Programação Orientada a Objetos para Dados

POO para extração de dados

Prof. Ms. Leonardo Rocha

- Unidade de Ensino: 3
- Competência da Unidade: Compreender e estruturar gráficos
- Resumo: Conhecendo funções para tratamento, leitura e escrita de dados.
- Palavras-chave: biblioteca, gráficos, funções
- Título da Teleaula: POO para extração de dados
- Teleaula nº: 3

Contextualização

Funções para leitura de dados

Funções para escrita de dados

tratamento de dados

Gráficos

Conceitos

Dados

Análise de dados

O trabalho com análise de dados exige, basicamente, o conhecimento do dado com relação à sua origem e a seus tipos.

Eles podem ser classificados em 3 princiapais categorias:

--> Estruturado

--> Não estruturado

--> Semi-estruturado

Categorias

Estruturados - possuem uma estrutura definida, estão apresentados em um formato tabular e podem ser armazenados e manipulados em um banco de dados relacional.

Não Estruturados - não estão disponíveis em um formato tabular, alguns exemplos de dados não estruturados são: arquivos de áudio mp3, imagens, textos, dentre outros

Semi-estruturados - não possuem um formato tabular, no entanto são estruturados por tags utilizadas para uma hierarquia dos dados. JSON e XML por exemplo.

Leitura de arquivos

Arquivo é a principal fonte para análise de dados. O pandas oferece uma gama de funções prontas para essa tarefa:

Format Type	Data Description	Reader	Writer
text	CSV	read_csv	to_csv
text	Fixed-Width Text File	read_fwf	
text	JSON	read_json	to_json
text	HTML	read_html	to_html
text	LaTeX		Styler.to_latex
text	XML	read_xml	to_xml

Fonte:https://pandas.pydata.org/pandas-docs/stable/user_guide/io.html

Vamos à prática

```
id_cliente, idade, status_civil
1, 20, solteiro
2, 35, casado
3, 50, solteiro
```

Criação de um arquivo .csv

In [9]:
```
1 import pandas as pd
2 df = pd.read_csv('exemplo_1.csv')
3 df
```

Out[9]:

	id_cliente	idade	status_civil
0	1	20	solteiro
1	2	35	casado
2	3	50	solteiro

Vamos à prática

valores separados por “;”

```
id_cliente; idade; status_civil
1; 20; solteiro
2; 35; casado
3; 50; solteiro
```

In [16]:

```
df = pd.read_csv('exemplo_2.csv', sep=';')
df
```

Out[16]:

	id_cliente	idade	status_civil
0	1	20	solteiro
1	2	35	casado
2	3	50	solteiro

Arquivos em planilha

	A	B	C
1	id_cliente	idade	status_civil
2	1	20	solteiro
3	2	35	casado
4	3	50	solteiro
5			

In [18]:
```
1 df = pd.read_excel('exemplo_3.xlsx')
2 df
```

Out[18]:

	id_cliente	idade	status_civil
0	1	20	solteiro
1	2	35	casado
2	3	50	solteiro

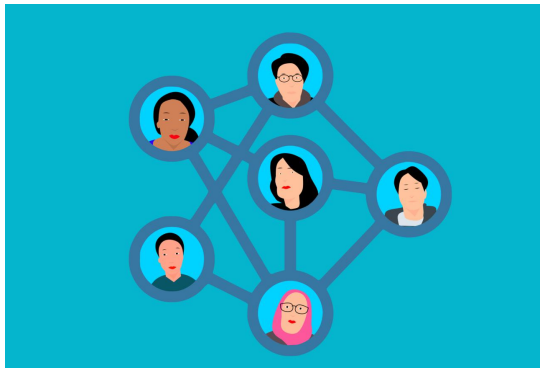
Conceitos

Escrevendo dados com Pandas

Escrita de dados

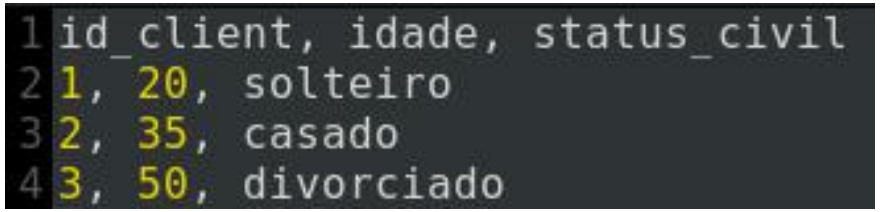
Dados em pandas, como já visto, é algo muito simples de se manipular. Nós vimos que a função para ler um arquivo csv no pandas é a **read_csv**. Da mesma forma, para ler um arquivo xlsx, é a **read_excel**.

