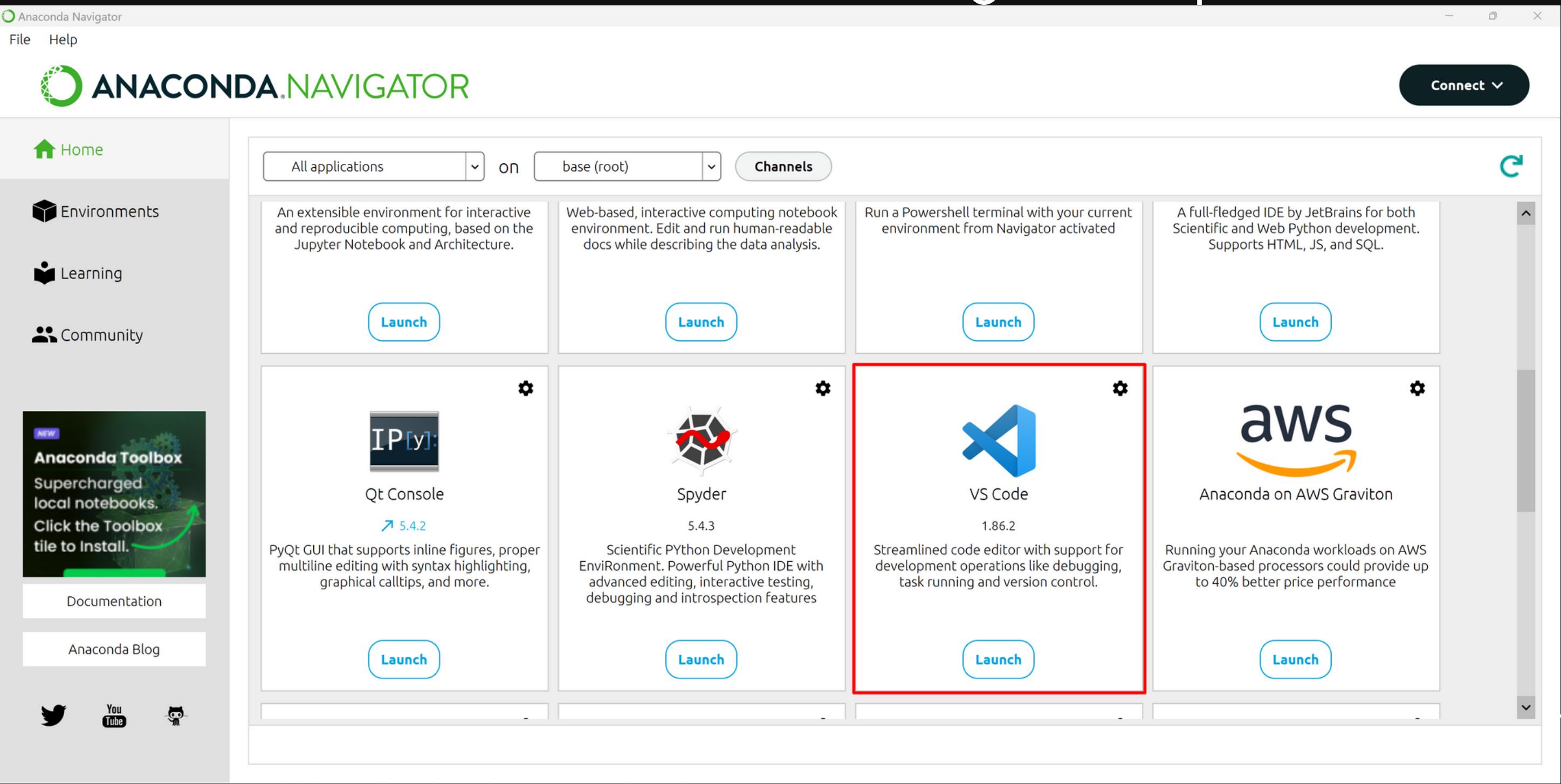
Temos duas formas de resolver isso, uma  
indo pelo caminho da Anaconda.

Outra indo da forma tradicional.

# IDEs PARA PYTHON

Instale um **editor de código** ou uma IDE  
Pelo caminho da Anaconda.

Só abrir o Anaconda navigator e pronto, resolvido!

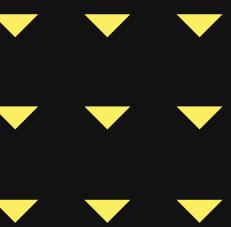


**Launch** = só meter  
bronca

**Install** = tem jeito  
não, vai ter que  
instalar

# IDEs PARA PYTHON

Instale um **editor de código** ou uma IDE  
Pelo caminho tradicional. Baixar e instalar.



code.visualstudio.com/download

Visual Studio Code Docs Updates Blog API Extensions FAQ Learn Search Docs

Version 1.86 is now available! Read about the new features and fixes from January.

## Download Visual Studio Code

Free and built on open source. Integrated Git, debugging and extensions.

[Windows](#) [.deb](#) [.rpm](#) [Mac](#)

