Engenharia de Software
Computacional thinking with Python
Aula 04 –
Dicionários e Exercícios Práticos

Prof. Dr. Francisco Elânio

Pandas vs NumPy

Pandas e NumPy são duas bibliotecas muito populares em Python para análise e manipulação de dados. Embora possam parecer semelhantes em certos aspectos, cada uma delas tem suas próprias características e é adequada para diferentes tipos de tarefas.

Pandas	NumPy
pode conter tipos de dados diferentes	tem dados homogêneos
operações tabulares, tarefas de pré-processamento semântico semelhantes a SQL	computação numérica, operações matriciais e vetoriais
duas dimensões	multidimensional (>2 possível)
mais memória	menos memória
Mais devagar	mais rápido

Exercícios Numpy
Estrutura Unidimensional

1. Exercício de Criação: Crie um NumPy array contendo os números de 1 a 10.
2. Exercício de Acesso: Acesse o quarto elemento de um NumPy array e imprima seu valor.
3. Exercício de Soma: Some dois NumPy arrays contendo os números de 1 a 5 e os números de 6 a 10, respectivamente.
4. Exercício de Filtragem: Crie um NumPy array contendo os números de 1 a 10 e exiba apenas os valores maiores que 5.
5. Exercício de Estatísticas: Calcule a média e o desvio padrão de um NumPy array contendo os números de 1 a 10.
6. Exercício de Concatenação: Concatene dois NumPy arrays contendo os números de 1 a 5 e 6 a 10, respectivamente, em um único array.
7. Exercício de Subtração: Subtraia um NumPy array contendo os números de 1 a 10 por outro array contendo os números de 10 a 1.
8. Exercício de Ordenação: Ordene um NumPy array contendo os números de 1 a 10 em ordem decrescente.
9. Exercício de Verificação de Valores Únicos: Verifique se há valores duplicados em um NumPy array contendo os números de 1 a 10.
10. Exercício de Slicing: Utilize slicing para selecionar os elementos de um NumPy array do terceiro ao sétimo elemento.

Exercícios Numpy
Estrutura Unidimensional

1. Exercício de Criação: Crie um NumPy array contendo os números de 1 a 10.

```
import numpy as np

# Exercício de Criação
array1to10 = np.array([1, 2, 3, 4, 5, 6, 7, 8, 9, 10])
```

2. Exercício de Acesso: Acesse o quarto elemento de um NumPy array e imprima seu valor.

```
# Exercício de Acesso
quarto_elemento = array1to10[3]
print("O quarto elemento do array é:",
quarto_elemento)
```

3. Exercício de Soma: Some dois NumPy arrays contendo os números de 1 a 5 e os números de 6 a 10, respectivamente.

```
# Exercício de Soma
array1to5 = np.array([1, 2, 3, 4, 5])
array6to10 = np.array([6, 7, 8, 9, 10])
soma_arrays = array1to5 + array6to10
print("A soma dos arrays é:", soma_arrays)
```

4. Exercício de Filtragem: Crie um NumPy array contendo os números de 1 a 10 e exiba apenas os valores maiores que 5.

```
# Exercício de Filtragem
array_maiores_que_5 = array1to10[array1to10 > 5]
print("Valores maiores que 5:", array_maiores_que_5)
```

Exercícios Numpy
Resolução

5. Exercício de Estatísticas: Calcule a média e o desvio padrão de um NumPy array contendo os números de 1 a 10.

```
print("Média:", arr.mean())

print("Desvio padrão:", arr.std())
```

6. Exercício de Concatenação: Concatene dois NumPy arrays contendo os números de 1 a 5 e 6 a 10, respectivamente, em um único array.

```
arr_concat = np.concatenate([arr1, arr2])

print("Concatenação:")

print(arr_concat)
```

7. Exercício de Subtração: Subtraia um NumPy array contendo os números de 1 a 10 por outro array contendo os números de 10 a 1.

```
import numpy as np

arr1 = np.arange(1, 11)

arr2 = np.arange(10, 0, -1)

resultado = arr1 - arr2

print("Resultado da Subtração:“, resultado)
```