Para a escrita de arquivo a função é outra. Para Arquivo csv, existe a **to_csv** e para planilha existe a **to_excel**



Na prática

Vamos criar um arquivo simples, pode ser com bloco de notas do seu computador. Vamos povoa-lo com os dados escritos como demonstra a figura abaixo. CUIDADO: os dados precisam ser escritos EXATAMENTE igual à figura. Depois, salve o aquivo com o nome “tabela.csv”.



Na prática

```
>>> import pandas as pd
p
>>> df = pd.read_csv('/tmp/tabela.csv')
>>> df
   id_client   idade  status_civil
0          1      20      solteiro
1          2      35        casado
2          3      50    divorciado
```

Escrevendo o dado em XLSX

Para escrever os dados, vamos utilizar a função **to_excel** disponível na biblioteca. O comando deverá ficar assim:

df.to_excel(‘caminho_do_seu_arquivo’)

```
>>>df.to_excel('tabela.xlsx')
```

Escrevendo o dado em CSV

Do mesmo modo, para escrever dados no formato CSV, temos a função **to_csv** que pode ser utilizada. Independente da origem do dado, é possível criar um arquivo .csv mesmo que ele já exista.

Muito cuidado pois você pode SOBRESCREVER um arquivo já existente se escrevê-lo com o mesmo nome.

Resolução da SP

Como criar um dicionário com dados de clientes e, utilizando pandas, um arquivo csv a partir dele?

Solução

```
>>> cadastro_cliente={'nome:':['Leonardo', 'Paula', 'Fernanda'],'CPF:':[11122
233356, 22211133345, 33322211199], 'Fone':[33334444, 44453322, 66655577]}
>>> data = pd.DataFrame(cadastro_cliente)
>>> data
      nome:         CPF:      Fone
0  Leonardo  11122233356  33334444
1     Paula  22211133345  44453322
2  Fernanda  33322211199  66655577
>>> data.to_csv('/tmp/cad_clientes.csv')
```

Interação

Dúvidas

Conceitos

Dados e formatos

Formato binário

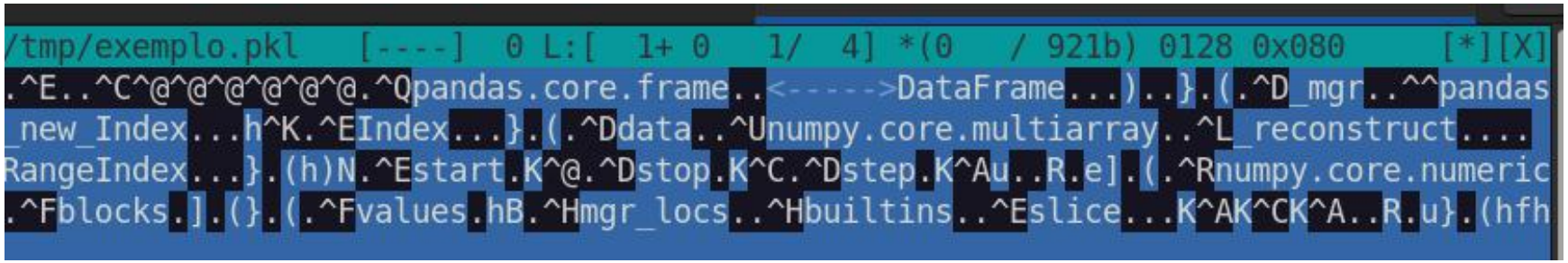
uma das formas mais fáceis de guardar os dados de forma eficiente em um formato binário é utilizando a serialização pickle nativa do python. (Mackinney, 2018)

O módulo pickle implementa protocolos binários para serializar e desserializar uma estrutura de objeto Python. “Pickling” é o processo pelo qual uma hierarquia de objetos Python é convertida em um fluxo de bytes, e “unpickling” é a operação inversa, em que um fluxo de bytes (de um arquivo binário ou objeto semelhante a bytes) é convertido de volta em uma hierarquia de objetos.

