📑 Lista de Exercícios Propostos – Estatística e Python

🔹 Capítulo 1 – Probabilidade

1. **Urna com bolas coloridas**
   Uma urna contém 5 bolas vermelhas e 7 azuis. Qual a probabilidade de retirar uma bola vermelha? Calcular a probabilidade de sair 2 vermelhas na sequência (com e sem reposição)

```
import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

from sympy import symbols, Eq, solve

from sklearn.linear_model import LinearRegression




# CAPÍTULO 1 - PROBABILIDADE

print("\n===== CAPÍTULO 1 - PROBABILIDADE =====\n")

# 1. Urna com bolas coloridas

vermelhas = 5

azuis = 7

total = vermelhas + azuis

# i) Probabilidade de uma vermelha

p_vermelha = vermelhas / total

print(f"\nProbabilidade de tirar 1 vermelha: {p_vermelha:.4f}
({p_vermelha*100:.2f}%)")

##i) Probabilidade de tirar 1 vermelha: 0.4167 (41.67%)
```

```
# ii) Duas vermelhas COM reposição

p_duas_com_reposicao = (vermelhas / total) * (vermelhas / total)

print(f"\nDuas vermelhas (com reposição): {p_duas_com_reposicao:.4f}
({p_duas_com_reposicao*100:.2f}%)")

##ii) Duas vermelhas (com reposição): 0.1736 (17.36%)

# iii) Duas vermelhas SEM reposição

p_duas_sem_reposicao = (vermelhas / total) * ((vermelhas - 1) / (total - 1))

print(f"\nDuas vermelhas (sem reposição): {p_duas_sem_reposicao:.4f}
({p_duas_sem_reposicao*100:.2f}%)")

##iii) Duas vermelhas (sem reposição): 0.1515 (15.15%)
```

2. **Lançamento de um dado**
   Qual a probabilidade de sair um número maior que 3 ao lançar um dado de 6 lados?

```
# 2. Lançamento de um dado
total_faces = 6
favoraveis = 3  # números maiores que 3: 4,5,6
prob_maior_3 = favoraveis / total_faces
print(f"\nProbabilidade de sair número > 3: {prob_maior_3:.2f}
({prob_maior_3*100:.1f}%)")

# Probabilidade de sair número > 3: 0.50 (50.0%)
```

3. **Lançamento de duas moedas**
   Qual a probabilidade de sair exatamente uma cara ao lançar duas moedas?

```
# 3. Lançamento de duas moedas (exatamente uma cara)
# Espaço amostral: CC, CX, XC, XX
# Casos favoráveis: CX, XC
prob_uma_cara = 2 / 4
print(f"\nProbabilidade de sair exatamente uma cara: {prob_uma_cara:.2f}
({prob_uma_cara*100:.1f}%)")

# Probabilidade de sair exatamente uma cara: 0.50 (50.0%)
```

4. **Probabilidade condicional**
   Uma urna tem 3 bolas vermelhas e 2 verdes. Se uma bola é retirada **sem reposição** e sai vermelha, qual a probabilidade da próxima ser verde?

```
# 4. Probabilidade condicional

# 3 vermelhas e 2 verdes
```

```python
# Sem reposição, primeira vermelha, qual a probabilidade da segunda ser verde?

P_vermelha_1 = 3/5

P_verde_2_dado_vermelha = 2/4

P_conjunta = P_vermelha_1 * P_verde_2_dado_vermelha

print(f"\nProbabilidade de tirar vermelha e depois verde (sem reposição):
{P_conjunta:.2f}")



# Probabilidade de tirar vermelha e depois verde (sem reposição): 0.30
```

🔹 Capítulo 2 – Estatística Descritiva com Pandas

5. **Média, mediana e moda**
   Com o conjunto de dados: [10, 15, 20, 20, 25, 30, 35], calcule média, mediana e moda utilizando **pandas**.

```python
# 5. Média, mediana e moda

dados = pd.Series([10, 15, 20, 20, 25, 30, 35])

print("\nMédia:", dados.mean())

# Média: 22.142857142857142

print("\nMediana:", dados.median())

# Mediana: 20.0

print("\nModa:", dados.mode()[0])

# Moda: 20
```

6. **Variância e desvio padrão**
   Calcule a variância e o desvio padrão do conjunto: [2, 4, 4, 4, 5, 5, 7, 9], usando **pandas**.

```python
# 6. Variância e desvio padrão
dados2 = pd.Series([2, 4, 4, 4, 5, 5, 7, 9])
print("\nVariância:", dados2.var())
# Variância: 4.571428571428571
print("\nDesvio padrão:", dados2.std())
# Desvio padrão: 2.138089935299395
```

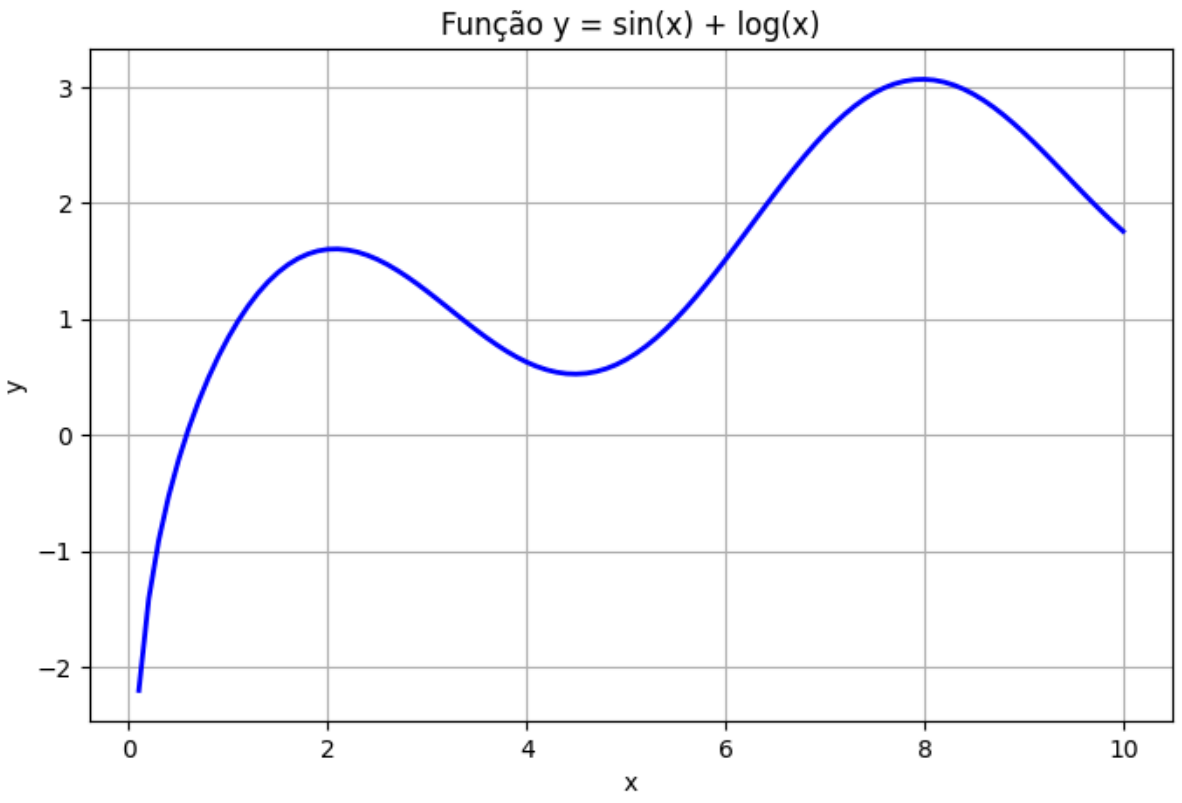
7. **Medidas resumo**
   Para os dados [5, 7, 8, 5, 10, 12, 15], calcule:
   - Média
   - Mediana
   - Valor mínimo
   - Valor máximo
   - Amplitude

```python
# 7. Medidas resumo

dados3 = pd.Series([5, 7, 8, 5, 10, 12, 15])

print("\nMédia:", dados3.mean())

# Média: 8.857142857142858

print("\nMediana:", dados3.median())

# Mediana: 8.0

print("\nValor mínimo:", dados3.min())

# Valor mínimo: 5

print("\nValor máximo:", dados3.max())

# Valor máximo: 15

print("\nAmplitude:", dados3.max() - dados3.min())

# Amplitude: 10
```

