**RELATÓRIO DE TESTES UNITÁRIOS COM PYTEST**

**082210016 – João Vitor Antunes Nascimento**

**OBJETIVO DA ATIVIDADE**

O objetivo desta atividade é identificar a importância e como construir testes unitários utilizando a biblioteca pytest. Esta prática visa garantir a qualidade, confiabilidade e manutenibilidade do código através da implementação de testes automatizados que validam o comportamento correto das funções estatísticas desenvolvidas.

**IMPORTÂNCIA DOS TESTES UNITÁRIOS NO CONTEXTO DE ESTATÍSTICA**

No contexto de análise estatística, os testes unitários são fundamentais e críticos pelas seguintes razões para garantir precisão das análises e dos cálculos, garantindo confiabilidade de dados que no contexto estatístico podem ser muitas vezes críticos para o ambiente e resistência contra as falhas previstas, além de detecção de outliers e valores extremos para garantir precisão e manutenibilidade dos métodos.

## DESCRIÇÃO DOS TESTES IMPLEMENTADOS

### \*\*Arquivo analisado:\*\* `estatisticas.py`

\*\*Funções testadas:\*\*

- `media(lista)` - Calcula a média aritmética

- `mediana(lista)` - Calcula a mediana

- `desvio\_padrao(lista)` - Calcula o desvio padrão

- `detectar\_outliers(lista)` - Detecta valores atípicos

### \*\*Arquivo de testes:\*\* `test\_estatisticas.py`

#### \*\*1. test\_media()\*\*

- \*\*Objetivo:\*\* Testa o cálculo da média de uma lista de números

- \*\*Cenários:\*\* Listas com números positivos

- \*\*Validações:\*\* Verifica se a média é calculada corretamente

#### \*\*2. test\_media\_valores\_negativos()\*\*

- \*\*Objetivo:\*\* Testa o cálculo da média com valores negativos

- \*\*Cenários:\*\* Listas contendo apenas números negativos

- \*\*Validações:\*\* Garante que a função funciona corretamente com valores negativos

#### \*\*3. test\_valores\_texto()\*\*

- \*\*Objetivo:\*\* Testa o tratamento de erros para entradas inválidas na função média

- \*\*Cenários:\*\* Lista vazia e lista com valores não numéricos

- \*\*Validações:\*\* Verifica se as exceções corretas são levantadas (TypeError, ValueError)

#### \*\*4. test\_mediana()\*\*

- \*\*Objetivo:\*\* Testa o cálculo da mediana de uma lista de números

- \*\*Cenários:\*\* Listas com números positivos (ímpares e pares)

- \*\*Validações:\*\* Verifica se a mediana é calculada corretamente

#### \*\*5. test\_mediana\_valores\_negativos()\*\*

- \*\*Objetivo:\*\* Testa o cálculo da mediana com valores negativos

- \*\*Cenários:\*\* Listas contendo apenas números negativos

- \*\*Validações:\*\* Garante funcionamento correto com valores negativos

#### \*\*6. test\_mediana\_texto()\*\*

- \*\*Objetivo:\*\* Testa o tratamento de erros para entradas inválidas na função mediana

- \*\*Cenários:\*\* Lista vazia e lista com valores não numéricos

- \*\*Validações:\*\* Verifica se as exceções corretas são levantadas

#### \*\*7. test\_desvio\_padrao()\*\*

- \*\*Objetivo:\*\* Testa o cálculo do desvio padrão

- \*\*Cenários:\*\* Listas com números positivos

- \*\*Validações:\*\* Verifica se o desvio padrão é calculado com precisão

#### \*\*8. test\_desvio\_padrao\_valores\_negativos()\*\*

- \*\*Objetivo:\*\* Testa o desvio padrão com valores negativos

- \*\*Cenários:\*\* Listas contendo apenas números negativos

- \*\*Validações:\*\* Garante funcionamento correto independente do sinal dos números

#### \*\*9. test\_desvio\_padrao\_texto()\*\*

- \*\*Objetivo:\*\* Testa o tratamento de erros na função desvio padrão

- \*\*Cenários:\*\* Lista vazia e lista com valores não numéricos

- \*\*Validações:\*\* Verifica se as exceções corretas são levantadas

#### \*\*10. test\_detectar\_outliers()\*\*

- \*\*Objetivo:\*\* Testa a detecção de valores atípicos (outliers)

- \*\*Cenários:\*\* Listas sem outliers, com outliers e casos específicos

- \*\*Validações:\*\* Verifica se outliers são identificados corretamente usando critério de 2 desvios padrão

#### \*\*11. test\_detectar\_outliers\_texto()\*\*

- \*\*Objetivo:\*\* Testa o tratamento de erros na função de detecção de outliers

