

Teste de Software para Web

Casos de Teste e Critérios de teste

MSc. Jonathas Silva dos Santos

Material adaptado, de acordo com a licença Creative Commons, do curso Testes - Teoria e Prática, de Auri Marcelo Rizzo Vincenzi, Márcio Eduardo Delamaro e José Carlos Maldonado. Todos os direitos reservados.



Legendas



Termo ou assunto que será abordado com detalhes em nas próximas aulas;



Termo ou assuntos que valem uma pesquisa posterior de vocês;

Casos de Teste

Casos de Teste - Conceito

É um conjunto de entradas de teste, condições de execução e resultados esperados desenvolvidos para um objetivo específico, como testar o caminho de determinado programa ou verificar um requisito específico;

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

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

Caso de Teste - 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?

1. Deve **iniciar com uma letra ou com _**;
2. Depois, pode haver uma seq. de caracteres alfanuméricos e o _;

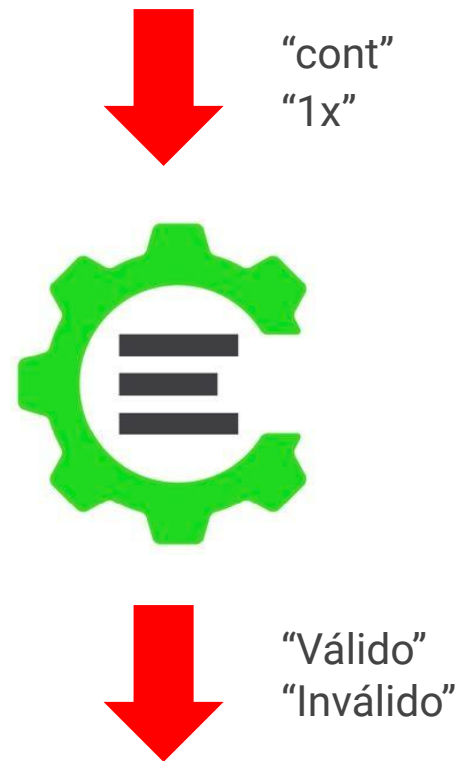


Caso de Teste - Exemplo

Quais casos de teste você usaria para esse programa?

CT01 ("cont", "Válido");

CT02 ("1x", "Inválido");



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 teste

Todas as formas de entrada e saída são relevantes;

Durante o projeto de um caso de teste, determinar a correção da saída esperada é função do oráculo (**oracle**);

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

Ordem de execução

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

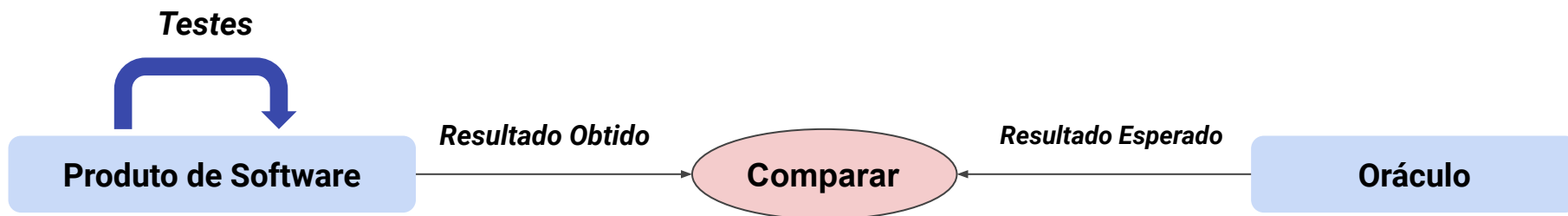
- Casos de teste em **cascata**, quando os casos de teste devem ser executados um após o outro, em uma ordem específica;
 - Ex: No teste de um CRUD, o teste de inserção deve vir antes do teste de remoção;
- Casos de teste **independentes**, a ordem não é importante;

Caso de Teste - Execução

O segredo do sucesso do teste está no projeto dos casos de teste, dado que um programa necessita de entradas para executar;

Relembrando o conceito de que ***“Teste é apenas uma amostragem”***;

O ato de testar envolve a comparação do resultado obtido na execução com o resultado esperado de acordo com o oráculo;



