NOTEBOOK DESTINADO A REGISTRAR AS TAREFAS DA DISCIPLINA DE AEDI - 1º/2025

PROFESSOR: JOÃO GABRIEL DE MORAES SOUZA

ALUNO: GUSTAVO PARREIRA LIMA CUNHA

TAREFA 2

QUESTÃO A

Escolha de 5 ativos e de um índice representativo:

Foram analisadas as carteiras recomendadas de 10 corretoras para o mês de abril/2025, e os ativos abaixo foram os mais recomendados:

Ordenadas por número de recomendações:

1. PETR4 (Petrobras PN) Aparece 6 vezes nas carteiras.
2. ITUB4 (Itaú Unibanco PN) Aparece 6 vezes nas carteiras.
3. VALE3 (Vale ON) Aparece 4 vezes nas carteiras.
4. JBSS3 (JBS ON) Aparece 4 vezes nas carteiras.
5. SBSP3 (Sabesp ON) Aparece 4 vezes nas carteiras.

As corretoras analisadas foram:

1. Ágora Investimentos
2. Ativa Investimentos
3. BB Investimentos
4. BTG Pactual
5. CM Capital
6. EQI Investimentos
7. Genial
8. Itaú
9. Planner Corretora
10. Santander Corretora

Dessa forma, os ativos foram escolhidos pela relevância que possuem no cenário de renda variável atual, o que inclui também a expectativa de bons rendimentos futuros. A análise dos referidos ativos é intressante pois será possível acompanhar, no transcorrer do tempo, se o cenário positivo de recomendação se confirmará.

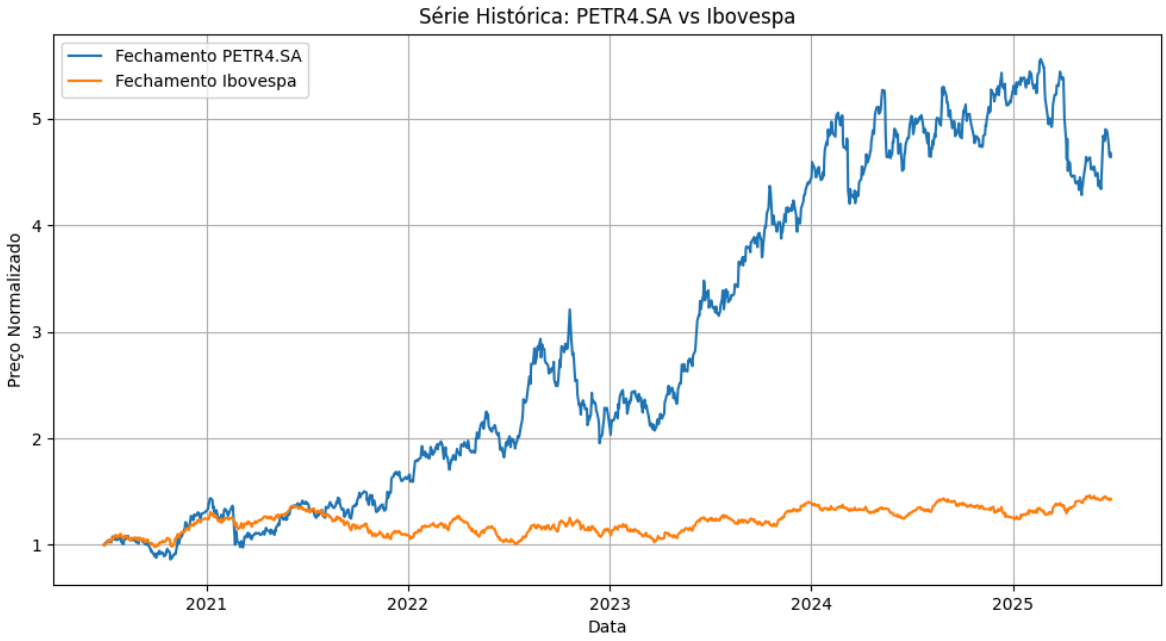
A escolha do índice representativo será o índice Bovespa, que representa uma cesta de ativos composta atualmente por 84 ações.

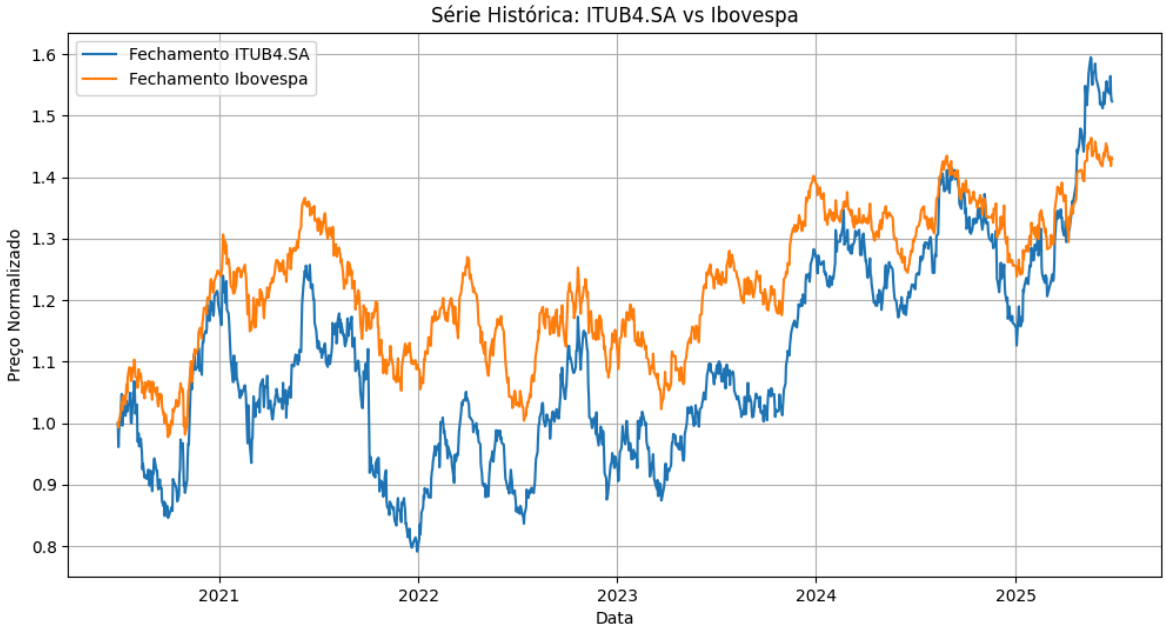
QUESTÃO B

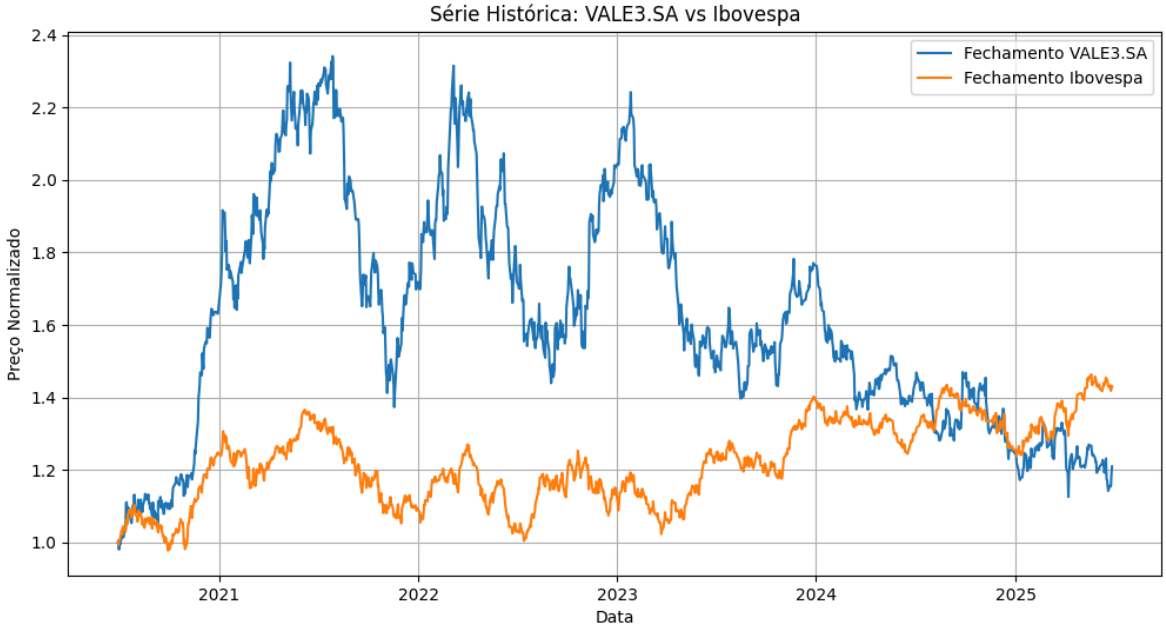
In [1]:
```
import yfinance as yf
import matplotlib.pyplot as plt
import pandas as pd


# Ativo e referência
ativo1 = "PETR4.SA"
referencia = "^BVSP"

# dados históricos de ambos
df_ativo1 = yf.Ticker(ativo1).history(period="5y")
df_ref = yf.Ticker(referencia).history(period="5y")

# Normalizar os preços para facilitar a comparação: cada DF é normalizado dividi

df_norm1 = df_ativo1 / df_ativo1.iloc[0]
df_norm2 = df_ref / df_ref.iloc[0]

# Plotar
plt.figure(figsize=(12,6))
plt.plot(df_norm1.index, df_norm1["Close"], label=f"Fechamento {ativo1}")
plt.plot(df_norm2.index, df_norm2["Close"], label="Fechamento Ibovespa")
plt.title(f"Série Histórica: {ativo1} vs Ibovespa")
plt.xlabel("Data")
plt.ylabel("Preço Normalizado")
plt.legend()
plt.grid()
plt.show()
print()


# Ativo
ativo2 = "ITUB4.SA"

# dados históricos de ambos
df_ativo2 = yf.Ticker(ativo2).history(period="5y")

# Normalizar os preços para facilitar a comparação
df_norm3 = df_ativo2 / df_ativo2.iloc[0]

# Plotar
plt.figure(figsize=(12,6))
```