Na prática

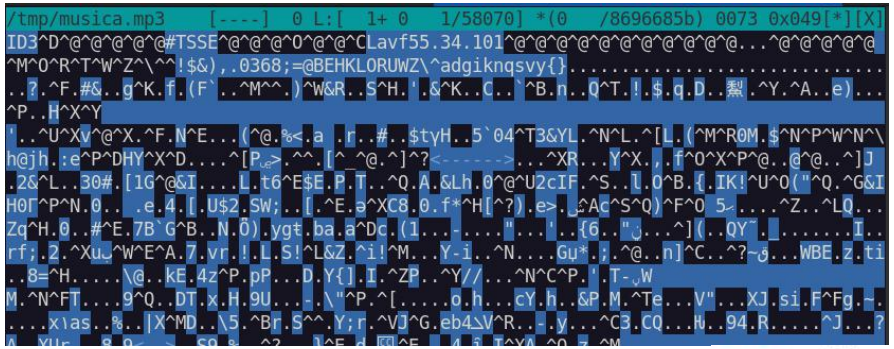
Muito popular e uma das mais utilizadas, oferece ao programador uma forma de trabalhar com análise e estruturas de dados de forma siplificada.

```
>>> data.to_pickle('/tmp/exemplo.pkl')
```



Na prática

Arquivo binário é todo arquivo que não está estruturado em formato de texto tal como o conhecemos. Pode ser de diversas fontes: programa de computador, imagem, som e outros. Exemplo de arquivo de música:




```
>>> df = pd.read_pickle('/tmp/exemplo.pkl')
>>> df
      nome:          CPF:      Fone
0  Leonardo  11122233356  33334444
1     Paula  22211133345  44453322
2  Fernanda  33322211199  66655577
```

Tabela relacional

Outra forma de consulta a dados utilizando Pandas é às tabelas relacionais. Elas são fruto da linguagem SQL, utilizada para manipulação de banco de dados.

CLIENTES

id_cliente	nome	Idade
1	Maria	42
2	Pedro	38

PRODUTOS

id_produto	nome_produto
101	Produto A
102	Produto B

VENDAS

id_transacao	id_cliente	id_produto	quantidade	data
1001	1	101	2	01/01/2021
1002	1	102	5	01/01/2021
1003	2	101	3	05/01/2021
1004	2	102	2	07/02/2021
1005	1	102	4	08/02/2021
1006	2	101	7	04/03/2021
1007	2	102	4	04/03/2021

Conexão a banco de dados

Com pandas, é possível conexão a banco para realizar consultas. Neste exemplo, uma conexão a banco mysql.

In [24]:

```
from sqlalchemy import create_engine
import pandas as pd

string_de_conexao = 'mysql+pymysql://mysql_user:mysql_password@mysql_host/mysql_db'
conexao = create_engine(string_de_conexao)

consulta = """
    SELECT ID_TRANSACAO, ID_PRODUTO, QUANTIDADE
    FROM VENDAS
    WHERE ID_CLIENTE = 1
"""
df = pd.read_sql(consulta, con=conexao)
```

Conceitos

DataFrame

DataFrame

São Dados tabulares bidimensionais, com tamanho mutável e potencialmente heterogêneos.

A estrutura de dados também contém eixos rotulados (linhas e colunas). As operações aritméticas se alinham nos rótulos de linha e coluna. Pode ser considerado um contêiner do tipo dict para objetos Series. A estrutura de dados principal do pandas.

DataFrame

É possível manipular um dataFrame e obter informações pontuais. Por exemplo, é possível obter somente informação de uma determinada coluna gerada por um DataFrame.

Exemplo

In [8]:
```
dados = {
    'id_cliente': [101, 102, 103, 104],
    'saldo': [500, 6500, 7000, 800],
    'tipo_de_conta': ['corrente', 'poupança', 'corrente', 'corrente']
}
df = pd.DataFrame(dados)
df
```

Out[8]:

	id_cliente	saldo	tipo_de_conta
0	101	500	corrente
1	102	6500	poupança
2	103	7000	corrente
3	104	800	corrente

In [9]:
```
df['id_cliente']
```

Out[9]:
```
0    101
1    102
2    103
3    104
Name: id_cliente, dtype: int64
```