🔹 Capítulo 3 – Funções Trigonométricas e Logarítmicas

8. **Enunciado:** Usando `numpy`, gere uma série de valores `x` (100 números entre 0.1 e 10) e calcule `y = sin(x) + log(x)`. Plote o gráfico.

```
# CAPÍTULO 3 – FUNÇÕES TRIGONOMÉTRICAS E LOGARÍTMICAS

print("\n===== CAPÍTULO 3 – FUNÇÕES TRIGONOMÉTRICAS E LOGARÍTMICAS =====\n")

x = np.linspace(0.1, 10, 100)
y = np.sin(x) + np.log(x)
plt.figure(figsize=(8,5))
plt.plot(x, y, color='blue', linewidth=2)
plt.title("Função y = sin(x) + log(x)")
plt.xlabel("x")
plt.ylabel("y")
plt.grid(True)
plt.show(block=False)
```



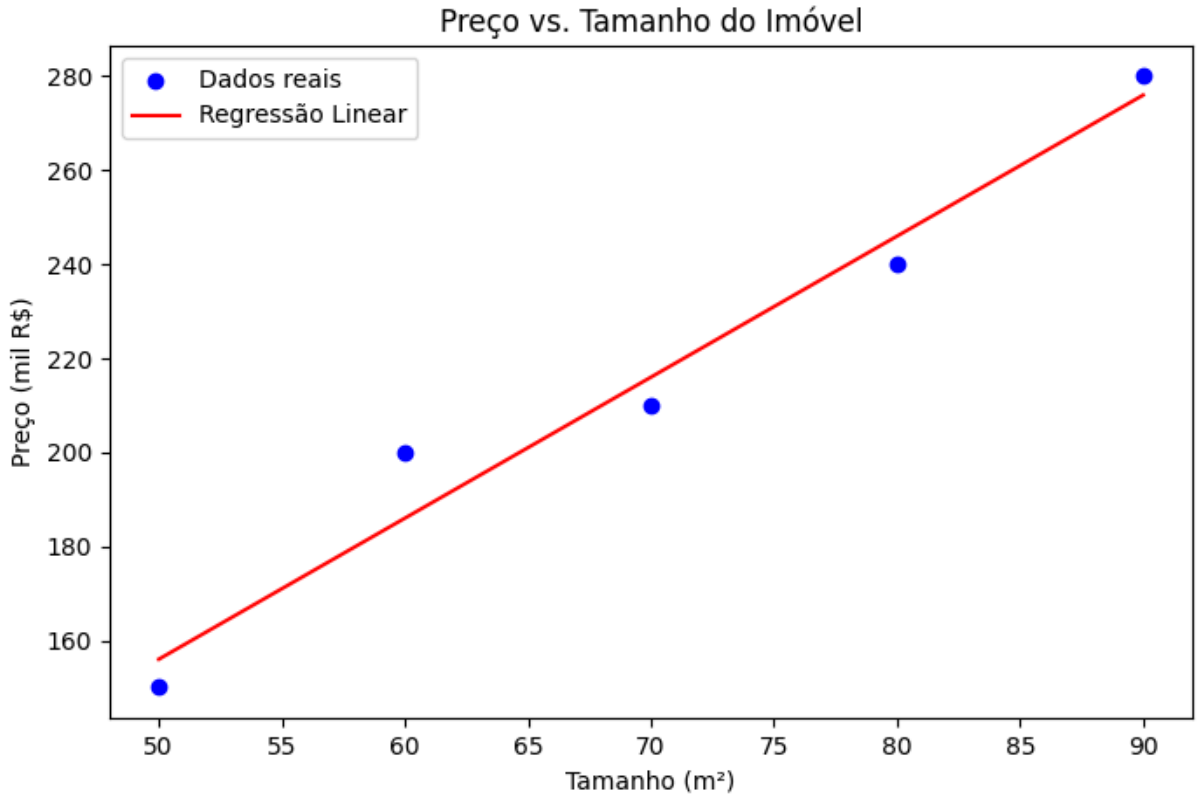
🔹 Capítulo 4 – Regressão Linear

9. **Horas de estudo vs. notas**
   Considere os dados:
   - Horas de estudo: [1, 2, 3, 4, 5]
   - Notas: [2, 4, 5, 4, 6]
     Ajuste uma **regressão linear simples** e interprete o coeficiente angular.

```
# 9. Horas de estudo vs. notas
X = np.array([1, 2, 3, 4, 5]).reshape(-1, 1)
y = np.array([2, 4, 5, 4, 6])
modelo = LinearRegression().fit(X, y)
coef = modelo.coef_[0]
intercepto = modelo.intercept_
print(f"\nCoeficiente angular (inclinação): {coef:.2f}")
# Coeficiente angular (inclinação): 0.80
print(f"\nIntercepto: {intercepto:.2f}")
# Intercepto: 1.80
```

10. **Preço vs. tamanho de imóveis**
    Considere os dados:
    - Tamanho (m²): [50, 60, 70, 80, 90]
    - Preço (mil reais): [150, 200, 210, 240, 280]
      Ajuste um modelo de regressão linear e estime o preço para um imóvel de 100 m².

```
# 10. Preço vs. tamanho de imóveis

X2 = np.array([50, 60, 70, 80, 90]).reshape(-1, 1)

y2 = np.array([150, 200, 210, 240, 280])

modelo2 = LinearRegression().fit(X2, y2)

preco_previsto = modelo2.predict([[100]])[0]

print(f"\nPreço estimado para imóvel de 100 m²: {preco_previsto:.2f} mil
reais") # Preço estimado para imóvel de 100 m²: 306.00 mil reais

plt.figure(figsize=(8,5))

plt.scatter(X2, y2, color='blue', label='Dados reais')

plt.plot(X2, modelo2.predict(X2), color='red', label='Regressão Linear')

plt.xlabel("Tamanho (m²)")

plt.ylabel("Preço (mil R$)")

plt.title("Preço vs. Tamanho do Imóvel")

plt.legend() plt.show(block=False)
```

