Introdução ao Teste de Software

Professor Diego Oliveira
IFRN

Introdução

- Objetivam provar que o software está fazendo o que foi programado para fazer
- Descobre erros antes da entrega
  - Entradas ou saídas inesperadas
  - Comportamento de acordo com a documentação
- São úteis tanto para o cliente quanto para os desenvolvedores
- Há vários tipos de teste
- Podem ser criados pelos desenvolvedores ou por um time de QA

Fases de Teste

- Análise e Validação
- Planejamento
- Detalhamento
  - Passos de execução
  - Decidir sobre automação
  - Ordem de execução
- Ambiente
  - Hardware
  - Software
  - Requisitos
  - Responsáveis
- Execução
- Relatório
  - Enviar resultados
  - Bugs encontrados

Teste Funcional x Teste Não-Funcional

- Teste Funcional: verifica o que o sistema faz
  - Verificar se a soma dos itens de um carrinho de compras está correta
- Teste Não-Funcional: verifica como o sistema se comporta
  - Verificar a segurança de um sistema

Técnicas de Teste Funcional

- Análise de valor limite
- Particionamento em classes de equivalência
- Grafo causa-efeito
- Error guessing
- Teste funcional sistemático
- Outros

Critérios de Análise do Valor Limite

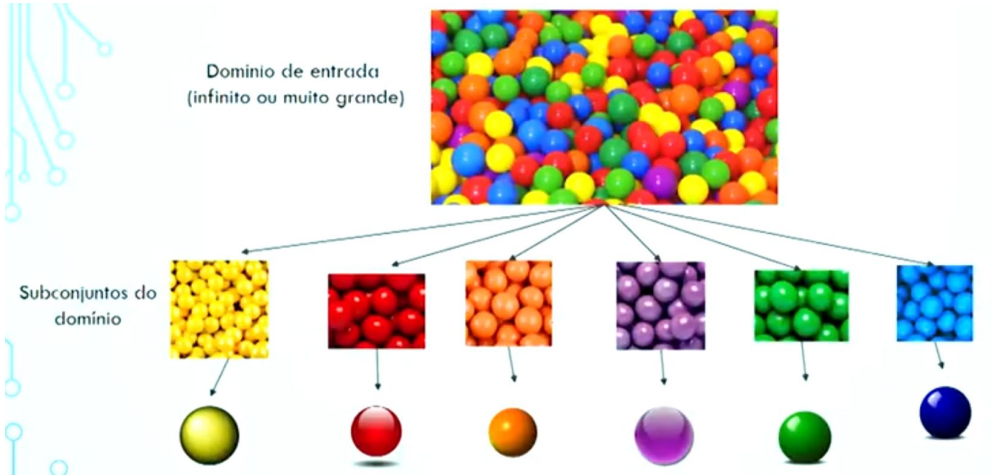
- Utilizada apenas para valores numéricos ou sequenciais
  - MAX
  - MIN

```
if(nota > 6){
    System.out.println("APROVADO");
}else{
    System.out.println("REPROVADO");
}
```

```
if(nota < 4){
    System.out.println("REPROVADO");
}else if(nota >= 4 && nota < 6){
    System.out.println("RECUPERAÇÃO");
}else{
    System.out.println("APROVADO");
}
```

Particionamento por Classes de Equivalência

- Se o domínio de entrada é muito grande, os testes ficam difíceis
- Por isso, dividem-se os testes em subconjuntos do domínio



Particionamento por Classes de Equivalência

- **Passo 1**: a partir da especificação do software, identificar as classes de equivalência
  - Procurar por termos como intervalo e conjunto na especificação do produto
- **Passo 2**: Gerar casos de teste selecionando um elemento de cada classe
  - Os casos de teste são definidos para as classes válidas e inválidas

Particionamento por Classes de Equivalência

- **Diretrizes para definir as classes:**
  - Se a condição de entrada especifica um intervalo, então define-se uma classe válida e duas inválidas
  - Se a condição de entrada especifica uma quantidade, então define-se uma classe válida e duas inválidas
  - Se a condição especifica conjuntos determinados de valores, define-se uma classe válida para cada conjunto e uma classe inválida com outro valor qualquer
    - Exemplo: tabela do IR
  - Se a condição de entrada é específica, então define-se uma classe válida e uma inválida
    - Exemplo: identificador deve iniciar com uma letra

Técnica Estrutural

- Observa-se a estrutura interna do código
- Teste de caixa branca (caixa transparente)
  - Usado para teste de unidade
  - Técnicas de seleção de dados de entrada
  - Pode-se observar o que está acontecendo durante o teste

Técnica Estrutural

- Testes estruturais
  - Teste do caminho básico
  - Teste de condição
  - Teste de fluxo de dados
  - Teste de ciclo