In [10]:
```
df[['id_cliente', 'saldo']]
```

Out[10]:

	id_cliente	saldo
0	101	500
1	102	6500
2	103	7000
3	104	800

Fatiamento

Assim como no python, é possível utilizar fatiamento com pandas para selecionar os dados a serem exibidos. Veja:

```
In [11]:  1 df[:2]
```

Out[11]:

	id_cliente	saldo	tipo_de_conta
0	101	500	corrente
1	102	6500	poupança

Sugestão de leituras

É possível selecionar dados com utilização de array booleano, ou seja, comparação:

```
In [12]: 1 df['tipo_de_conta'] == 'poupança'

Out[12]: 0    False
         1     True
         2    False
         3    False
         Name: tipo_de_conta, dtype: bool
```

```
In [13]: 1 df[df['tipo_de_conta'] == 'poupança']
```

Out[13]:

	id_cliente	saldo	tipo_de_conta
1	102	6500	poupança

Conceitos

Gráfico com Pandas

Matplotlib

A geração de gráficos é outro recurso indispensável no tratamento e análise de dados. Para isso, é utilizada a biblioteca Matplotlib que conta com uma série de recursos para plotagem de gráficos.

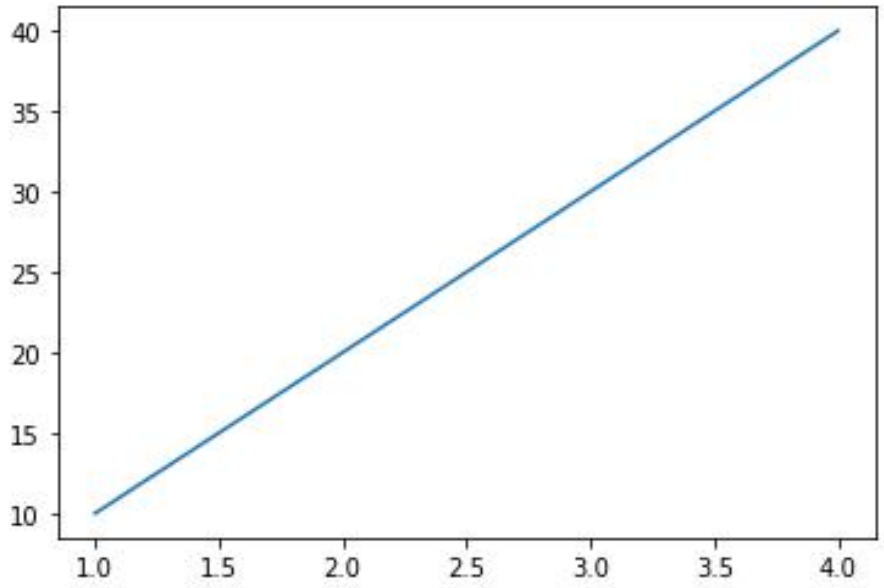
A importação é feita da seguinte maneira:

```
In [1]:  1  import matplotlib.pyplot as plt
```

Exemplo

In [2]:
```
1 x = [1, 2, 3, 4]
2 y = [10, 20, 30, 40]
3
4 plt.plot(x, y)
```

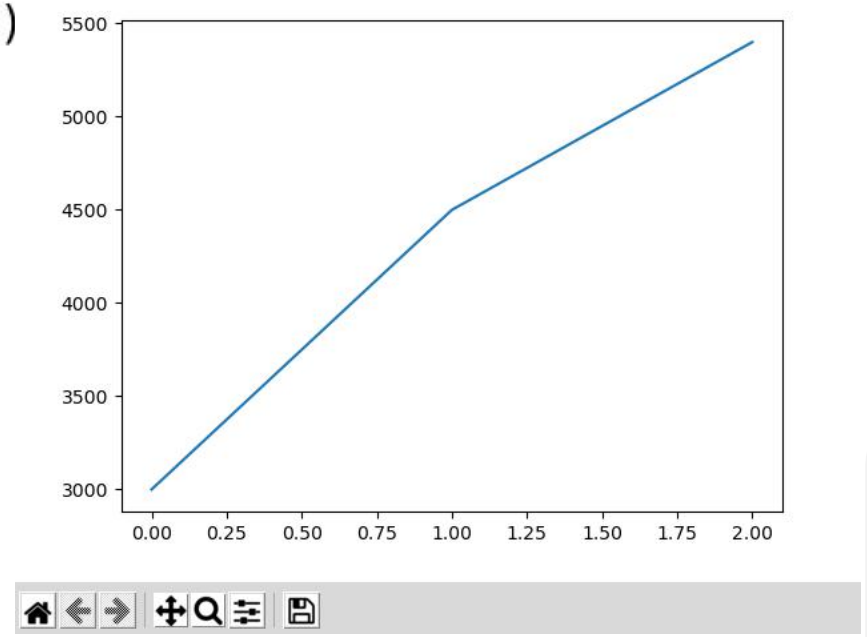
Out[2]: [<matplotlib.lines.Line2D at 0x1d6dc1c76c8>]