Windows 10, 11   Debian, Ubuntu   Red Hat, Fedora, SUSE   macOS 10.15+

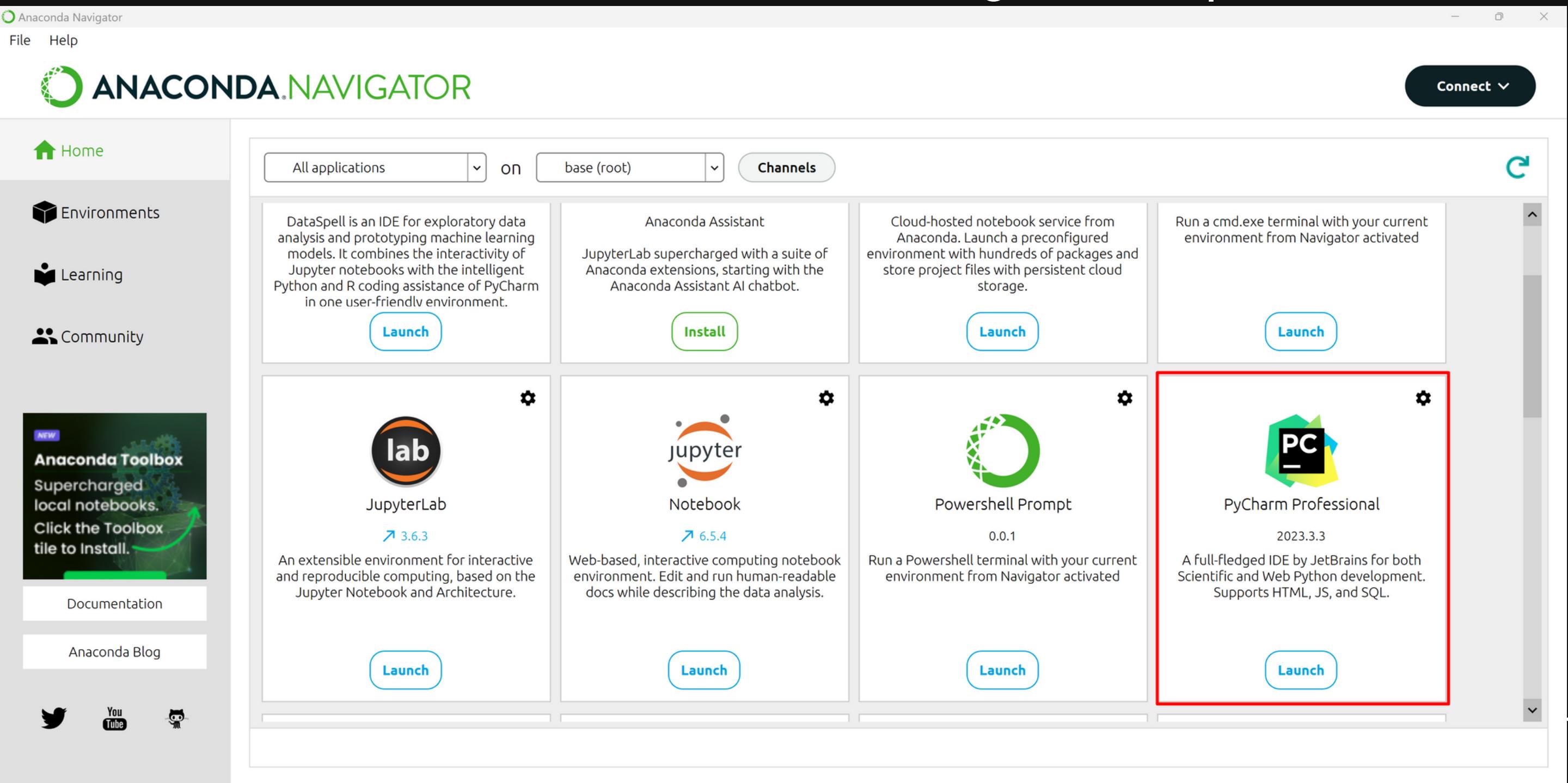
User Installer   .deb   .zip   .tar.gz   Snap   CLI  
System Installer   x64   Arm64   x64   Arm32   x64   Intel chip  
.zip   x64   Arm64   x64   Arm32   x64   Apple silicon  
CLI   x64   Arm64   x64   Arm32   x64   Universal



# IDEs PARA PYTHON

Instale um editor de código ou uma **IDE**  
Pelo caminho da Anaconda.

Só abrir o Anaconda navigator e pronto, resolvido!



Launch = só meter  
bronca

Install = tem jeito  
não, vai ter que  
instalar

# IDEs PARA PYTHON

Instale um editor de código ou uma **IDE**  
Pelo caminho tradicional. Baixar e instalar.

The screenshot shows the official JetBrains PyCharm download page for Windows. At the top, there's a navigation bar with links like 'Para desenvolvimento', 'Para equipes', 'Educação', 'Soluções', 'Suporte', 'Loja', and 'Aprenda'. Below that is a main menu with 'PyCharm' selected, followed by 'JetBrains IDEs', 'Na nova versão', 'Casos de uso', 'Novidades', 'Funcionalidades', 'Aprenda', 'Preços', and a prominent green 'Baixar' button. A red box highlights the 'Baixar' button. The page also features a 'Windows' tab, 'macOS' and 'Linux' tabs, and a section for 'PyCharm Professional' which describes it as 'O IDE Python para desenvolvedores profissionais'. It includes a download link for 'Baixar .exe' and a note about a 30-day free trial. On the right side of the page, there's a large screenshot of the PyCharm IDE interface displaying Python code for a Django application. Below the screenshot, there are links for 'Versão: 2023.3.3', 'Build: 233.13763.11', '25 de janeiro de 2024', 'Requisitos do sistema', 'Outras versões', 'Instruções de instalação', and 'Softwares de terceiros'.

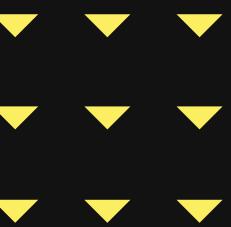
Vocês,  
enquanto alunos  
tem acesso a  
versão  
**Professional.**



# IDEs PARA PYTHON

Instale um editor de código ou uma **IDE**

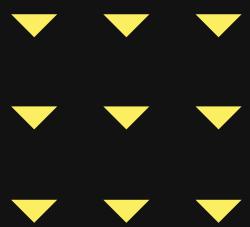
Pelo caminho tradicional. Baixar e instalar.



The screenshot shows a web browser window with the URL [jetbrains.com/pt-br/pycharm/download/?section=windows](https://jetbrains.com/pt-br/pycharm/download/?section=windows). The page is for the PyCharm Community Edition. It features a dark header with navigation links like 'Na nova versão', 'Casos de uso', 'Novidades', 'Funcionalidades', 'Aprenda', 'Preços', and a prominent green 'Baixar' button. The main content area has a dark background with white text. It starts with a paragraph about valuing the Python community and offering the PyCharm Community Edition for free. Below this, there's a section for 'PyCharm Community Edition' featuring its logo (a stylized 'PC'), a brief description, and download links for 'Baixar .exe'. A red box highlights this download section. At the bottom, it says 'Gratuito, com base em open source'.

Ou podem usar  
a versão  
gratuita mesmo.

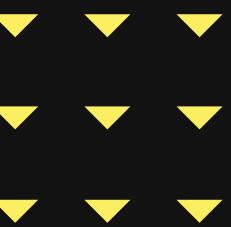




Mas qual a diferença entre um editor de código e uma IDE?

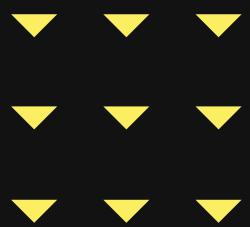
E onde a Anaconda entra nisso?

Mas qual a diferença entre um editor de código e uma IDE?



