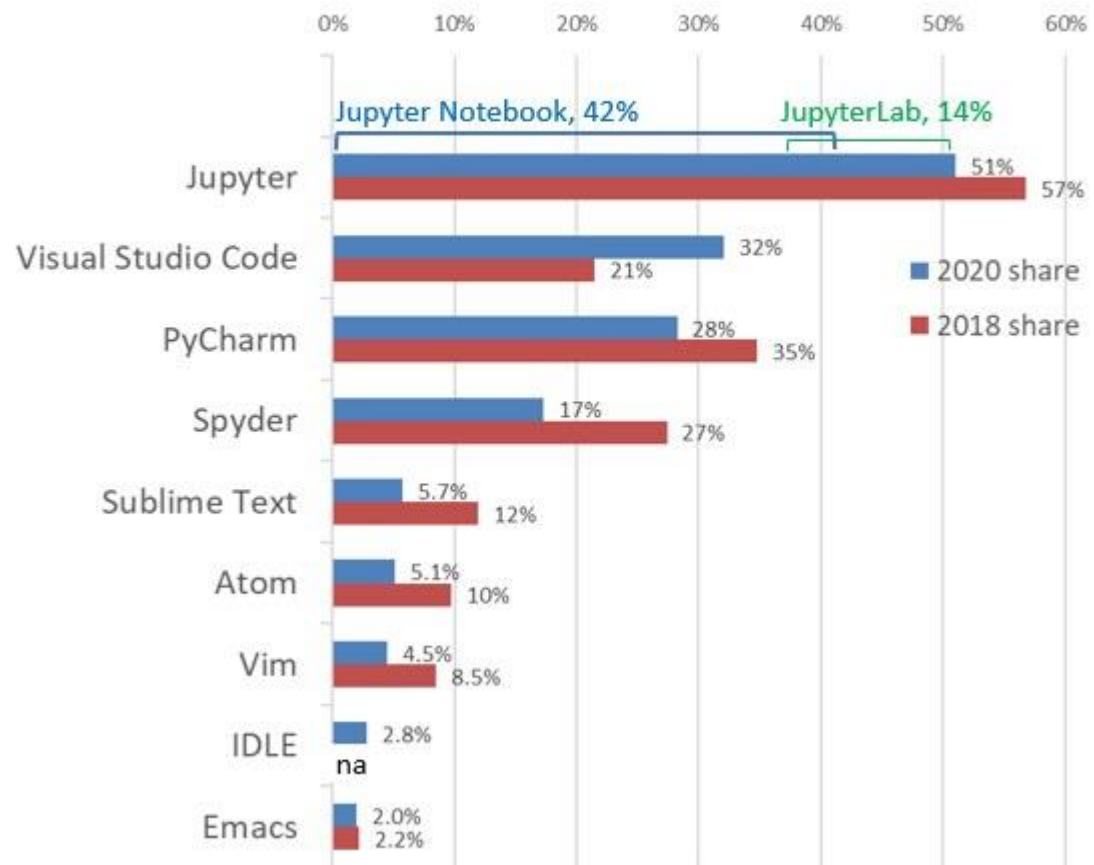
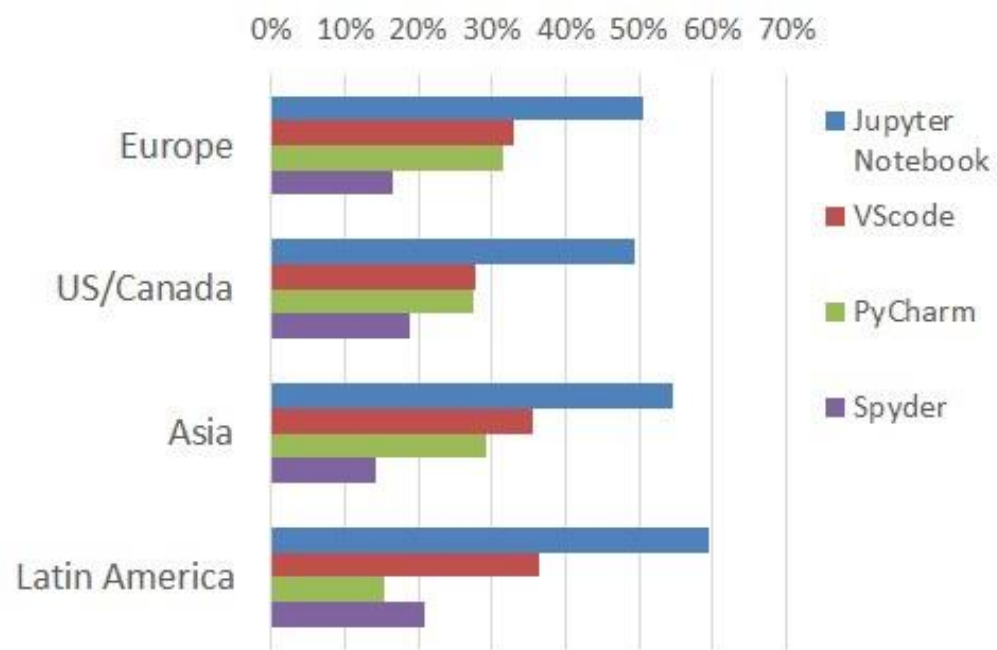