Técnica de Teste Funcional

Técnica Funcional - Relembrando o conceito

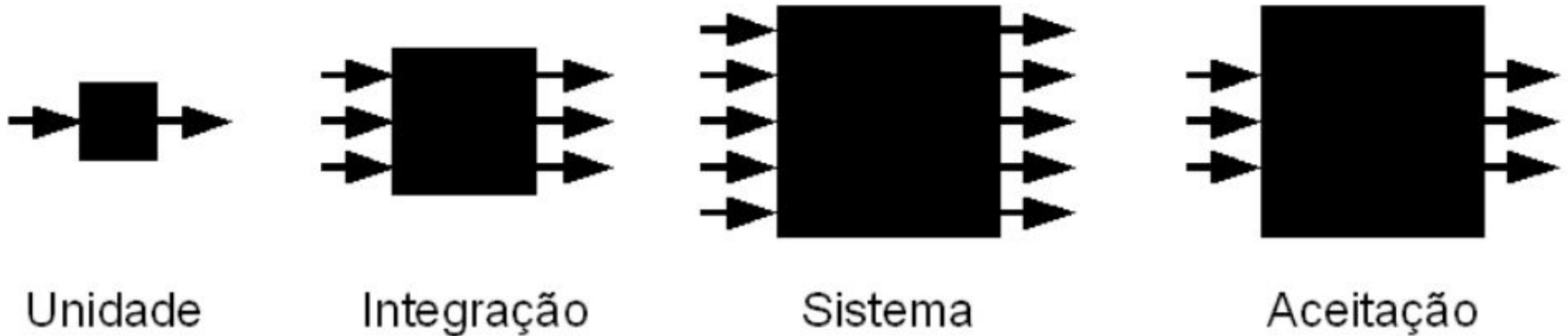
- Conforme mencionado, a Técnica Funcional considera o produto em teste como uma **caixa-preta** da qual só se conhece a entrada e saída, ou seja, nenhum conhecimento de como o produto é internamente é utilizado;
- **Crítérios** dessa técnica baseiam-se somente na especificação de requisitos para derivar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

Aplicação da técnica de 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

Técnica Funcional - Uso nos níveis de teste

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



Técnica Funcional - 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**).

Técnica Funcional - 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

Critérios da Técnica Funcional

Critérios da Técnica Funcional

- Particionamento de Equivalência (*Equivalence Partition*).
- Análise do Valor Limite (*Boundary Value Analysis*).
- Tabela de Decisão (*Decision Table*).
- Teste de Todos os Pares (*Pairwise Testing*).
- Teste de Transição de Estado (*State-Transition Testing*)
- Teste de Análise de Domínio (*Domain Analysis Testing*).
- Teste de Caso de Uso (*Use Case Testing*).



Particionamento de Equivalência

Particionamento de Equivalência - Conceito

- 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 - Exemplo

- ***Como criar casos de teste para esse módulo?***

0 – 16	Não empregar.
16 – 18	Pode ser empregado tempo parcial.
18 – 55	Pode ser empregado tempo integral.
55 – 99	Não empregar.

Particionamento de Equivalência - Exemplo

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

0 – 16	Não empregar.
16 – 18	Possível tempo parcial.
18 – 55	Possível tempo integral.
55 – 99	Não empregar.

Qual seria esse valor?

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

- 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**).

0 – 16	Não empregar.
16 – 18	Pode ser empregado tempo parcial.
18 – 55	Pode ser empregado tempo integral.
55 – 99	Não empregar.

Classes de Equivalência - Passos para aplicação

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

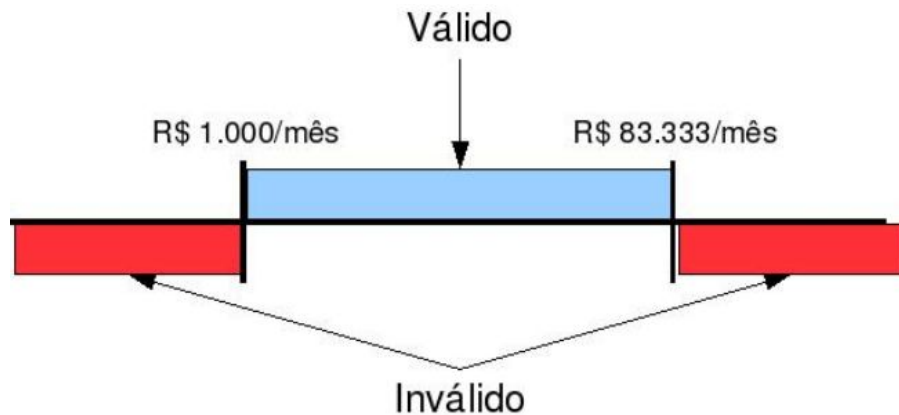
Particionamento de equivalência - Exemplo