Prática

```
>>> valor_vendas={'vendedor':['Leonardo', 'Paula', 'Fernanda'],
'valor':[3000,4500,5400]}
>>> df = pd.DataFrame(valor_vendas)
>>> df
   vendedor  valor
0  Leonardo   3000
1     Paula   4500
2  Fernanda   5400
```

```
>>> plt.plot(df['valor'])
>>> plt.show()
```

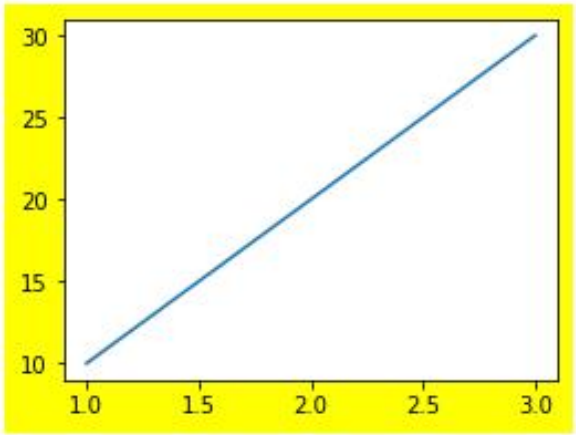


plot() para criar um gráfico de linha com as coordenadas das listas x e y

Figure

```
In [3]:
1 fig = plt.figure(figsize=(4, 3), facecolor='yellow')
2 ax = fig.add_subplot(1, 1, 1)
3 ax.plot([1, 2, 3], [10, 20, 30])

Out[3]: [<matplotlib.lines.Line2D at 0x15e1514bc48>]
```



figsize - define tamanho da figura

facecolor - define cor de fundo da figura

Tipos de gráficos

Para um gráfico de pontos e barra basta uma pequena alteração.

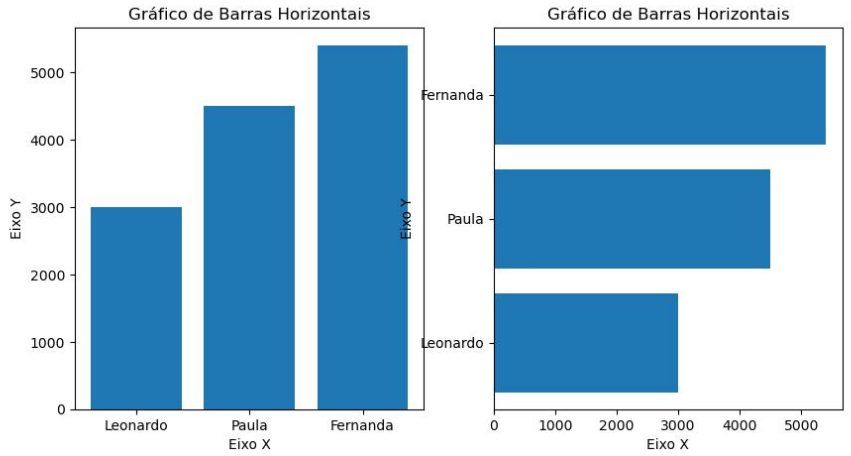
```
>>> plt.plot(df['valor'],'o')
<matplotlib.lines.Line2D object at 0x7f42b2a775b0>
>>> plt.show()
```

