

# PSICOLOGIA DOS TESTES DE SOFTWARE

February 24, 2025

## AGENDA

- Introdução à Psicologia dos Testes
- Definição Correta do Teste
- Objetivo Psicológico dos Testes
- Analogia Médica
- Princípios de Teste de Software
- Princípio 1: Definição de Resultados Esperados
- Princípio 2: Teste por Outros
- Princípio 3: Independência nos Testes
- Princípio 4: Inspeção Minuciosa
- Princípio 5: Casos de Teste Inesperados
- Princípio 6: Efeitos Colaterais Indesejados
- Princípio 7: Casos de Teste Reutilizáveis
- Princípio 8: Planejamento de Testes
- Princípio 9: Erros Agrupados
- Princípio 10: Criatividade nos Testes
- Resumo dos Princípios de Teste





PSICOLOGIA

# INTRODUÇÃO À PSICOLOGIA DOS TESTES



Uma das principais causas de testes de software deficientes é o fato de que a maioria dos programadores começa com uma definição errada do termo. Eles podem dizer: "Testar é o processo de demonstrar que não há erros". Essa visão distorcida leva a uma abordagem que falha em agregar valor ao software. A verdadeira intenção do teste é encontrar erros, partindo da suposição de que o programa contém falhas. Portanto, "Testar é o processo de executar um programa com a intenção de encontrar erros." Essa compreensão é crucial para o sucesso dos esforços de teste.



PRINCÍPIOS  
DE TESTE

Testar é o  
processo de  
executar um  
programa com a  
intenção de  
encontrar erros.

Uma das principais causas de testes de software deficientes é a definição errada do termo. Muitos acreditam que "testar é o processo de demonstrar que não há erros". A definição correta é que o teste deve partir da suposição de que o programa contém erros e o objetivo é encontrá-los.



PSICOLOGIA

# OBJETIVO PSICOLÓGICO DOS TESTES



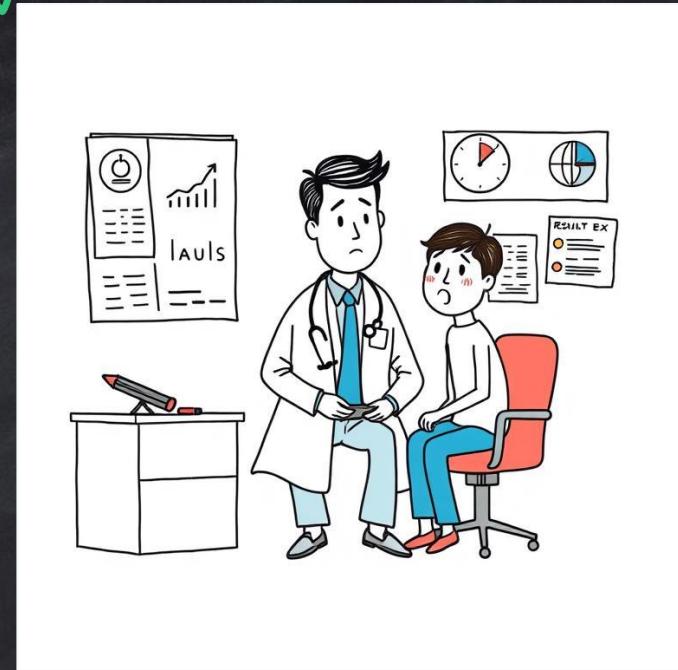
O objetivo de encontrar erros em um programa é fundamental para aumentar a qualidade e a confiabilidade do software. Ao partir da suposição de que o programa contém erros, os testadores são incentivados a explorar todas as possibilidades falhas, o que resulta em um teste mais eficaz. Como mencionado, 'se nosso objetivo é demonstrar que um programa tem erros, nossos dados de teste terão uma probabilidade maior de encontrar erros.' Assim, essa abordagem não só melhora a qualidade do software, mas também a confiança dos usuários, garantindo que ele funcione conforme o esperado e esteja livre de falhas críticas.



# ANALOGIA MÉDICA

## Importância de Encontrar Erros

- A analogia entre testes de software e exames médicos destaca a necessidade de identificar problemas antes que se tornem críticos.
- Assim como um médico utiliza exames laboratoriais para descobrir a causa de um mal-estar, os testadores buscam erros em um programa para garantir sua funcionalidade.
- Se um exame laboratorial não localiza o problema, não é considerado bem-sucedido; da mesma forma, um teste que não encontra erros pode falhar em seu objetivo.
- Um teste bem-sucedido é aquele que revela um problema existente, semelhante a um exame que identifica uma condição médica, permitindo o tratamento adequado.



# PRINCÍPIOS DE TESTE DE SOFTWARE

PRINCÍPIOS



## Definição de Resultados Esperados

Uma parte necessária de um caso de teste é a definição do resultado ou saída esperada. "Se o resultado esperado de um caso de teste não foi previamente definido, é provável que um resultado plausível, mas errado, seja interpretado como correto."

