TESTES AUTOMATIZADOS PARA WEB

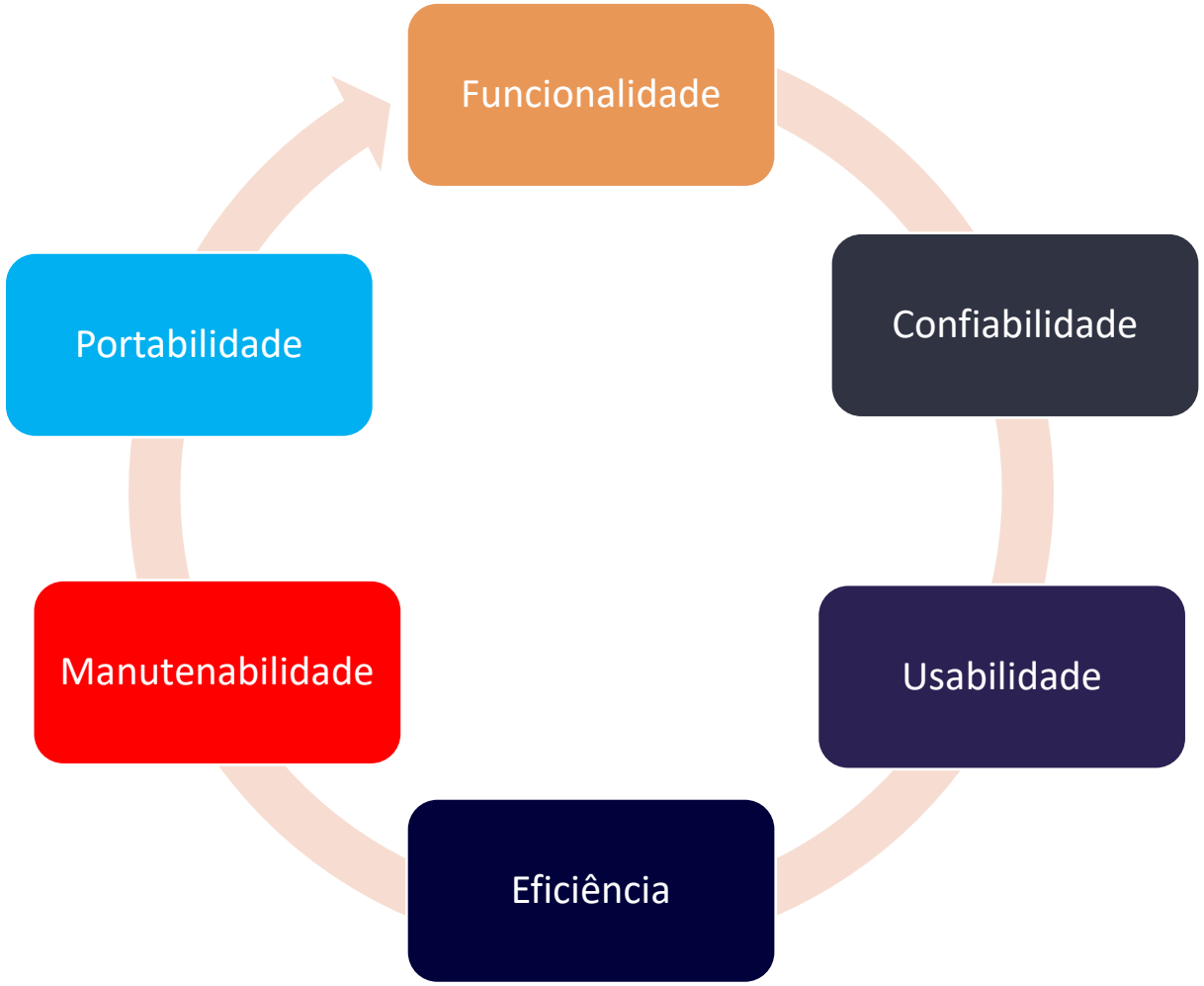
CASOS DE TESTES

PROF. MESTRANDO ÉRICO BORGONOVE

AGENDA

- INTRODUÇÃO
- CONCEITOS
- TIPOS DE TESTES
- CERTIFICAÇÕES
- PRINCÍPIOS DO TESTE
- TÉCNICAS
- PARTCIONAMENTO POR EQUIVALÊNCIA
- ANÁLISE DO VALOR LIMITE
- TABELA DE DECISÃO
- ROTEIRO DE TESTES
- EXERCÍCIOS