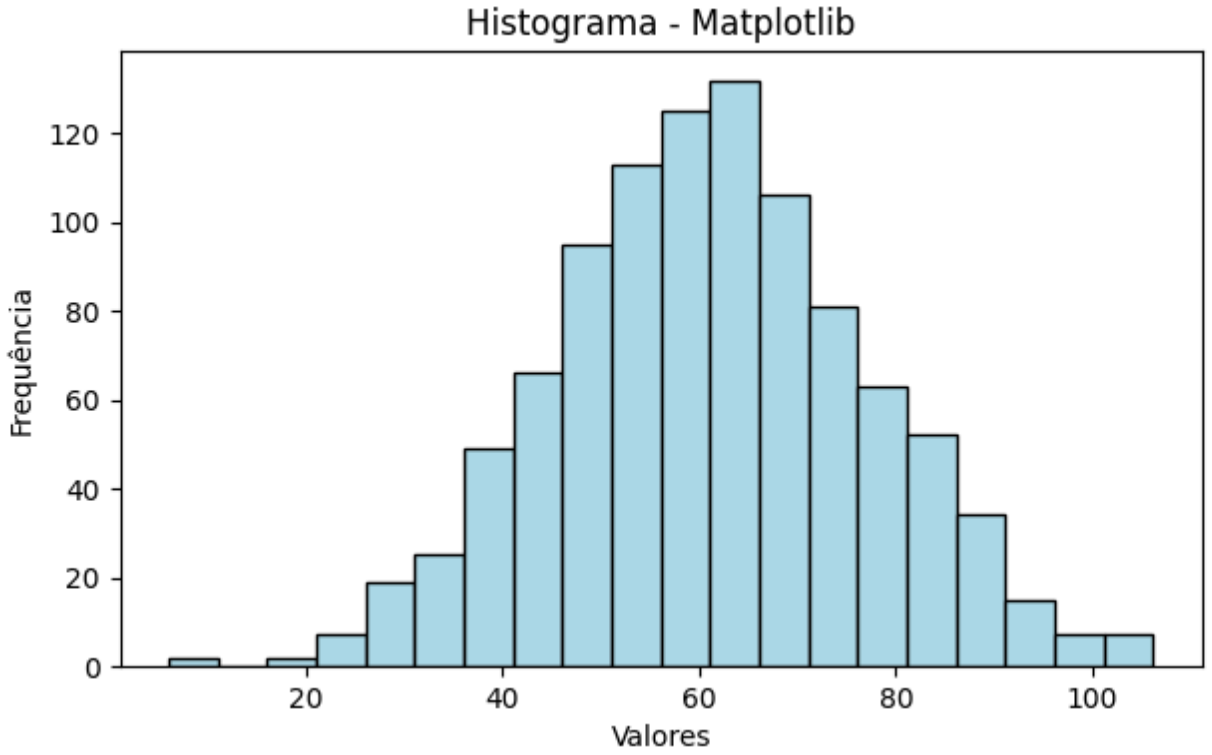


Capítulo 5 – Visualizações com Matplotlib e Seaborn

11.Histograma (Matplotlib e Seaborn)
Gere 1000 números aleatórios com distribuição normal (média 60, desvio padrão 15).

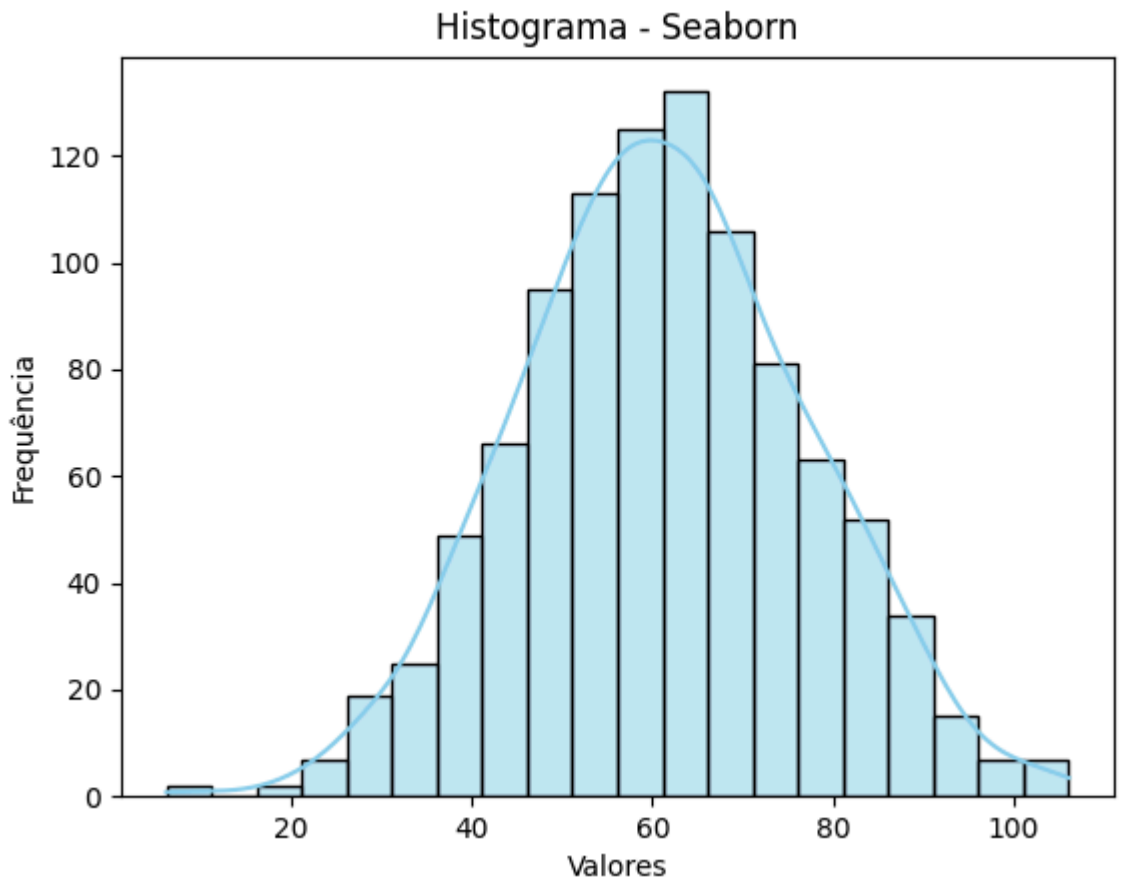
```
# 11. Histograma com Matplotlib e Seaborn

dados_normais = np.random.normal(60, 15, 1000)



plt.figure(figsize=(7,4))

plt.hist(dados_normais, bins=20, color='lightblue', edgecolor='black')

plt.title("Histograma - Matplotlib")

plt.xlabel("Valores")

plt.ylabel("Frequência")

plt.show(block=False)
```

- Plote um histograma com **matplotlib**.



- Plote o mesmo histograma com **seaborn**.

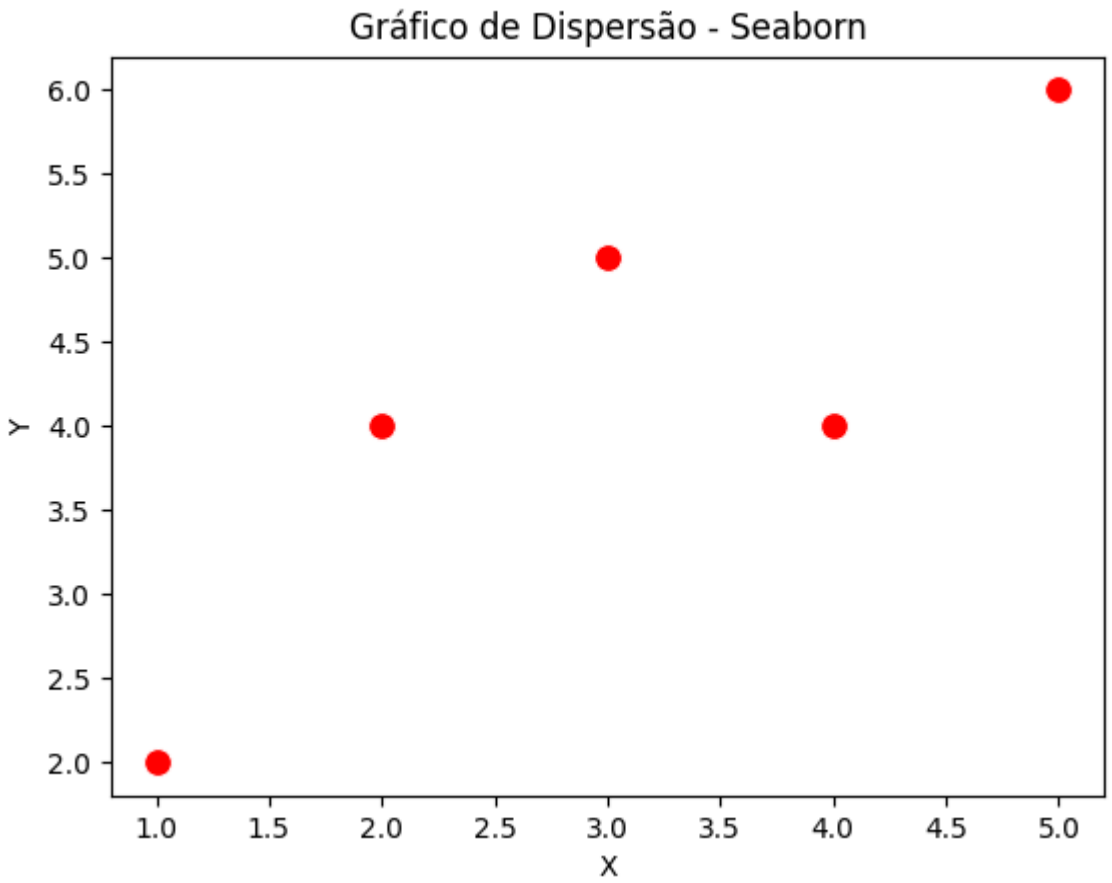
```
sns.histplot(dados_normais, bins=20, kde=True, color='skyblue')

plt.title("Histograma - Seaborn")

plt.xlabel("Valores")

plt.ylabel("Frequência")

plt.show(block=False)
```