### Most Popular Python IDEs, 2020 vs 2018



### Top Python IDEs, by region



# Frameworks



Api	Dados	Web Scrapping e gráficos	Estatística e Inteligência artificial	Aws
<ul style="list-style-type: none"> <li>• Request (get,post, put e etc)</li> <li>• Django (crud)</li> <li>• Flask (crud)</li> <li>• PyTest</li> </ul>	<ul style="list-style-type: none"> <li>• Numpy (Array)</li> <li>• Pandas (Dataframe)</li> <li>• PySpark (Dataframe com sql)</li> </ul>	<ul style="list-style-type: none"> <li>• BeautifulSoup</li> <li>• Scrapy</li> <li>• PyTesseract</li> <li>• Selenium</li> </ul> <p>Graficos</p> <ul style="list-style-type: none"> <li>• Matplotlib</li> <li>• Seaborn</li> <li>• Ploty</li> </ul>	<ul style="list-style-type: none"> <li>• Sklearn (Vp-Vn – Fp – Vp)</li> <li>• NLTK</li> <li>• Scipy</li> <li>• PyTesseract</li> </ul>	<ul style="list-style-type: none"> <li>• Boto, boto3</li> <li>• dynamodb</li> <li>• Aws glue</li> <li>• Aws rds</li> <li>• <b>aws-cdk.aws-elasticsearch</b></li> </ul>