- Testar um módulo que calcula a validade da negociação de hipotecas residenciais. O módulo deve receber como entrada a **renda mensal** dos moradores, a **quantidade de casas** da negociação, o **CPF** do titular da hipoteca e o **tipo da residência**. As seguintes regras são implementadas nesse módulo:
 - A renda dos moradores deve ser maior ou igual a R\$1000 e menor ou igual a R\$83000;
 - A negociação deve envolver no mínimo 1 e no máximo 5 casas;
 - Hipoteca somente para pessoas físicas;
 - Três tipos de hipoteca são válidas (condomínio, sobrado e casa térrea);

Particionamento de equivalência - Exemplo

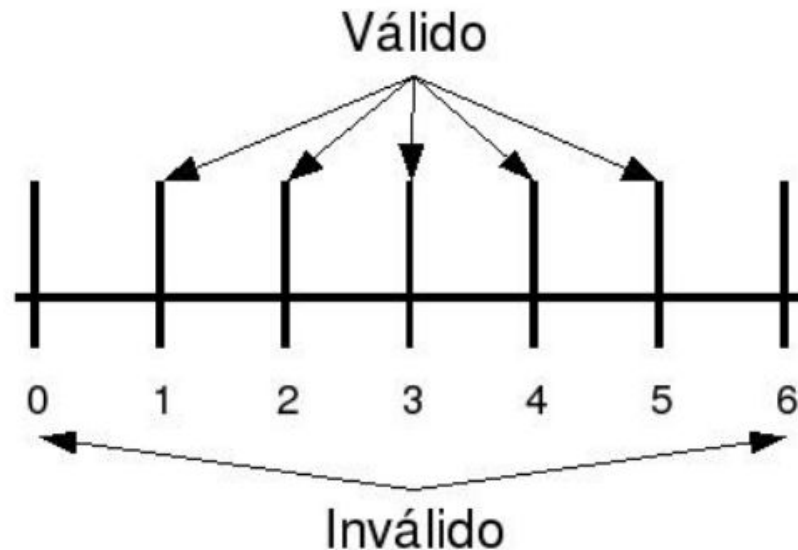
A renda para hipoteca de R\$1.000 a 83.000/mês

- Em geral são definidas duas classes inválidas e uma válida.
- Para a classe válida poderia ser escolhido R\$1.342/mês.
- Para as classes inválidas poderia ser: R\$123/mês e R\$90.000/mês.



Particionamento de equivalência - Exemplo

- Exemplo: Quantidade de casas na negociação:
 - Em geral são definidas duas classes inválidas e uma válida;
 - Para a classe válida poderia ser escolhido 2;
 - Para as classes inválidas poderia ser: -2 e 8.



(extraído de Copeland (2004))

Exemplo - Intervalo de dados simples

- Em geral são definidas uma classe inválida e uma válida;
- Para a classe válida poderia ser escolhida uma pessoa qualquer;
- Para a classe inválida deve ser escolhida uma companhia ou associação.




Válido



Inválido

Exemplo - Intervalos de múltipla escolha

- Para o intervalo válido pode-se escolher: condomínio, sobrado ou casa térrea;
- Escolher somente um ou os três?
 - Depende da criticalidade do programa em teste;
- Se forem poucos itens vale a pena selecionar um de cada;
- O mesmo para a classe inválida.



Condomínio
Sobrado
Casa térrea

Válido



Duplex
Triplex
Trailer

Inválido

Particionamento de equivalência - Exemplo

- Em geral, não há tempo para a criação de um caso de teste para cada classe válida;
 - Solução: criar o menor número possível de casos de teste que cubram todas as classes válidas.
 - Criar um caso de teste para cada classe inválida.