12.Gráfico de dispersão (Seaborn)
Considere os dados:

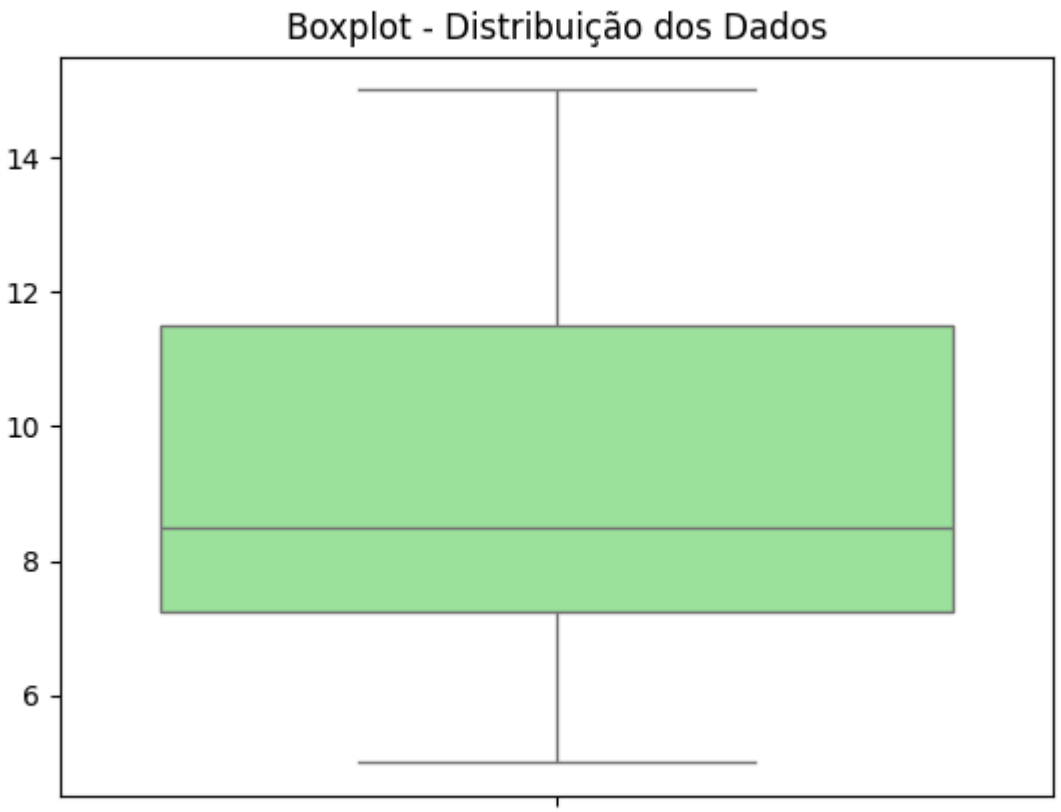
- X = [1, 2, 3, 4, 5]
- Y = [2, 4, 5, 4, 6]
  Plote o gráfico de dispersão (scatter plot) usando **seaborn**.

```
# 12. Gráfico de dispersão (Seaborn)

X = [1, 2, 3, 4, 5]

Y = [2, 4, 5, 4, 6]

sns.scatterplot(x=X, y=Y, color='red', s=100)

plt.title("Gráfico de Dispersão - Seaborn")

plt.xlabel("X")

plt.ylabel("Y")

plt.show(block=False)
```



13.Boxplot (Seaborn)
Construa um **boxplot** para os dados: [7, 8, 5, 6, 12, 14, 15, 8, 9, 10] com **seaborn**.

```
# 13. Boxplot (Seaborn)

dados_box = [7, 8, 5, 6, 12, 14, 15, 8, 9, 10]

sns.boxplot(data=dados_box, color='lightgreen')

plt.title("Boxplot - Distribuição dos Dados")

plt.show(block=False)
```


Boxplot - Distribuição dos Dados
14
12
10
8
6

📝 Exercícios – CSV “industria.csv”

Instruções:

- Carregue o arquivo `industria.csv` no Python usando `pandas`.
- Realize os cálculos solicitados e gere os gráficos pedidos usando `pandas` e `seaborn`.
- Use `groupby` para agrupar os dados quando necessário.

```
# EXERCÍCIO 14 – ANÁLISE DO ARQUIVO industria.csv



# Carregar o arquivo CSV

df = pd.read_csv("industria.csv")



print("\nPrimeiras linhas do arquivo:")

print(df.head())



# Primeiras linhas do arquivo:

#          Data    Fabrica Produto  Quantidade_Produzida  Quantidade_Vendida  Receita
Custo

# 0  2025-01-10  Fabrica A   Motor                   120                 100    50000
32000

# 1  2025-01-10  Fabrica B  Painel                    80                  75    25000
15000

# 2  2025-01-15  Fabrica A   Motor                   140                 130    65000
40000

# 3  2025-02-05  Fabrica C    Roda                   200                 180    36000
21000

# 4  2025-02-12  Fabrica B  Painel                    95                  90    30000
18000



print("\nInformações gerais:")

print(df.info())
```

```
# Informações gerais:
# <class 'pandas.core.frame.DataFrame'>
# RangeIndex: 13 entries, 0 to 12
# Data columns (total 7 columns):
#  #   Column               Non-Null Count  Dtype
# ---  ------               --------------  -----
#  0   Data                 13 non-null     object
#  1   Fabrica              13 non-null     object
#  2   Produto              13 non-null     object
#  3   Quantidade_Produzida  13 non-null     int64
#  4   Quantidade_Vendida    13 non-null     int64
#  5   Receita              13 non-null     int64
#  6   Custo                13 non-null     int64
```