8. Exercício de Ordenação: Ordene um NumPy array contendo os números de 1 a 10 em ordem decrescente.

```
arr1 = np.arange(1, 11)

arr_ordenado = np.sort(arr1)[::-1]

print(arr_ordenado)
```

Exercícios Numpy
Resolução

9. Exercício de Verificação de Valores Únicos: Verifique se há valores duplicados em um NumPy array contendo os números de 1 a 10.

```
arr = np.array([0, 4, 2, 6, 5, 9, 6])

print("Valores Duplicados:", np.unique(arr).size != arr.size)
```

10. Exercício de Slicing: Utilize slicing para selecionar os elementos de um NumPy array do terceiro ao sétimo elemento.

```
print("Slicing do terceiro ao sétimo elemento:")

print(arr[2:7])
```

Exercícios Desafio
Pandas Series

Exercício Prático: Análise de Dados de Vendas de uma Loja de Eletrônicos

Descrição do Problema:

Uma loja de eletrônicos deseja analisar seus dados de vendas para entender melhor o desempenho de seus produtos ao longo do tempo. Eles possuem um arquivo CSV com informações sobre as vendas de vários produtos nos últimos meses. O objetivo é usar o Pandas Series para realizar algumas análises básicas e responder a algumas perguntas sobre os dados de vendas.

Exercícios Desafio
Pandas Series

Tarefas

1. Carregar os dados do arquivo CSV em um Pandas Series, onde cada entrada representa o número de vendas de um produto em um determinado mês.

2. Identificar o produto mais vendido e o menos vendido ao longo de todo o período.

3. Calcular a média, mediana, máximo e mínimo de vendas mensais.

4. Crie um gráfico para apresentar as vendas dos produtos ao longo do tempo.

5. Calcular a variação percentual entre as vendas de um mês para o próximo e identificar os meses com maior e menor variação percentual.

Exercícios Desafio
Pandas Series

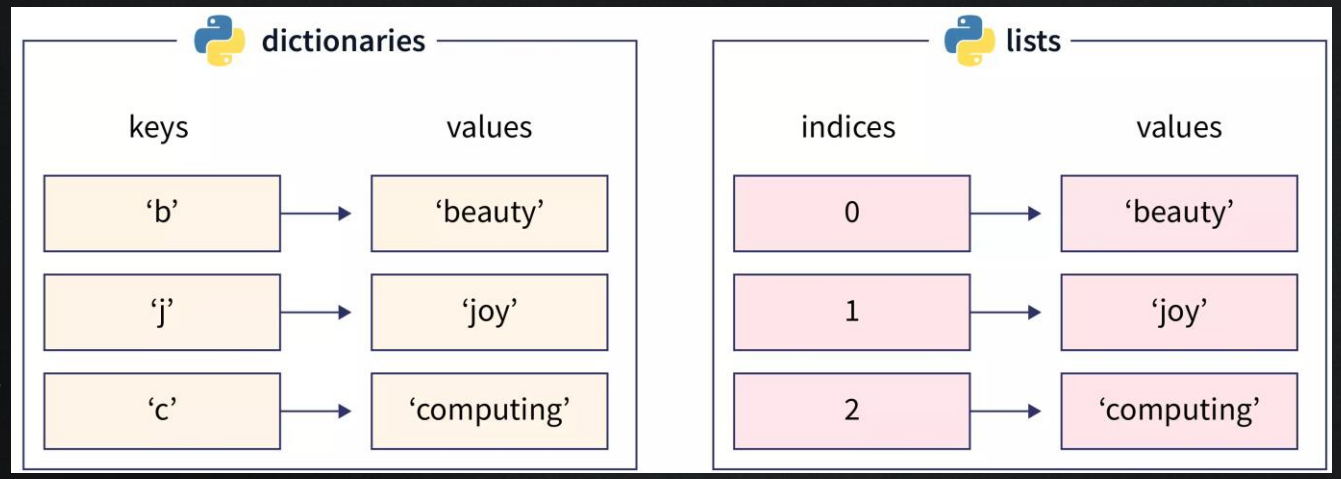
```
Produto,Mês,Vendas
Celular,01-01-2023,150
Celular, 01-02-2023,180
Celular, 01-03-2023,200
Tablet, 01-01-2023,100
Tablet, 01-02-2023,120
Tablet, 01-03-2023,130
Notebook, 01-01-2023,80
Notebook, 01-02-2023,90
Notebook, 01-03-2023,100
```