Renda	# Moradores	Aplicante	Tipo	Resultado
\$5.000	2	Pessoas	Condomínio	Válido
\$100	1	Pessoas	Uma família	Inválido
\$90.000	1	Pessoas	Uma família	Inválido
\$1.342	0	Pessoas	Condomínio	Inválido
\$1.342	6	Pessoas	Condomínio	Inválido
\$1.342	1	Corporação	Sobrado	Inválido
\$1.342	1	Pessoas	Duplex	Inválido

Hora de praticar

Considera-se que triângulo é válido quando nenhum de seus lados é negativo

Gere casos de teste de acordo com o formato aprendido em sala de aula

20min sozinhos, depois vamos corrigir juntos!



```
public int TipoTriang(int L1, int L2, int L3)
// Entradas: L1, L2 e L3 são os lados de
//          um triângulo
// Saídas produzidas:
//   TipoTriang = 1 se triângulo escaleno
//   TipoTriang = 2 se triângulo isósceles
//   TipoTriang = 3 se triângulo equilátero
//   TipoTriang = 4 se não é triângulo
```

Análise do Valor Limite

Análise do Valor Limite - Conceito

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

$0 < idade < 16$	Não empregar.
<i>Como derivar casos de teste para o exemplo anterior, usando esse critério?</i>	
$55 \leq idade < 99$	Não empregar.

Análise do Valor Limite - Passos para aplicação

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 - Exemplo

$0 \leq idade < 16$	Não empregar.
$16 \leq idade < 18$	Pode ser empregado tempo parcial.
$18 \leq idade < 55$	Pode ser empregado tempo integral.
$55 \leq idade < 99$	Não empregar.

Valores limites a serem considerados:

- $\{-1, 0, 1\}, \{14, 15, 16\}$
- $\{15, 16, 17\}, \{16, 17, 18\}$
- $\{17, 18, 19\}, \{53, 54, 55\}$
- $\{54, 55, 56\}, \{98, 99, 100\}$

Hora de praticar

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 ≥ 4.0 , o aluno irá para uma **prova final**
- Se nota < 4.0 , o aluno já está **reprovado**

Gere casos de teste de acordo com o formato aprendido em sala de aula

20min sozinhos, depois vamos corrigir juntos!



Tabela de Decisão

Tabela de Decisão - Conceito

- 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.

	Regra ₁	Regra ₂	...	Regra _r
Condições				
Condição ₁	<p><i>Condições de 1 a c representam as várias condições de entrada do sistema.</i></p> <p><i>Cada Regra de 1 a r representa uma combinação única de entradas que resultam na execução (“disparo”) das ações relacionadas.</i></p>			
Condição ₂				
...				
Condição _c				
Ações				
Ação ₁	<p><i>Ações de 1 a a correspondem às ações a serem tomadas pelo sistema em função das combinações das condições de entrada.</i></p>			
Ação ₂				
...				
Ação _a				

	Regra ₁	Regra ₂	...	Regra _r
Condições	<ul style="list-style-type: none"> ● Observe que as ações não dependem da ordem na qual as condições são avaliadas, apenas de seus valores ● Assume-se que todos os valores estão disponíveis simultaneamente ● Do mesmo modo, as ações dependem apenas das condições especificadas e não de entradas passadas ou do estado do sistema 			
Condição ₁				
Condição ₂				
...				
Condição _c				
Ações				
Ação ₂				
Ação ₂				
...				
Ação _a				

Tabela de decisão - Exemplo

- 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.

	Regra₁	Regra₂	Regra₃	Regra₄
Condições				
Casado(a)?	Sim	Sim	Não	Não
Bom desempenho escolar?	Sim	Não	Sim	Não

Tabela de decisão - Exemplo

- 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.

	Regra₁	Regra₂	Regra₃	Regra₄
Condições				
Casado(a)?	Sim	Sim	Não	Não
Bom desempenho escolar?	Sim	Não	Sim	Não
Ações				
Desconto (R\$)?	60	25	50	0

Hora de praticar



The image shows a mockup of a login window titled "Sistema - Tela de Entrada". It features two identical login sections. Each section has a label ("Login" or "Senha"), a text input field, and two buttons labeled "Entrar" and "Cancelar". The window has a standard title bar with minimize, maximize, and close buttons.