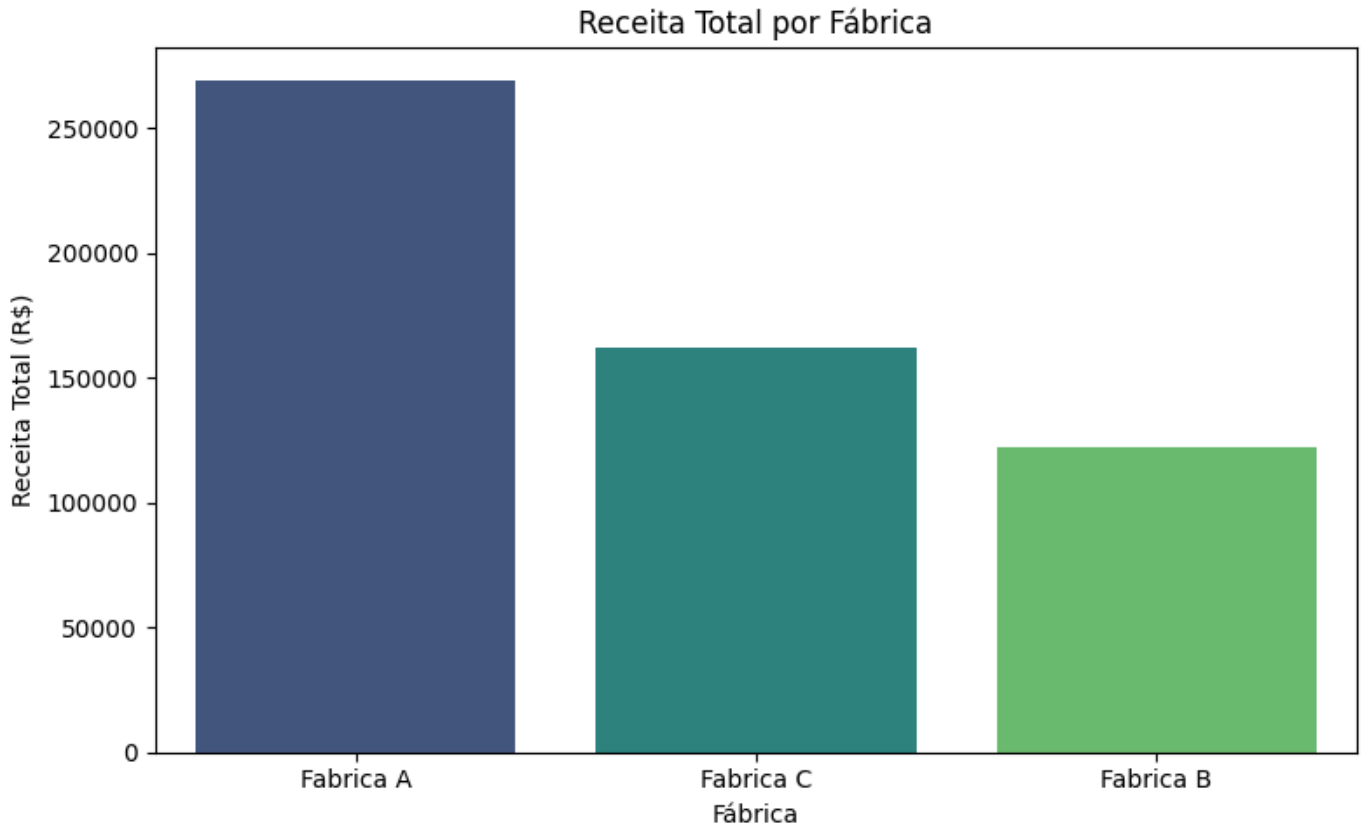
Exercício 1 – Receita total por fábrica

1. Calcule a **receita total** de cada fábrica.

```
# ================================================================

# EXERCÍCIO 1 – Receita total por fábrica

# ================================================================



receita_fabrica = df.groupby("Fabrica")["Receita"].sum().sort_values(ascending=False)

print("\nReceita total por fábrica:\n", receita_fabrica, "\n")



# Receita total por fábrica:

#  Fabrica

# Fabrica A    269000

# Fabrica C    162000

# Fabrica B    122000
```

2. Gere um **gráfico de barras** mostrando a receita total por fábrica.

```
# Gráfico de barras
plt.figure(figsize=(8,5))
sns.barplot(x=receita_fabrica.index, y=receita_fabrica.values, hue=receita_fabrica.index,
palette="viridis", legend=False)
plt.title("Receita Total por Fábrica")
plt.ylabel("Receita Total (R$)")
plt.xlabel("Fábrica")
plt.tight_layout()
plt.show(block=False)
```



3. Perguntas:
    - Qual fábrica teve a maior receita?
    - Qual a diferença entre a fábrica com maior receita e a de menor receita?

```
# Perguntas:

fabrica_maior = receita_fabrica.idxmax()

fabrica_menor = receita_fabrica.idxmin()

dif_receita = receita_fabrica.max() - receita_fabrica.min()

print(f"Fábrica com maior receita: {fabrica_maior}\n")

# Fábrica com maior receita: Fabrica A

print(f"Diferença entre maior e menor receita: R$ {dif_receita:,.2f}\n")

# Diferença entre maior e menor receita: R$ 147,000.00
```

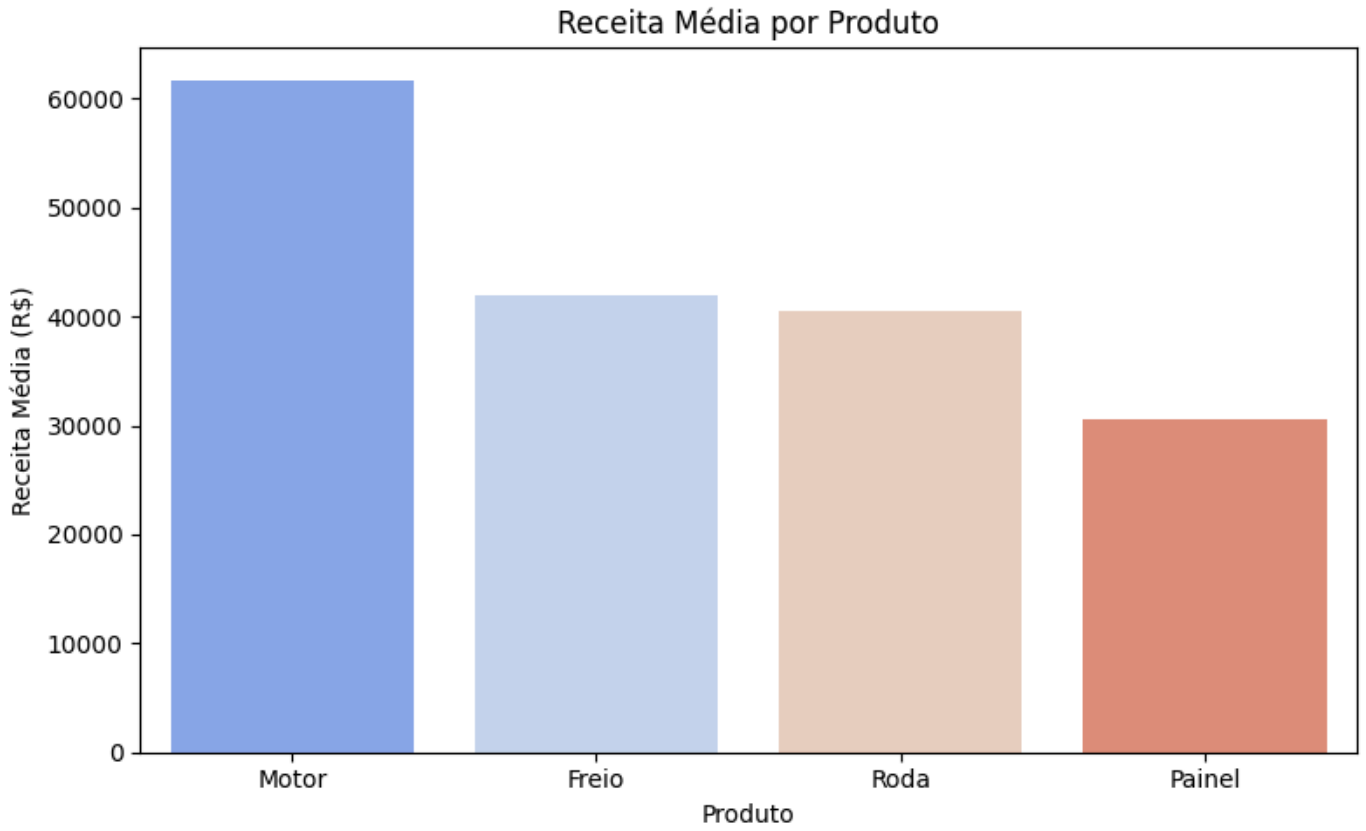
Exercício 2 – Receita média por produto

1. Calcule a **receita média** de cada produto.

```
# ================================================================

# EXERCÍCIO 2 – Receita média por produto

# ================================================================



receita_media_produto =
df.groupby("Produto")["Receita"].mean().sort_values(ascending=False)

print("\nReceita média por produto:\n", receita_media_produto, "\n")



# Receita média por produto:

#  Produto

# Motor     61666.666667

# Freio     42000.000000

# Roda      40500.000000

# Painel    30500.000000
```