- \*\*Cenários:\*\* Lista vazia e lista com valores não numéricos

- \*\*Validações:\*\* Verifica se as exceções corretas são levantadas

## RESULTADOS DOS TESTES

### \*\*Quantidade de Testes Realizados:\*\* 11 testes

### \*\*Resultado:\*\* ✅ \*\*TODOS OS 11 TESTES PASSARAM\*\*

```

================================== test session starts ===================================

platform win32 -- Python 3.13.5, pytest-8.4.2, pluggy-1.6.0

collected 11 items

estatisticas/test\_estatisticas.py::test\_media PASSED [ 9%]

estatisticas/test\_estatisticas.py::test\_media\_valores\_negativos PASSED [ 18%]

estatisticas/test\_estatisticas.py::test\_valores\_texto PASSED [ 27%]

estatisticas/test\_estatisticas.py::test\_mediana PASSED [ 36%]

estatisticas/test\_estatisticas.py::test\_mediana\_valores\_negativos PASSED [ 45%]

estatisticas/test\_estatisticas.py::test\_mediana\_texto PASSED [ 54%]

estatisticas/test\_estatisticas.py::test\_desvio\_padrao PASSED [ 63%]

estatisticas/test\_estatisticas.py::test\_desvio\_padrao\_valores\_negativos PASSED [ 72%]

estatisticas/test\_estatisticas.py::test\_desvio\_padrao\_texto PASSED [ 81%]

estatisticas/test\_estatisticas.py::test\_detectar\_outliers PASSED [ 90%]

estatisticas/test\_estatisticas.py::test\_detectar\_outliers\_texto PASSED [100%]

=================================== 11 passed in 0.05s ===================================

```

## ANÁLISE DE CODE COVERAGE (COBERTURA DE CÓDIGO)

### \*\*Relatório de Cobertura:\*\*

```

Name Stmts Miss Cover Missing

-----------------------------------------------------------------

estatisticas\estatisticas.py 26 1 96% 23

estatisticas\test\_estatisticas.py 44 0 100%

-----------------------------------------------------------------

TOTAL 70 1 99%

```

### \*\*Discussão sobre Code Coverage:\*\*

#### \*\*Cobertura do Arquivo Principal (estatisticas.py):\*\*

- \*\*96% de cobertura\*\* - Excelente resultado

- \*\*26 statements totais, apenas 1 não coberto\*\*

- \*\*Linha 23 não coberta:\*\* Corresponde a uma linha de código que não foi executada durante os testes

#### \*\*Cobertura do Arquivo de Testes:\*\*

- \*\*100% de cobertura\*\* - Todos os testes foram executados

#### \*\*Cobertura Total do Projeto:\*\*

- \*\*99% de cobertura geral\*\* - Resultado excepcional

### \*\*Análise Crítica:\*\*

#### \*\*Pontos Positivos:\*\*

1. \*\*Cobertura quase completa (96%)\*\* do código principal

2. \*\*Todos os testes passaram\*\* sem falhas

3. \*\*Cobertura abrangente\*\* de cenários positivos e negativos

4. \*\*Tratamento adequado de exceções\*\* testado

5. \*\*Funções críticas\*\* (média, mediana, desvio padrão) totalmente validadas

#### \*\*Área de Melhoria:\*\*

- \*\*Linha 23 não coberta:\*\* Investigar se representa um cenário de edge case não testado

- Considerar adicionar teste para cobrir essa linha específica

#### \*\*Qualidade dos Testes:\*\*

- \*\*Testes bem estruturados\*\* com cenários diversos

- \*\*Validação de exceções\*\* adequadamente implementada

- \*\*Casos de teste representativos\*\* para uso real

- \*\*Nomenclatura clara\*\* dos testes facilitando manutenção

## CONCLUSÃO

A implementação de testes unitários com pytest para o módulo de estatísticas demonstra \*\*excelente qualidade e cobertura\*\*. Com 96% de cobertura do código principal e 100% de sucesso nos testes, o código está bem protegido contra regressões e bugs.

Os testes cobrem adequadamente:

- ✅ Funcionalidades principais (cálculos estatísticos)

- ✅ Tratamento de erros e exceções

- ✅ Casos extremos (valores negativos, listas vazias)

- ✅ Validação de tipos de entrada

Esta abordagem garante que as funções estatísticas são \*\*confiáveis, precisas e robustas\*\*, características essenciais para análises estatísticas críticas.

---

\*\*Relatório gerado em:\*\* $(Get-Date)

\*\*Ferramenta utilizada:\*\* pytest com pytest-cov

\*\*Ambiente:\*\* Python 3.13.5, Windows