NORMA ISO 9126

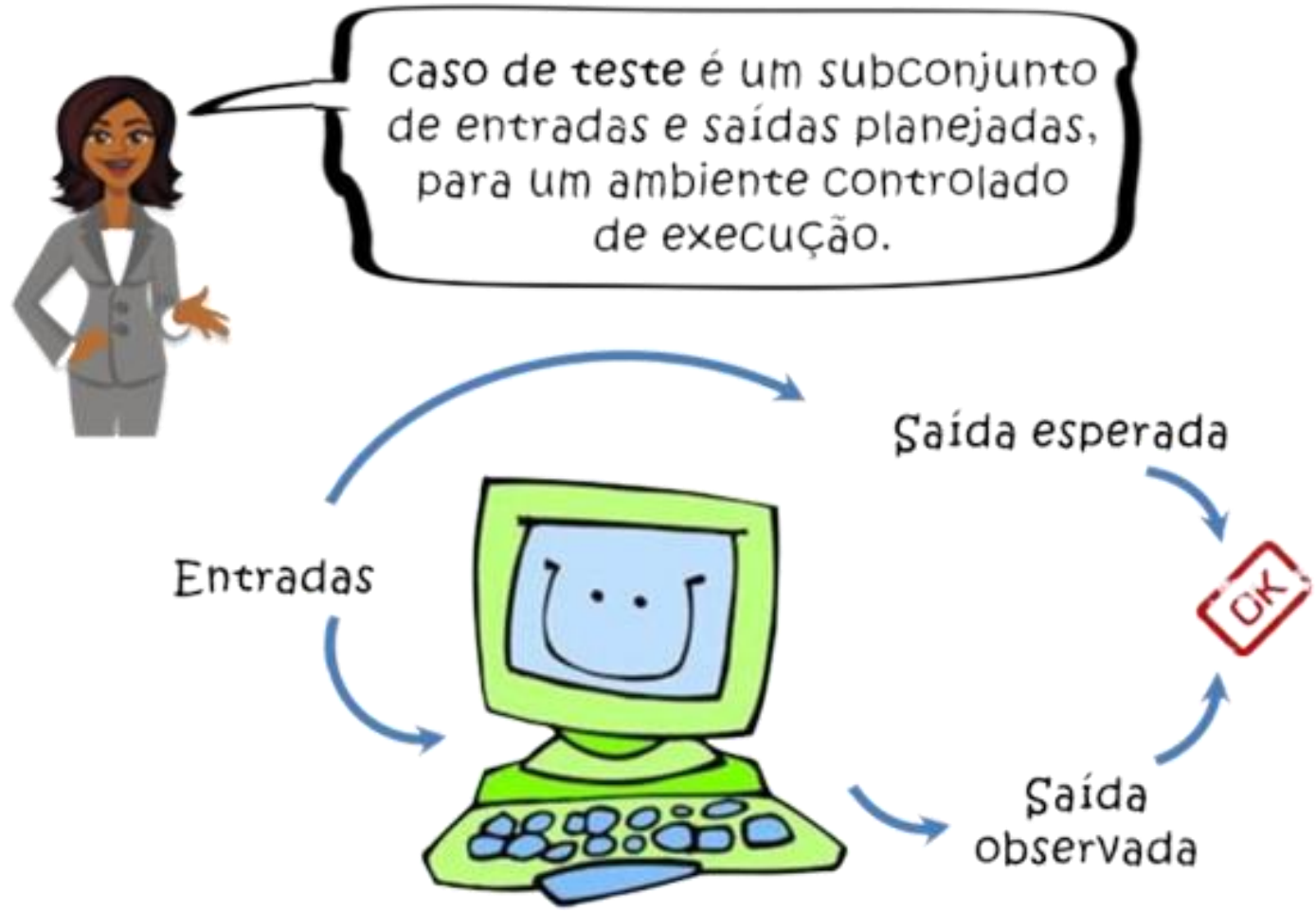


O QUE SÃO CASOS DE TESTE?

Os casos de teste são os **cenários** usados para **medir a funcionalidade** do aplicativo em um conjunto de determinadas ações ou condições para verificar os resultados esperados. Em outras palavras, um caso de teste é um **conjunto de ações executadas para autenticar a funcionalidade do seu aplicativo de software.**

O QUE SÃO CASOS DE TESTE?

Um caso de teste consiste em várias coisas, como **etapas de teste, dados de teste e pré e pós-condições desenvolvidas para um cenário de teste específico.** Os casos de teste podem ser aplicados a qualquer aplicativo de software. Isso pode ser feito por meio de testes manuais e automatizados ou qualquer ferramenta de gerenciamento de testes.


Caso de teste é um subconjunto de entradas e saídas planejadas, para um ambiente controlado de execução.
Saída esperada
Entradas
OK
Saída observada

Artefatos de Teste

Todo o conjunto de documentação gerado pelo processo de teste de software.

Caso de Teste

É composto por um conjunto de entradas, por Caso de Teste passos de execução e um resultado esperado

Roteiro de Teste

É composto por um conjunto de casos de teste definidos para uma determinada especificação.

Segundo Craig e Jaskiel (2002), estes artefatos do processo de teste de software podem ser definidos da seguinte forma:

- **Caso de Teste:** descreve uma condição particular a ser testada e é composto por valores de entrada, restrições para a sua execução e um resultado ou comportamento esperado.
- **Procedimento de Teste:** é uma descrição dos passos necessários para executar um caso (ou um grupo de casos) de teste.

CASOS X PROCEDIMENTOS

CASO DE TESTE DE FUNCIONALIDADE

- Os casos de teste de funcionalidade, como o nome sugere, são usados para analisar se o sistema está funcionando conforme o esperado ou não.
- A equipe de QA é responsável por escrever os casos de teste funcionais. Esse tipo de teste pode ser feito assim que a equipe de desenvolvimento disponibilizar a primeira função do aplicativo para teste.
- Por exemplo, verificar se o usuário pode fazer upload de uma foto de perfil.

CASO DE TESTE DE INTEGRAÇÃO

- Os casos de teste de integração são usados para analisar se os diferentes módulos da aplicação estão interagindo corretamente ou não.
- A equipe de testes é responsável por separar as áreas que devem passar por testes de integração. Por outro lado, a equipe de desenvolvimento ajuda a escrever os casos de teste de integração.
- Por exemplo, verificar se a página de login aparece quando clicamos no botão 'login' na página inicial.

CASO DE TESTE DE USABILIDADE

- Casos de teste de usabilidade, também conhecidos como tarefas ou cenários, são casos em que os testadores apresentam cenários ou tarefas de alto nível a serem concluídos, em vez de instruções passo a passo para realizar o teste.
- Esses casos de teste são usados para analisar como os usuários geralmente abordam e usam um aplicativo.
- Por exemplo, verificar se o usuário pode adicionar mais de um item ao carrinho de compras em um site de compras online e como é essa experiência?

CASO DE TESTE X CENÁRIO DE TESTE

Caso de Teste (Test Case)

- **Propósito:** Um caso de teste é uma descrição detalhada de um conjunto específico de entradas, condições de execução e resultados esperados para validar uma funcionalidade específica do sistema. Ele é geralmente focado em testar uma única funcionalidade ou requisito do software.
- **Detalhes:** Um caso de teste inclui informações específicas, como pré-condições (estado inicial do sistema), passos a serem seguidos durante a execução do teste, dados de entrada necessários, e critérios de sucesso (resultados esperados).
- **Exemplo:** Para um sistema de login, um caso de teste poderia incluir passos como inserir um nome de usuário e uma senha válidos, clicar no botão de login e verificar se o sistema redireciona o usuário para a página inicial.

Cenário de Teste (Test Scenario)

- **Propósito:** Um cenário de teste é uma descrição de alto nível de uma funcionalidade completa do sistema ou de um processo de negócios. Ele geralmente abrange uma série de casos de teste e se concentra em um fluxo geral de eventos ou em um caminho de execução completo através do sistema.
- **Detalhes:** Um cenário de teste é menos detalhado do que um caso de teste individual. Ele fornece uma visão geral dos eventos que ocorrem em uma funcionalidade específica, sem entrar nos detalhes específicos de cada passo ou entrada de dados.
- **Exemplo:** Para um sistema de comércio eletrônico, um cenário de teste para o processo de compra poderia incluir passos de alto nível, como selecionar um produto, adicionar ao carrinho, fornecer informações de pagamento e endereço de entrega, finalizar a compra e receber uma confirmação de pedido.