2. Gere um **gráfico de barras** mostrando a receita média por produto.

```
plt.figure(figsize=(8,5))
sns.barplot(x=receita_media_produto.index, y=receita_media_produto.values,
            hue=receita_media_produto.index, palette="coolwarm", legend=False)
plt.title("Receita Média por Produto")
plt.ylabel("Receita Média (R$)")
plt.xlabel("Produto")
plt.tight_layout()
plt.show(block=False)
```



3. Perguntas:
    - Qual produto tem a maior receita média?

```
print(f"Produto com maior receita média: {receita_media_produto.idxmax()}\n")
# Produto com maior receita média: Motor
```

    - Algum produto apresenta receita média significativamente menor? Qual?

```
# Análise de receita significativamente menor

media_geral = receita_media_produto.mean()

limite_baixo = 0.75 * media_geral  # 75% da média geral

produtos_baixos = receita_media_produto[receita_media_produto < limite_baixo]

if not produtos_baixos.empty:

    print("\nProduto(s) com receita média significativamente menor que a média geral:")

    for p, valor in produtos_baixos.items():

        print(f"   - {p}: R$ {valor:,.2f} (média geral = R$ {media_geral:,.2f})")

else:

    print("\nNenhum produto apresentou receita média significativamente menor.")
```

```
# Produto(s) com receita média significativamente menor que a média geral:

#   - Painel: R$ 30,500.00 (média geral = R$ 43,666.67)
```

Exercício 3 – Quantidade vendida total por mês

1. Crie uma coluna `Mes` a partir da data de cada registro.

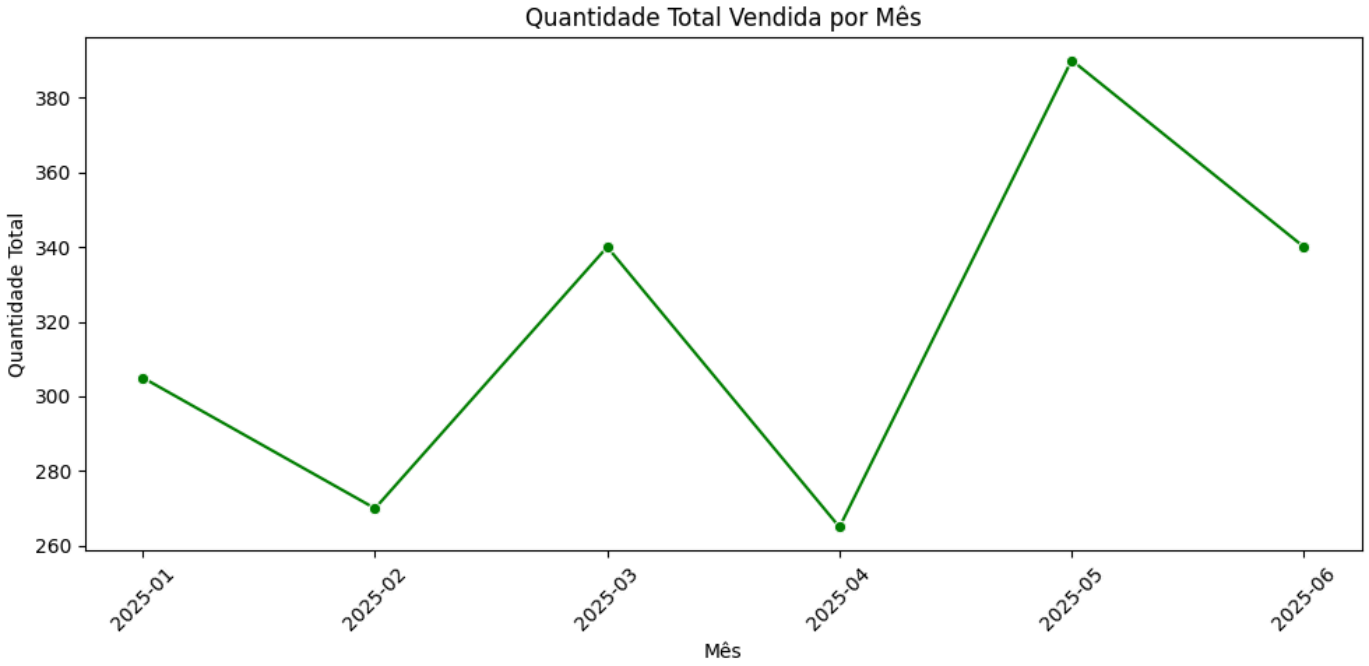
```
# ================================================================

# EXERCÍCIO 3 – Quantidade vendida total por mês

# ================================================================

# Converter datas

df["Data"] = pd.to_datetime(df["Data"], dayfirst=True, errors="coerce", format="mixed")

df["Mes"] = df["Data"].dt.to_period("M").astype(str)
```

2. Calcule a **quantidade total vendida** por mês.

```
# Agrupar pela quantidade vendida
vendas_mes = df.groupby("Mes")["Quantidade_Vendida"].sum()
print("\nQuantidade total vendida por mês:\n", vendas_mes)

# Quantidade total vendida por mês:
#  Mes
# 2025-01    305
# 2025-02    270
# 2025-03    340
# 2025-04    265
# 2025-05    390
# 2025-06    340
```

3. Gere um **gráfico de linha** mostrando a evolução da quantidade vendida ao longo dos meses.

```
plt.figure(figsize=(10,5))
sns.lineplot(x=vendas_mes.index, y=vendas_mes.values, marker="o", color="green")
plt.title("Quantidade Total Vendida por Mês")
plt.ylabel("Quantidade Total Vendida")
plt.xlabel("Mês")
plt.xticks(rotation=45)
plt.tight_layout()
plt.show(block=False)
```



4. Perguntas:
    - Qual mês teve a maior quantidade vendida?

```
mes_mais_vendas = vendas_mes.idxmax()
print(f"\nMês com maior quantidade vendida: {mes_mais_vendas}\n")
# Mês com maior quantidade vendida: 2025-05
```

    - Existe tendência de aumento ou diminuição ao longo do período?

```
# Análise de tendência

if len(vendas_mes) > 1:

    # Regressão simples para verificar tendência

    X_tempo = np.arange(len(vendas_mes)).reshape(-1, 1)

    y_vendas = vendas_mes.values

    modelo_tendencia = LinearRegression().fit(X_tempo, y_vendas)

    inclinacao = modelo_tendencia.coef_[0]

    if inclinacao > 0:

        print("Tendência observada: aumento das vendas ao longo do tempo.\n")

    elif inclinacao < 0:

        print("Tendência observada: diminuição das vendas ao longo do tempo.\n")

    else:

        print("Não há tendência significativa de variação nas vendas.\n")
```

```
else:

    print("Dados insuficientes para analisar tendência temporal.\n")

# Tendência observada: aumento das vendas ao longo do tempo.
```