Editores de código são **editores de texto** com poderosos recursos integrados e funcionalidades especializadas para simplificar e acelerar o processo de edição de código.

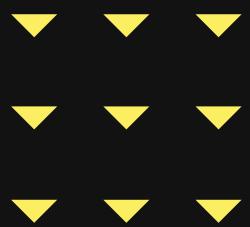
Já um IDE, por outro lado, é um **conjunto de ferramentas de desenvolvimento** de software projetadas para tornar a codificação mais fácil.



Essa guerra é eterna



### E onde a Anaconda entra nisso?



Anaconda é uma **distribuição de linguagem de programação Python e R**

Ela vem com um grande número de bibliotecas e ferramentas pré-instaladas para pesquisa de dados. É especialmente uma ferramenta popular entre cientistas de dados, analistas e pesquisadores. Se você deseja iniciar a ciência de dados, o Anaconda pode permitir que você comece de forma rápida e simples.

Você pode usar o **Conda**, o gerenciador de pacotes incluído no Anaconda, para instalar, atualizar e gerenciar bibliotecas de maneira conveniente.

# IDEs PARA PYTHON

E onde a Anaconda entra nisso?

The screenshot shows the Anaconda Navigator application window. On the left, there's a sidebar with links like Home, Environments (which is selected and highlighted in green), Learning, Community, Documentation, and Anaconda Blog. Below the sidebar is an advertisement for the Anaconda Toolbox. At the bottom are social media links for Twitter, YouTube, and GitHub.

The main area has a header with "Search Environments" and a search icon. Below it, there's a dropdown menu set to "Installed", a "Channels" button, and a "Update index..." button. To the right is a "Search Packages" input field with a magnifying glass icon.

The central part of the interface is divided into two sections. On the left, under "Environments", there's a list with "base (root)" selected and highlighted with a red box. A red arrow points from the text "AMBIENTE" to this list item. At the bottom of this section are buttons for Create, Clone, Import, Backup, and Remove.

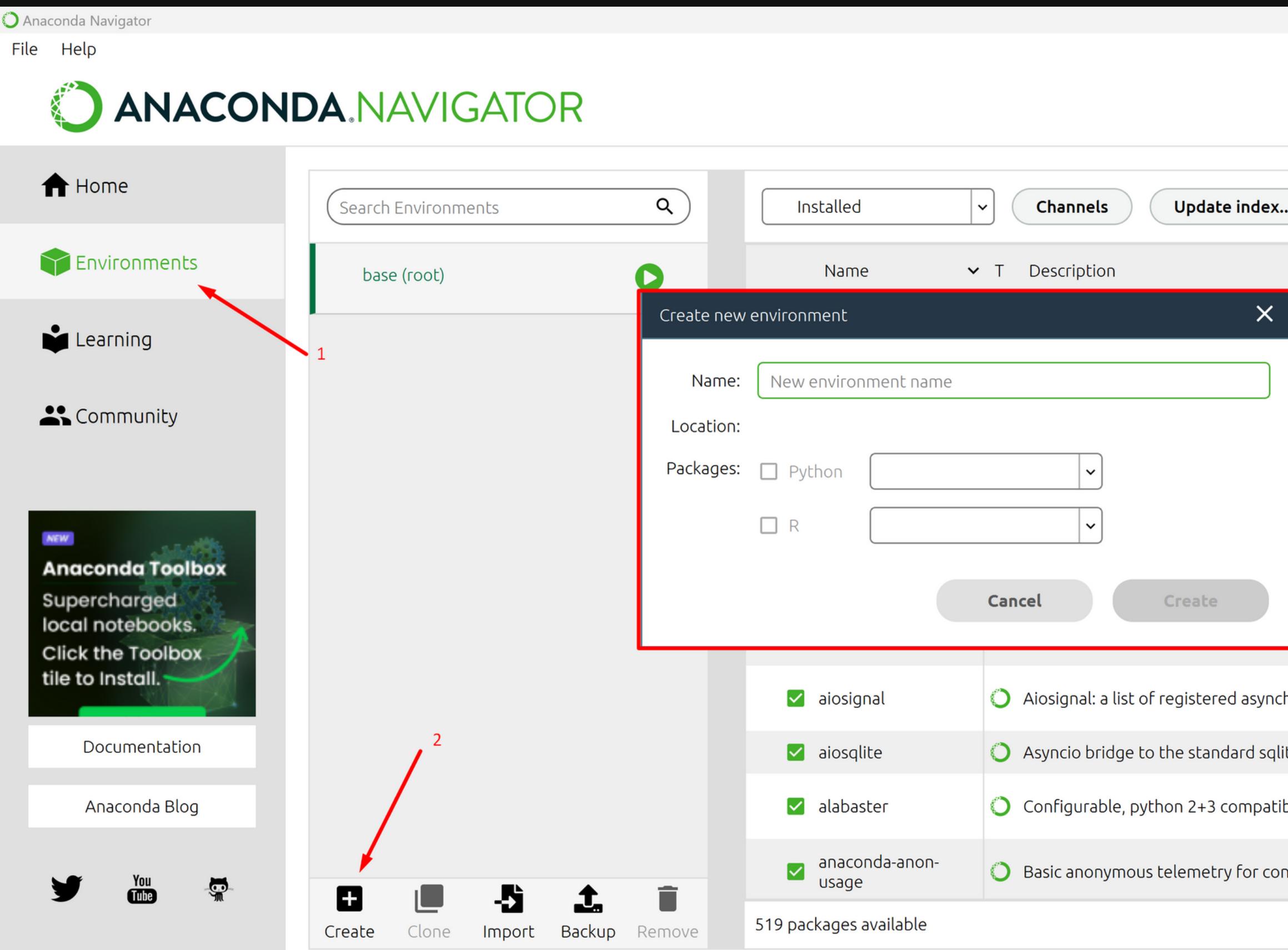
On the right, there's a table titled "Installed" with columns for Name, Description, and Version. The table is surrounded by a large red box. A red arrow points from the text "BIBLIOTECAS INSTALADAS" to the start of this table. The table contains the following data:

Name	Description	Version
_anaconda_depends	Simplifies package management and deployment of anaconda	2023.09
abseil-cpp	Abseil common libraries (c++)	2021110
aiobotocore	Async client for aws services using botocore and aiohttp	2.5.0
aiofiles	File support for asyncio	22.1.0
aiohttp	Async http client/server framework (asyncio)	3.8.5
aioitertools	Asyncio version of the standard multiprocessing module	0.7.1
aiosignal	Aiosignal: a list of registered asynchronous callbacks	1.2.0
aiosqlite	Asyncio bridge to the standard sqlite3 module	0.18.0
alabaster	Configurable, python 2+3 compatible sphinx theme.	0.7.12
anaconda-anon-usage	Basic anonymous telemetry for conda clients	0.4.2

At the bottom of the main area, it says "519 packages available".