Resolver todos exercícios utilizando pandas series

Criando Dicionários em Python

Um dicionário é uma estrutura de dados que permite armazenar coleções de pares chave-valor. Cada elemento no dicionário consiste em uma chave e seu valor correspondente. As chaves em um dicionário são únicas, o que significa que não pode haver chaves duplicadas no mesmo dicionário.

Os elementos de um dicionário são acessados por meio de suas chaves. Isso torna os dicionários ideais para mapear chaves a valores associados, fornecendo uma maneira eficiente de recuperar e manipular dados.



Criando Dicionários em Python
Exemplo 1

```
notas = {'João': 85, 'Maria': 90, 'Pedro': 88}

print(notas['Maria'])

print(notas[‘João'])
```

```
# Criando um dicionário de notas de alunos

# Acessando o valor associado à chave 'Maria'

# Acessando o valor associado à chave ‘João'
```

Criando Dicionários em Python
Exemplo 2

```
notas_alunos = {
    'João': {'matemática': 90, 'português': 85, 'ciências': 88},
    'Maria': {'matemática': 95, 'português': 92, 'ciências': 89},
    'Pedro': {'matemática': 80, 'português': 75, 'ciências': 82}
}

    nota_maria_matematica = notas_alunos['Maria']['matemática']
    print("A nota de Maria em matemática é:",
    nota_maria_matematica)

    notas_alunos['Ana'] = {'matemática': 88, 'português': 91,
    'ciências': 85}
    notas_alunos
```

```
# Criando um dicionário para armazenar informações sobre estudantes e suas notas
```

```
# Acessando as notas de um aluno específico
```

```
# Adicionando um novo aluno ao dicionário
```

Criando Dicionários em Python
Exemplo 3 – Dicionário para Dataframe

```
# Cria um dataframe do dicionário notas_alunos

df = pd.DataFrame(notas_alunos)
```

	João	Maria	Pedro	Ana
matemática	90	95	80	88
português	85	92	75	91
ciências	88	89	82	85

Criando Dicionários em Python
Exemplo 4 –Dataframe acessando informações

```
# Cria um dataframe do dicionário notas_alunos

df = pd.DataFrame(notas_alunos)

df['João']

df['João']['matemática']

dados_dois_alunos = df.loc[:, ['João', 'Maria']]
```

```
# Acessando informações do João

# Acessando informações da nota de
matemática de João

# Selecionar dados
apenas de João e Maria
```

Criando Dicionários em Python
Exemplo 5 – Inserindo novas informações

```
# Criar uma lista de datas no formato desejado
datas = ['01-01-2024', '01-02-2024', '01-03-2024']

# Converter as datas para o formato datetime
datas_formatadas = pd.to_datetime(datas, format='%d-%m-%Y')

# Inserir a coluna de datas no DataFrame
df['Data'] = datas_formatadas
df
```

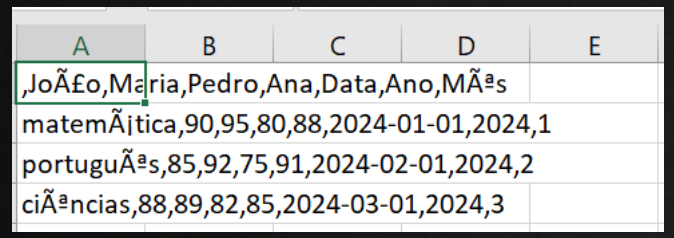
Criando Dicionários em Python
Exemplo 6 – Extraindo mês e ano

```
df['Data'] = pd.to_datetime(df['Data'], format='%d-%m-%Y')
df['Ano'] = df['Data'].dt.year
df['Mês'] = df['Data'].dt.month
```