ou

```
>>> plot.bar(df['vendedor'],df['valor'])
>>> plt.show()
```

Dois gráficos

```
>>> fig, (ax1, ax2) = plt.subplots(1,2, figsize=(10,5))
>>> ax1.bar(df['vendedor'],df['valor'])
<BarContainer object of 3 artists>
>>> ax2.barh(df['vendedor'],df['valor'])
<BarContainer object of 3 artists>
>>> ax1.set(title='Gráfico de Barras Horizontais', xlabel='Eixo
X', ylabel='Eixo Y')
[Text(0.5, 1.0, 'Gráfico de Barras Horizontais'), Text(0.5, 0, '
Eixo X'), Text(0, 0.5, 'Eixo Y')]
>>> ax2.set(title='Gráfico de Barras Horizontais', xlabel='Eixo
X', ylabel='Eixo Y')
[Text(0.5, 1.0, 'Gráfico de Barras Horizontais'), Text(0.5, 0, '
Eixo X'), Text(0, 0.5, 'Eixo Y')]
>>> plot.show()
```

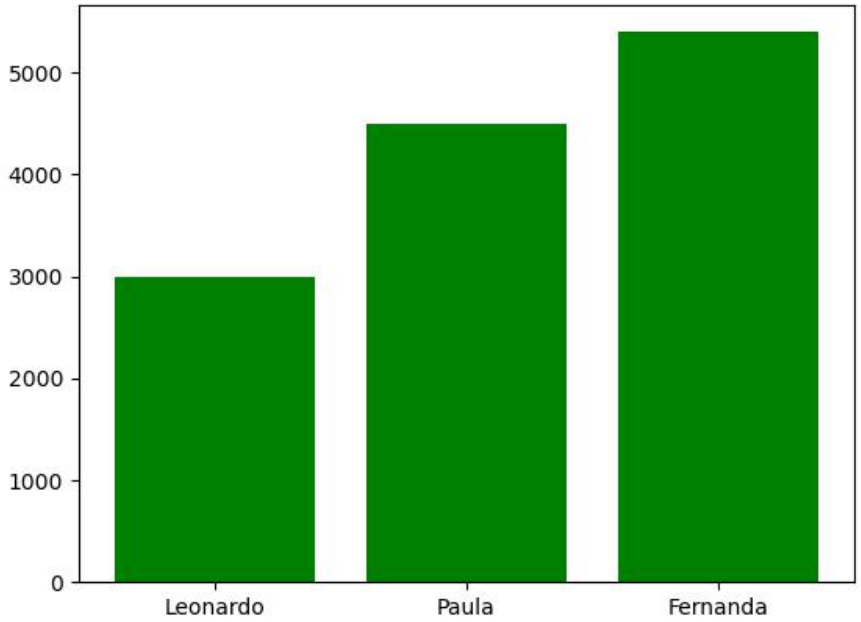


Resolução da SP

Como mudar a cor do gráfico desejado?

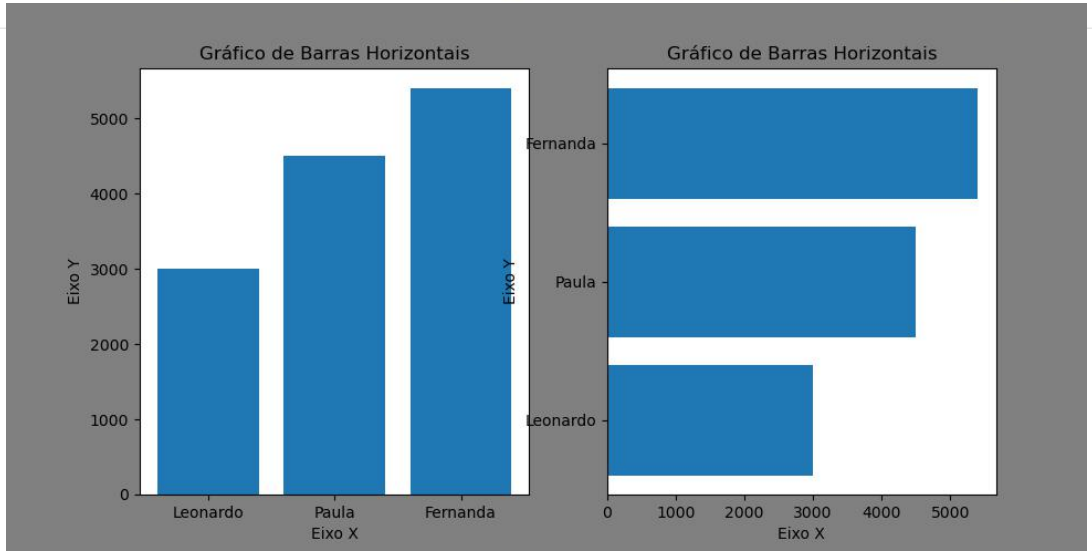
1ª forma - cor do gráfico

```
>>> plt.bar(df['vendedor'],df['valor'], color='gr
een')
<BarContainer object of 3 artists>
>>> plt.show()
```