# IDEs PARA PYTHON

E onde a Anaconda entra nisso?



Anaconda tem uma grande vantagem que é o gerenciamento de ambientes criados, é algo totalmente **user friendly**.

Facilmente dá para criar e excluir um ambiente

# IDEs PARA PYTHON

The screenshot shows the Anaconda Navigator interface. On the left, there's a sidebar with links like Home, Environments, Learning, Community, Documentation, and Anaconda Blog. A prominent 'Anaconda Toolbox' tile is also present. The main area has tabs for 'Search Environments', 'Installed' (selected), 'Channels', and 'Update index...'. Below these are buttons for 'Create', 'Clone', 'Import', 'Backup', and 'Remove'. A search bar for packages is at the top right. A modal window titled 'Create new environment' is open, showing fields for 'Name' (set to 'testel'), 'Location' (set to 'C:\Users\caio\conda\envs\teste'), and 'Packages'. Under 'Python', '3.11.7' is selected. Under 'R', '3.6.1' is listed. At the bottom of the modal are 'Cancel' and 'Create' buttons. In the background, a list of packages is visible, including aiosignal, aiosqlite, alabaster, anaconda-anon-usage, Aiosignal, Asyncio bridge to the standard sqlite3 module, Configurable, python 2+3 compatible sphinx theme, and Basic anonymous telemetry for conda clients. The version column shows various versions like 2023.09, 2021.11.0, 2.5.0, 22.1.0, 3.8.5, 0.7.1, 1.2.0, 0.18.0, 0.7.12, and 0.4.2.

Anaconda Navigator

File Help

ANACONDA NAVIGATOR Connect

Home Environments Learning Community

NEW Anaconda Toolbox Supercharged local notebooks. Click the Toolbox tile to Install.

Documentation Anaconda Blog

Create Clone Import Backup Remove

Search Environments Installed Channels Update index... Search Packages

base (root)

Name T Description Version

Create new environment

Name: testel

Location: C:\Users\caio\conda\envs\teste

Packages:

- Python 3.11.7
- R 3.6.1

Cancel Create

<input checked="" type="checkbox"/>	aiosignal	Aiosignal: a list of registered asynchronous callbacks
<input checked="" type="checkbox"/>	aiosqlite	Asyncio bridge to the standard sqlite3 module
<input checked="" type="checkbox"/>	alabaster	Configurable, python 2+3 compatible sphinx theme.
<input checked="" type="checkbox"/>	anaconda-anon-usage	Basic anonymous telemetry for conda clients

519 packages available

# IDEs PARA PYTHON

Anaconda Navigator

File Help

**ANACONDA NAVIGATOR**

Connect ▾

Home Environments Learning Community

**Anaconda Toolbox**  
Supercharged local notebooks.  
Click the Toolbox tile to Install.

Documentation Anaconda Blog

SOMENTE O BÁSICO

Installed Channels Update index... Search Packages

Name	T	Description	Version
bzip2	<input checked="" type="checkbox"/>	High-quality data compressor	1.0.8
ca-certificates	<input checked="" type="checkbox"/>	Certificates for use with other packages.	2023.12.
libffi	<input checked="" type="checkbox"/>	A portable foreign function interface library	3.4.4
openssl	<input checked="" type="checkbox"/>	OpenSSL is an open-source implementation of the SSL and TLS protocols	3.0.13
pip	<input checked="" type="checkbox"/>	Pypa recommended tool for installing Python packages	23.3.1
python	<input checked="" type="checkbox"/>	General purpose programming language	3.11.7
setuptools	<input checked="" type="checkbox"/>	Download, build, install, upgrade, and uninstall Python packages	68.2.2
sqlite	<input checked="" type="checkbox"/>	Implements a self-contained, zero-configuration, SQL database engine	3.41.2
tk	<input checked="" type="checkbox"/>	A dynamic programming language with GUI support. Bundles tcl and tk.	8.6.12

15 packages available

Create Clone Import Backup Remove

# IDEs PARA PYTHON

Anaconda Navigator

File Help

ANACONDA NAVIGATOR Connect ▾

Home Environments Learning Community

**NEW** Anaconda Toolbox Supercharged local notebooks. Click the Toolbox tile to Install.

Documentation Anaconda Blog

SOMENTE O BÁSICO

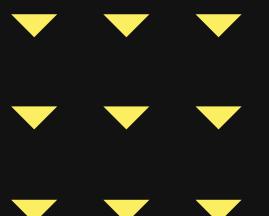
Search Environments

Installed  Channels  Update index...

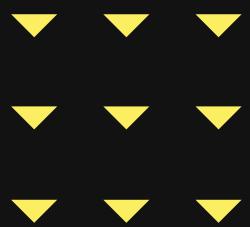
Name	T	Description	Version
bzip2	<input checked="" type="checkbox"/>	High-quality data compressor	1.0.8
ca-certificates	<input checked="" type="checkbox"/>	Certificates for use with other packages.	2023.12.
libffi	<input checked="" type="checkbox"/>	A portable foreign function interface library	3.4.4
openssl	<input checked="" type="checkbox"/>	OpenSSL is an open-source implementation of the SSL and TLS protocols	3.0.13
pip	<input checked="" type="checkbox"/>	Pypa recommended tool for installing Python packages	23.3.1
python	<input checked="" type="checkbox"/>	General purpose programming language	3.11.7
setuptools	<input checked="" type="checkbox"/>	Download, build, install, upgrade, and uninstall Python packages	68.2.2
sqlite	<input checked="" type="checkbox"/>	Implements a self-contained, zero-configuration, SQL database engine	3.41.2
tk	<input checked="" type="checkbox"/>	A dynamic programming language with GUI support. Bundles tcl and tk.	8.6.12

Create  Clone  Import  Backup  Remove

15 packages available



Alguma dúvida até aqui



Cada novo ambiente, traz consigo ferramentas próprias.  
Logo, dependendo do tipo de aplicação que for... ao mudar  
o ambiente de desenvolvimento você precisará novamente  
instalar aquela aplicação