	João	Maria	Pedro	Ana	Data	Ano	Mês
matemática	90	95	80	88	2024-01-01	2024	1
português	85	92	75	91	2024-02-01	2024	2
ciências	88	89	82	85	2024-03-01	2024	3

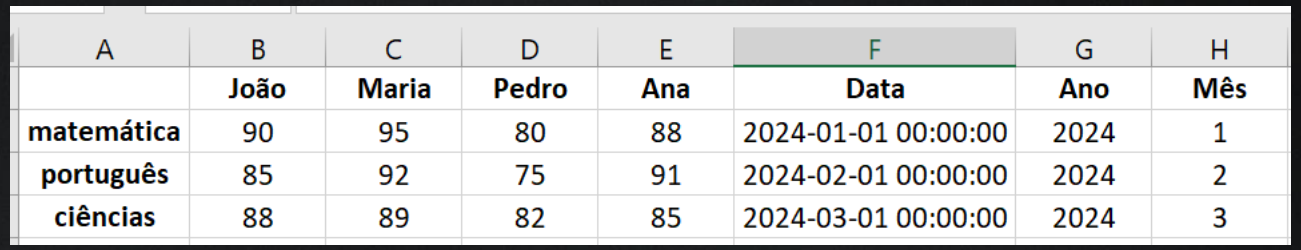
Criando Dicionários em Python
Exemplo 7 – Salvando arquivos

```
df.to_csv('exemplo11.csv')
```



A	B	C	D	E
,JoÃ£o,Maria,Pedro,Ana,Data,Ano,MÃªs				
matemÃ¡tica,90,95,80,88,2024-01-01,2024,1				
portuguÃªs,85,92,75,91,2024-02-01,2024,2				
ciÃªncias,88,89,82,85,2024-03-01,2024,3				

```
df.to_excel('exemplo2.xlsx')
```



A	B	C	D	E	F	G	H
	João	**Maria**	**Pedro**	**Ana**	**Data**	**Ano**	**Mês**
matemática	90	95	80	88	2024-01-01 00:00:00	2024	1
português	85	92	75	91	2024-02-01 00:00:00	2024	2
ciências	88	89	82	85	2024-03-01 00:00:00	2024	3

Exercícios Desafio
Pandas Series - Resolução

1. Carregar os dados do arquivo CSV em um Pandas Series, onde cada entrada representa o número de vendas de um produto em um determinado mês.

```
import pandas as pd

# Carregar os dados do arquivo CSV em um Pandas DataFrame
data = {
    'Produto': ['Celular', 'Celular', 'Celular', 'Tablet', 'Tablet', 'Tablet', 'Notebook', 'Notebook', 'Notebook'],
    'Mês': ['Jan-2023', 'Fev-2023', 'Mar-2023', 'Jan-2023', 'Fev-2023', 'Mar-2023', 'Jan-2023', 'Fev-2023',
'Mar-2023'],
    'Vendas': [150, 180, 200, 100, 120, 130, 80, 90, 100]
}
df = pd.DataFrame(data)
```

Exercícios Desafio
Pandas Series - Resolução

2. Identificar o produto mais vendido e o menos vendido ao longo de todo o período.

```
df['Mês'] = pd.to_datetime(df['Mês'], format='%d-%m-%Y')
df['Ano'] = df['Mês'].dt.year
df['Mês'] = df['Mês'].dt.month

# Converter o DataFrame para um Pandas Series, indexado pelos meses
vendas_series = pd.Series(df['Vendas'].values, index=df['Mês'])

# 2. Identificar o produto mais vendido e o menos vendido
produto_mais_vendido = df.groupby('Produto')['Vendas'].sum().idxmax()
produto_menos_vendido = df.groupby('Produto')['Vendas'].sum().idxmin()
```