Exercício 4 – Lucro médio por fábrica

1. Crie uma coluna `Lucro = Receita - Custo`.

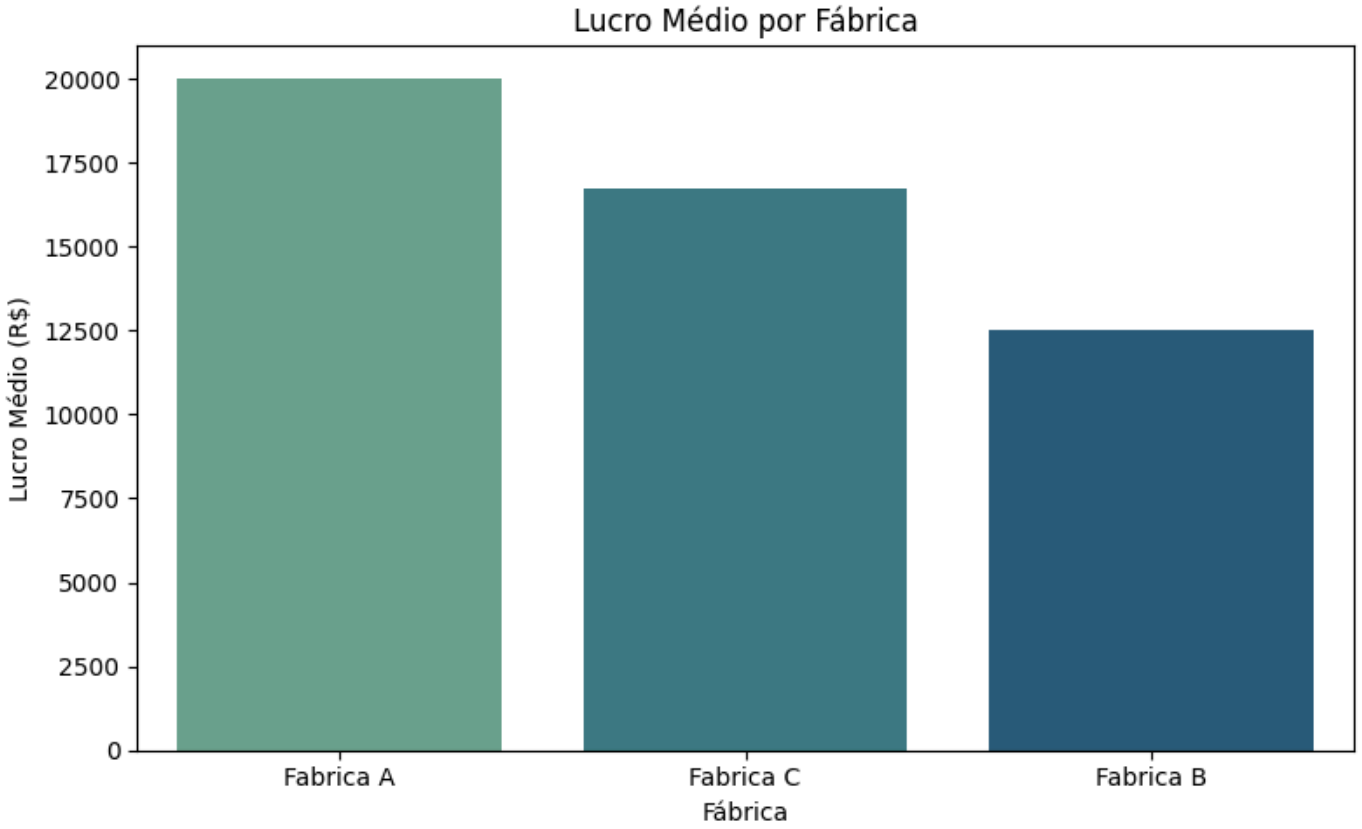
```
df["Lucro"] = df["Receita"] - df["Custo"]
```

2. Calcule o **lucro médio por fábrica** usando `groupby`.

```
lucro_medio = df.groupby("Fabrica")["Lucro"].mean().sort_values(ascending=False)
print("\nLucro médio por fábrica:\n", lucro_medio)

# Lucro médio por fábrica:
#  Fabrica
# Fabrica A    20000.0
# Fabrica C    16750.0
# Fabrica B    12500.0
```

3. Gere um **gráfico de barras** usando `seaborn` mostrando o lucro médio por fábrica.

```
plt.figure(figsize=(8,5))
sns.barplot(x=lucro_medio.index, y=lucro_medio.values, hue=lucro_medio.index,
palette="crest", legend=False)
plt.title("Lucro Médio por Fábrica")
plt.ylabel("Lucro Médio (R$)")
plt.xlabel("Fábrica")
plt.tight_layout()
plt.show(block=False)
```



4. Perguntas:
    - Qual fábrica é mais lucrativa em média?

```
fabrica_lucrativa = lucro_medio.idxmax()
print(f"Fábrica mais lucrativa: {fabrica_lucrativa}\n")
# Fábrica mais lucrativa: Fabrica A
```

    - Alguma fábrica apresenta lucro negativo em algum registro?

```
if (df["Lucro"] < 0).any():

    print("Existe pelo menos um registro com lucro negativo.\n")

else:

    print("Nenhuma fábrica apresentou lucro negativo.\n")



# Nenhuma fábrica apresentou lucro negativo.
```

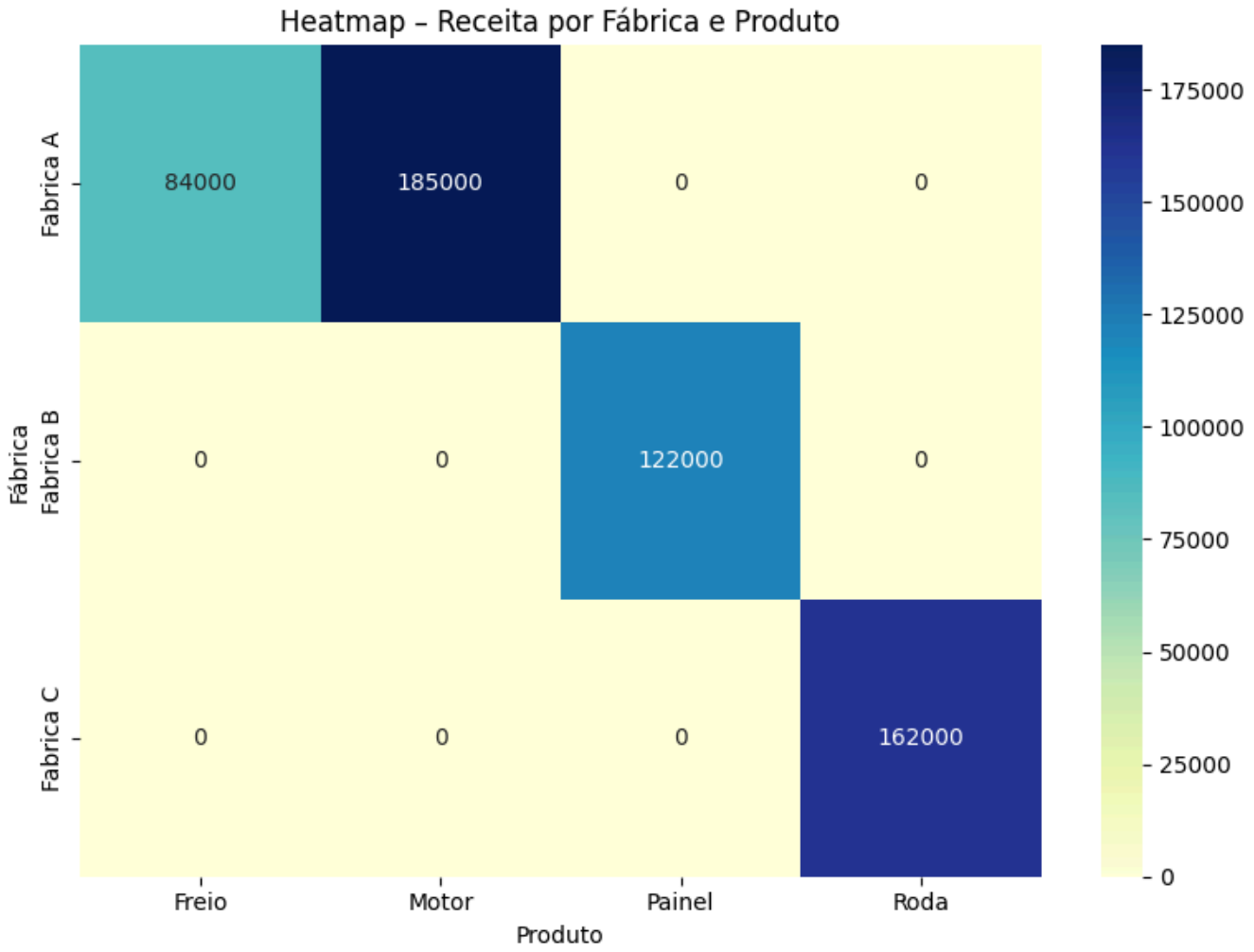
Exercício 5 – Receita total por fábrica e produto