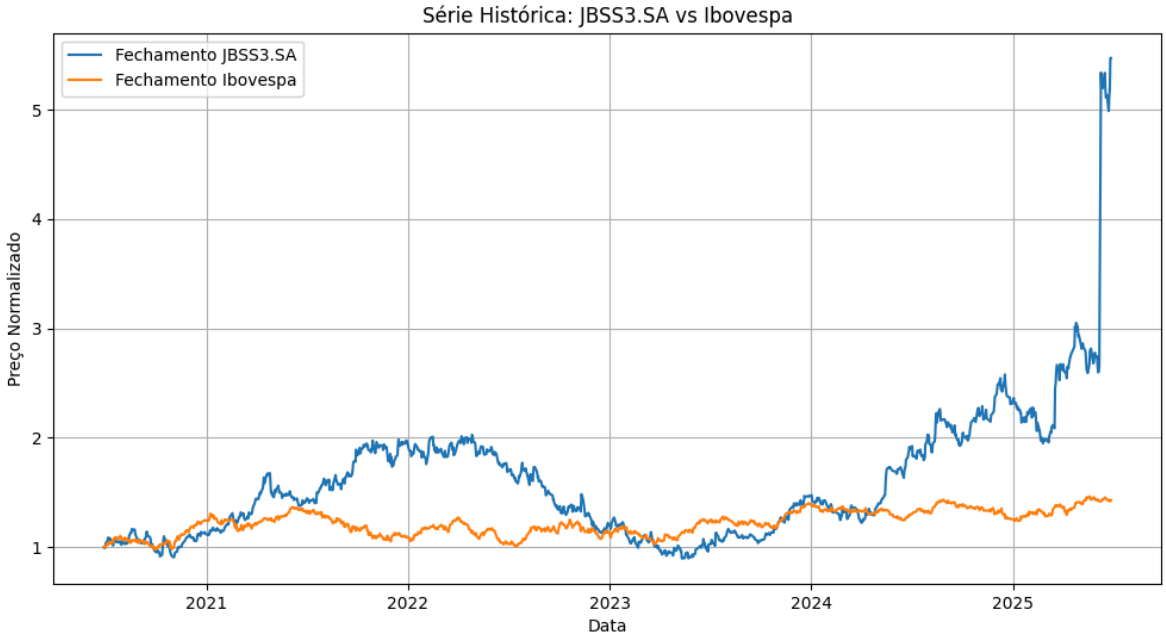
```
plt.plot(df_norm3.index, df_norm3["Close"], label=f"Fechamento {ativo2}")
plt.plot(df_norm2.index, df_norm2["Close"], label="Fechamento Ibovespa")
plt.title(f"Série Histórica: {ativo2} vs Ibovespa")
plt.xlabel("Data")
plt.ylabel("Preço Normalizado")
plt.legend()
plt.grid()
plt.show()
print()

# Ativo
ativo3 = "VALE3.SA"

# dados históricos de ambos
df_ativo3 = yf.Ticker(ativo3).history(period="5y")

# Normalizar os preços para facilitar a comparação
df_norm4 = df_ativo3 / df_ativo3.iloc[0]

# Plotar
plt.figure(figsize=(12,6))
plt.plot(df_norm4.index, df_norm4["Close"], label=f"Fechamento {ativo3}")
plt.plot(df_norm2.index, df_norm2["Close"], label="Fechamento Ibovespa")
plt.title(f"Série Histórica: {ativo3} vs Ibovespa")
plt.xlabel("Data")
plt.ylabel("Preço Normalizado")
plt.legend()
plt.grid()
plt.show()
print()


# Ativo
ativo4 = "JBSS3.SA"

# dados históricos de ambos
df_ativo4 = yf.Ticker(ativo4).history(period="5y")

# Normalizar os preços para facilitar a comparação
df_norm5 = df_ativo4 / df_ativo4.iloc[0]

# Plotar
plt.figure(figsize=(12,6))
plt.plot(df_norm5.index, df_norm5["Close"], label=f"Fechamento {ativo4}")
plt.plot(df_norm2.index, df_norm2["Close"], label="Fechamento Ibovespa")
plt.title(f"Série Histórica: {ativo4} vs Ibovespa")
plt.xlabel("Data")
plt.ylabel("Preço Normalizado")
plt.legend()
plt.grid()
plt.show()
print()


# Ativo
ativo5 = "SBSP3.SA"

# dados históricos de ambos
df_ativo5 = yf.Ticker(ativo5).history(period="5y")
```

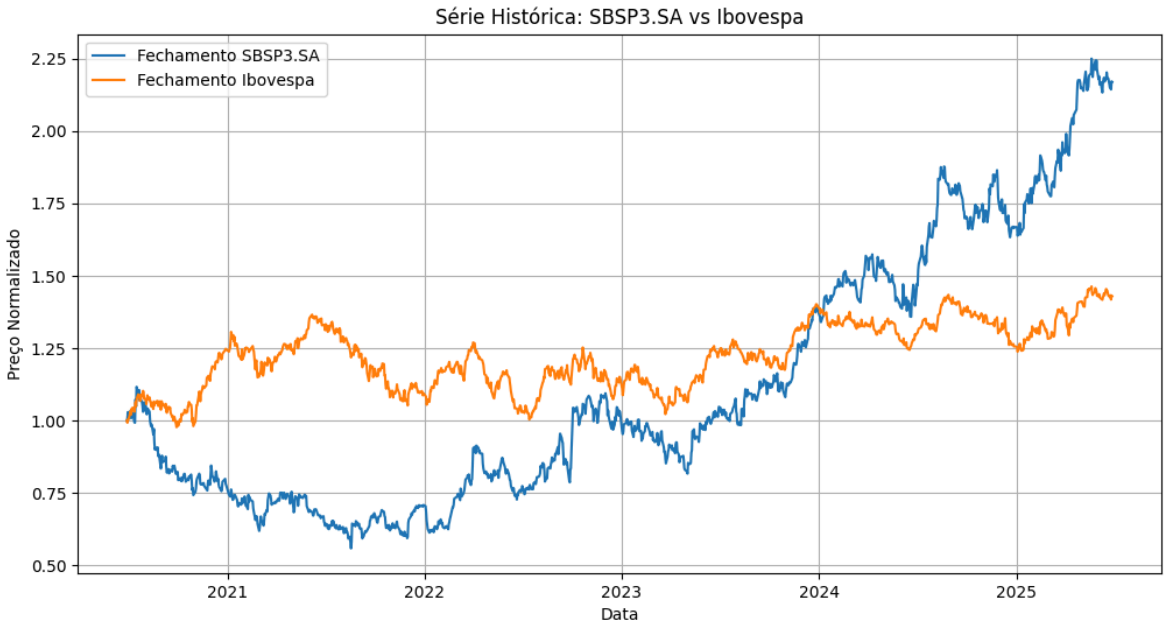
```
# Normalizar os preços para facilitar a comparação
df_norm6 = df_ativo5 / df_ativo5.iloc[0]

# Plotar
plt.figure(figsize=(12,6))
plt.plot(df_norm6.index, df_norm6["Close"], label=f"Fechamento {ativo5}")
plt.plot(df_norm2.index, df_norm2["Close"], label="Fechamento Ibovespa")
plt.title(f"Série Histórica: {ativo5} vs Ibovespa")
plt.xlabel("Data")
plt.ylabel("Preço Normalizado")
plt.legend()
plt.grid()
plt.show()
print()
```






Série Histórica: VALE3.SA vs Ibovespa
Fechamento VALE3.SA
Fechamento Ibovespa
Preço Normalizado
Data
2.4
2.2
2.0
1.8
1.6
1.4
1.2
1.0
2021
2022
2023
2024
2025


Série Histórica: JBSS3.SA vs Ibovespa
Fechamento JBSS3.SA
Fechamento Ibovespa
Preço Normalizado
Data
5
4
3
2
1
2021
2022
2023
2024
2025

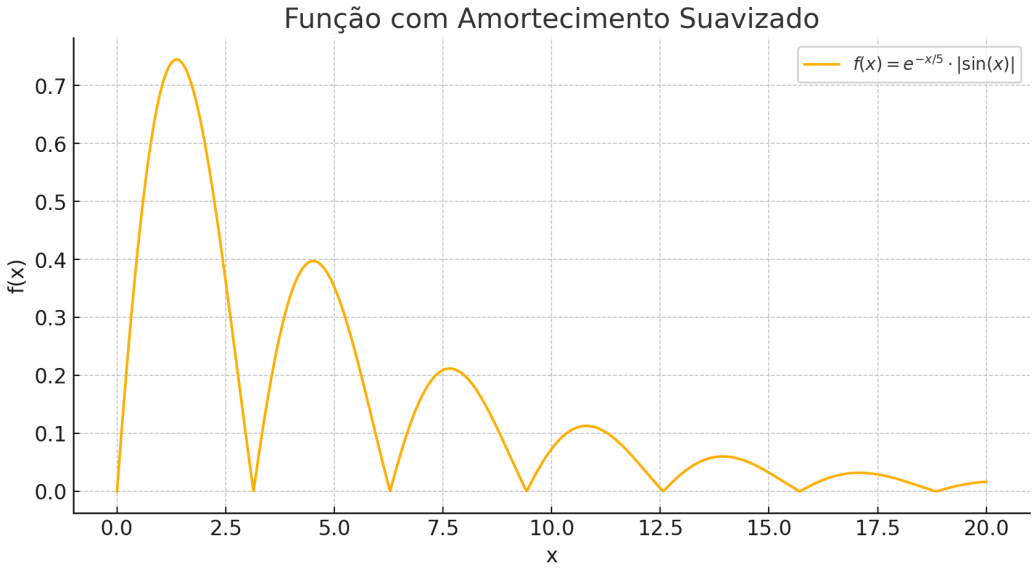

Série Histórica: SBSP3.SA vs Ibovespa
Fechamento SBSP3.SA
Fechamento Ibovespa
Preço Normalizado
Data
2.25
2.00
1.75
1.50
1.25
1.00
0.75
0.50
2021
2022
2023
2024
2025

Análises

- PETR4: Pode-se notar que, aproximadamente após a metade do ano de 2021, o ativo performou melhor que o Ibovespa, tendo obtido maior rentabilidade dentro do período de 5 anos analisado. Após o início de 2023, o ativo apresentou viés de alta sustentado por cerca de um ano, o que elevou o preço do ativo significativamente. Observa-se que o Ibovespa se comporta de forma menos volátil, com movimentos menores ao longo do tempo, se mostrando um ativo mais estável. Por fim, nota-se uma queda expressiva ao final de 2022, que mostra uma reação negativa do mercado às últimas eleições presidenciais.