# IDEs PARA PYTHON

on  Channels

4.0.11 Environment for interactive computing, based on the Jupyter Notebook and Architecture.	7.0.8 Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.	3.34.0 Component based data mining framework. Data visualization and data analysis for novice and expert. Interactive workflows with a large toolbox.	0.0.1 Run a Powershell terminal with your current environment from Navigator activated
<input type="button" value="Install"/>	<input type="button" value="Install"/>	<input type="button" value="Install"/>	<input type="button" value="Install"/>
shortcut_minicond: 0.0.1 A Powershell prompt	Qt Console 5.5.0 PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.	RStudio 1.1.456 A set of integrated tools designed to help you be more productive with R. Includes R essentials and notebooks.	Spyder 5.4.3 Scientific PYthon Development EnviRonment. Powerful Python IDE with advanced editing, interactive testing, debugging and introspection features
<input type="button" value="Install"/>	<input type="button" value="Install"/>	<input type="button" value="Install"/>	<input type="button" value="Install"/>

Anaconda Navigator

File Help

## ANACONDA NAVIGATOR

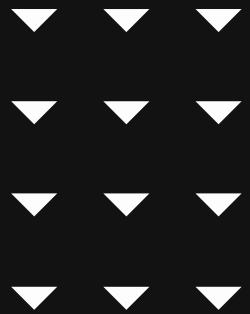
Home Environments Learning Community

All applications on  Channels

 Qt Console 3.6.3 An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.	 Spyder 6.5.4 Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.
<input type="button" value="Launch"/>	<input type="button" value="Launch"/>
	 Qt Console 5.4.2 PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.
<input type="button" value="Launch"/>	 Spyder 5.4.3 Scientific PYthon Development EnviRonment. Powerful Python IDE with advanced editing, interactive testing, debugging and introspection features
<input type="button" value="Launch"/>	<input type="button" value="Launch"/>

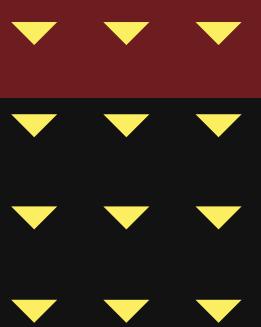


↓



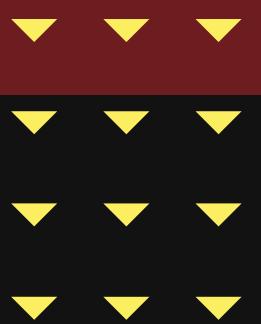
GOOGLE COLAB

# Google Colab - uma solução em nuvem.



- Não precisa configurar ambiente de desenvolvimento!
- Só logar e usar!

# Google Colab - uma solução em nuvem.



O principal objetivo do Google Colab é criar um ambiente rico e colaborativo que incentive a pesquisa e criação relacionadas à Inteligência Artificial e Machine Learning. Esse ecossistema funciona em Python, mas, com alguns ajustes, é possível usar outras linguagens de programação também.

## Google Colab - uma solução em nuvem.



O Colab oferece acesso gratuito a unidades de processamento gráfico (GPUs), o que é benéfico para tarefas que exigem computação intensiva, como treinamento de modelos de aprendizado de máquina.

## Google Colab - uma solução em nuvem.



Por ser amigável, é também utilizado por quem está começando na área de programação.

# Google Colab - uma solução em nuvem.

▼ ▼ ▼

Olá, este é o Colaboratory - Colab

colab.research.google.com Navegação anónima

Olá, este é o Colaboratory

Arquivo Editar Ver Inserir Ambiente de execução Ferramentas Ajuda

Índice Vamos começar Ciência de dados Machine learning Mais recursos Exemplos em destaque + Seção

+ Código + Texto Copiar para o Drive

Acesso Login Compartilhar Fazer login Conectar

Conectem-se após o login

Conheça o Colab

(Novidade) Teste a API Gemini

- [Generate a Gemini API key](#)
- [Talk to Gemini with the Speech-to-Text API](#)
- [Compare Gemini with ChatGPT](#)
- [More notebooks](#)

Se você já conhece bem o Colab, confira este vídeo para saber mais sobre as tabelas interativas, a visualização do histórico de código executado e o Palette de comandos.



[ ]

Google Colab - uma solução em nuvem.



Neste curso, podemos utilizar 3 ferramentas.

- Colab
- Pycharm
- VS Code

Em alguns momento eu, por questões didáticas, utilizarei o VS Code, em outras o Pycharm e no restante o Colab.

Sintam-se a vontade para **inicialmente** usarem a ferramenta que quiserem.

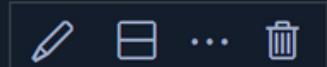
Mão no código!



# Manipulação de variáveis e constantes



## Manipulação de variáveis e constantes



### Variáveis e constantes

- Por meio de variáveis que um algoritmo “guarda” os dados do problema
- Todo dado que tem a possibilidade de ser alterado no decorrer do tempo deverá ser tratado como uma variável
- Quando um dado não tem nenhuma possibilidade de variar no decorrer do tempo, deverá ser tratado como constante

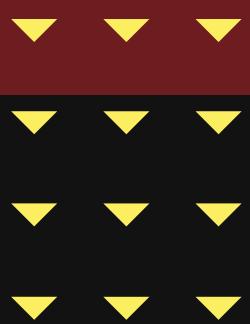
Exemplo: calcular a área de um triângulo. Sabemos que a fórmula para o cálculo da área de um triângulo é  $\text{BASE} * \text{ALTURA} / 2$ . **Base** e **altura** são dados que irão variar no decorrer do “tempo de execução”. O número 2 da fórmula é um dado constante, pois sempre terá o mesmo valor



## Tipo de variáveis

- Inteiros: valores positivos ou negativos, que não possuem uma parte fracionária. Exemplos: 1, 30, 40, 12, -50
- Float (real): valores positivos ou negativos, que podem possuir uma parte fracionária (também podem ser inteiros). Exemplos: 1.4, 6.7, 10.3, 100, -47
- Caracteres (char ou string): qualquer elemento presente no teclado. Exemplos: "Maria", "João", 'M', 'F'
- Lógico (booleano): verdadeiro ou falso. Exemplos: true, false, 1, 0