## Exemplos

**pip install requests**

**Pip uninstall flask**

**Pip install pandas**

**Pip install matplotlib**

**Pip install pymongo**

**Pip install boto3**

**Pip install aws-cdk.aws-elasticsearch**

**Pip install aws glue**

# Bibliotecas mais famosas

```

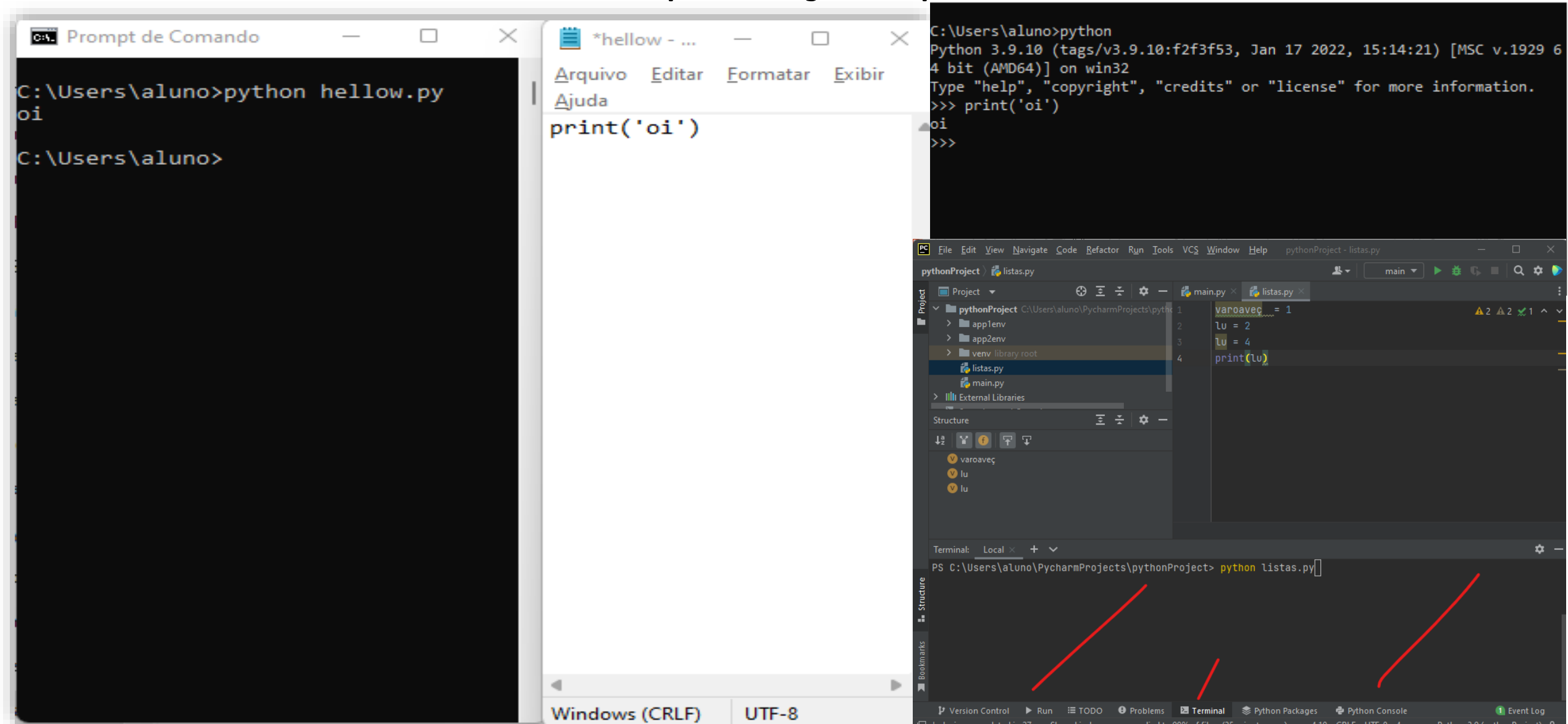
C:\Windows\System32\cmd.exe
Microsoft Windows [versão 10.0.22000.434]
(c) Microsoft Corporation. Todos os direitos reservados.

C:\Users\aluno\OneDrive\Imagens\env_arquivos\json>pip install pytesseract

```

- Aplicações em Python normalmente usam pacotes e módulos que não vêm como parte da instalação padrão. Aplicações às vezes necessitam uma versão específica de uma biblioteca, porque ela requer que algum problema em particular tenha sido consertado ou foi escrita utilizando-se de uma versão obsoleta da interface da biblioteca.
- Isso significa que talvez não seja possível que uma instalação Python preencha os requisitos de qualquer aplicação. Se uma aplicação A necessita a versão 1.0 de um módulo particular mas a aplicação B necessita a versão 2.0, os requisitos entrarão em conflito e instalar qualquer uma das duas versões 1.0 ou 2.0 fará com que uma das aplicações não consiga executar.
- A solução para este problema é criar um [ambiente virtual](#), uma árvore de diretórios que contém uma instalação Python para uma versão particular do Python, além de uma série de pacotes adicionais.