- ITUB4: É possível notar que o ativo tende a acompanhar o Ibovespa, reproduzindo, de forma proporcional, os movimentos de subida e queda do índice. Comportou-se de forma lateralizada até meados de 2023 (aproximadamente 1 trimestre), até o momento em que iniciou uma tendência de alta até meados (julho) de 2024. Teve, ainda, alta expressiva no ano de 2025.

- VALE3: Apresentou alta expressiva ao final de 2020, possivelmente decorrente da recuperação do mercado após a estabilização da pandemia, tendo dobrado seu valor em um ano. Apresenta comportamento similar ao de uma senoide amortecida, do tipo



Apresenta em alguns momentos variação condizente com a variação do Ibovespa, porém ocorrendo antes da variação do referido índice.

- JBSS3: Entre metade de 2021 e o final de 2022 o ativo apresentou melhores resultados quando comparado ao Ibovespa, porém não se sustentou. Somente em meados de 2024 a rentabilidade do ativo passou a superar consistentemente à do Ibovespa, que se mostrou relativamente lateralizado no último ano. Esse ativo deixou de ser negociado na bolsa brasileira no dia 06/06/2025, devido à reestruturação societária e processo de listagem internacional do ativo.

- SBSP3: Após metade de 2020, o ativo performou pior que o Ibovespa durante longo período, até o início de 2022. Em seguida, aos poucos reduziu a distância entre as curvas, até que superou a rentabilidade do Ibovespa, após 2024. Desde meados de 2023 o ativo respeita uma linha de tendência de alta (LTA).

QUESTÃO C

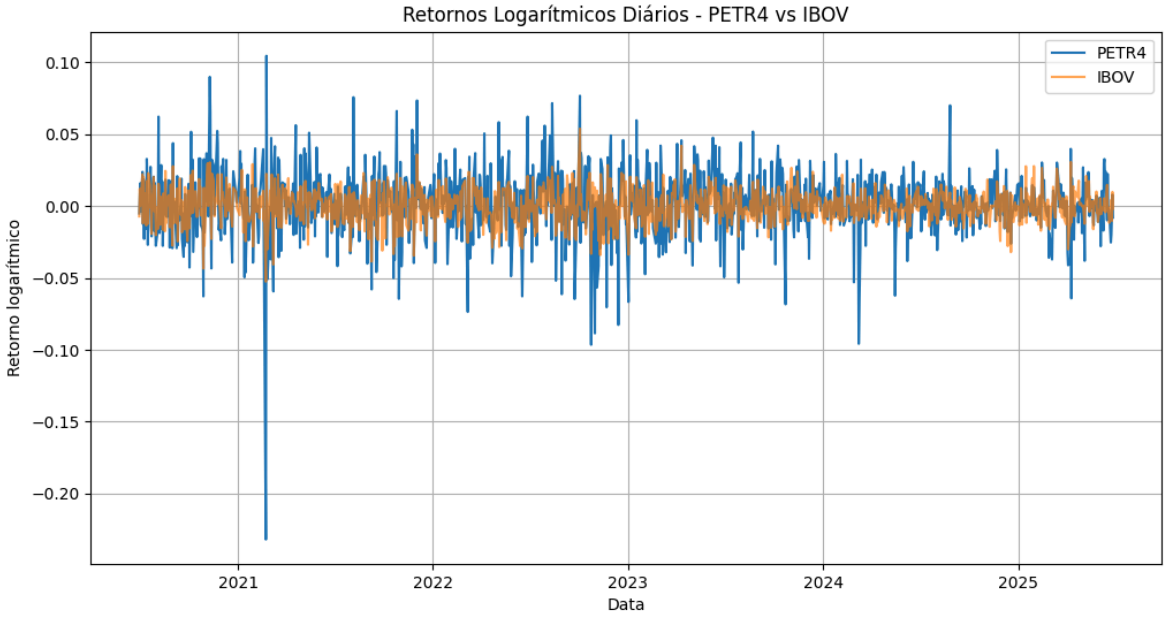
Calcular os retornos logarítmicos diários

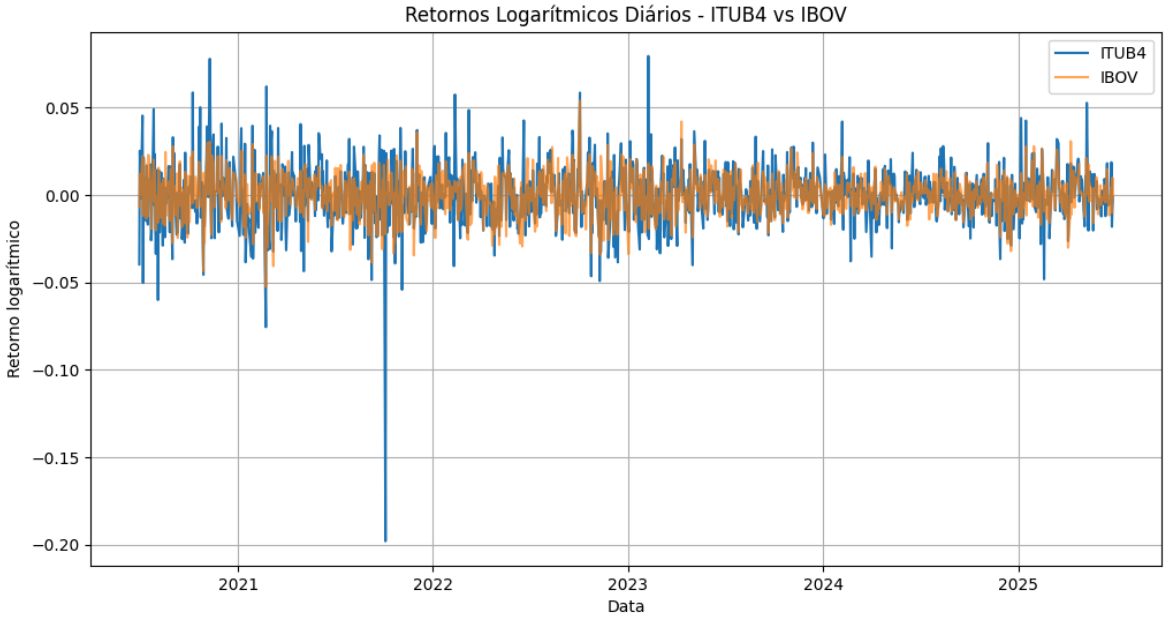
In [2]:
```
import yfinance as yf
import numpy as np
import matplotlib.pyplot as plt

# Baixar dados
ativo1 = yf.download("PETR4.SA", period="5y", auto_adjust=True)["Close"]
ativo2 = yf.download("ITUB4.SA", period="5y", auto_adjust=True)["Close"]
ativo3 = yf.download("VALE3.SA", period="5y", auto_adjust=True)["Close"]
ativo4 = yf.download("JBSS3.SA", period="5y", auto_adjust=True)["Close"]
ativo5 = yf.download("SBSP3.SA", period="5y", auto_adjust=True)["Close"]
ibov = yf.download("^BVSP", period="5y", auto_adjust=True)["Close"]

# Calcular retornos logarítmicos
ret_ativo1 = np.log(ativo1 / ativo1.shift(1))
ret_ativo2 = np.log(ativo2 / ativo2.shift(1))
ret_ativo3 = np.log(ativo3 / ativo3.shift(1))
ret_ativo4 = np.log(ativo4 / ativo4.shift(1))
ret_ativo5 = np.log(ativo5 / ativo5.shift(1))
ret_ibov = np.log(ibov / ibov.shift(1))

# Plotar
plt.figure(figsize=(12,6))
plt.plot(ret_ativo1, label="PETR4")
plt.plot(ret_ibov, label="IBOV", alpha=0.7)
plt.title("Retornos Logarítmicos Diários - PETR4 vs IBOV")
plt.xlabel("Data")
plt.ylabel("Retorno logarítmico")
plt.legend()
plt.grid()
plt.show()
print()
# Plotar
plt.figure(figsize=(12,6))
plt.plot(ret_ativo2, label="ITUB4")
plt.plot(ret_ibov, label="IBOV", alpha=0.7)
plt.title("Retornos Logarítmicos Diários - ITUB4 vs IBOV")
plt.xlabel("Data")
plt.ylabel("Retorno logarítmico")
plt.legend()
plt.grid()
plt.show()
print()
# Plotar
plt.figure(figsize=(12,6))
plt.plot(ret_ativo3, label="VALE3")
plt.plot(ret_ibov, label="IBOV", alpha=0.7)
```