## Evitar Testar o Próprio Código

Um programador deve evitar tentar testar seu próprio programa. "É extremamente difícil mudar repentinamente de perspectiva para olhar o programa com um olhar destrutivo."

## Inspeção Minuciosa

Inspecione minuciosamente os resultados de cada teste. "Erros encontrados em testes posteriores muitas vezes são perdidos nos resultados dos testes anteriores."

## Testar Condições Inválidas

Os casos de teste devem ser escritos para condições de entrada que sejam inválidas e inesperadas. "Casos de teste representando condições de entrada inesperadas e inválidas parecem ter uma maior taxa de detecção de erros."

## Criatividade nos Testes

Testar é uma tarefa extremamente criativa e intelectualmente desafiadora. "A criatividade necessária para testar um grande programa supera a criatividade exigida para projetá-lo."

## PRINCÍPIO 1: DEFINIÇÃO DE RESULTADOS ESPERADOS

Uma parte necessária de um caso de teste é a definição do resultado ou saída esperada. Este princípio óbvio é um dos erros mais frequentes em testes de programas. Se o resultado esperado de um caso de teste não foi previamente definido, é provável que um resultado plausível, mas errado, seja interpretado como correto devido ao fenômeno do: "os olhos veem o que querem ver".

## PRINCÍPIO 1: DEFINIÇÃO DE RESULTADOS ESPERADOS

Portanto, um caso de teste deve consistir de dois componentes:

- uma descrição dos dados de entrada para o programa e
- uma descrição precisa da saída correta do programa para aquele conjunto de dados de entrada.



PRINCÍPIOS DE TESTE DE  
SOFTWARE

## PRINCÍPIO 2: TESTE POR OUTROS



Um programador deve evitar tentar testar seu próprio programa.

Qualquer escritor sabe—ou deveria saber—que é uma má ideia tentar editar ou revisar o próprio trabalho. Você sabe o que a peça deveria dizer e pode não perceber quando ela diz algo diferente. E você realmente não quer encontrar erros no seu próprio trabalho. O mesmo se aplica aos autores de software.





PRINCÍPIOS DE TESTE DE  
SOFTWARE

## PRINCÍPIO 2: TESTE POR OUTROS



Outro problema surge com a mudança de foco em um projeto de software. Depois que um programador projetou e codificou um programa de maneira construtiva, é extremamente difícil mudar repentinamente de perspectiva para olhar o programa com um olhar destrutivo. Portanto, o teste é mais eficaz e bem-sucedido quando feito por outra pessoa.





PRINCÍPIOS DE TESTE DE  
SOFTWARE

## PRINCÍPIO 2: TESTE POR OUTROS



Há ainda um segundo problema significativo: o programa pode conter erros devido ao mal-entendido do programador sobre a declaração do problema ou a especificação. Se for esse o caso, é provável que o programador leve o mesmo mal-entendido para os testes de seu próprio programa.

Isso não significa que seja impossível para um programador testar seu próprio programa. Em vez disso, implica que o teste é mais eficaz e bem-sucedido quando feito por outra pessoa. Vale ressaltar que esse argumento não se aplica à depuração (correção de erros conhecidos); a depuração é mais eficientemente realizada pelo programador original.



## PRINCÍPIO 3: INDEPENDÊNCIA NOS TESTES

**Uma organização de programação não deve testar seus próprios programas.**

O argumento aqui é similar ao argumento anterior. Um projeto ou organização de programação é, em muitos sentidos, uma organização viva com problemas psicológicos semelhantes aos dos programadores individuais. Além disso, uma organização de programação ou um gerente de projeto é amplamente avaliado pela capacidade cumprir prazos e por um custo específico.

Portanto, é difícil para uma organização de programação ser objetiva ao testar seus próprios programas, porque o processo de teste, se abordado com a definição adequada, pode ser visto como algo que diminui a probabilidade de atender ao cronograma e aos objetivos de custo.



PRINCÍPIOS DE TESTE

## PRINCÍPIO 4: INSPEÇÃO MINUCIOSA



Este é provavelmente o princípio mais óbvio, mas, novamente, é algo que frequentemente é negligenciado. Vimos diversos experimentos que mostram que muitos participantes não conseguiram detectar certos erros, mesmo quando os sintomas desses erros eram claramente observáveis nas listagens de saída. Em outras palavras, erros encontrados em testes posteriores muitas vezes são perdidos nos resultados dos testes anteriores.





PRINCÍPIOS DE TESTE

## PRINCÍPIO 5: CASOS DE TESTE INESPERADOS E ESPERADOS



Os casos de teste devem ser escritos para condições de entrada que sejam inválidas e inesperadas, assim como para aquelas que são válidas e esperadas.