# Mão no código!



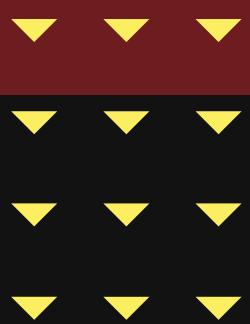
## Variáveis inteiras

```
1 numero = -3
2 numero_jogos = 14
3 numero_convidados = 15
[1] ✓ 0.0s
```

```
1 numero
[2] ✓ 0.0s
... -3
```

```
1 print(numero)
[3] ✓ 0.0s
... -3
```

```
▷ ▾ 1 print(numero, numero_convidados)
[4] ✓ 0.0s
... -3 15
```



# Variáveis float (ponto flutuante)

```
1 pi = 3.14
2 numero_euler = 2.71
3 escala_terremoto = -4.55
[5] ✓ 0.0s
```

```
1 print(pi)
2 print(numero_euler)
3 print(escala_terremoto)
[6] ✓ 0.0s
```

```
... 3.14
    2.71
    -4.55
```

# Mão no código!



## Strings e chars

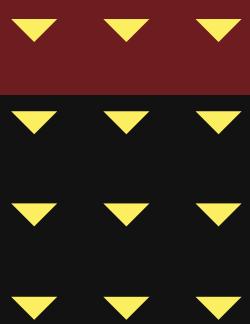
```
1 letra = 'a'  
2 palavra1 = 'linguagem de programação'  
3 palavra2 = 'Python'  
[7] ✓ 0.0s
```

```
1 print(letra, palavra1)  
[8] ✓ 0.0s  
... a linguagem de programação
```

```
1 print('Estou aprendendo uma', palavra1)  
[9] ✓ 0.0s  
... Estou aprendendo uma linguagem de programação
```

```
1 print('Esta', palavra1, 'se chama', palavra2)  
[10] ✓ 0.0s  
... Esta linguagem de programação se chama Python
```

# Mão no código!



```
1 idade = int(input('Digite sua idade:'))
2 print('Sua idade é ', idade)
```

[11] ✓ 3m 20.5s

... Sua idade é 24

```
1 pH = float(input('Qual o pH do solo durante a última medição?'))
2 print('O pH medido foi ', pH)
```

[12] ✓ 5.3s

... O pH medido foi 4.0

```
1 nome = str(input('Qual o seu nome?'))
2 print('Seu nome é', nome)
```

[13] ✓ 3.7s

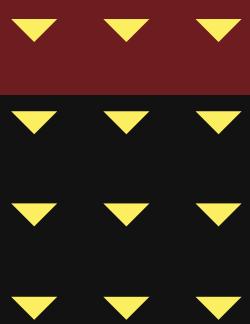
... Seu nome é Caio

```
1 idade
```

[14] ✓ 0.0s

... 24

Mão no código!



# Manipulação de strings

# Mão no código!



## Manipulação de strings

```
1 a = 'casaco'  
2 print(a)
```

.. casaco

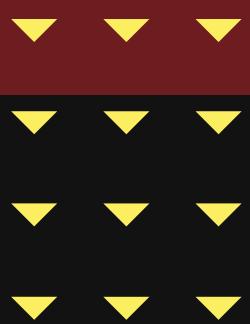
```
1 maiuscula = a.upper()  
2 print(maiuscula)
```

.. CASACO

```
1 minuscula = maiuscula.lower()  
2 print(minuscula)
```

.. casaco

# Mão no código!



```
1 capital = a.capitalize()  
2 print(capital)
```

... Casaco

```
1 metade_palavra = a[0:4]  
2 print(metade_palavra)
```

... casa

```
1 ultimas_letras = a[3:]  
2 print(ultimas_letras)
```

... aco

▷ ▾

```
1 b = a.replace('aco', 'inha')  
2 print(a)  
3 print(b)
```

... casaco  
casinha

# Mão no código!



```
1 c = a.replace('o', 'a')
2 print(c)
```

... casaca

```
1 c.find('s')
```

... 2

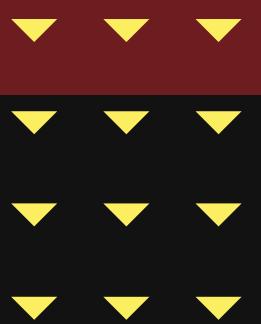
```
1 c.find('a')
```

... 1

```
1 c.find('b')
```

... -1

# Mão no código!



```
1 e = ' casaco '
2 print(len(e))
```

... 8

```
▷ 
1 f = e.strip() #strip retira espaços em branco antes e depois da palavra
2 print(f)
```

... casaco

```
1 print(len(f))
```

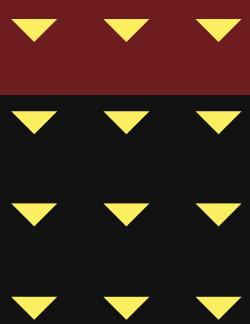
... 6

```
1 n1 = 14
2 n2 = 16
```

```
▷ 
1 print(f'Dividindo {n1} por {n2} o resultado é {n1/n2}')
```

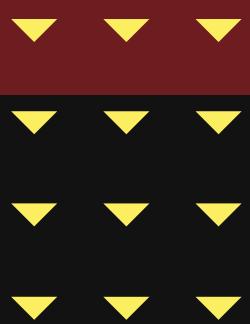
... Dividindo 14 por 16 o resultado é 0.875

Mão no código!



# Operações matemáticas

# Mão no código!



## Operações matemáticas

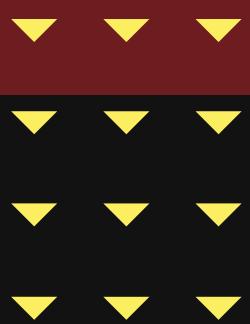
### Operações matemáticas

```
> 
  1 a = 5
  2 b = 3
  3 print(a, b)
[1] ✓ 0.0s
...
... 5 3
```

```
1 a + b
[2] ✓ 0.0s
...
... 8
```

```
1 print('A soma é', a + b)
[3] ✓ 0.0s
...
... A soma é 8
```

# Mão no código!



```
1 print('A subtração é', a - b)
[4] ✓ 0.0s
...
... A subtração é 2
```

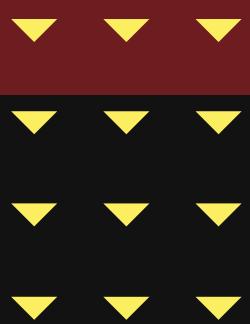
```
1 print('A divisão é', a / b)
[5] ✓ 0.0s
...
... A divisão é 1.6666666666666667
```

```
1 print('A multiplicação é', a * b)
[6] ✓ 0.0s
...
... A multiplicação é 15
```

```
1 print('O resto da divisão é', 10 % 3)
[7] ✓ 0.0s
...
... O resto da divisão é 1
```

```
▷ 
1 5 * 5 * 5 * 5
[8] ✓ 0.0s
...
... 625
```