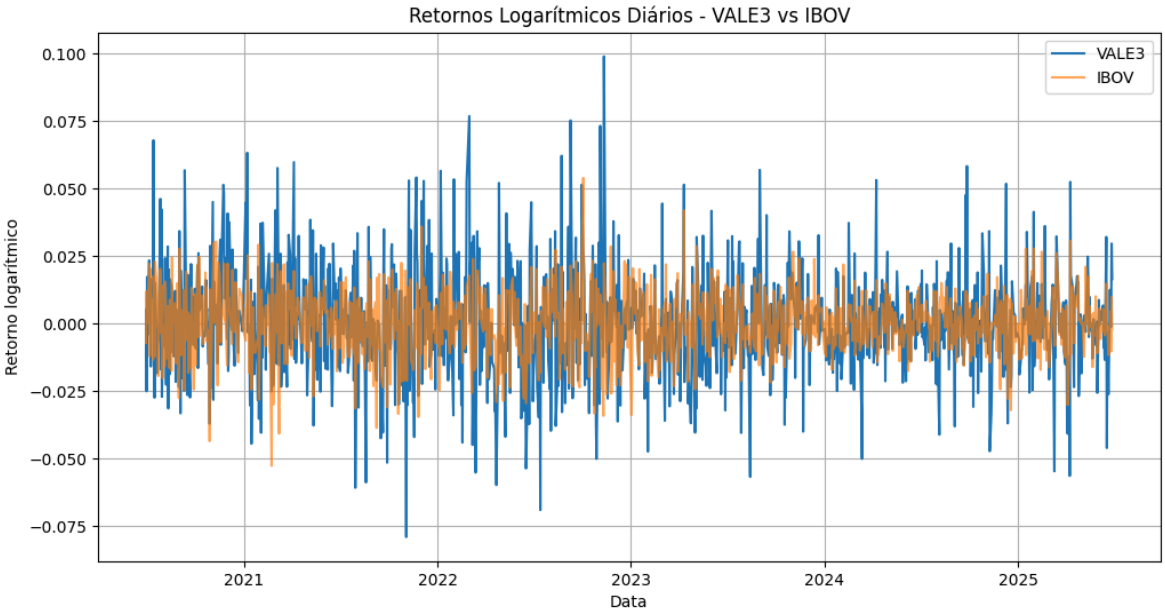
```
plt.title("Retornos Logarítmicos Diários - VALE3 vs IBOV")
plt.xlabel("Data")
plt.ylabel("Retorno logarítmico")
plt.legend()
plt.grid()
plt.show()
print()
# Plotar
plt.figure(figsize=(12,6))
plt.plot(ret_ativo4, label="JBSS3")
plt.plot(ret_ibov, label="IBOV", alpha=0.7)
plt.title("Retornos Logarítmicos Diários - JBSS3 vs IBOV")
plt.xlabel("Data")
plt.ylabel("Retorno logarítmico")
plt.legend()
plt.grid()
plt.show()
print()
# Plotar
plt.figure(figsize=(12,6))
plt.plot(ret_ativo5, label="SBSP3")
plt.plot(ret_ibov, label="IBOV", alpha=0.7)
plt.title("Retornos Logarítmicos Diários - SBSP3 vs IBOV")
plt.xlabel("Data")
plt.ylabel("Retorno logarítmico")
plt.legend()
plt.grid()
plt.show()
print()
print("DISTRIBUIÇÃO DOS RETORNOS OBTIDOS")

plt.hist(ret_ativo1.dropna(), bins=30, alpha=0.7, color='blue')
plt.title(f"Histograma dos Retornos Logarítmicos - {'PETR4'}")
plt.xlabel("Retorno logarítmico")
plt.ylabel("Frequência")
plt.grid(True)
plt.show()
print()

plt.hist(ret_ativo2.dropna(), bins=30, alpha=0.7, color='blue')
plt.title(f"Histograma dos Retornos Logarítmicos - {'ITUB4'}")
plt.xlabel("Retorno logarítmico")
plt.ylabel("Frequência")
plt.grid(True)
plt.show()
print()

plt.hist(ret_ativo3.dropna(), bins=30, alpha=0.7, color='blue')
plt.title(f"Histograma dos Retornos Logarítmicos - {'VALE3'}")
plt.xlabel("Retorno logarítmico")
plt.ylabel("Frequência")
plt.grid(True)
plt.show()
print()

plt.hist(ret_ativo4.dropna(), bins=30, alpha=0.7, color='blue')
plt.title(f"Histograma dos Retornos Logarítmicos - {'JBSS3'}")
plt.xlabel("Retorno logarítmico")
plt.ylabel("Frequência")
plt.grid(True)
```

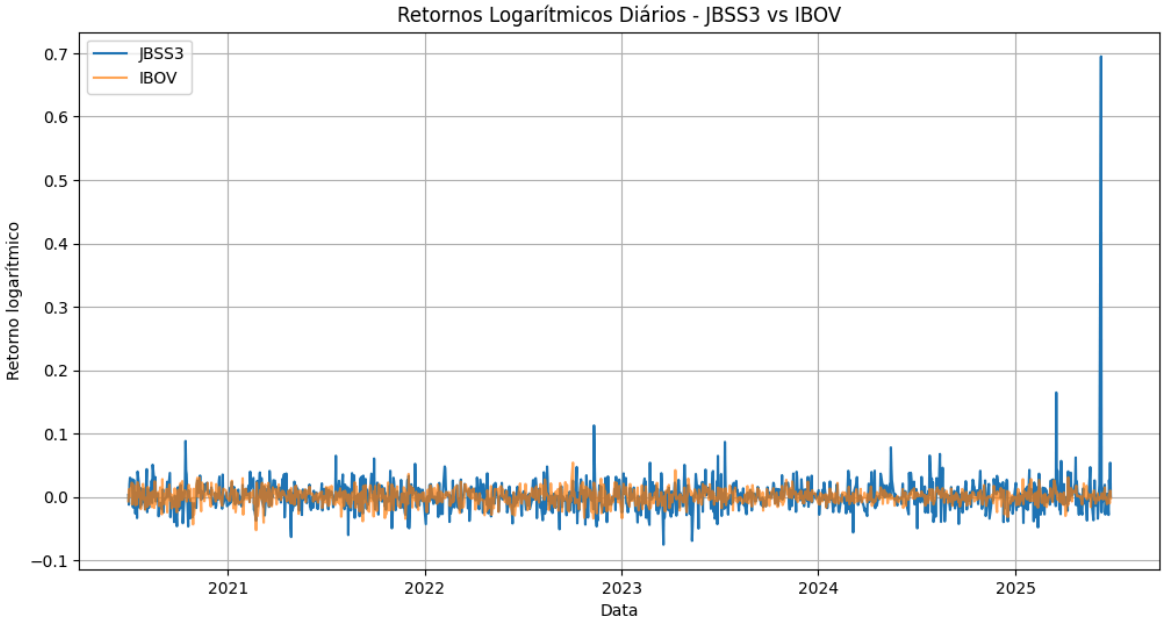
```
plt.show()
print()

plt.hist(ret_ativo5.dropna(), bins=30, alpha=0.7, color='blue')
plt.title(f"Histograma dos Retornos Logarítmicos - {'SBSP3'}")
plt.xlabel("Retorno logarítmico")
plt.ylabel("Frequência")
plt.grid(True)
plt.show()
print()
```

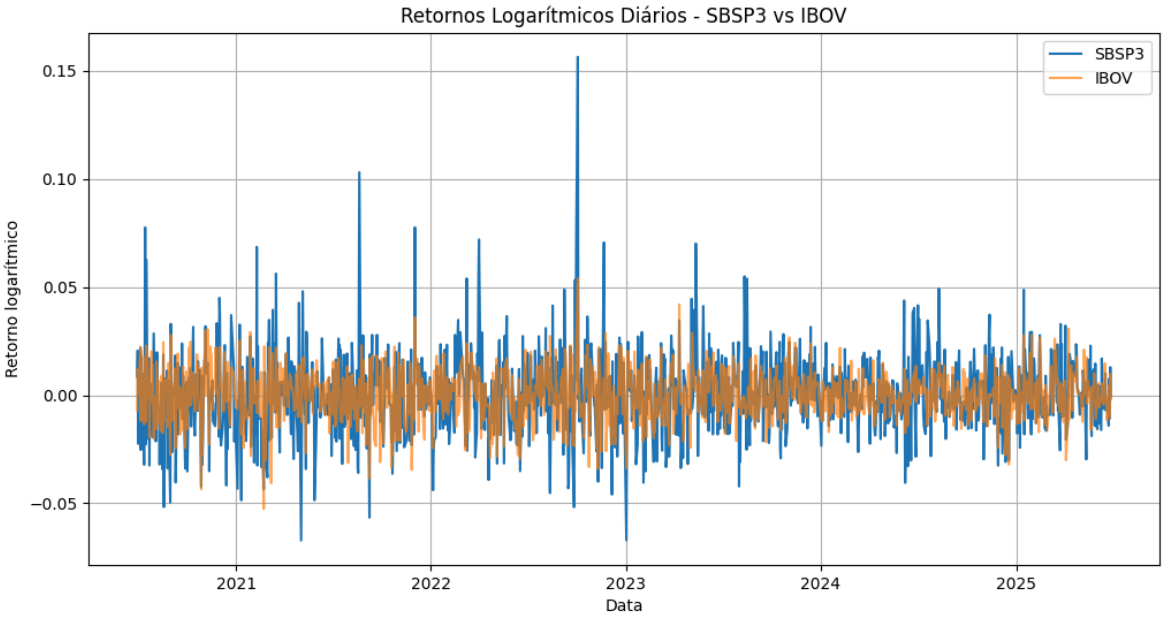
```
[**********100%**********]  1 of 1 completed
[**********100%**********]  1 of 1 completed
[**********100%**********]  1 of 1 completed
[**********100%**********]  1 of 1 completed
[**********100%**********]  1 of 1 completed
[**********100%**********]  1 of 1 completed
```