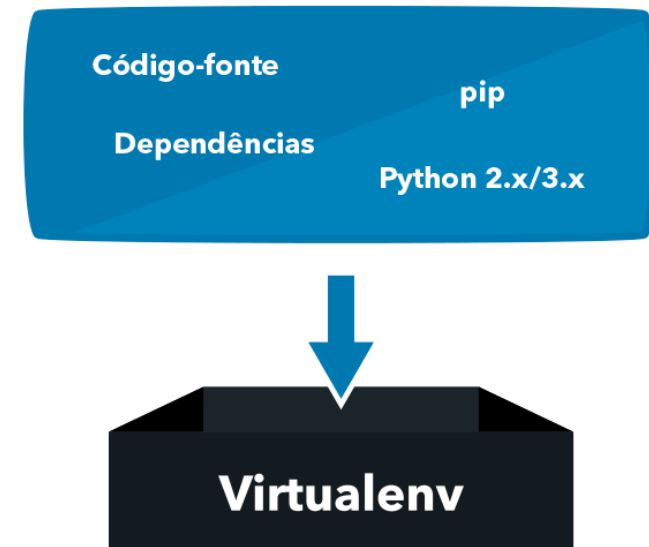
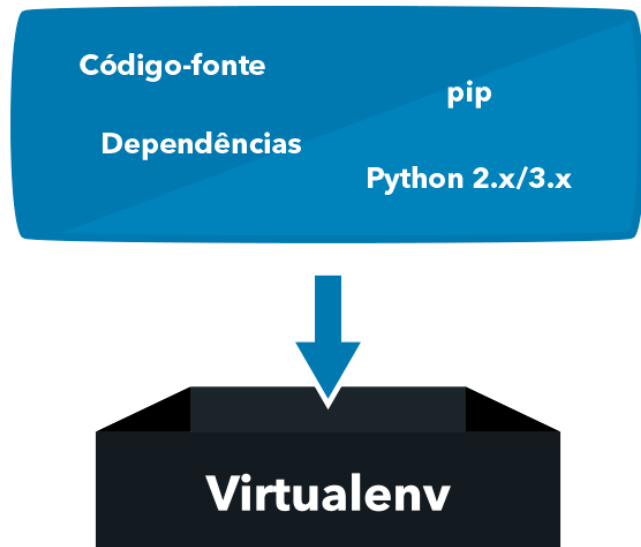
# Como rodar uma aplicação python



# Teoria

```
python -m venv app1env  
pip install pymysql  
pip install requests==2.6.0
```

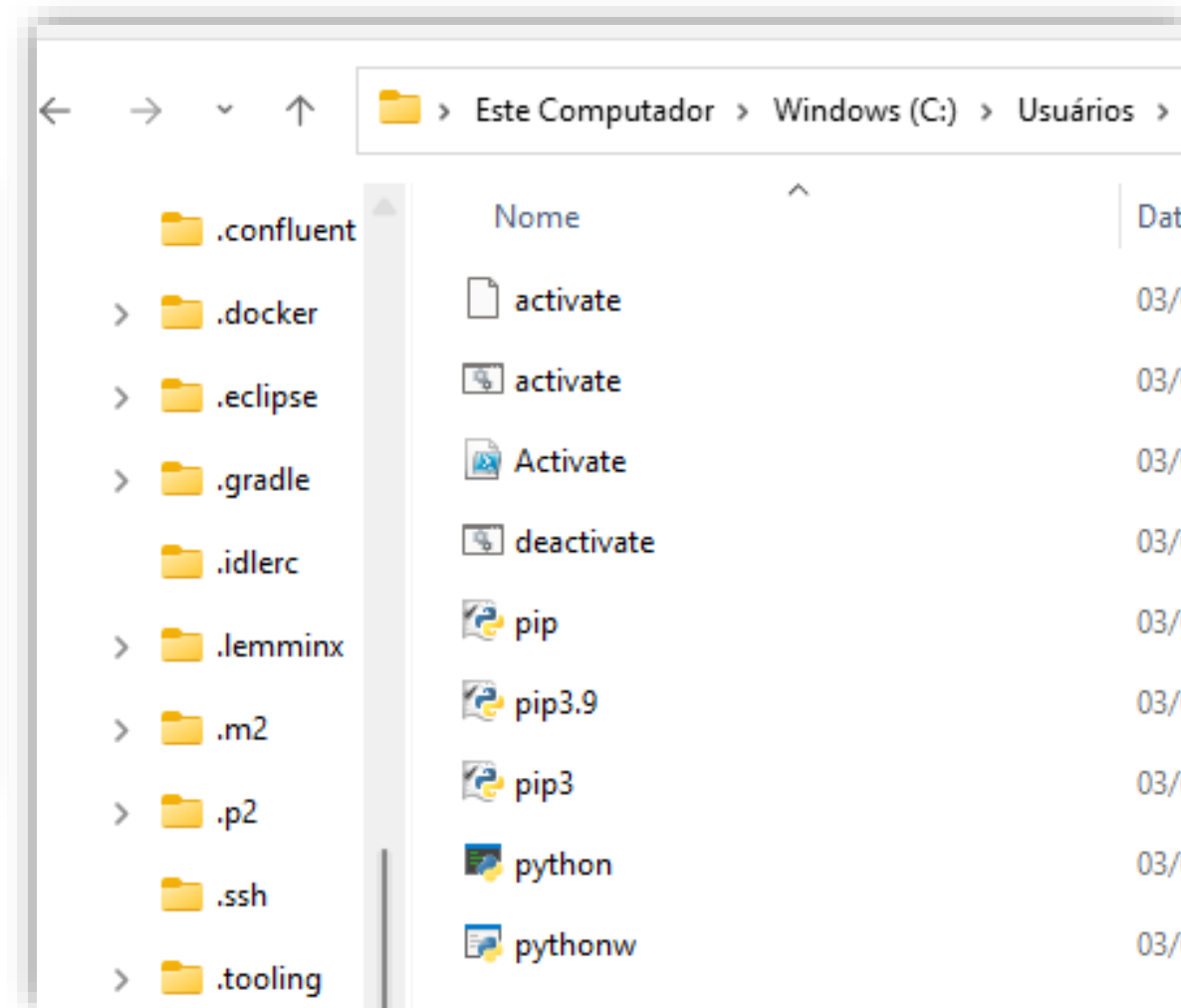
```
python3 -m venv app2env  
pip install pymongo  
pip install requests==2.7.1  
pip install --upgrade requests
```