# Mão no código!



```
1 print('5 elevado a 4 é', 5**4)
```

[9] ✓ 0.0s

... 5 elevado a 4 é 625

```
1 import math
2 math.sqrt(81)
```

[10] ✓ 0.0s

... 9.0

Importar uma  
biblioteca específica

# Arredondamento

```
1 casos_doenca_vass = 1430
2 numero_habitantes_vass = 347135
3 casos_por_habitante = casos_doenca_vass / numero_habitantes_vass
4 print(casos_por_habitante)
```

[24] ✓ 0.0s

... 0.004119434802022268

```
▷ 
1 round(casos_por_habitante, 6)
```

[25] ✓ 0.0s

... 0.004119

```
1 print('O número de casos por habitante é de', round(casos_por_habitante, 4))
```

[26] ✓ 0.0s

... O número de casos por habitante é de 0.0041

Mão no código!



# Exercícios

Mão no código!

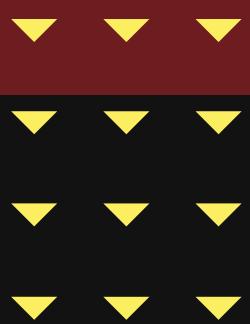


**1. Ler dois números inteiros, executar e mostrar o resultado das seguintes operações: adição, subtração, multiplicação, divisão e potenciação do primeiro elevado ao segundo número.**

Mão no código!

**2. Efetuar o cálculo da quantidade de litros de combustível gasto em uma viagem, utilizando um automóvel que faz 12 Km por litro. Para obter o cálculo, o usuário deve fornecer o tempo gasto na viagem e a velocidade média durante ela. Desta forma, será possível obter a distância percorrida com a fórmula DISTANCIA = TEMPO \* VELOCIDADE. Tendo o valor da distância, basta calcular a quantidade de litros de combustível utilizada na viagem, com a fórmula: LITROS\_USADOS = DISTANCIA / 12. O programa deve apresentar os valores da velocidade média, tempo gasto na viagem, a distância percorrida e a quantidade de litros utilizada na viagem**

Mão no código!



# Resolução

# Mão no código!



## Exercício 1

Ler dois números inteiros, executar e mostrar o resultado das seguintes operações: adição, subtração, multiplicação, divisão e potenciação do primeiro pelo segundo número.

```
[10] 1 numero1 = int(input("Digite o primeiro número: "))
      2 numero2 = int(input("Digite o segundo número: "))
```

✓ 3.7s

```
[11] 1 print(numero1, numero2)
```

✓ 0.0s

... 5 10

```
[12] 1 adicao = numero1 + numero2
      2 subtracao = numero1 - numero2
      3 multiplicacao = numero1 * numero2
      4 divisao = numero1 / numero2
      5 potenciacao = numero1**numero2
```

Pyt

```
[13] 1 print("Adição: ", adicao)
      2 print("Subtração: ", subtracao)
      3 print("Multiplicação: ", multiplicacao)
      4 print("Divisão: ", round(divisao, 2))
      5 print("Potenciação: ", potenciacao)
```

Pyt

```
... Adição: 15
Subtração: -5
Multiplicação: 50
Divisão: 0.5
Potenciação: 9765625
```

## Exercício 2

Efetuar o cálculo da quantidade de litros de combustível gasto em uma viagem, utilizando um automóvel que faz 12 Km por litro. Para obter o cálculo, o usuário deve fornecer o tempo gasto na viagem e a velocidade média durante ela. Desta forma, será possível obter a distância percorrida com a fórmula  $DISTANCIA = TEMPO * VELOCIDADE$ . Tendo o valor da distância, basta calcular a quantidade de litros de combustível utilizada na viagem, com a fórmula:  $LITROS\_USADOS = DISTANCIA / 12$ . O programa deve apresentar os valores da velocidade média, tempo gasto na viagem, a distância percorrida e a quantidade de litros utilizada na viagem

```
[14] 1 tempo = float(input("Digite o tempo gasto na viagem: "))
2 velocidade = float(input("Digite a velocidade média: "))
```

✓ 2.6s

Python

```
[15] 1 print(tempo, velocidade)
```

✓ 0.0s

Python

... 2.0 80.0

```
[16] 1 distancia = tempo * velocidade
2 litros_usados = distancia / 12
```

✓ 0.0s

Python

```
[17] 1 print("Velocidade média: ", velocidade)
2 print("Tempo gasto na viagem: ", tempo)
3 print("Distância percorrida:", distancia)
4 print("Quantidade de litros:", round(litros_usados, 1))
```

✓ 0.0s

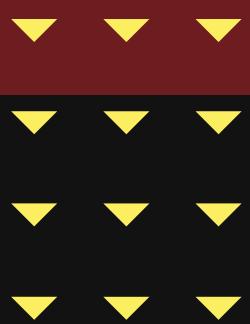
Python

.. Velocidade média: 80.0
Tempo gasto na viagem: 2.0
Distância percorrida: 160.0
Quantidade de litros: 13.3

Mão no código!



# Operadores lógicos e relacionais



## ▼ Operadores lógicos e relacionais

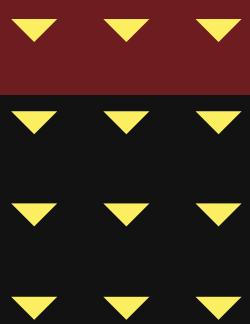
### Operadores lógicos

```
▷ ▾  
 1 a = True  
 2 b = False  
[1] ✓ 0.0s
```

```
1 print(a, b)  
[2] ✓ 0.0s  
... True False
```

```
1 a and b  
[3] ✓ 0.0s  
... False
```

# Mão no código!



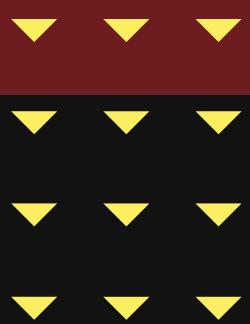
```
1 a & b  
[4] ✓ 0.0s  
... False
```

```
1 c = a and b  
2 print("'A' e 'B' são iguais é", c)  
[5] ✓ 0.0s  
... 'A' e 'B' são iguais é False
```

```
1 a or b  
[6] ✓ 0.0s  
... True
```

```
▷ ▾ 1 a | b # essa barra vertical tem o mesmo significado do OR (ou)|  
[7] ✓ 0.0s  
... True
```

# Mão no código!



```
1 d = a or b
2 print("'A' ou 'B' é igual a", d)
```

[8] ✓ 0.0s

... 'A' ou 'B' é igual a True

```
1 not a
```

[9] ✓ 0.0s

... False

```
1 not b
```

[10] ✓ 0.0s

... True

Mão no código!