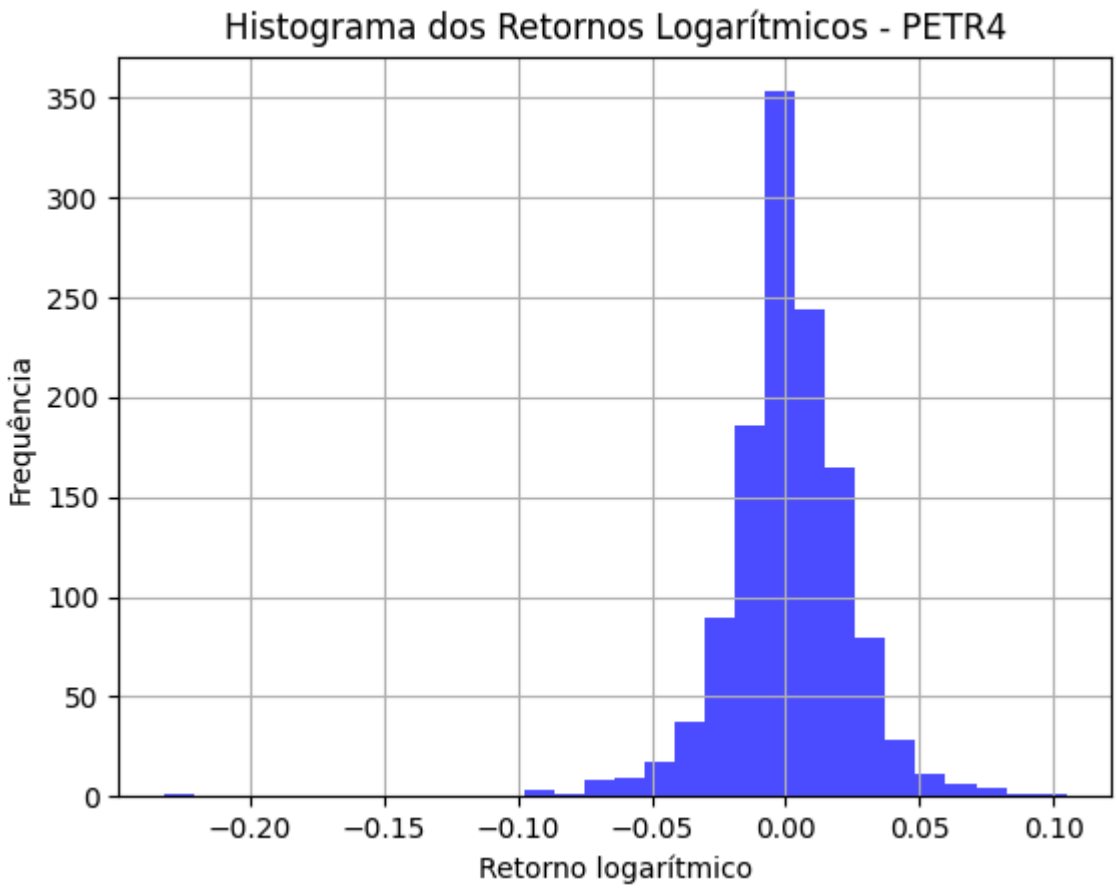


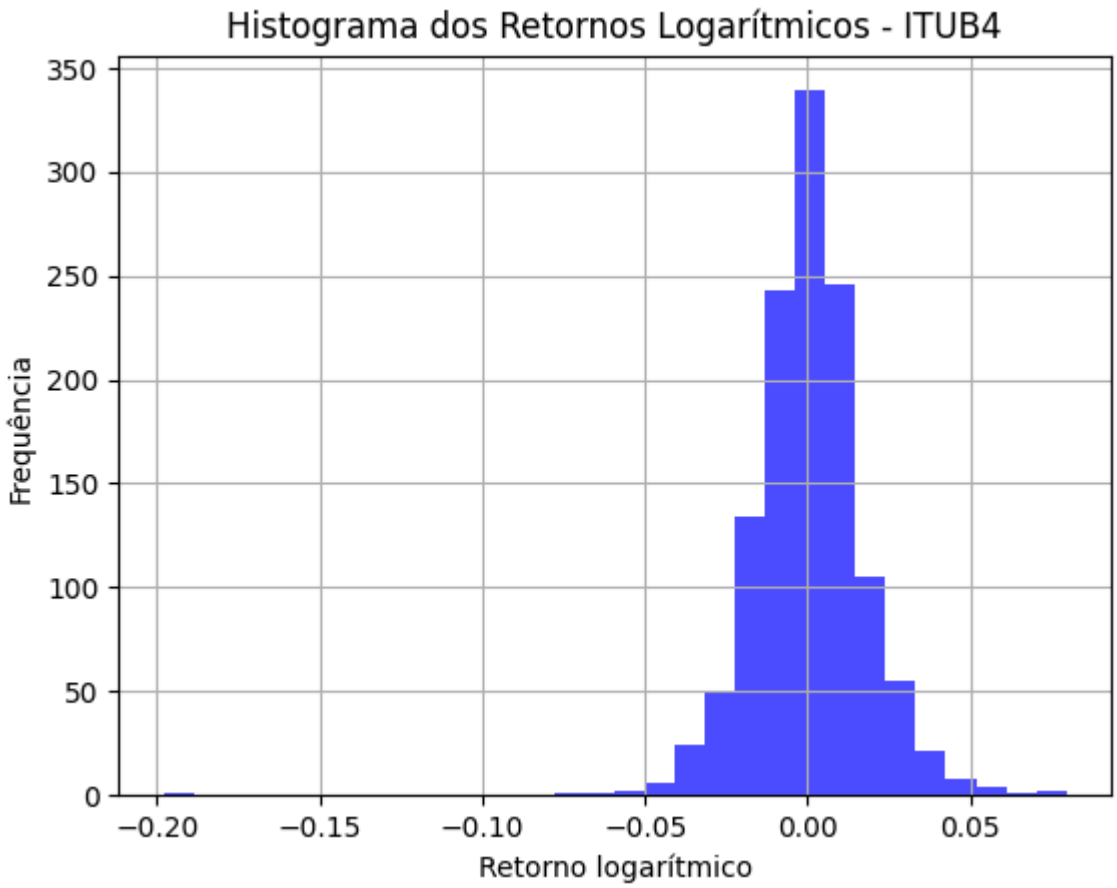


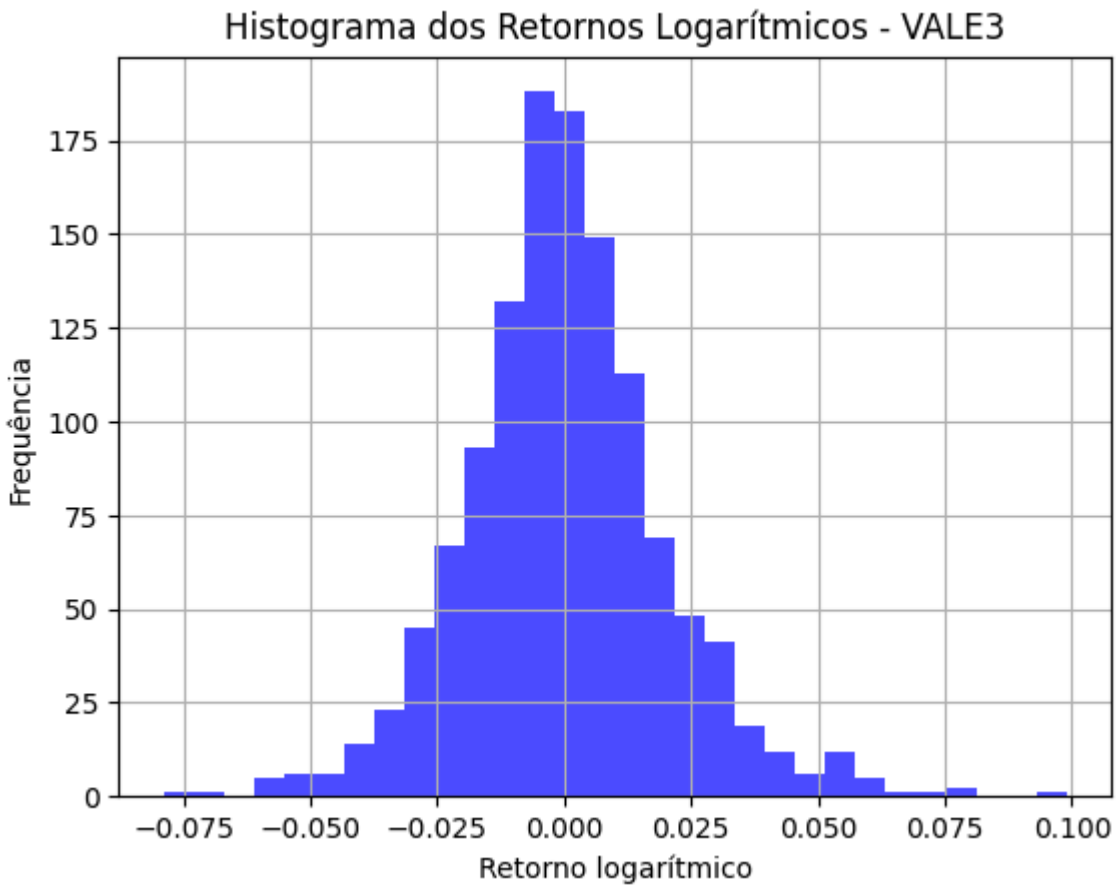


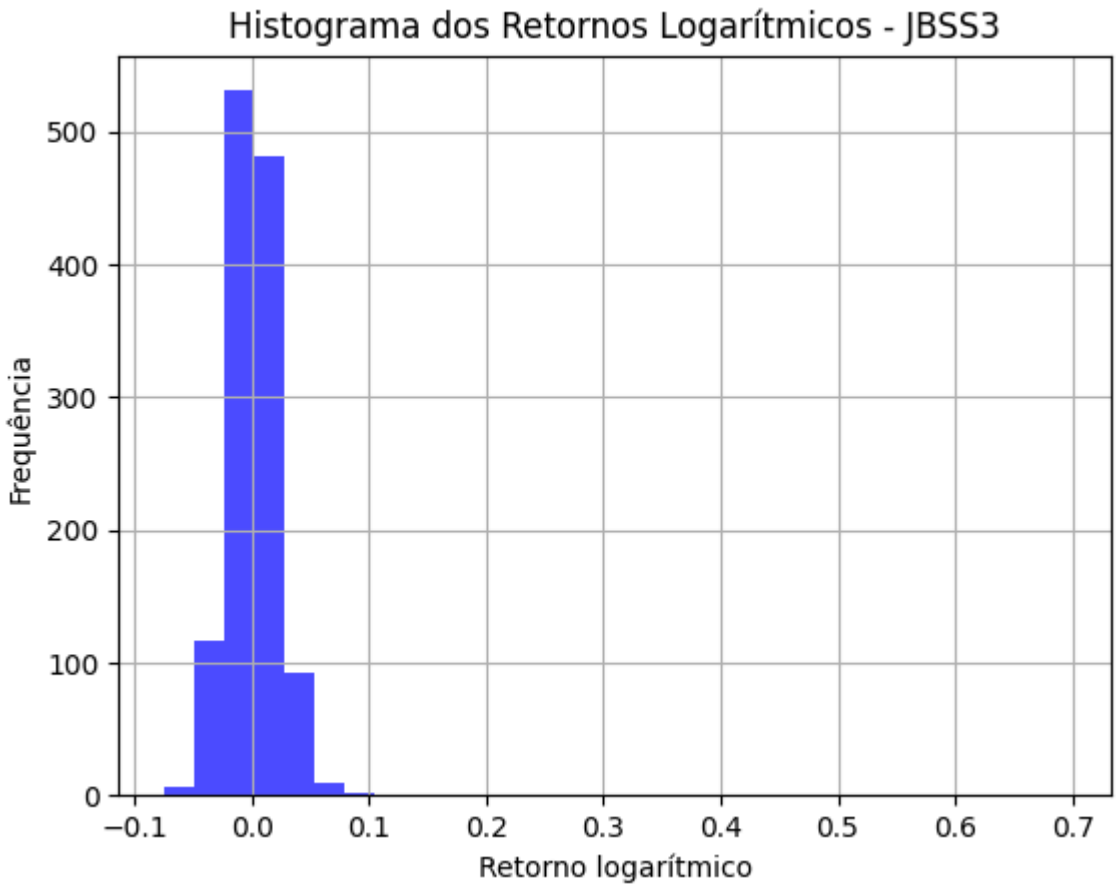


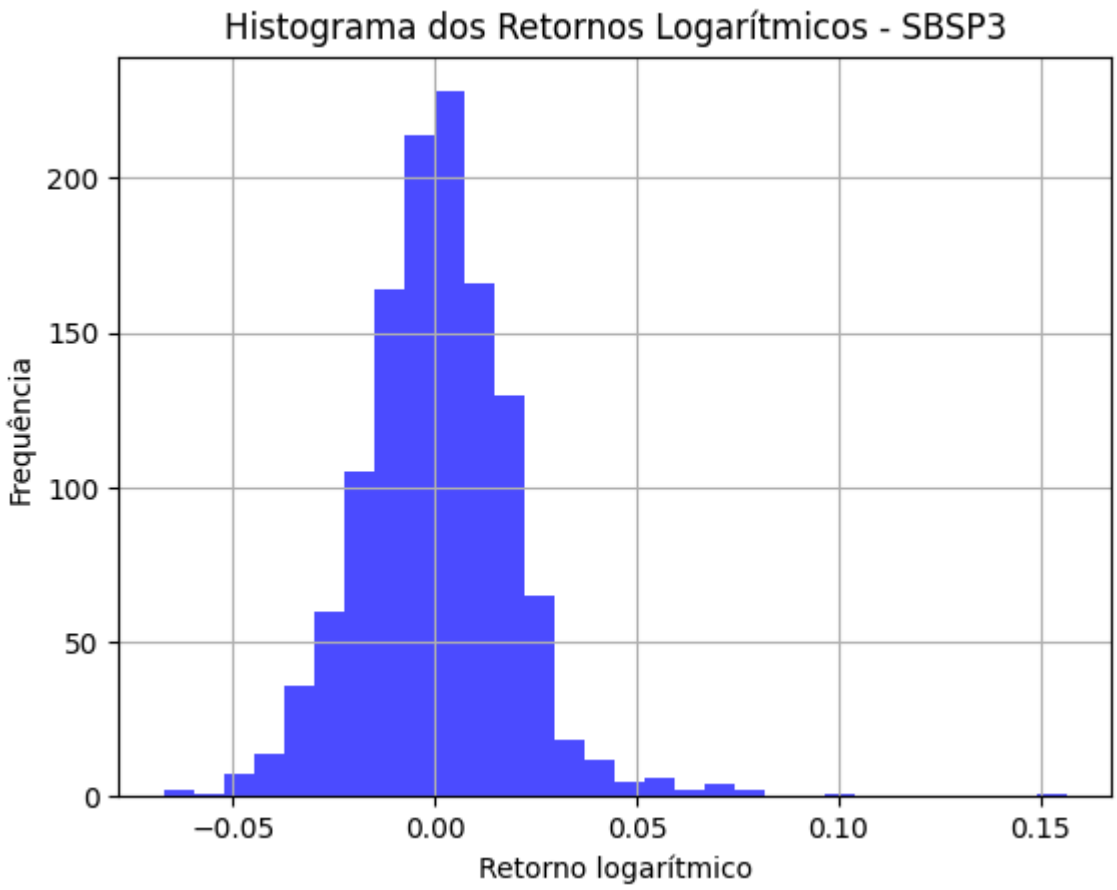
DISTRIBUIÇÃO DOS RETORNOS OBTIDOS


Histograma dos Retornos Logarítmicos - PETR4
Frequência
Retorno logarítmico
0
50
100
150
200
250
300
350
−0.20
−0.15
−0.10
−0.05
0.00
0.05
0.10


Histograma dos Retornos Logarítmicos - ITUB4
Frequência
Retorno logarítmico
0
50
100
150
200
250
300
350
−0.20
−0.15
−0.10
−0.05
0.00
0.05


Histograma dos Retornos Logarítmicos - VALE3
Frequência
Retorno logarítmico
0
25
50
75
100
125
150
175
−0.075
−0.050
−0.025
0.000
0.025
0.050
0.075
0.100


Histograma dos Retornos Logarítmicos - JBSS3
Frequência
Retorno logarítmico
0
100
200
300
400
500
−0.1
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7



Análises:

1. Sobre retorno logarítmico e volatilidade

- É possível observar, pelo gráfico de retorno, que os ativos mais voláteis apresentam curvas com maior dispersão com relaçao ao eixo x (zero). Momentos em que essa dispersão em torno do zero é mais acentuada podem ser explicados por incertezas no mercado, seja por fatos políticos ou relacionados à empresa objeto do ativo. Em comparação com o retorno do índice, é possível confirmar que esse possui comportamento mais controlado, o que é esperado, uma vez que traduz uma média ponderada de vários ativos.

2. Sobre distribuição dos retornos

- A distribuição dos retornos é também uma forma de visualizar a volatilidade das ações. Uma vez que trata-se de retorno logaritmo, um valor zero significa que não houve variação entre o preço anterior e o atual, ou seja, a ação não variou de valor. É possível observar que a distribuição dos retornos tende a respeitar uma distribuição normal de diferentes parâmetros. O desvio padrão dessa distribuição relaciona-se à volatilidade do ativo, pois quanto mais volátil for, mais "espalhada" será a curva, indicando muitas ocorrência de retornos extremos.