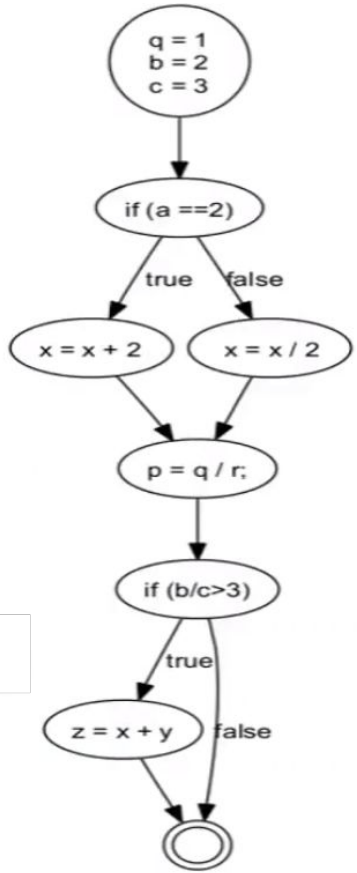
Critérios Baseados em Fluxo de Controle

- Grafo do Fluxo de Controle
  - Notação utilizada para abstrair o fluxo de controle lógico de um programa
  - Composto por **nós** e **arcos**
  - Um nó representa uma ou mais instruções as quais são executadas em sequência
  - Um arco, também chamado de ramo/**aresta**, representa o fluxo de controle entre blocos de comandos (nós)

Critérios Baseados em Fluxo de Controle

- Exemplo de Grafo do Fluxo de Controle

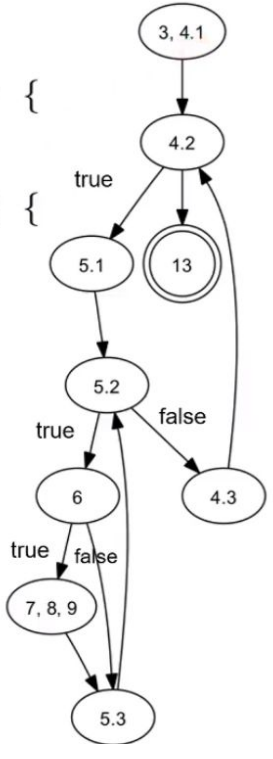
```
1   q = 1;
2   b = 2;
3   c = 3;
4   if (a == 2) {
5       x = x + 2;
6   } else {
7       x = x / 2;
8   }
9   p = q / r;
10  if (b/c>3) {
11      z = x + y;
12  }
```



Critérios Baseados em Fluxo de Controle

- Exemplo de Grafo do Fluxo de Controle por Linha

```
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      }
```



Critérios Baseados em Fluxo de Dados

- Complementar ao critério baseado em fluxo de controle
- Testa o uso de variáveis e valores resultantes em um programa
  - Exemplos:
    - referenciar variáveis não inicializadas
    - atribuição de valor a uma variável mais de uma vez
    - liberação da memória reservada por uma variável
    - atribuição de novo endereço a um ponteiro sem que este tenha sido liberado

Critérios Baseados em Fluxo de Dados

- Tipos de ocorrências de variáveis
  - **~** não existe a variável
  - **d** a variável está sendo definida
  - **u** a variável está sendo usada
  - **k** a variável está sendo destruída
- Existem 3 possibilidades de ocorrência em um dado caminho do programa:
  - **~d** a variável não existe e é definida
  - **~u** a variavél não existe e é usada
  - **~k** a variável não existe e é destruída

Critérios Baseados em Fluxo de Dados

- Tipos de ocorrências de variáveis

dd	Variável definida e redefinida	(provavelmente incorreto)
du	Variável definida e usada	(correto)
dk	Variável definida e destruída	(provavelmente incorreto)
ud	Variável usada e definida	(aceitável)
uu	Variável usada e reusada	(aceitável)
uk	Variável usada e destruída	(aceitável)
kd	Variável destruída e redefinida	(aceitável)
ku	Variável destruída e reusada	(incorreto)
kk	Variável destruída e redestruída	(provavelmente incorreto)

Tipos de Teste

- Teste Unitário
- Teste Funcional
- Teste de Integração
- Teste de Carga
- Teste de Aceitação
- Teste de Segurança
- Teste de Ponta a Ponta
- Teste de Fumaça
- Teste de Serviço
- Teste de UI
- Teste de Lançamento
- Teste de Regressão

Ferramentas e Bibliotecas de Teste

- JUnit
- Mockito
- JMeter
- Selenium
- Artillery
- ReadyAPI
- Postman
- Jenkins
- AWS Lambda/Pipeline
- Robot Framework
- Cypress

Perguntas?