# Exemplo hands on

- `python3 -m venv app1env`

```
C:\Users\aluno>python3 -m venv app1env
C:\Users\aluno>app1env\Scripts\activate.bat
(app1env) C:\Users\aluno>pip install pandas
Collecting pandas
  Downloading pandas-1.4.0-cp39-cp39-win_amd64.whl (10.5 MB)
    |████████████████████████████████████████| 10.5 MB 2.2 MB/s
```





`pip show` irá mostrar informações sobre um pacote em particular:

```
(tutorial-env) $ pip show requests
---
Metadata-Version: 2.0
Name: requests
Version: 2.7.0
Summary: Python HTTP for Humans.
Home-page: http://python-requests.org
Author: Kenneth Reitz
Author-email: me@kennethreitz.com
License: Apache 2.0
Location: /Users/akuchling/envs/tutorial-env/lib/python3.4/site-packages
Requires:
```

`pip list` irá apresentar uma lista de todos os pacotes instalados no ambiente virtual.

```
(tutorial-env) $ pip list
novas (3.1.1.3)
numpy (1.9.2)
pip (7.0.3)
requests (2.7.0)
setuptools (16.0)
```

```
(app1env) C:\Users\aluno>pip install -r requirements.txt
Collecting flask
  Using cached Flask-2.0.2-py3-none-any.whl (95 kB)
Collecting sqlalchemy
  Downloading SQLAlchemy-1.4.31-cp39-cp39-win_amd64.whl (1.6 MB)
    | 1.6 MB 1.6 MB/s
Requirement already satisfied: pandas in c:\users\aluno\app1env\lib\site-packages (from -r requirements.txt (line 3)) (1.4.0)
Collecting numpy==1.9.2
  Downloading numpy-1.9.2.zip (4.5 MB)
    | 4.5 MB 819 kB/s
Collecting requests==2.7.0
  Downloading requests-2.7.0-py2.py3-none-any.whl (470 kB)
    | 470 kB 1.7 MB/s
Collecting Werkzeug>=2.0
  Using cached Werkzeug-2.0.2-py3-none-any.whl (288 kB)
Collecting click>=7.1.2
  Using cached click-8.0.3-py3-none-any.whl (97 kB)
Collecting Jinja2>=3.0
  Using cached Jinja2-3.0.3-py3-none-any.whl (133 kB)
Collecting itsdangerous>=2.0
  Using cached itsdangerous-2.0.1-py3-none-any.whl (18 kB)
Collecting greenlet!=0.4.17
```

```
flask
sqlalchemy
pandas
numpy==1.9.2
requests==2.7.0
```