- **login:** código alfanumérico de 8 caracteres. Se o código é inválido ou não é reconhecido pelo sistema, este solicita ao usuário que o forneça novamente, até que um código válido seja fornecido.
- **senha:** código alfanumérico de 5 caracteres. Se a senha é incorreta, o usuário tem uma chance a mais para fornecê-la. Se ambas as tentativas falharem, o usuário deve recomeçar todo o processo.



Técnica de Teste Estrutural

Técnica Estrutural - Relembrando o conceito

- Baseia-se nos caminhos internos, estrutura e implementação do produto em teste;
- Conhecido na área por **teste caixa branca**;
- Requer conhecimento do código do produto em teste para ser aplicada;

Critérios da Técnica Funcional

- Critérios Baseados em Fluxo de Controle;
- Critérios Baseados em Fluxo de Dados.



Aplicação da técnica de Teste Estrutural

1. A implementação do produto em teste é analisada;
2. Caminhos através da implementação são escolhidos;
3. Valores de entradas são selecionados de modo que os caminhos selecionados sejam executados;
4. As saídas esperadas para as entradas escolhidas são determinadas;
5. Os casos de testes são construídos
6. As saídas obtidas são comparadas com as saídas esperadas;

Técnica Estrutural - Uso nos níveis de teste

- Critérios da técnica estrutural também podem ser utilizados em todas as fases de teste;
- São mais comuns no teste de unidade e de integração;
- Teste de caminhos:
 - Caminhos dentro de uma unidade.
 - Caminhos entre unidades.
 - Caminhos entre sub-sistemas.
 - Caminhos entre o sistema todo

Técnica Estrutural - Desvantagens

- O número de caminhos a serem executados pode ser infinito (semelhante ao teste exaustivo);
- Assume fluxo de controle correto (ou próximo do correto).
- Casos de testes são baseados em caminhos existentes:
 - Caminhos inexistentes não podem ser descobertos;
 - Determinação de caminhos não executáveis pode ser um problema.
 - Dificuldade de automação;
 - Habilidades de programação avançadas exigidas para compreender o código e decidir pela executabilidade ou não de um caminho.

Técnica Estrutural - Vantagens

- É possível garantir que partes essenciais ou críticas do programa sejam executadas;
 - **Requisito mínimo de teste:** garantir que o programa foi liberado tendo seus comandos executados ao menos uma vez por pelo menos um caso de teste;

Critério de Fluxo de Controle

Conceitos básicos

Fluxo de Controle - Conceito

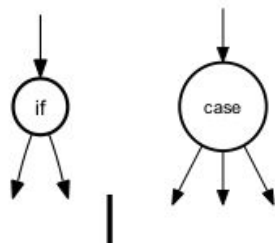
- Identificam requisitos de testes (caminhos de execução) a partir da implementação do produto em teste;
- Requer a criação e execução de casos de testes que exercite tais requisitos;
 - **Caminho**: sequência de execução de comandos que se inicia em um ponto de entrada e termina em um ponto de saída do produto em teste;
- Necessária uma abstração do código-fonte para facilitar a visualização dos caminhos

Grafo do Fluxo de Controle (CFG)

- Notação utilizada para abstrair o fluxo de controle lógico de um programa;
- Composto de **nós** e **arcos**;
- Um **nó** representa uma ou mais instruções as quais são sempre executadas em sequência, ou seja, uma vez executada a primeira instrução de um nó todas as demais instruções daquele nó também são executadas;
- Um **arco**, também chamado de ramo ou aresta, representa o fluxo de controle entre blocos de comandos (nós).



grupo de comandos executados sequencialmente do início ao fim.