Exercícios Desafio
Pandas Series - Resolução

3. Calcular a média, mediana, máximo e mínimo de vendas mensais.

```
media_vendas = vendas_series.mean()
mediana_vendas = vendas_series.median()
maximo_vendas = vendas_series.max()
minimo_vendas = vendas_series.min()

print("Média de vendas mensais:", media_vendas)
print("Mediana de vendas mensais:", mediana_vendas)
print("Máximo de vendas mensais:", maximo_vendas)
print("Mínimo de vendas mensais:", minimo_vendas)
```

Exercícios Desafio
Pandas Series - Resolução

4. Visualizar um gráfico de linha mostrando a tendência de vendas ao longo do tempo.

```
# Agrupar por mês e produto e calcular a soma das vendas
df_grouped = df.groupby(['Mês', 'Produto'])['Vendas'].sum().unstack()

# Plotar o gráfico de barras
plt.figure(figsize=(10, 6))
df_grouped.plot(kind='bar', stacked=False)
plt.title("Vendas por Produto ao longo do Tempo")
plt.xlabel("Mês")
plt.ylabel("Vendas")
plt.xticks(rotation=45)
plt.legend(title='Produto')
plt.grid(axis='y')
plt.tight_layout()
plt.show()
```

unstack(): reorganiza o DataFrame de modo que o índice resultante contenha os meses e as colunas correspondam aos produtos.

Exercícios Desafio
Pandas Series - Resolução

5. Calcular a variação percentual entre as vendas de um mês para o próximo e identificar os meses com maior e menor variação percentual.

```
# Calcular a variação percentual entre as vendas de um mês para o próximo
df['Variação Percentual'] = df.groupby('Produto')['Vendas'].pct_change()

# Identificar os meses com maior e menor variação percentual
mes_maior_variacao = df.loc[df['Variação Percentual'].idxmax()]['Mês']
mes_menor_variacao = df.loc[df['Variação Percentual'].idxmin()]['Mês']
mes_menor_variacao
```

.pct_change(): calcula a variação percentual entre os elementos consecutivos de cada grupo. No caso, estamos calculando a variação percentual das vendas de cada produto em relação ao mês anterior.

Exercícios Desafio
Pandas Series

Exercício Prático: vendas de produtos de climatização

Descrição do Problema:

Uma loja de eletrônicos deseja analisar seus dados de vendas de produtos de climatização para entender melhor o desempenho ao longo do tempo. Eles possuem um arquivo CSV com informações sobre as vendas de ar condicionado, ventilador e ar quente nos últimos cinco meses. O objetivo é usar o Pandas Series para realizar algumas análises básicas e responder a algumas perguntas sobre os dados de vendas.

Exercícios Desafio
Pandas Series

1. Carregue os dados do arquivo CSV em um Pandas DataFrame, onde cada entrada representa o número de vendas de um produto em um determinado mês.

2. Identifique o produto mais vendido e o menos vendido ao longo dos cinco meses.

3. Calcule a média, mediana, máximo e mínimo de vendas mensais para cada produto.

4. Crie um gráfico para apresentar as vendas dos produtos ao longo do tempo.

5. Calcule a variação percentual entre as vendas de um mês para o próximo para cada produto e identifique os meses com maior e menor variação percentual.

Exercício Desafio
Tabela de venda de produtos

data	Produto	Quantidade
01/01/2024	Ar Frio	100
01/02/2024	Ar Frio	120
01/03/2024	Ar Frio	130
01/04/2024	Ar Frio	110
01/05/2024	Ar Frio	150
01/01/2024	Ventilador	150
01/02/2024	Ventilador	140
01/03/2024	Ventilador	160
01/04/2024	Ventilador	130
01/05/2024	Ventilador	170
01/01/2024	Ar Quente	80
01/02/2024	Ar Quente	90
01/03/2024	Ar Quente	100
01/04/2024	Ar Quente	85
01/05/2024	Ar Quente	95