Há uma tendência natural, ao testar um programa, de se concentrar nas condições de entrada válidas e esperadas, negligenciando as condições inválidas e inesperadas. Por exemplo, pouca atenção é dada ao testar o programa de triângulos com entradas que não formam um triângulo.





PRINCÍPIOS DE TESTE

## PRINCÍPIO 5: CASOS DE TESTE INESPERADOS E ESPERADOS



Muitos erros que são descobertos em programas em produção surgem quando o programa é usado de uma maneira nova ou inesperada. Portanto, casos de teste representando condições de entrada **inesperadas e inválidas** parecem ter uma maior taxa de detecção de erros do que casos de teste para condições de entrada válidas.



## PRINCÍPIO 6: EFEITOS COLATERAIS INDESEJADOS

"Examinar um programa para ver se ele não faz o que deveria fazer é apenas metade da batalha; a outra metade é verificar se o programa faz o que não deveria fazer."

Isso significa que, além de garantir que o software funcione conforme o esperado, é crucial identificar efeitos colaterais indesejados que possam surgir.

## PRINCÍPIO 6: EFEITOS COLATERAIS INDESEJADOS



Por exemplo, um programa de folha de pagamento que produz contracheques corretos ainda é considerado errôneo se também gerar cheques extras para empregados inexistentes ou sobrescrever registros importantes.

Portanto, uma abordagem abrangente nos testes deve incluir a verificação de comportamentos inesperados em todas as funcionalidades do programa.

## PRINCÍPIO 7: CASOS DE TESTE REUTILIZÁVEIS

Evitar casos de teste descartáveis é crucial, a menos que o programa seja realmente um programa descartável. Um problema comum observado em sistemas interativos é a prática de "sentar-se em um terminal e inventar casos de teste na hora". Os casos de teste representam um investimento valioso que, nesse ambiente, desaparecem após a conclusão dos testes.

## PRINCÍPIO 7: CASOS DE TESTE REUTILIZÁVEIS

Sempre que o programa precisar ser testado novamente, como após a correção de um erro ou melhoria, os casos de teste precisam ser reinventados. Isso exige uma quantidade considerável de trabalho, levando as pessoas a evitar essa reinvenção. Portanto, o reteste do programa raramente é tão rigoroso quanto o teste original, resultando em erros que podem passar despercebidos.

PRINCÍPIOS

## PRINCÍPIO 8: PLANEJAMENTO DE TESTES

Não planeje um esforço de testes sob a suposição tácita de que nenhum erro será encontrado. Este é um erro comum cometido por gerentes de projeto e é um sinal do uso da definição incorreta de testes — ou seja, a suposição de que o teste é o processo de mostrar que o programa funciona corretamente. Mais uma vez, a definição de testes é o processo de executar um programa com a intenção de encontrar erros.

## PRINCÍPIO 9: ERROS AGRUPADOS

"A probabilidade da existência de mais erros em uma seção de um programa é proporcional ao número de erros já encontrados nessa seção."

Este fenômeno indica que "os erros tendem a surgir em agrupamentos", e que, em um programa típico, algumas seções parecem ser muito mais propensas a erros do que outras.

## PRINCÍPIO 9: ERROS AGRUPADOS

Portanto, se um módulo possui vários erros, isso sugere que é necessário concentrar os esforços de teste nessa área problemática. Essa análise ajuda a priorizar o teste e a alocar recursos de maneira mais eficaz, aumentando as chances de identificar e corrigir falhas críticas.

## PRINCÍPIO 10: CRIATIVIDADE NOS TESTES

Testar é uma tarefa extremamente criativa e intelectualmente desafiadora. Provavelmente, a criatividade necessária para testar um grande programa supera a criatividade exigida para projetá-lo. Já vimos que é impossível testar um programa o suficiente para garantir a ausência total de erros. As metodologias que vamos discutir ao longo da disciplina permitem desenvolver um conjunto razoável de casos de teste para um programa, mas essas metodologias ainda exigem uma quantidade significativa de criatividade.

# PRINCÍPIOS

# RESUMO DOS PRINCÍPIOS DE TESTE



## Intenção de Encontrar Erros

Testar é o processo de executar um programa com a intenção de encontrar erros. Essa definição é fundamental para ter um foco claro durante o processo de teste.

## Alta Probabilidade de Detectar Erros

Um bom caso de teste é aquele que tem alta probabilidade de detectar um erro ainda não descoberto, aumentando assim a confiabilidade do software.

## Sucesso ao Encontrar Erros

Um caso de teste bem-sucedido é aquele que detecta um erro ainda não descoberto, contribuindo significativamente para a melhoria da qualidade do programa.

THANK  
YOU





Copy and paste these squiggles to  
decorate your slides.

Swap them out or add more – it's up  
to you!