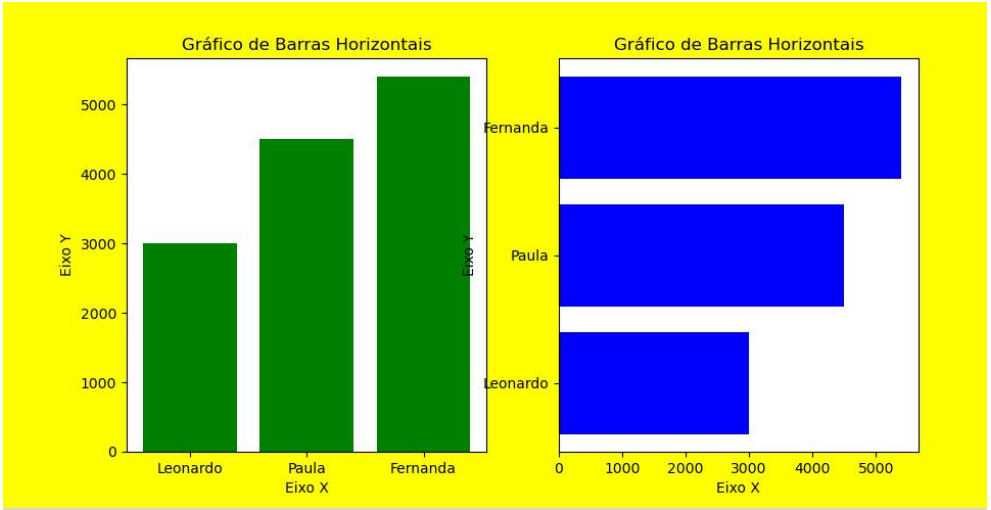
2ª forma - cor do fundo do gráfico

```
>>> fig, (ax1, ax2) = plt.subplots(1,2, figsize=(10,5), facecolor='gray')
>>> ax1.bar(df['vendedor'],df['valor'])
<BarContainer object of 3 artists>
>>> ax2.barh(df['vendedor'],df['valor'])
<BarContainer object of 3 artists>
>>> ax1.set(title='Gráfico de Barras Horizontais', xlabel='Eixo X', ylabel='Eixo Y')
[Text(0.5, 1.0, 'Gráfico de Barras Horizontais'), Text(0.5, 0, 'Eixo X'), Text(0, 0.5, 'Eixo Y')]
>>> ax2.set(title='Gráfico de Barras Horizontais', xlabel='Eixo X', ylabel='Eixo Y')
[Text(0.5, 1.0, 'Gráfico de Barras Horizontais'), Text(0.5, 0, 'Eixo X'), Text(0, 0.5, 'Eixo Y')]
>>> plt.show()
```



As duas

```
>>> fig, (ax1, ax2) = plt.subplots(1,2, figsize=(10,5), facecolor='yellow')
>>> ax1.bar(df['vendedor'],df['valor'], color='green')
<BarContainer object of 3 artists>
>>> ax2.barh(df['vendedor'],df['valor'], color='blue')
<BarContainer object of 3 artists>
>>> ax1.set(title='Gráfico de Barras Horizontais', xlabel='Eixo X', ylabel='Eixo Y')
[Text(0.5, 1.0, 'Gráfico de Barras Horizontais'), Text(0.5, 0, 'Eixo X'), Text(0, 0.5, 'Eixo Y')]
>>> ax2.set(title='Gráfico de Barras Horizontais', xlabel='Eixo X', ylabel='Eixo Y')
[Text(0.5, 1.0, 'Gráfico de Barras Horizontais'), Text(0.5, 0, 'Eixo X'), Text(0, 0.5, 'Eixo Y')]
>>> plt.show()
```



Interação

Dúvidas

Conceitos

Recapitulando

Vimos as funções para leitura e escrita de dados

Tipos de dados como salvo em binário

Construção de gráficos

Modificação de características de gráficos