1. Use `groupby` ou `pivot_table` para calcular a **receita total** para cada combinação de fábrica e produto.

```
# ================================================================

# EXERCÍCIO 5 - Receita total por fábrica e produto

# ================================================================



tabela_receita = df.pivot_table(values="Receita", index="Fabrica", columns="Produto",
aggfunc="sum", fill_value=0)

print("\nReceita total por fábrica e produto:\n", tabela_receita, "\n")



# Receita total por fábrica e produto:

#  Produto    Freio   Motor  Painel    Roda

# Fabrica

# Fabrica A  84000  185000       0       0

# Fabrica B      0       0  122000       0

# Fabrica C      0       0       0  162000
```

2. Gere um **heatmap** com `seaborn` mostrando a receita por fábrica (linhas) e produto (colunas).

```
plt.figure(figsize=(8,6))
sns.heatmap(tabela_receita, annot=True, fmt=".0f", cmap="YlGnBu")
plt.title("Heatmap - Receita por Fábrica e Produto")
plt.xlabel("Produto")
plt.ylabel("Fábrica")
plt.tight_layout()
plt.show(block=False)
```



3. Perguntas:
   - Qual produto gera mais receita em cada fábrica?

```
for fabrica in tabela_receita.index:
    produto_top = tabela_receita.loc[fabrica].idxmax()
    print(f"\n{fabrica}: produto com maior receita = {produto_top}")

# Fabrica A: produto com maior receita = Motor
# Fabrica B: produto com maior receita = Painel
# Fabrica C: produto com maior receita = Roda
```

   - Existe algum produto que não foi produzido ou vendido em alguma fábrica?

```
# Verificar combinação de todos os produtos e fábricas possíveis

fabrica_produto = pd.crosstab(df["Fabrica"], df["Produto"])



print("\nTabela de presença (1 = existe registro, 0 = não existe):")

print(fabrica_produto.applymap(lambda x: 1 if x > 0 else 0))
```

```
# Produto    Freio  Motor  Painel  Roda

# Fabrica

# Fabrica A      1      1       0     0

# Fabrica B      0      0       1     0

# Fabrica C      0      0       0     1



# Encontrar produtos ausentes em alguma fábrica

ausentes = []

for fabrica in df["Fabrica"].unique():

    produtos_fabrica = set(df.loc[df["Fabrica"] == fabrica, "Produto"])

    todos_produtos = set(df["Produto"].unique())

    faltando = todos_produtos - produtos_fabrica

    if faltando:

        ausentes.append((fabrica, ", ".join(faltando)))

if ausentes:

    print("\nExistem produtos que não foram produzidos/vendidos em certas fábricas:")

    for fab, prods in ausentes:

        print(f" - {fab}: não produziu/vendeu {prods}")

else:

    print("\nTodas as fábricas possuem registros para todos os produtos.")

# Existem produtos que não foram produzidos/vendidos em certas fábricas:

#  - Fabrica A: não produziu/vendeu Painel, Roda

#  - Fabrica B: não produziu/vendeu Freio, Motor, Roda

#  - Fabrica C: não produziu/vendeu Painel, Freio, Motor
```

===== CAPÍTULO 1 – PROBABILIDADE =====

Probabilidade de tirar 1 vermelha: 0.4167 (41.67%)

Duas vermelhas (com reposição): 0.1736 (17.36%)

Duas vermelhas (sem reposição): 0.1515 (15.15%)

Probabilidade de sair número > 3: 0.50 (50.0%)

Probabilidade de sair exatamente uma cara: 0.50 (50.0%)

Probabilidade de tirar vermelha e depois verde (sem reposição): 0.30

===== CAPÍTULO 2 – ESTATÍSTICA DESCRITIVA =====

Média: 22.142857142857142

Mediana: 20.0

Moda: 20

Variância: 4.571428571428571

Desvio padrão: 2.138089935299395

Média: 8.857142857142858

Mediana: 8.0

Valor mínimo: 5

Valor máximo: 15

Amplitude: 10

===== CAPÍTULO 3 – FUNÇÕES TRIGONOMÉTRICAS E LOGARÍTMICAS =====

===== CAPÍTULO 4 – REGRESSÃO LINEAR =====

Coeficiente angular (inclinação): 0.80

Intercepto: 1.80

Preço estimado para imóvel de 100 m²: 306.00 mil reais

===== CAPÍTULO 5 – VISUALIZAÇÕES =====

Primeiras linhas do arquivo:

	Data	Fabrica	Produto	Quantidade_Produzida	Quantidade_Vendida	Receita	Custo
0	2025-01-10	Fabrica A	Motor	120	100	50000	32000
1	2025-01-10	Fabrica B	Painel	80	75	25000	15000
2	2025-01-15	Fabrica A	Motor	140	130	65000	40000
3	2025-02-05	Fabrica C	Roda	200	180	36000	21000
4	2025-02-12	Fabrica B	Painel	95	90	30000	18000

Informações gerais:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 13 entries, 0 to 12
Data columns (total 7 columns):

#	Column	Non-Null Count	Dtype
0	Data	13 non-null	object
1	Fabrica	13 non-null	object
2	Produto	13 non-null	object
3	Quantidade_Produzida	13 non-null	int64
4	Quantidade_Vendida	13 non-null	int64
5	Receita	13 non-null	int64
6	Custo	13 non-null	int64

dtypes: int64(4), object(3)
memory usage: 924.0+ bytes
None

Receita total por fábrica:
Fabrica
Fabrica A 269000
Fabrica C 162000
Fabrica B 122000
Name: Receita, dtype: int64

Fábrica com maior receita: Fabrica A

Diferença entre maior e menor receita: R$ 147,000.00

Receita média por produto:
Produto
Motor 61666.666667
Freio 42000.000000
Roda 40500.000000
Painel 30500.000000
Name: Receita, dtype: float64

Produto com maior receita média: Motor

Produto(s) com receita média significativamente menor que a média geral:
- Painel: R$ 30,500.00 (média geral = R$ 43,666.67)

Quantidade total vendida por mês:
Mes
2025-01 305
2025-02 270
2025-03 340
2025-04 265
2025-05 390
2025-06 340
Name: Quantidade_Vendida, dtype: int64

Mês com maior quantidade vendida: 2025-05

Tendência observada: aumento das vendas ao longo do tempo.

Lucro médio por fábrica:
Fabrica
Fabrica A 20000.0
Fabrica C 16750.0
Fabrica B 12500.0
Name: Lucro, dtype: float64
Fábrica mais lucrativa: Fabrica A

Nenhuma fábrica apresentou lucro negativo.

Receita total por fábrica e produto:

Fabrica	Freio	Motor	Painel	Roda
Fabrica A	84000	185000	0	0
Fabrica B	0	0	122000	0
Fabrica C	0	0	0	162000

Pressione Enter para fechar tudo...

Fabrica A: produto com maior receita = Motor

Fabrica B: produto com maior receita = Painel

Fabrica C: produto com maior receita = Roda

Tabela de presença (1 = existe registro, 0 = não existe):

Fabrica	Freio	Motor	Painel	Roda
Fabrica A	1	1	0	0
Fabrica B	0	0	1	0
Fabrica C	0	0	0	1

Existem produtos que não foram produzidos/vendidos em certas fábricas:
- Fabrica A: não produziu/vendeu Painel, Roda
- Fabrica B: não produziu/vendeu Freio, Motor, Roda
- Fabrica C: não produziu/vendeu Painel, Freio, Motor

Análise concluída com sucesso!