Caso de teste	Cenário de Teste
Caso de teste é um conjunto de ações executadas para verificar um determinado recurso ou funcionalidade do aplicativo.	Um cenário de teste é qualquer funcionalidade do aplicativo que pode ser testada.
Caso de teste foca em como testar	O cenário de teste se concentra no que testar
O caso de teste inclui etapas de teste, dados e resultados esperados de teste	O cenário de teste inclui funcionalidades de ponta a ponta que precisam ser testadas
As equipes de controle de qualidade e desenvolvimento são responsáveis por escrever os casos de teste	Os cenários de teste são revisados por analistas de negócios ou gerentes de negócios
Isso ajuda em testes exaustivos	Isso ajuda na metodologia ágil de testar a personalidade geral
Requer mais tempo, esforço e recursos	Requer menos tempo e esforços em comparação

CASO DE TESTE X CENÁRIO DE TESTE

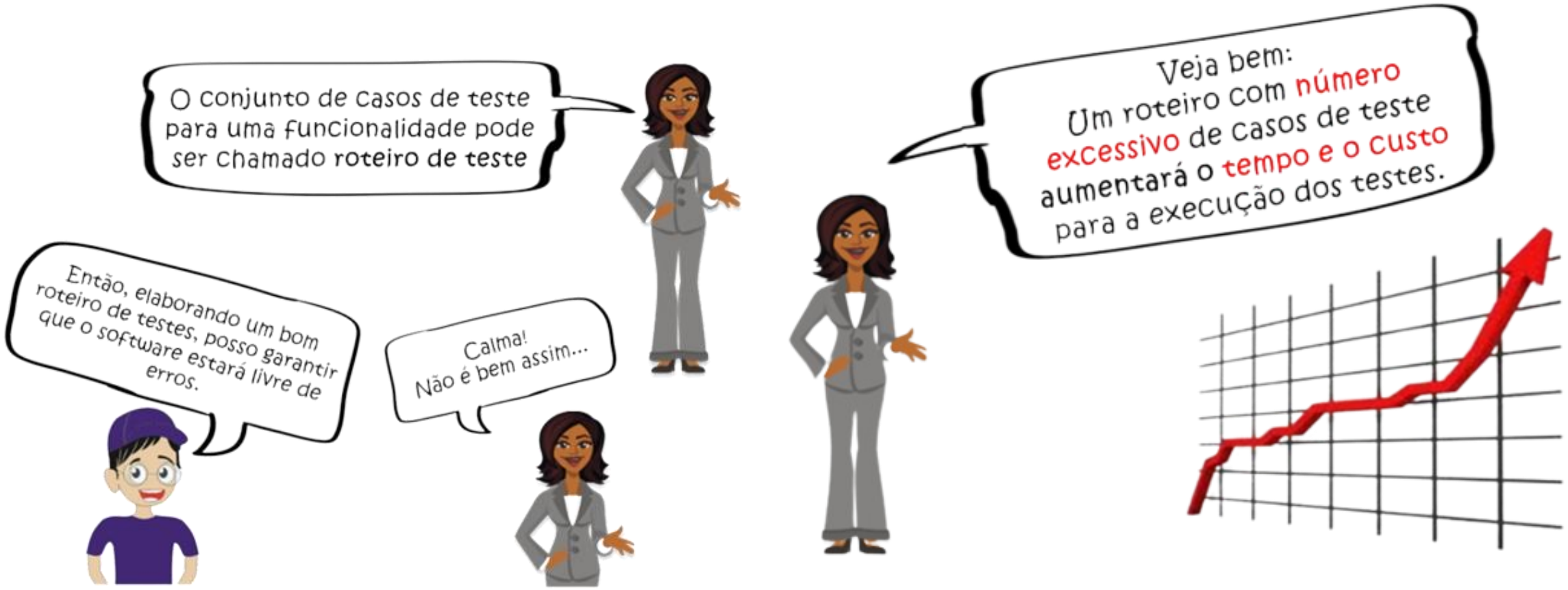
CASOS DE TESTE - ESCRITA

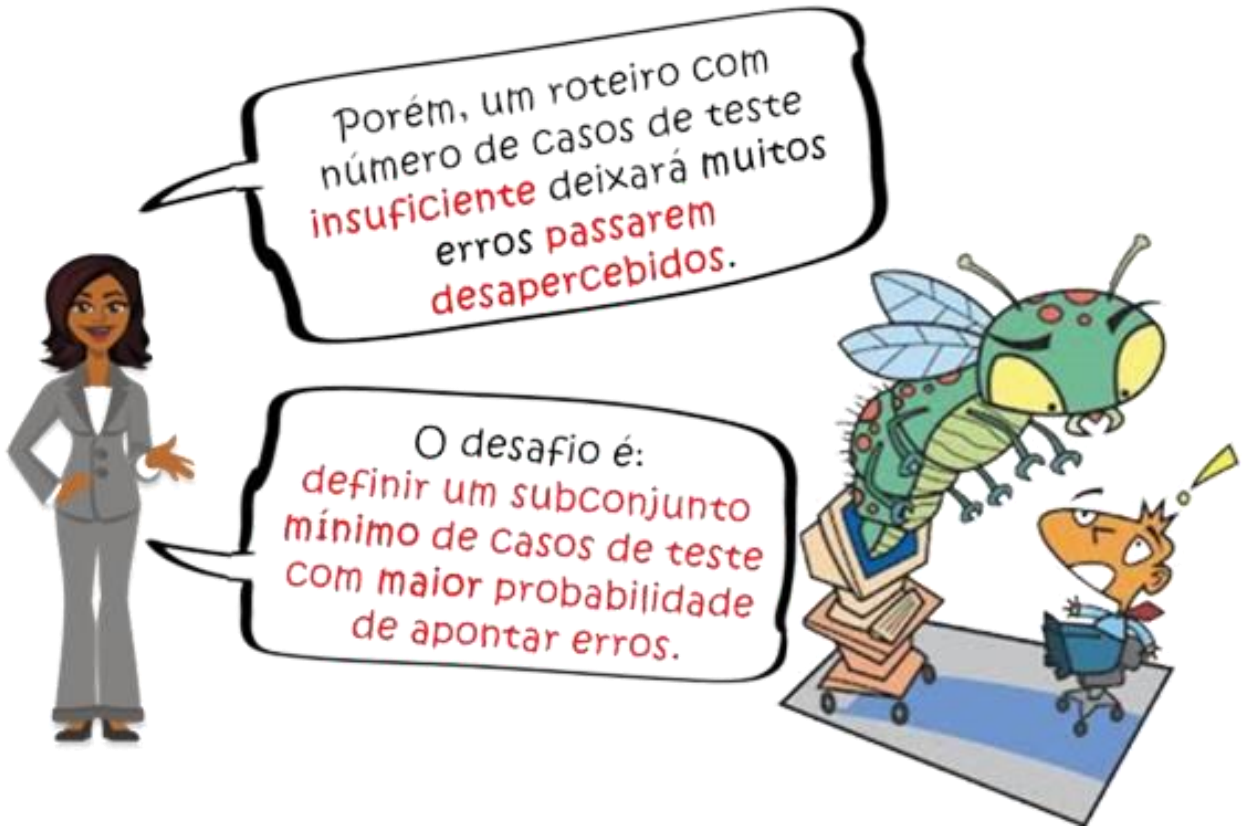
- **ID do caso de teste –** Todo e qualquer caso de teste deve ser representado com um ID exclusivo. Isso torna mais fácil e conveniente entender e distinguir entre eles.
- **Descrição do caso de teste –** Cada caso de teste deve ter uma descrição adequada que consiste em detalhes importantes, como qual recurso, unidade ou função está sendo testada e o que deve ser verificado.
- **Pré-condições –** Acumulamos também as condições que devem ser observadas durante a execução do teste.
- **Etapas de teste –** Para poder executar um teste corretamente, você deve entender corretamente como executar esse caso de teste específico. Portanto, escreva as etapas para executar o caso de teste em uma linguagem fácil e compreensível.
- **Informações de teste –** Reúna os dados necessários para a execução do caso de teste de forma precisa e concisa. Acumule esses dados em um documento claro e abrangente.
- **Resultado desejado –** Explicar os resultados desejados do caso de teste. Por exemplo, ao entrar no botão de login, o usuário será logado no site com sucesso.
- **Pós-condições –** Essas são as condições que devem ser atendidas após a execução bem-sucedida do caso de teste.
- **Resultado real –** Esses são os resultados que obtemos após a execução do caso de teste. Com base nesses resultados, descobrimos se o caso de teste foi aprovado ou reprovado.
- **Status –** Finalmente, encontramos o status do caso de teste como aprovado ou reprovado com base nos resultados reais que obtemos da execução do caso de teste. Se obtivermos os resultados desejados, o teste será marcado como aprovado, caso contrário, será marcado como reprovado. Se o teste falhar, ele passa pelo ciclo de vida do bug para ser corrigido pelos desenvolvedores.