# Requirements.txt

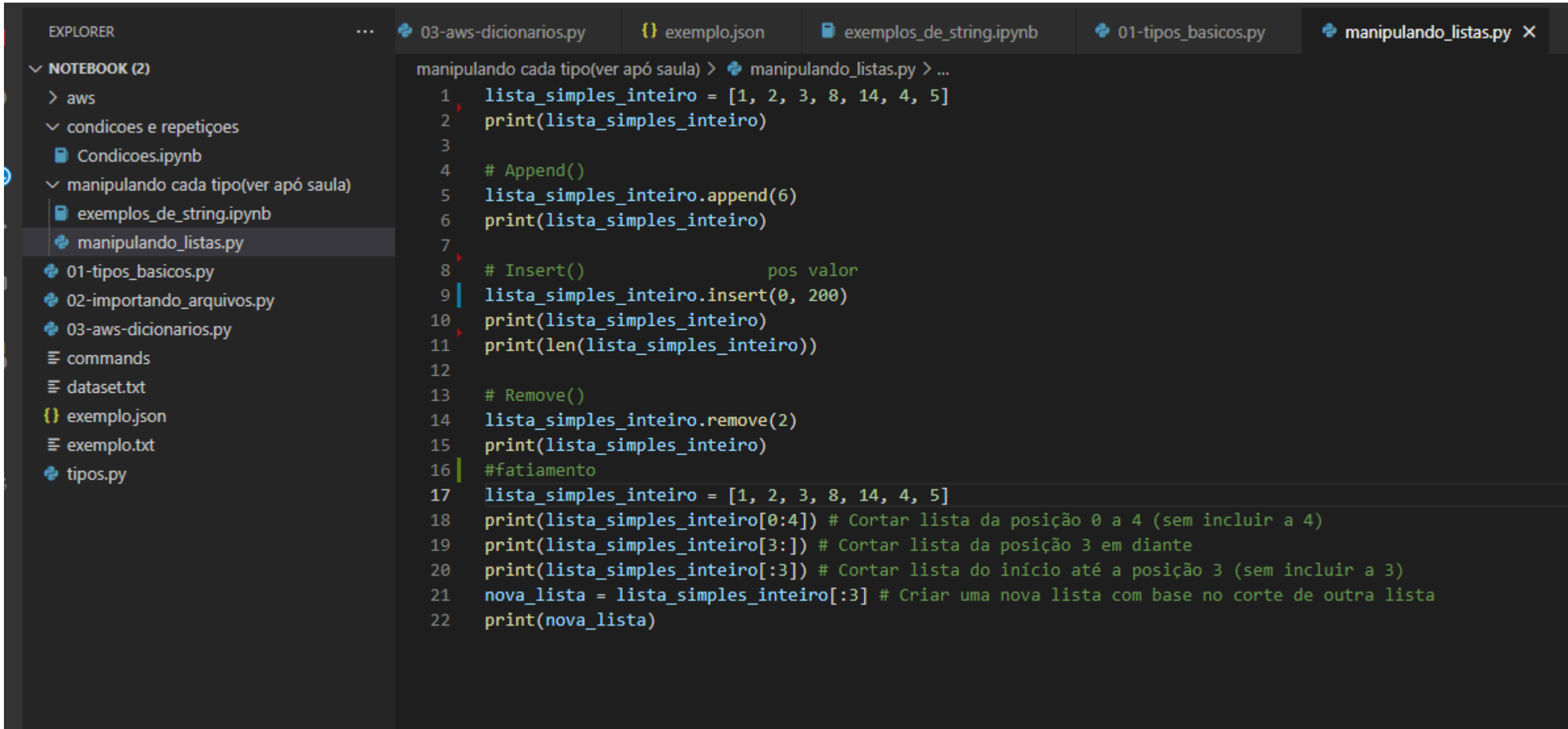
- flask
- sqlalchemy
- pandas
- numpy==1.9.2
- requests==2.7.0

```
C:\Windows\System32\cmd.exe
(app1env) C:\Users\aluno\app1env\Scripts>pip freeze
colorama==0.4.4
MarkupSafe==2.0.1
numpy==1.22.1
pandas==1.4.0
python-dateutil==2.8.2
pytz==2021.3
six==1.16.0
Werkzeug==2.0.2

(app1env) C:\Users\aluno\app1env\Scripts>
```