QUESTÃO D

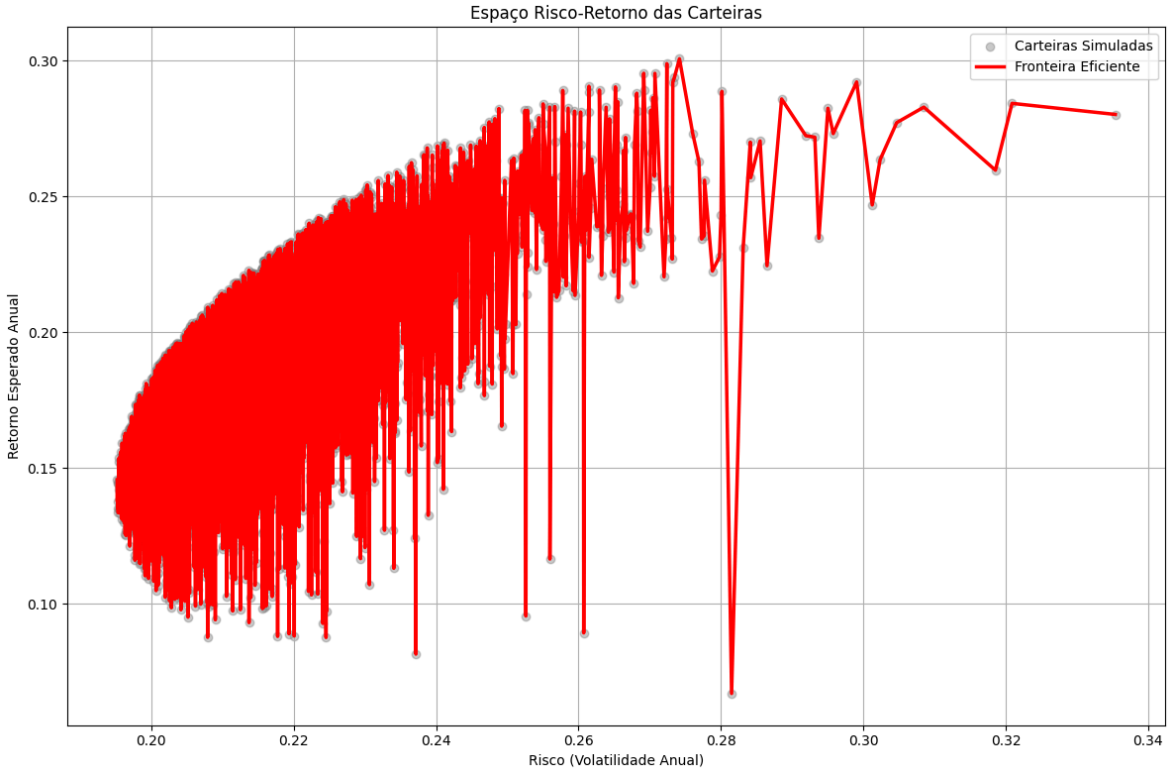
In [3]:
```
import numpy as np
import yfinance as yf
import pandas as pd
import matplotlib.pyplot as plt


# Download dos dados

tickers = ["PETR4.SA", "ITUB4.SA", "VALE3.SA", "JBSS3.SA", "SBSP3.SA"]

ativos = [yf.download(t, period="5y", auto_adjust=True)["Close"] for t in ticker

n_carteiras = 5000 # número de simulações de carteiras aleatórias (cada carteira
n_ativos = len(ativos)

np.random.seed(123) #definição do padrão de aleatoriedade

# Gerar carteiras
carteiras = [] # lista que guardará 50000 arrays, cada um contendo os pesos dos

for _ in range(n_carteiras):
  pesos = np.random.random(n_ativos)   # retorna um array numpy de 5 números ale
  pesos /= pesos.sum()                  # normaliza para somar 1
  carteiras.append(pesos) #adiciona ao final da lista carteiras o array de pesos

# Agora carteiras é uma lista de arrays com pesos de cada ativo


# CÁLCULO DO RETORNO ESPERADO:

# Cálculo do retorno médio anual de cada ativo
retornos_anuais = [] # será uma lista com 5 elementos, ou seja, para cada ativo
for acao in ativos:
    ret = float(np.log(acao / acao.shift(1)).mean() * 252) #anualizando, pois há
    retornos_anuais.append(ret) #adiciona o retorno médio anual na lista de reto

# Calcular retorno esperado para cada carteira
retornos = []
for p in range(n_carteiras):
  retornos_aux = 0
  for i in range(n_ativos):
    retornos_aux += retornos_anuais[i] * carteiras[p][i]
  retornos.append(retornos_aux)

# Exibir exemplo

# número da carteira para a qual se deseja exibir informações específicas. X dev


x = 1000

print(f"\n\nSerão impressos valores de exemplo para a carteira de número {x}, co

print(f"Exemplo de pesos para a carteira {x}:\n Peso do ativo {tickers[0]} = {ca

print(f"Retorno médio anual de cada ativo: \n{retornos_anuais}")

#prazo para cálculo do retorno, em anos:
```

```
p_ret = 5

# cálculo do retorno acumulado composto
retorno_acumulado = (1 + retornos[x])**p_ret - 1

print(f"\nRetorno esperado anual da carteira {x}: {retornos[x]*100:.2f}%\n")
print(f"Retorno esperado acumulado da carteira {x} em {p_ret} anos: {retorno_acu

#cálculo do desvio padrão da carteira
#para isso, pensar na distribuição dos retornos em cada carteira em um período f

# o retorno da carteira no dia é a soma dos retornos ponderados dos ativos naque


# retornos logarítmicos dos ativos
retornos_diarios = [np.log(acao / acao.shift(1)) for acao in ativos]

# DataFrame com os tickers como colunas
df_retornos = pd.concat(retornos_diarios, axis=1)
df_retornos.columns = tickers
df_retornos.dropna(inplace=True)  # remove o primeiro dia com NaN

# retorno diário de cada carteira
ret_cart = []

for j in range(n_carteiras):
    ret_dia = pd.Series(0, index=df_retornos.index)
    for n in range(n_ativos):
        ret_dia += df_retornos.iloc[:, n] * carteiras[j][n]
    ret_cart.append(ret_dia)


# retorno anualizado e risco (desvio padrão) anualizado
retornos_anuais = [serie.mean() * 252 for serie in ret_cart]
volatilidades = [serie.std() * np.sqrt(252) for serie in ret_cart]



# Índice de Sharpe assumindo taxa livre de risco = 0 (como se o investimento mai
sharpe_ratios = [ret / vol if vol > 0 else 0 for ret, vol in zip(retornos_anuais

# exibindo desvio padrão e índice de sharp para uma dada carteira x

print(f"O desvio padrão da carteira {x} é: {volatilidades[x]}")
print()
print(f"O índice de sharp da carteira {x} é: {sharpe_ratios[x]}")
print()

# Carteira com maior índice de Sharpe
idx_max_sharpe = np.argmax(sharpe_ratios)
ret_max_sharpe = retornos_anuais[idx_max_sharpe]
vol_max_sharpe = volatilidades[idx_max_sharpe]

# --- FRONTEIRA EFICIENTE ---

df_resultados = pd.DataFrame({
    'retorno': retornos_anuais,
    'risco': volatilidades,
    'sharpe': sharpe_ratios
})
```

```
df_efficiente = df_resultados.sort_values('risco').drop_duplicates('risco', keep

# --- PLOTAGEM ---

plt.figure(figsize=(12, 8))
plt.scatter(df_resultados['risco'], df_resultados['retorno'],
            c='gray', alpha=0.4, label='Carteiras Simuladas')
plt.plot(df_efficiente['risco'], df_efficiente['retorno'],
         color='red', linewidth=2.5, label='Fronteira Eficiente')
plt.xlabel('Risco (Volatilidade Anual)')
plt.ylabel('Retorno Esperado Anual')
plt.title('Espaço Risco-Retorno das Carteiras')
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```

```
[**********************100%**********************]  1 of 1 completed
[**********************100%**********************]  1 of 1 completed
[**********************100%**********************]  1 of 1 completed
[**********************100%**********************]  1 of 1 completed
[**********************100%**********************]  1 of 1 completed
C:\Users\Gustavo Cunha\AppData\Local\Temp\ipykernel_2072\2115231657.py:34: Future
Warning: Calling float on a single element Series is deprecated and will raise a
TypeError in the future. Use float(ser.iloc[0]) instead
  ret = float(np.log(acao / acao.shift(1)).mean() * 252) #anualizando, pois há 25
2 pregões no ano
```

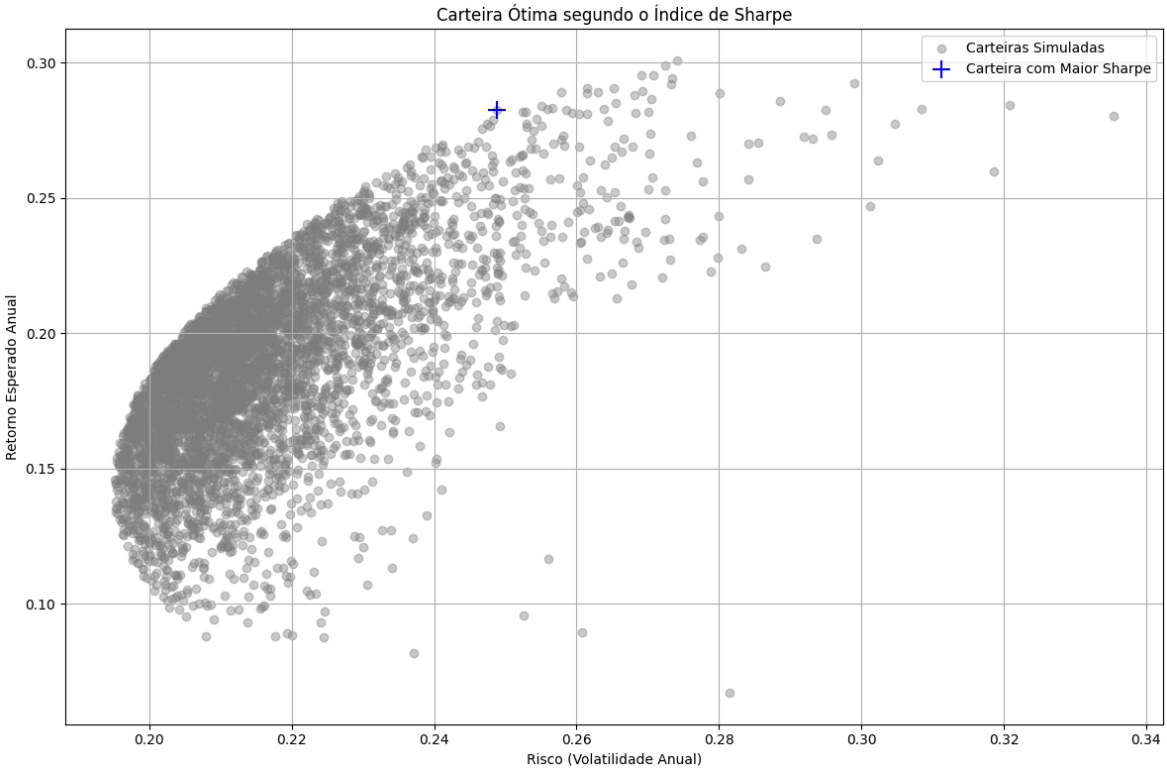
```
Serão impressos valores de exemplo para a carteira de número 1000, conforme defin
ido acima.
Exemplo de pesos para a carteira 1000:
 Peso do ativo PETR4.SA = 0.20
 Peso do ativo ITUB4.SA = 0.08
 Peso do ativo VALE3.SA = 0.31
 Peso do ativo JBSS3.SA = 0.22
 Peso do ativo SBSP3.SA = 0.18
Retorno médio anual de cada ativo:
[0.3111992882691153, 0.08528363168620948, 0.03856830612569235, 0.3445442806798827
4, 0.15699281125271533]