DICAS

Algumas dicas que se deve ter em mente ao escrever um:

- Escreva em linguagem fácil, compreensível e abrangente
- Seu caso de teste deve ressoar com a perspectiva do usuário
- Aloque um ID exclusivo para cada caso de teste. Isso ajudará na rastreabilidade eficiente
- Os pré-requisitos devem ser citados de forma adequada e clara.
- Os dados de teste devem ser definidos adequadamente para avaliar as áreas funcionais
- Certifique-se de que os resultados esperados sejam declarados claramente
- As etapas do teste também devem ser listadas claramente
- Você deve garantir que fornece detalhes adequados sobre o ambiente necessário para executar o teste
- Consulte alguns veteranos e colegas para revisar os casos de teste que você escreve e reconheça suas opiniões também.


O conjunto de casos de teste para uma funcionalidade pode ser chamado roteiro de teste
Então, elaborando um bom roteiro de testes, posso garantir que o software estará livre de erros.
Calma!
Não é bem assim...
Veja bem:
Um roteiro com número excessivo de casos de teste aumentará o tempo e o custo para a execução dos testes.


Porém, um roteiro com número de casos de teste insuficiente deixará muitos erros passarem desapercebidos.
O desafio é: definir um subconjunto mínimo de casos de teste com maior probabilidade de apontar erros.

CASOS DE TESTE - COMPOSIÇÃO

Casos de teste são compostos de **duas partes obrigatórias** e outras opcionais:

- **Entrada;**
- **Saída esperada;**
- Pré-condição;
- Ordem de execução.

ENTRADAS

Geralmente identificadas como dados fornecidos via teclado para o programa executar;

Entretanto, os dados de entrada podem ser fornecidos por outros meios, tais como:

- Dados oriundos de outro sistema servem de entrada para o programa;
- Dados fornecidos por outro dispositivo;
- Dados lidos de arquivos ou banco de dados;
- O estado do sistema quando os dados são recebidos;
- O ambiente no qual o programa está executando.

SAÍDAS OBTIDAS

A mais comum é aquela apresentada na tela do computador;

Além dessa, as saídas podem ser enviadas para:

- Outro sistema interagindo com o programa em teste;
- Dados escritos em arquivos ou banco de dados;
- O estado do sistema ou o ambiente de execução podem ser alterados durante a execução do programa.

ORÁCULO DE TESTES

Corresponde a um mecanismo (programa, processo ou documento) que indica ao projetista de casos de testes se a saída obtida para um caso de teste **é aceitável** ou não.

Durante o projeto de um caso de teste, ele determina a corretude da saída obtida pelo teste.

Responsável por verificar e decidir após a execução de um programa em teste se o resultado obtido está de acordo com o resultado esperado (isto é, se o caso de teste passou ou falhou).

Esse papel pode ser desempenhado pelo próprio testador ou por alguma ferramenta de teste automatizado.

ORDEM DE EXECUÇÃO

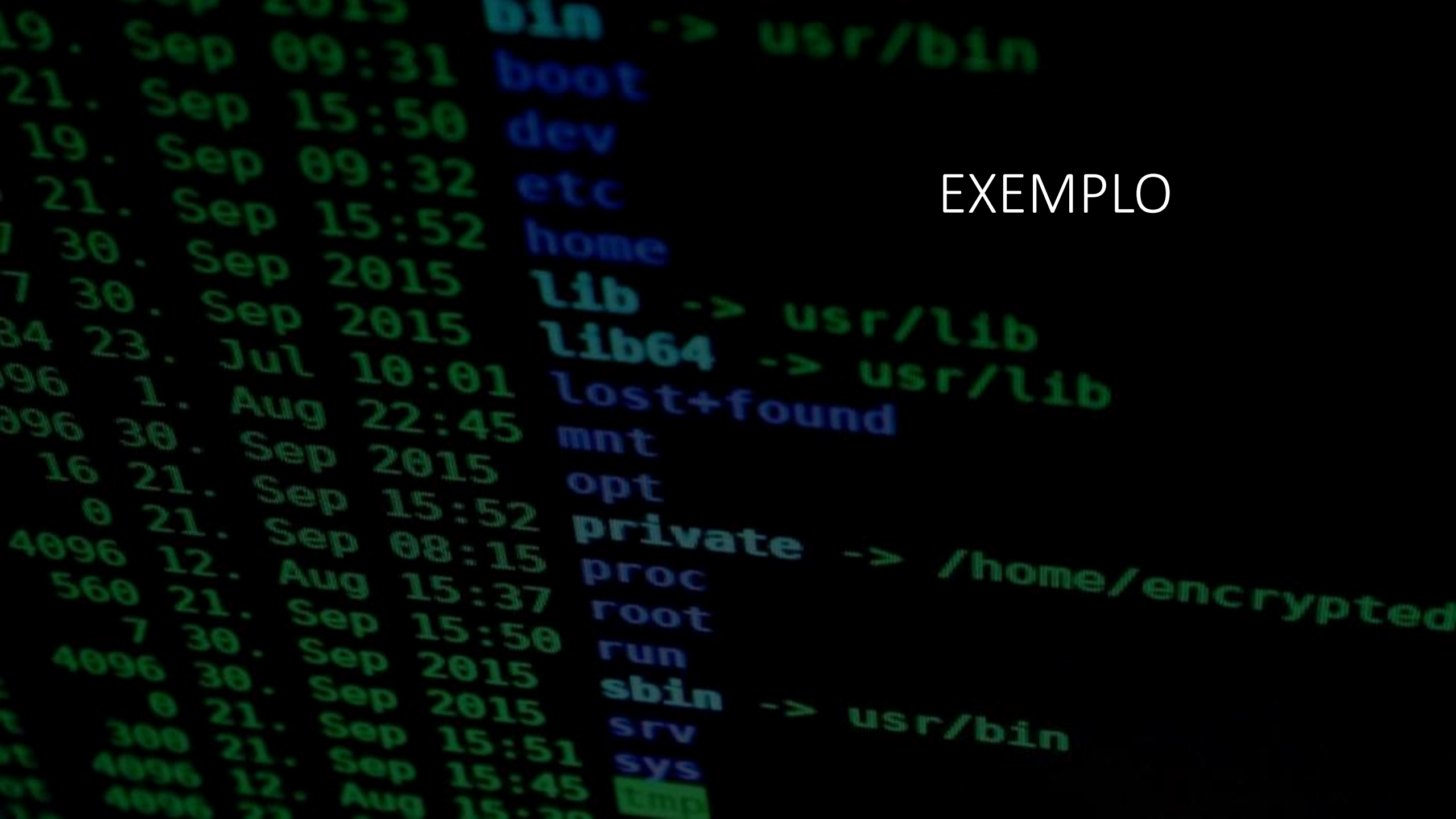
Existem dois estilos de projeto de casos de teste relacionados com a ordem de execução:

- **Dependente da ordem de execução** - quando os casos de teste devem ser executados um após o outro, em uma ordem específica
  - EX: No teste de um CRUD, o teste de inserção deve vir antes do teste de remoção
- **Independentes da ordem de execução** – onde a ordem não é importante

PRÉ-CONDIÇÕES

Referem-se aos estados ou configurações específicas que devem ser estabelecidas antes da execução de um caso de teste. Elas são cruciais para garantir que o teste seja realizado sob condições adequadas e que os resultados sejam válidos e confiáveis.

- **Ambiente de Teste Configurado:** O ambiente no qual o teste será executado deve estar configurado corretamente. Isso pode incluir a instalação de software específico, configurações de rede, servidores de banco de dados, entre outros.
- **Dados de Teste Preparados:** Dados específicos podem ser necessários para testar determinadas funcionalidades. Isso inclui, mas não se limita a, contas de usuário, dados de entrada para testar validações de formulário, ou arquivos com formatos específicos.
- **Estado do Sistema:** O sistema pode precisar estar em um estado particular antes de um teste ser iniciado. Por exemplo, um teste que verifica a funcionalidade de logout pode exigir que um usuário esteja logado inicialmente.
- **Dependências Satisfeitas:** Algumas funcionalidades podem depender de outros componentes ou serviços. As pré-condições podem incluir a disponibilidade e o correto funcionamento dessas dependências.
- **Permissões e Acessos:** Dependendo do que está sendo testado, pode ser necessário garantir que o perfil de teste tenha as permissões adequadas para realizar ações específicas no software.


bin -> usr/bin
boot
dev
etc
home
lib -> usr/lib
lib64 -> usr/lib
lost+found
mnt
opt
private -> /home/encrypted
proc
root
run
sbin -> usr/bin
srv
sys
tmp

EXEMPLO

EXEMPLO

- **Um programa, que verifica se o identificador de uma variável é válido para a Linguagem C**

Lembram das especificações para o identificador de uma variável em C?

- Deve iniciar com uma letra ou com _
- Depois, pode haver uma sequência de caracteres alfanuméricos e o _

EXEMPLO

- Quais casos de teste você usaria para esse programa?

CT01 *(“cont”, “Válido”);*

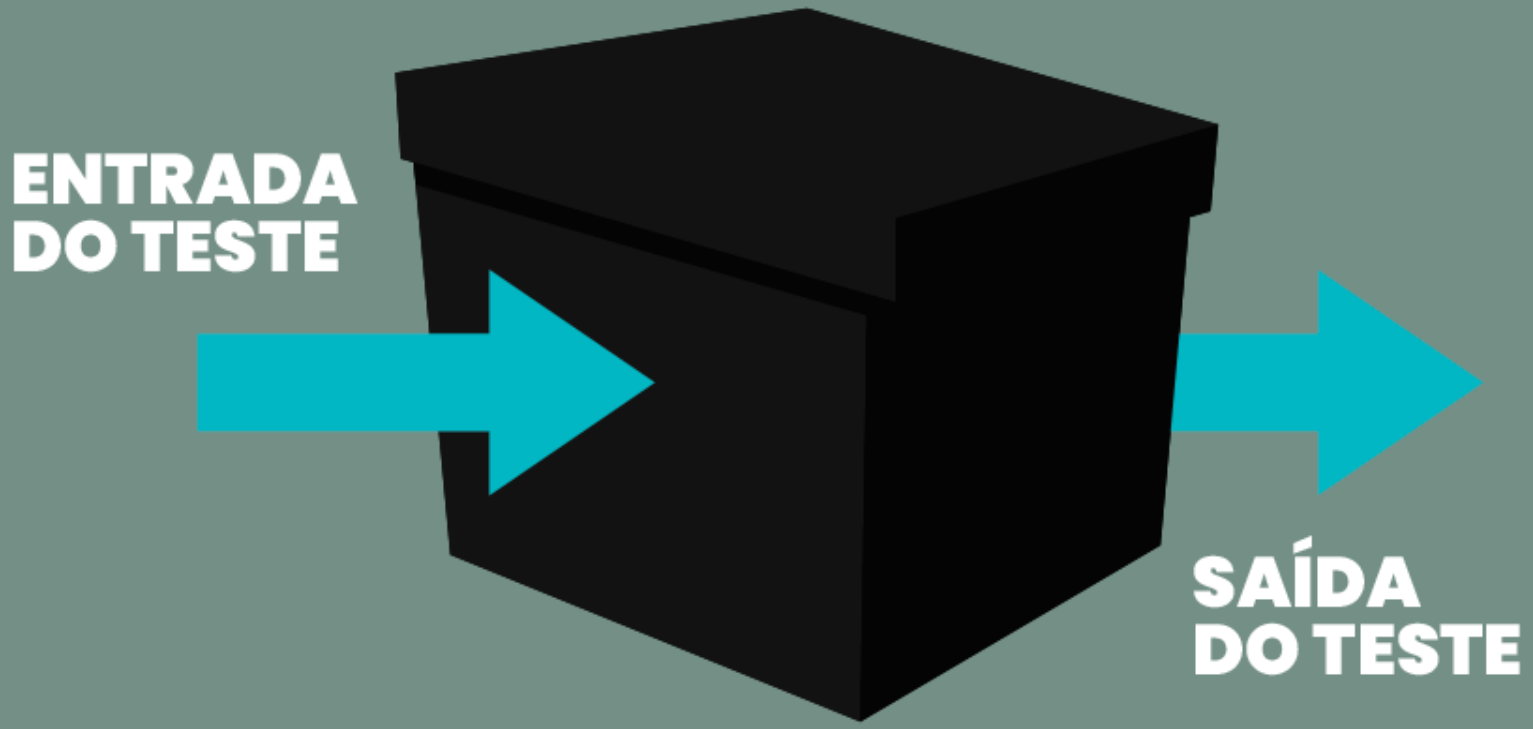
CT02 *(“1x”, “Inválido”);*

- Esses 2 casos de teste são suficientes para o teste?