Exercícios Desafio
Pandas Series

1. Carregue os dados do arquivo CSV em um Pandas DataFrame, onde cada entrada representa o número de vendas de um produto em um determinado mês.

```
import pandas as pd

# Dados do DataFrame
data = {
    'Data': ['01/01/2024', '01/02/2024', '01/03/2024', '01/04/2024', '01/05/2024', '01/01/2024', '01/02/2024', '01/03/2024',
'01/04/2024', '01/05/2024', '01/01/2024', '01/02/2024', '01/03/2024', '01/04/2024', '01/05/2024'],
    'Produto': ['Ar Frio', 'Ar Frio', 'Ar Frio', 'Ar Frio', 'Ar Frio', 'Ventilador', 'Ventilador', 'Ventilador', 'Ventilador', 'Ventilador',
                'Ar Quente', 'Ar Quente', 'Ar Quente', 'Ar Quente', 'Ar Quente'],
    'Quantidade': [100, 120, 130, 110, 150, 150, 140, 160, 130, 170, 80, 90, 100, 85, 95]
}

# Criar o DataFrame
df = pd.DataFrame(data)
```

Exercícios Desafio
Pandas Series

2. Identifique o produto mais vendido e o menos vendido ao longo dos cinco meses.

```
df['Data'] = pd.to_datetime(df['Data'], format='%d-%m-%Y')
df['Ano'] = df['Data'].dt.year
df['Mês'] = df['Data'].dt.month

# Converter o DataFrame para um Pandas Series, indexado pelos meses
vendas_series = pd.Series(df['Quantidade'].values, index=df['Mês'])

# 2. Identificar o produto mais vendido e o menos vendido
produto_mais_vendido = df.groupby('Produto')['Quantidade'].sum().idxmax()
produto_menos_vendido = df.groupby('Produto')['Quantidade'].sum().idxmin()
produto_mais_vendido
```

Exercícios Desafio
Pandas Series

3. Calcule a média, mediana, máximo e mínimo de vendas mensais para cada produto.

```
media_vendas = vendas_series.mean()
mediana_vendas = vendas_series.median()
maximo_vendas = vendas_series.max()
minimo_vendas = vendas_series.min()

print("Média de vendas mensais:", media_vendas)
print("Mediana de vendas mensais:", mediana_vendas)
print("Máximo de vendas mensais:", maximo_vendas)
print("Mínimo de vendas mensais:", minimo_vendas)
```

Exercícios Desafio
Pandas Series

4. Crie um gráfico para apresentar as vendas dos produtos ao longo do tempo.

```
# Agrupar por mês e produto e calcular a soma das vendas
df_grouped = df.groupby(['Mês', 'Produto'])['Quantidade'].sum().unstack()

# Plotar o gráfico de barras
plt.figure(figsize=(10, 6))
df_grouped.plot(kind='bar', stacked=False)
plt.title("Vendas por Produto ao longo do Tempo")
plt.xlabel("Mês")
plt.ylabel("Quantidade")
plt.xticks(rotation=45)
plt.legend(title='Produto')
plt.grid(axis='y')
plt.tight_layout()
plt.show()
```

Exercícios Desafio
Pandas Series

5. Calcule a variação percentual entre as vendas de um mês para o próximo para cada produto e identifique os meses com maior e menor variação percentual.

```
# Calcular a variação percentual entre as vendas de um mês para o próximo
df['Variação Percentual'] = df.groupby('Produto')['Quantidade'].pct_change()

# Identificar os meses com maior e menor variação percentual
mes_maior_variacao = df.loc[df['Variação Percentual'].idxmax()]['Mês']
mes_menor_variacao = df.loc[df['Variação Percentual'].idxmin()]['Mês']
mes_maior_variacao
```

“O que sabemos é uma gota; o que ignoramos é um oceano.”
(Issac Newton)

Referências

- ASCENCIO, A. F. G, CAMPOS, E. A. V. Fundamentos da Programação de Computadores: algoritmos, Pascal, C/C++ e Java, 2ª Edição, São Paulo: Pearson 2007.
- FURGERI, Sérgio. Introdução à Programação em Python. São Paulo: Editora Senac, 2021.
- MENEZES, Nilo. Introdução à Programação em Python. São Paulo: Novatec, 2019
- SALVETTI, Dirceu Douglas; BARBOSA, Lisbete Madson. Algoritmos. São Paulo: Pearson, 2004.