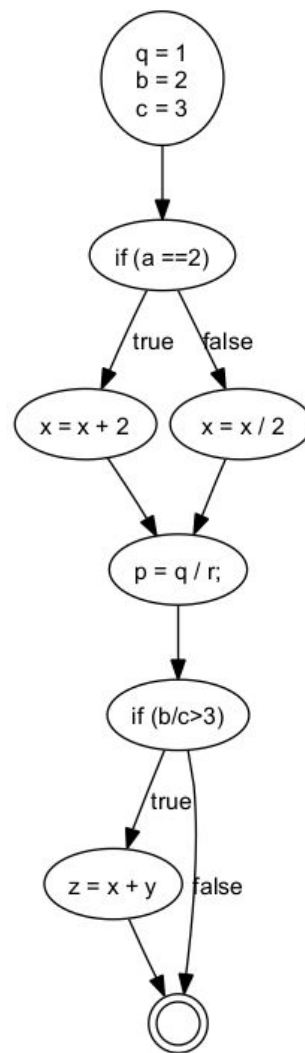
ponto de mudança de fluxo de controle.



ponto no qual fluxos de controle são unidos.

Exemplo de CFG

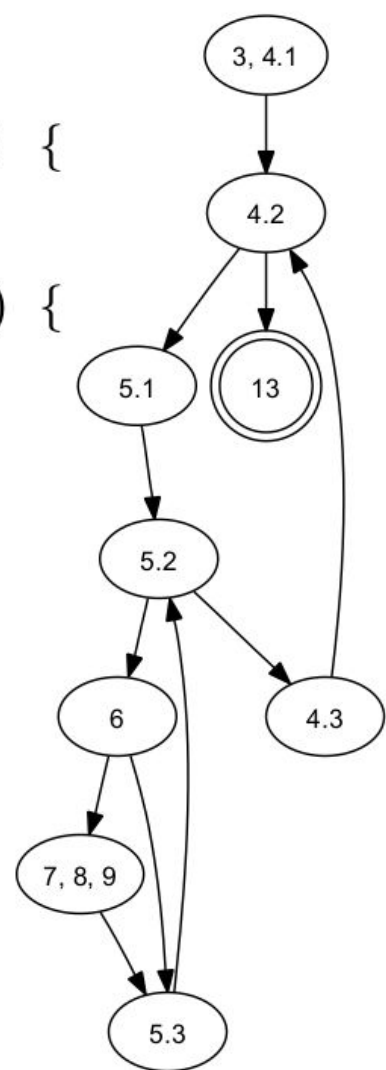
```
1  q = 1;  
2  b = 2;  
3  c = 3;  
4  if (a == 2) {  
5      x = x + 2;  
6  } else {  
7      x = x / 2;  
8  }  
9  p = q / r;  
10 if (b/c > 3) {  
11     z = x + y;  
12 }
```



```

2  public void bolha(int [] a, int size) {
3      int i, j, aux;
4      for (i = 0; i < size; i++) {
5          for (j = size - 1; j > i; j--) {
6              if (a[j - 1] > a[j]) {
7                  aux = a[j - 1];
8                  a[j - 1] = a[j];
9                  a[j] = aux;
10             }
11         }
12     }
13 }

```



Níveis de Cobertura

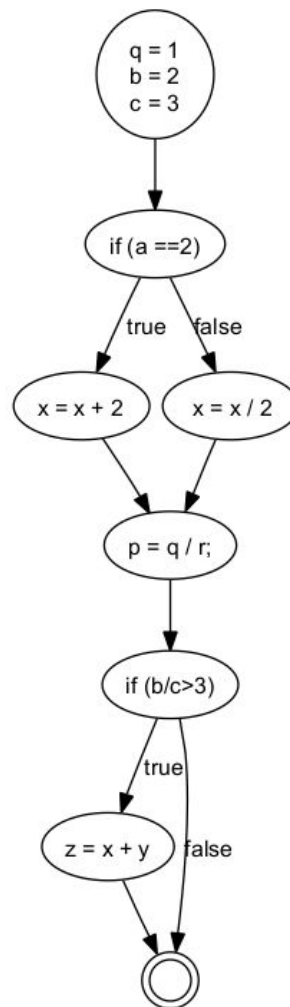
Níveis de Cobertura - Conceito

- Diferentes níveis de cobertura podem ser definidos em função dos elementos do GFC;
- **Cobertura:** porcentagem dos caminhos que foram testados versus o total de caminhos gerados.
- Oito diferentes níveis de cobertura são definidos por Copeland (2004).
- Quanto maior o nível, maior o rigor do critério de teste, ou seja, mais caso de teste ele exige para ser satisfeito
 - Nível 0 ← Nível 1 ← Nível 2 ← Nível 3 ← Nível 4 ← Nível 5 ← Nível 6
← Nível 7