CT01 *(“cont”, “Válido”);*
CT02 *(“1x”, “Inválido”);*
CT03 *(“cont#”, “Inválido”);*
CT04 *(“_1”, “Válido”);*
CT05 *(“_cont”, “Válido”);*
CT06 *(“cont_”, “Válido”);*

Roteiro de teste

TÉCNICA DO TESTE FUNCIONAL



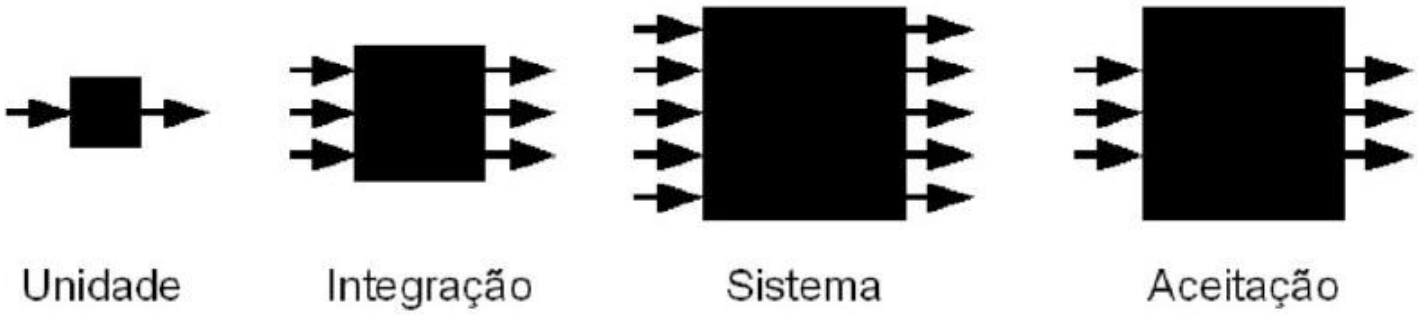
- A Técnica Funcional considera o produto em teste como uma caixa-preta da qual só se conhece a entrada e saída, ou seja, nenhum conhecimento de como o produto é internamente é utilizado;
- Critérios dessa técnica baseiam-se somente na especificação de requisitos para derivar os requisitos de testes;

APLICAÇÃO DA TÉCNICA DO TESTE FUNCIONAL

1. A especificação de requisitos é analisada
2. Entradas válidas são escolhidas (com base na especificação) para determinar se o produto em teste se comporta corretamente
3. Entradas inválidas também são escolhidas para verificar se são detectadas e manipuladas adequadamente
4. As saídas esperadas para as entradas escolhidas são determinadas
5. Os casos de testes são construídos, e o conjunto de teste é executado
6. As saídas obtidas são comparadas com as saídas esperadas
7. Um relatório é gerado para avaliar o resultado dos testes

APLICAÇÃO DA TÉCNICA DO TESTE FUNCIONAL

Por ser independente da implementação, critérios da técnica funcional podem ser utilizados em todas as fases de teste;



DESVANTAGENS

- Dependente de uma boa especificação de requisitos o que, em geral, não é bem feito.
- Não é possível garantir que partes essenciais ou críticas do software sejam executadas.
- Para encontrar todos os defeitos utilizando a técnica funcional é necessário o teste exaustivo.
  - Como testar todas as possíveis entradas para um compilador? (Myers, 1979).

VANTAGENS

- Pode ser utilizado em todas as fases de teste;
- Independente do paradigma de programação utilizado;
- Eficaz em detectar determinados tipos de erros;
  - Funcionalidade ausente, por exemplo

PARTICIONAMENTO DE EQUIVALÊNCIA

PARTICIONAMENTO DE EQUIVALÊNCIA

- Critério utilizado para reduzir o número de casos de teste procurando garantir uma boa cobertura das funcionalidades do produto em teste;
- Empregado intuitivamente pelos programadores mesmo sem conhecer o critério;
- A ideia é dividir as entradas em conjuntos que satisfaçam as especificações do software/módulo a ser testado;

PARTICIONAMENTO DE EQUIVALÊNCIA

- Como criar casos de teste para esse módulo?

FAIXA ETÁRIA	SITUAÇÃO
0-15	NÃO EMPREGAR
16-17	PODE SER EMPREGADO EM TEMPO PARCIAL
18-64	PODE SER EMPREGADO EM TEMPO INTEGRAL
65-99	NÃO EMPREGAR

- Dada a especificação do programa, fica claro que não é necessário testar para todos os valores 0, 1, 2, · · · , 14, 15 e 16, por exemplo;
- Apenas um valor desse intervalo precisa ser testado;

PARTICIONAMENTO DE EQUIVALÊNCIA

- Qualquer valor dentro do intervalo tem a mesma importância, ou seja, qualquer valor escolhido é adequado;
- O mesmo se aplica para os demais intervalos de dados;
- Tais intervalos determinam o que é chamado de classe de equivalência
- Qualquer valor no intervalo de uma classe é considerado equivalente em termos de teste. Assim sendo:
    - Se um caso de teste de uma classe de equivalência revela um erro, qualquer caso de teste da mesma classe também revelaria e vice-versa.

PARTICIONAMENTO DE EQUIVALÊNCIA

FAIXA ETÁRIA	SITUAÇÃO
0-15	NÃO EMPREGAR
16-17	PODE SER EMPREGADO EM TEMPO PARCIAL
18-64	PODE SER EMPREGADO EM TEMPO INTEGRAL
65-99	NÃO EMPREGAR

- Observe que com esse critério de teste o número de casos de teste é reduzido de 100 para 4 (um para cada classe de equivalência).

1. Identificar as classes de equivalência (requisitos de teste do critério).
2. Criar um caso de teste para cada classe de equivalência válidas
3. (usando entradas válidas);
4. Criar um caso de teste para cada classe de equivalência inválida (entradas inválidas são grandes fontes de defeitos);
5. Casos de teste adicionais podem ser criados caso haja tempo e recursos suficientes.
6. Com base em sua experiência, o(a) testador(a) pode criar casos de teste adicionais

EXERCÍCIOS

1. **Validação de Idade:** Uma aplicação exige que o usuário tenha entre 18 e 65 anos para se registrar.
2. **Validação de Número de Telefone:** Uma aplicação requer que o número de telefone digitado tenha exatamente 10 dígitos.
3. **Validação de Nota de Exame:** Um sistema acadêmico requer que as notas dos alunos estejam entre 0 e 100. Qualquer nota fora deste intervalo é considerada inválida.
4. **Validação de Senha:** Uma aplicação exige que a senha tenha entre 8 e 16 caracteres, incluindo pelo menos uma letra maiúscula, uma letra minúscula e um número.