## Operadores relacionais

```
1 5 > 3
[11] ✓ 0.0s
```

... True

```
1 5 < 3
[12] ✓ 0.0s
```

... False

```
▷ 1 5 ≥ 5
[13] ✓ 0.0s
```

... True

```
1 5 ≤ 3
[14] ✓ 0.0s
```

... False

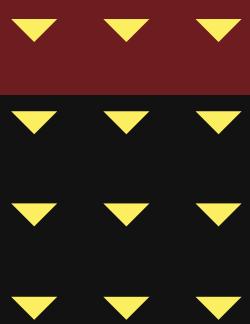
```
1 5 = 3
[15] ✓ 0.0s
```

... False

```
1 5 ≠ 3
[16] ✓ 0.0s
```

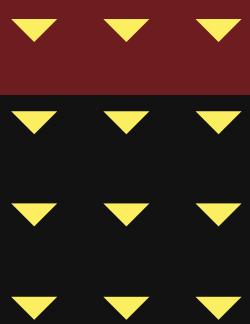
... True

Mão no código!



# Operadores condicionais

# Mão no código!



## ▼ Operador condicional

```
1 5 > 3  
[1] ✓ 0.0s
```

... True

```
▷ ▼  
1 if 5 > 3:  
2 |   print("5 é maior que 3")  
3 # print('teste')  
[2] ✓ 0.0s
```

... 5 é maior que 3

```
1 if 5 > 4:  
2 |   print("5 é maior")  
3 else:  
4 |   print("5 não é maior")  
[3] ✓ 0.0s
```

... 5 é maior

# Mão no código!

```
1 n = 9
2 if n == 4:
3     print("n é igual a 4")
4 else:
5     if n == 3:
6         print("n é igual a 3")
7     else:
8         print("n não é igual a 4 nem 3")
[4] ✓ 0.0s
```

.. n não é igual a 4 nem 3

```
1 x = 1
2 y = 5
3 if (x > 1) or (y % 2 == 0):
4     print("x é maior que 1 e y é par")
5 else:
6     print("Uma ou nenhuma das condições foram satisfeitas")
[5] ✓ 0.0s
```

.. Uma ou nenhuma das condições foram satisfeitas

Mão no código!



# Exercícios

Mão no código!



## 1. Leia a idade do usuário e classifique-o em:

- Criança - 0 a 12 anos
  - Adolescente - 13 a 17 anos
  - Adulto - acima de 18 anos
- Se o usuário digitar um número negativo, mostrar a mensagem que a idade é inválida

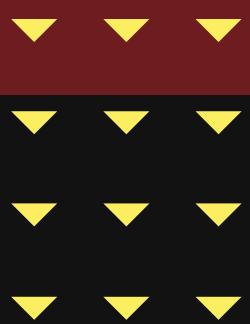


**2. Calcular a média de um aluno que cursou a disciplina de Programação I, a partir da leitura das notas M1, M2 e M3; passando por um cálculo da média aritmética. Após a média calculada, devemos anunciar se o aluno foi aprovado, reprovado ou pegou exame**

Mão no código!

- Se a média estiver entre 0.0 e 4.0, o aluno está reprovado
- Se a média estiver entre 4.1 e 6.0, o aluno pegou exame
- Se a média for maior do que 6.0, o aluno está aprovado
- Se o aluno pegou exame, deve ser lida a nota do exame. Se a nota do exame for maior do que 6.0, está aprovado, senão; está reprovado

Mão no código!



# Resolução

# Mão no código!

1. Leia a idade do usuário e classifique-o em:

- Criança – 0 a 12 anos
- Adolescente – 13 a 17 anos
- Adulto – acima de 18 anos -Se o usuário digitar um número negativo, mostrar a mensagem que a idade é inválida

```
> ▾
  1 idade = int(input('Digite a idade: '))
  2 print(idade)
[3] ✓ 1.7s
...
29
```

```
1 if idade ≥ 0 and idade ≤ 12:
2     print ('Criança')
3 elif idade > 12 and idade ≤ 18:
4     print ('Adolescente')
5 elif idade > 18:
6     print ('Adulto')
7 else:
8     print ( ' Idade inválida' )
[4] ✓ 0.0s
...
Adulto
```

# Mão no código!

2. Calcular a média de um aluno que cursou a disciplina de Programação I, a partir da leitura das notas N1, N2 e N3; passando por um cálculo da média aritmética. Após a média calculada, devemos anunciar se o aluno foi aprovado, reprovado ou pegou exame

- Se a média estiver entre 0.0 e 4.0, o aluno está reprovado
- Se a média estiver entre 4.1 e 6.0, o aluno pegou exame
- Se a média for maior do que 6.0, o aluno está aprovado
- Se o aluno pegou exame, deve ser lida a nota do exame. Se a nota do exame for maior do que 6.0, está aprovado, senão; está reprovado

```
1 n1 = float(input('Digite a nota: '))
2 n2 = float(input("Digite a nota: "))
3 n3 = float(input("Digite a nota: "))
4 print(f"A nota 01 foi {n1}, A nota 02 foi {n2} e a nota 03 foi {n3}.")
[8] ✓ 12.3s
...
... A nota 01 foi 10.0, A nota 02 foi 2.7 e a nota 03 foi 8.9.
```

```
▷ ▾
1 media= (n1+n2+n3)/3
2 print(media)
[9] ✓ 0.0s
...
... 7.2
```

# Mão no código!

```
▶ ▾
1 if media <= 4.0:
2     print ('Aluno reprovado')
3 elif media >= 4.1 and media <= 6.0:
4     exame = float(input('Digite a nota do exame: '))
5     if exame >= 6.0:
6         print ('Aprovado no exame')
7     else :
8         print ('Reprovado no exame')
9 else:
10    print('Aluno aprovado')
11
[10] ✓ 0.0s
... Aluno aprovado
```

*That's all Folks!*

ATÉ A PRÓXIMA!