# Pratica de tipos

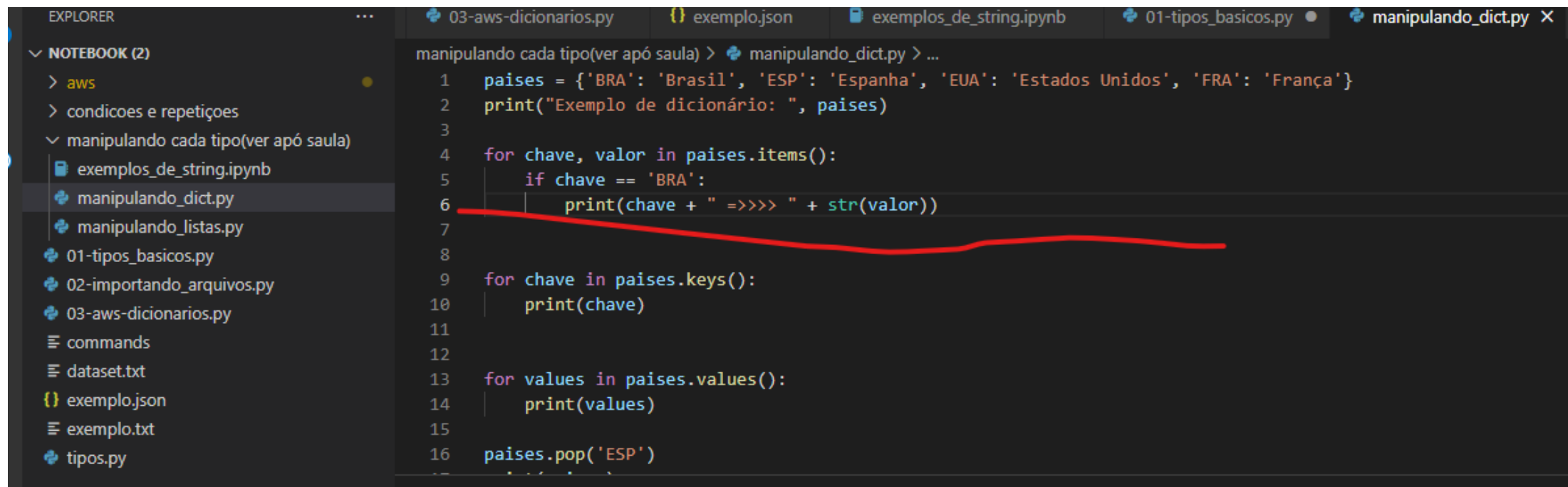
# Pratica de manipulação de listas



The image shows a code editor interface with a sidebar on the left and a main workspace on the right. The sidebar, titled 'EXPLORER', shows a file tree with a 'NOTEBOOK (2)' section containing files like 'aws', 'condicoes e repeticoes', 'Condicoes.ipynb', 'exemplos\_de\_string.ipynb', and 'manipulando\_listas.py'. The main workspace displays the code for 'manipulando\_listas.py'.

```
manipulando cada tipo(ver após saula) > manipulando_listas.py > ...
1  lista_simples_inteiro = [1, 2, 3, 8, 14, 4, 5]
2  print(lista_simples_inteiro)
3
4  # Append()
5  lista_simples_inteiro.append(6)
6  print(lista_simples_inteiro)
7
8  # Insert()                pos valor
9  lista_simples_inteiro.insert(0, 200)
10 print(lista_simples_inteiro)
11 print(len(lista_simples_inteiro))
12
13 # Remove()
14 lista_simples_inteiro.remove(2)
15 print(lista_simples_inteiro)
16 #fatiamento
17 lista_simples_inteiro = [1, 2, 3, 8, 14, 4, 5]
18 print(lista_simples_inteiro[0:4]) # Cortar lista da posição 0 a 4 (sem incluir a 4)
19 print(lista_simples_inteiro[3:]) # Cortar lista da posição 3 em diante
20 print(lista_simples_inteiro[:3]) # Cortar lista do início até a posição 3 (sem incluir a 3)
21 nova_lista = lista_simples_inteiro[:3] # Criar uma nova lista com base no corte de outra lista
22 print(nova_lista)
```

# Manipulando dicionários (fazer linha 9:16)



```
manipulando cada tipo(ver após saula) > manipulando_dict.py > ...
1  paises = {'BRA': 'Brasil', 'ESP': 'Espanha', 'EUA': 'Estados Unidos', 'FRA': 'França'}
2  print("Exemplo de dicionário: ", paises)
3
4  for chave, valor in paises.items():
5      if chave == 'BRA':
6          print(chave + " =>>>> " + str(valor))
7
8
9  for chave in paises.keys():
10     print(chave)
11
12
13  for values in paises.values():
14     print(values)
15
16  paises.pop('ESP')
```

- Exemplo aws
- Exemplo importação