ANÁLISE DO VALOR LIMITE

ANÁLISE DO VALOR LIMITE

- Basicamente, de acordo com a especificação, exercitar os limites do domínio de entrada;
- Um dos critérios de teste mais básico que existe;

FAIXA ETÁRIA	SITUAÇÃO
0-15	NÃO EMPREGAR
16-17	PODE SER EMPREGADO EM TEMPO PARCIAL
18-64	PODE SER EMPREGADO EM TEMPO INTEGRAL
65-99	NÃO EMPREGAR

1. Identificar as classes de equivalência;
2. Identificar os limites de cada classe;
3. Criar casos de teste para os limites escolhendo:
    a) Um ponto abaixo do limite;
    b) O limite;
    c) Um ponto acima do limite;
4. Observe que “acima” e “abaixo” são termos relativos e dependente do valor dos dados.
    a) Números inteiros: limite = 16; abaixo = 15; acima = 17.
    b) Números reais: limite = $5,00; abaixo = $4,99; acima = $5,01.

ANÁLISE DO VALOR LIMITE

- Valores limites a serem considerados:

FAIXA ETÁRIA	SITUAÇÃO
0-15	NÃO EMPREGAR
16-17	PODE SER EMPREGADO EM TEMPO PARCIAL
18-64	PODE SER EMPREGADO EM TEMPO INTEGRAL
65-99	NÃO EMPREGAR

- {−1, 0, 1}, {13, 14, 15}
- {16, 17, 18}, {16, 17, 18}
- {17, 18, 19}, {63, 64, 65}
- {64, 65, 66}, {98, 99, 100}

EXERCÍCIO

- Considera-se um sistema de notas de uma universidade de Manaus:
- Entrada de dados: N1 e N2, as notas das duas avaliações bimestrais, que estão entre 0.0 e 10.0
- Realiza-se uma média aritmética das notas;
  - Se nota >= 7.0 o aluno é considerado aprovado direto
  - Se nota < 7.0 e >= 5.0, o aluno irá para uma prova final
  - Se nota < 5.0, o aluno já está reprovado
- Gere casos de teste de acordo com o formato aprendido em sala de aula.
  - 7 min sozinhos, depois vamos corrigir juntos!

EXERCÍCIO

a) Validação de Idade: Uma aplicação exige que a idade do usuário esteja entre 18 e 65 anos para se registrar.

a) Quais são os valores limite para essa faixa de idade?

b) Crie casos de teste com base na análise de valor limite.

c) Teste valores abaixo, nos limites e acima da faixa permitida.

b) Quantidade de Itens no Carrinho: Uma loja virtual permite que o cliente adicione entre 1 e 50 itens no carrinho.

a) Identifique os valores limite para a quantidade de itens no carrinho.

b) Proponha 6 casos de teste, testando os valores abaixo, nos limites e acima do intervalo permitido.

TABELA DE DECISÃO

TABELA DE DECISÃO

- Ferramenta excelente para capturar certos tipos de requisitos do sistema e documentar soluções interna de projetos;
- Utilizados para armazenar regras de negócio complexas que o sistema deve implementar;
- Além disso, podem ser utilizadas pelos testadores como um guia para a criação de casos de teste.

TABELA DE DECISÃO

- Suponha que uma companhia de seguros oferece desconto especial para motoristas que são casados e/ou com bom desempenho escolar;
- Desse modo, têm-se duas condições, cada uma com dois possíveis valores, resultando em $2^2 = 4$ regras;
- O testador deve verificar se todas as combinações foram definidas.

	REGRA 1	REGRA 2	REGRA 3	REGRA 4
CONDIÇÕES				
CASADO(A)?	SIM	SIM	NÃO	NÃO
BOM DESEMPENHO ESCOLAR ?	SIM	NÃO	SIM	NÃO

TABELA DE DECISÃO

- Em seguida, para cada regra, uma determinada ação deve ser disparada.
- No caso do sistema, essa ação corresponde a um determinado valor de desconto no seguro.

	REGRA 1	REGRA 2	REGRA 3	REGRA 4
CONDIÇÕES				
CASADO(A)?	SIM	SIM	NÃO	NÃO
BOM DESEMPENHO ESCOLAR ?	SIM	NÃO	SIM	NÃO
AÇÕES				
DESCONTO (R$)	60 %	25 %	50 %	0