Critérios de cobertura

Podemos resumir basicamente em duas:

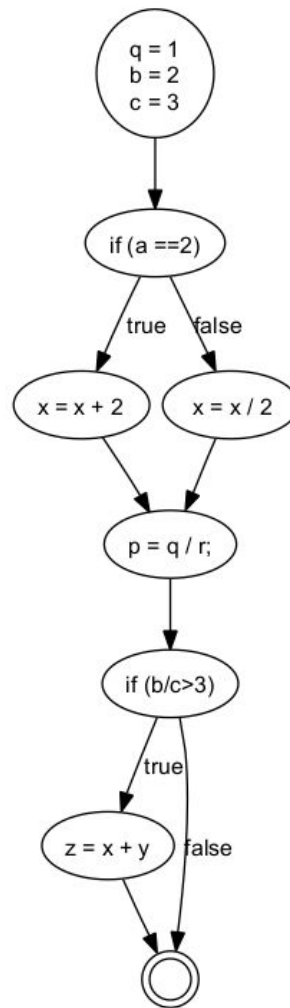
- Cobertura de nós ou comandos;
- Cobertura de condições;



Critérios de cobertura

Podemos resumir basicamente em duas:

- Cobertura de nós ou comandos;
- Cobertura de condições;



Hora de praticar


Com base no algoritmo ao lado, desenvolva casos de teste a partir da técnica caixa branca

20min sozinhos, depois vamos corrigir juntos!



Algoritmo 9 Verifica aprovação de alunos.

```
1: var frequencia, media: real
2: escreva ("digite a media e a frequencia")
3: leia(media, frequencia)
4: se frequencia  $\geq$  0.75 então
5:     se media  $\geq$  7 então
6:         escreva ("voce esta APROVADO")
7:     senão
8:         se media  $\geq$  3 então
9:             escreva ("voce esta em RECUPERACAO")
10:        senão
11:            escreva ("voce esta REPROVADO POR MEDIA")
12:        fim-se
13:    fim-se
14: senão
15:    escreva ("voce esta reprovado por FALTAS")
16: fim-se
```

E então, qual o
melhor critério?

Referências

[1] BSTQB – Brazilian Software Testing Qualifications Board.

http://www.bstqb.org.br/uploads/docs/syllabus_2007br.pdf.

[2] INFO Online. “Defeito de Software põe doentes em perigo”.

<http://info.abril.com.br/aberto/infonews/012009/16012009-19.shl>. Publicado em 16/01/2009.

[3] G1. “Show de Madonna: fãs que tiveram cartão debitado terão ingresso”.

<http://g1.globo.com/Noticias/Musica/0,,MUL749176-7085,00.html>. Publicado em 05/09/2008.

[4] Base de Conhecimento em Teste de Software. 2ª Edição. Rios, Emerson; Cristalli, Ricardo; Moreira, Trayahú & Bastos, Aderson. – S. Paulo, Martins Fontes, 2007.

[5] Alexandre Bartie. “Processo de Teste de Software – Parte 01”.

http://imasters.uol.com.br/artigo/6102/des_de_software/processo_de_teste_de_software_parte_01/. Publicado em 07/05/2007.

Referências

[6] Gustavo Quezada. “Papéis e Responsabilidades na Equipe de Testes”.

<http://www.alats.org.br/Default.aspx?tabid=206&mid=1073&ctl=Details&ItemID=109>. Publicado em 05/09/2008.

[7] Alexandre Bartie. Fábrica de Testes – Parte 01.

http://imasters.uol.com.br/artigo/4435/des_de_software/fabrica_de_testes_parte_01/. Publicado em 26/07/2006.

[8] Roberto Murillo. A evolução do teste de software.

http://imasters.uol.com.br/artigo/9369/des_de_software/a_evolucao_do_teste_de_software/. Publicado em 11/07/2008.

Read more: <http://www.linhadecodigo.com.br/artigo/2775/introducao-ao-teste-de-software.aspx#ixzz6LsVlFi2k6>