Retorno esperado anual da carteira 1000: 18.58%

Retorno esperado acumulado da carteira 1000 em 5 anos: 134.42%

O desvio padrão da carteira 1000 é: 0.20746431050296438

O índice de sharp da carteira 1000 é: 0.8953898490167771
```



QUESTÃO E

In [4]:
```
# Índice da carteira com maior Sharpe
idx_melhor = np.argmax(sharpe_ratios)

print(f"Carteira com maior índice de Sharpe: {idx_melhor}")
print(f"Índice de Sharpe: {sharpe_ratios[idx_melhor]:.4f}")
print(f"Retorno esperado anual: {ret_max_sharpe:.4f}")
print(f"Risco (volatilidade anual): {vol_max_sharpe:.4f}")
print("Pesos alocados em cada ativo:")

for ticker, peso in zip(tickers, carteiras[idx_melhor]):
    print(f"  {ticker}: {peso:.4f}")

# Gráfico destacando a carteira ótima (maior Sharpe)

plt.figure(figsize=(12, 8))
plt.scatter(volatilidades, retornos_anuais, c='gray', alpha=0.4, label='Carteira
plt.scatter(volatilidades[idx_melhor], ret_max_sharpe,
            color='blue', s=150, marker='+', label='Carteira com Maior Sharpe')
plt.xlabel('Risco (Volatilidade Anual)')
plt.ylabel('Retorno Esperado Anual')
plt.title('Carteira Ótima segundo o Índice de Sharpe')
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```

```
Carteira com maior índice de Sharpe: 577
Índice de Sharpe: 1.1349
Retorno esperado anual: 0.2824
Risco (volatilidade anual): 0.2488
Pesos alocados em cada ativo:
  PETR4.SA: 0.3977
  ITUB4.SA: 0.0052
  VALE3.SA: 0.0071
  JBSS3.SA: 0.3481
  SBSP3.SA: 0.2420
```



Racionalidade da composição da carteira ótima sob a ótica risco-retorno.

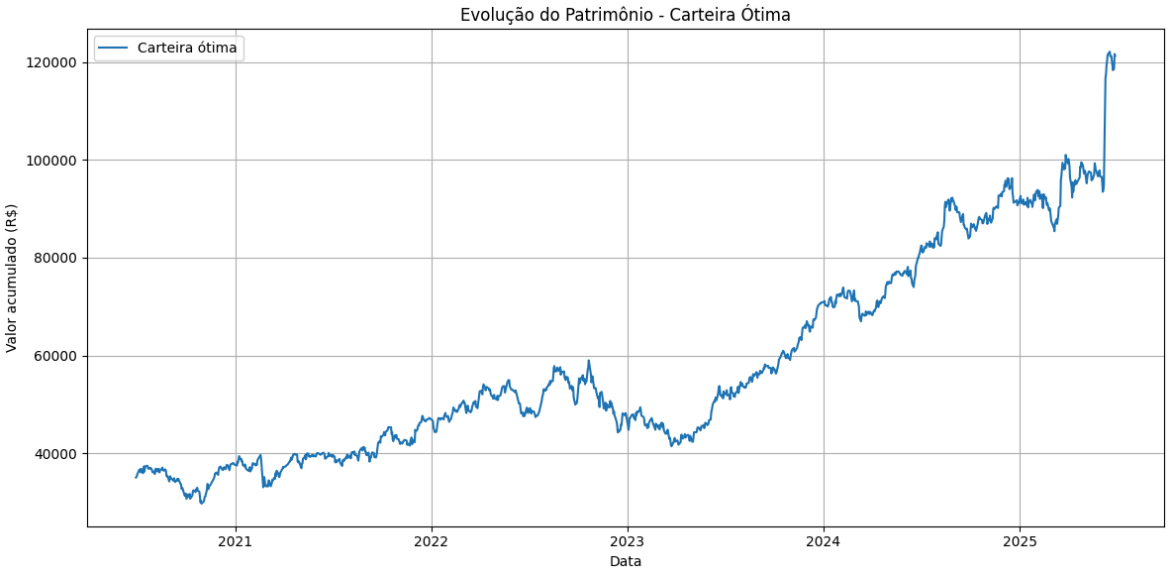
A carteira ótima, considerando as proporções de cada ativo em sua composição, apresenta índice de Sharp de 1,0513, isso significa que a carteira gera 1,0513 unidades de retorno excedente por unidade de risco, o que é considerado bom em finanças. Outro fator a ser considerado é a correlação entre ativos, pois ativos que possuem alta correlação positiva, ou seja, retorno positivo do primeiro indica fortemente retorno positivo do segundo, se tornam "redundantes", e não contribuem na diversificação da carteira. Dessa forma, a baixa correlação entre SBSP3 e PETR4 pode explicar a alta proporção de cada um na composição da carteira ótima, pois se equilibram e reduzem a volatilidade anual (risco).

QUESTÃO F

In [5]:
```
# SIMULAÇÃO DA EVOLUÇÃO PATRIMONIAL

inv_inicial = 35000

print("Simulação temporal do patrimônio com base na carteira ótima\n")
```

```
print(f"Pesos da carteira ótima: {carteiras[idx_melhor]}")
print(f"Retorno anual esperado da carteira ótima: {ret_max_sharpe:.4f}")
print("Janela temporal: 5 anos")
print(f"Investimento inicial: {inv_inicial}")

# Retorno diário total
retornos_serie_otima = pd.Series(0, index=df_retornos.index)
for n in range(n_ativos):
    retornos_serie_otima += df_retornos.iloc[:, n] * carteiras[idx_melhor][n]


# Calcular evolução do patrimônio
patrimonio = (1 + retornos_serie_otima).cumprod() * inv_inicial

# Calcular retorno acumulado
retorno_acumulado = patrimonio.iloc[-1] / inv_inicial - 1
print(f"\nRetorno acumulado da carteira ótima no período: {retorno_acumulado*100

# Curva de evolução do patrimônio
plt.figure(figsize=(12, 6))
plt.plot(patrimonio, label="Carteira ótima")
plt.title("Evolução do Patrimônio - Carteira Ótima")
plt.ylabel("Valor acumulado (R$)")
plt.xlabel("Data")
plt.grid(True)
plt.legend()
plt.tight_layout()
plt.show()
```

```
Simulação temporal do patrimônio com base na carteira ótima

Pesos da carteira ótima: [0.39766681 0.00519199 0.00707877 0.34809046 0.24197198]
Retorno anual esperado da carteira ótima: 0.2824
Janela temporal: 5 anos
Investimento inicial: 35000

Retorno acumulado da carteira ótima no período: 246.51%
```



QUESTÃO G

In [6]:
```
import matplotlib.pyplot as plt

# Retornos diários da carteira ótima, pois VaR é baseado na distribuição dos ret
retornos_serie_otima = pd.Series(0, index=df_retornos.index)
for n in range(n_ativos):
    retornos_serie_otima += df_retornos.iloc[:, n] * carteiras[idx_melhor][n]

# VaR histórico
var_95_hist = np.percentile(retornos_serie_otima, 5)
var_99_hist = np.percentile(retornos_serie_otima, 1)

# em valores monetários
inv_inicial = 35000
var_95_monetario = -var_95_hist * inv_inicial
var_99_monetario = -var_99_hist * inv_inicial

print("VaR histórico: ")
print(f"VaR 95%: {var_95_hist:.4%} ou R$ {var_95_monetario:.2f}")
print(f"VaR 99%: {var_99_hist:.4%} ou R$ {var_99_monetario:.2f}")

# VaR por Simulação de Monte Carlo
media = retornos_serie_otima.mean()
desvio = retornos_serie_otima.std()
simulacoes = np.random.normal(loc=media, scale=desvio, size=100000)

var_95_mc = np.percentile(simulacoes, 5)
var_99_mc = np.percentile(simulacoes, 1)

var_95_mc_monetario = -var_95_mc * inv_inicial
var_99_mc_monetario = -var_99_mc * inv_inicial

print("\nVaR via SIMULAÇÃO DE MONTE CARLO: ")
print(f"VaR 95%: {var_95_mc:.4%} ou R$ {var_95_mc_monetario:.2f}")
print(f"VaR 99%: {var_99_mc:.4%} ou R$ {var_99_mc_monetario:.2f}")
```

```
VaR histórico:
VaR 95%: -2.2818% ou R$ 798.63
VaR 99%: -3.3979% ou R$ 1189.28

VaR via SIMULAÇÃO DE MONTE CARLO:
VaR 95%: -2.4618% ou R$ 861.62
VaR 99%: -3.5340% ou R$ 1236.91
```

Interpretação do risco envolvido:

Conceito simples de VaR: Com x % de confiança, espera-se que a perda máxima diária da carteira não ultrapasse um determinado valor. Pode ser expresso de forma percentual ou monetária.

Portanto, os resultados do VaR histórico indicam que, com 95% de confiança, a perda máxima diária não deve ultrapassar R$ 806,00 ou -2,3029%.

Adicionalmente, acrescenta-se que, com 99% de confiança, a perda máxima diária não deve ultrapassar R$ 1210,87 ou -3,4596%.

Ressalta-se que os resultados obtidos com o VaR não considera a eventual ocorrência de eventos extremos.