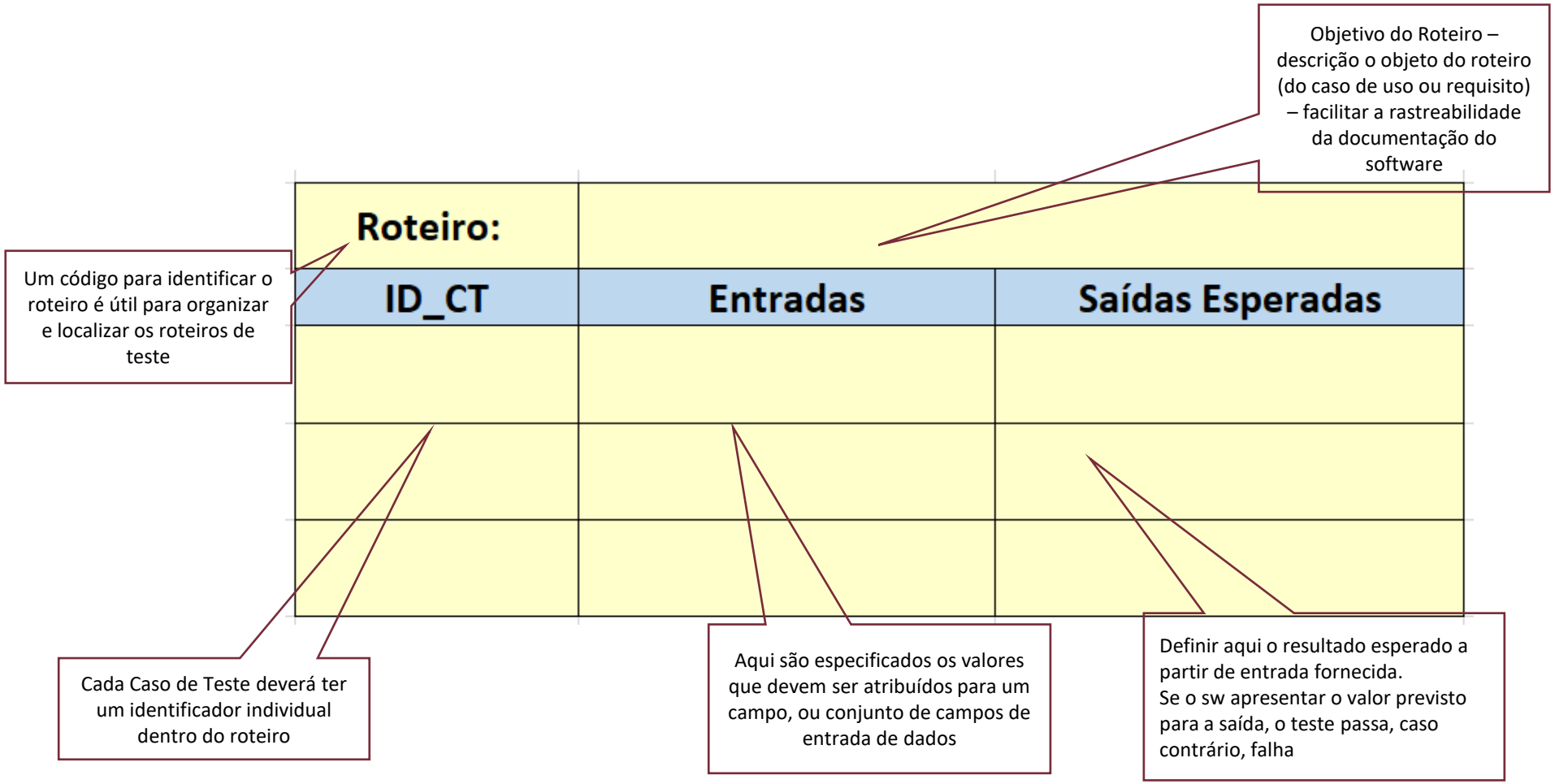
ROTEIRO DE TESES

ROTEIRO DE TESTE

- Um roteiro é um conjunto de casos de teste necessárias para testar alguma funcionalidade do software
- Na automação os roteiros serão convertidos em scripts de teste, mas no processo manual pode-se usar o template ao lado para escrever os casos de teste

Roteiro:		
ID_CT	Entradas	Saídas Esperadas

ROTEIRO DE TESTE



ROTEIRO DE TESTE

- Roteiro de teste para testar o módulo do sistema de venda de ingresso, que concede desconto de acordo com o tipo de pessoa, conforme o seguinte:
- Se estudante ou Idoso ganha 50% de desconto, se PCD não paga, e valor integral para qualquer outro caso.

Roteiro: **R1**	Testar módulo de cálculo de desconto	
ID_CT	**Entradas**	**Saídas Esperadas**
CT01	Tipo: Estudante ValorIngresso:R$ 30,00	Valor a Pagar: R$ 15,00
CT02	Tipo: Não estudante ValorIngresso:R$ 30,00	Valor a Pagar: R$ 30,00
CT03	Tipo: PCD ValorIngresso: R$ 30,00	Valor a Pagar: R$ 0,00
CT03	Tipo: Idoso ValorIngresso: R$ 30,00	Valor a Pagar: R$ 15,00

EXERCÍCIOS

1. O programa deverá calcular 2% de multa, se a data de pagamento for posterior a de vencimento.
2. Dada a idade do leitor, o programa informa se ele: Não pode votar, É obrigado a votar ou Tem voto facultativo.
3. Validação de Saldo Bancário: Um banco permite que o saldo de uma conta corrente esteja entre R$0,00 e R$100.000,00.
    a) Quais são os valores limite para esse intervalo de saldo?
    b) Crie 4 casos de teste para verificar valores no limite e valores inválidos fora do limite.

EXERCÍCIOS

1. Peso de Bagagem: Uma companhia aérea permite que a bagagem despachada tenha entre 0kg e 23kg.
    a) Identifique os valores limite para o peso da bagagem.
    b) Proponha casos de teste que testem valores abaixo, dentro e acima do limite.
2. **Validação de Número de Telefone:** Um formulário exige que o número de telefone digitado tenha exatamente 10 dígitos.
3. **Validação de Quantidade de Itens no Carrinho:** Um sistema de compras online permite que o usuário adicione de 1 a 50 itens ao carri

EXERCÍCIOS

1. **Regras de Desconto em Loja Virtual:** Uma loja virtual tem as seguintes regras de desconto:
    - Se o cliente é VIP, ele recebe 10% de desconto.
    - Se o cliente compra mais de R$ 500,00, ele recebe 5% de desconto.
    - Se é uma promoção especial, todos recebem 3% de desconto.

As ações possíveis são:

- Aplicar 10% de desconto.
- Aplicar 5% de desconto.
- Aplicar 3% de desconto.
- Não aplicar desconto.

Tarefa:

- Construa uma tabela de decisão com todas as combinações dessas três condições.
- Especifique as ações que devem ser tomadas para cada combinação.
- Crie os casos de teste baseados nas combinações da tabela.

EXERCÍCIOS

1. **Sistema de Aprovação de Empréstimo:** Um banco possui um sistema que aprova empréstimos com base nas seguintes condições:
    - O cliente deve ter uma renda mensal superior a R$ 3.000,00.
    - O cliente deve ter mais de 18 anos.
    - O cliente não pode estar com o nome sujo (restrições no crédito).

As ações possíveis são:

- Aprovar empréstimo.
- Negar empréstimo.

Tarefa:

- Construa uma tabela de decisão para todas as combinações dessas três condições.
- Indique quais ações devem ser tomadas para cada combinação.
- Crie os casos de teste para as combinações resultantes.

REFERÊNCIAS

- WILLIAMS, M., SUCCI, G. e MARCHESI, L. Traditional and Agile Software Engineering. Ch 8 — Black Box Testing. Ed. Addison-Wesley, 2003
- Syllabus Foundation Level — BSTQB — https://www.bstqb.org.br/uploads/syllabus/syllabus_ctfl_2018br.pdf
- International Software Testing Qualifications Board (ISTQB). (2018). ISTQB® Glossary of Testing Terms. ISTQB.
- GONÇALVEZ, Priscila F.; BARRETO, Jeanine S.; ZENKER, Aline M.; et al. Testes de software e gerência de configuração. Porto Alegre: Grupo A, 2019. E-book. ISBN 9788595029361.
- DELAMARO, Marcio. Introdução ao Teste de Software. Rio de Janeiro: Grupo GEN, 2016. E-book. ISBN 9788595155732.
- SILVA, Fabiana B. Gerenciamento de Projetos Fora da Caixa. Rio de Janeiro: Editora Alta Books, 2016. E-book. ISBN 9788550809632.
- XAVIER, Carlos Magno da S. Gerenciamento de projetos: como definir e controlar o escopo do projeto. Rio de Janeiro: Grupo GEN, 2018. E-book. ISBN 9788553131204.