**Estratégias de Testes de Software**

Nesta aula, você aprenderá sobre as diferentes estratégias e técnicas de testes aplicadas ao desenvolvimento de software, com foco nas abordagens funcionais (caixa preta) e estruturais (caixa branca).

1. **Técnicas de Teste Funcional (Caixa Preta):**
   * Baseia-se nas especificações e funcionalidades do sistema, sem levar em conta a implementação interna (código-fonte).
   * O objetivo é verificar se as funcionalidades do sistema funcionam corretamente, criando casos de teste que simulem situações de uso reais.
   * Essa técnica é ideal para cenários em que o código-fonte não está disponível, como quando o sistema é entregue por terceiros.
   * Exemplo prático: Testar o processo de login em um sistema de vendas online, verificando a usabilidade e segurança da interface.
2. **Técnicas de Teste Estrutural (Caixa Branca):**
   * Foca na análise da arquitetura interna do software, acessando o código-fonte para realizar uma cobertura completa dos caminhos e fluxos do programa.
   * Utiliza ferramentas automatizadas para gerar grafos que representam o fluxo do código, permitindo que os testes sejam planejados com base nesses grafos.
   * O objetivo é cobrir ao máximo os possíveis caminhos lógicos do sistema para encontrar falhas.
   * Exemplo prático: Validar a estrutura de um código que verifica a validade de identificadores com base em regras pré-definidas (como o tamanho e os caracteres permitidos).

**Planejamento de Testes:**

* Planejamento é essencial para garantir a qualidade dos testes.
* Inclui a solicitação do documento de requisitos, a escolha da técnica de teste adequada (funcional no caso da ausência de código-fonte) e a participação de desenvolvedores para garantir um entendimento profundo do sistema testado.

Essas técnicas proporcionam uma abordagem abrangente, permitindo testar tanto as funcionalidades visíveis para o usuário quanto a estrutura interna do código, garantindo assim a qualidade do software.

4o

Claro, aqui estão mais alguns pontos importantes da aula sobre Estratégias de Testes:

1. **Revisões de Software:**
   * Antes da execução dos testes, realizar revisões eficazes do código pode ajudar a identificar e corrigir problemas antecipadamente.
2. **Progressão do Teste:**
   * Os testes devem começar com unidades individuais do software (teste de unidade) e evoluir para testes mais integrados (teste de integração e teste de sistema).
3. **Depuração:**
   * A depuração, apesar de ser uma atividade distinta, deve estar associada ao processo de teste para corrigir problemas encontrados durante os testes.
4. **Tipos de Testes:**
   * **Teste de Unidade:** Verifica funcionalidades individuais ou componentes do software.
   * **Teste de Integração:** Avalia a interação entre diferentes unidades ou componentes do sistema.
   * **Teste de Sistema:** Testa o sistema como um todo, verificando se todas as partes funcionam corretamente juntas.
   * **Teste de Aceitação:** Valida se o software atende aos requisitos do cliente e está pronto para produção.
   * **Teste de Regressão:** Assegura que alterações no código não introduziram novos problemas em funcionalidades existentes.
5. **Técnicas de Teste Funcional:**
   * **Particionamento de Equivalência:** Divide o input em grupos que devem ser tratados da mesma forma pelo sistema.
   * **Análise de Valores-Limites:** Testa os limites extremos dos valores de entrada.
   * **Baseado em Casos de Uso:** Cria casos de teste baseados em como o usuário interage com o sistema.
6. **Técnicas de Teste Estrutural:**
   * **Teste de Caminhos:** Verifica se todos os caminhos lógicos no código são cobertos.
   * **Teste de Condições:** Avalia a execução de cada condição dentro do código.
   * **Teste de Condições Múltiplas:** Testa combinações de condições para garantir que todas as possibilidades sejam verificadas.
7. **Testes para Diferentes Tipos de Aplicações:**
   * **Aplicações Móveis:** Considerações específicas para testar em dispositivos móveis, como diferentes tamanhos de tela e sistemas operacionais.
   * **Aplicações Web:** Testes relacionados à funcionalidade e compatibilidade em diferentes navegadores e dispositivos.
   * **Aplicações Orientadas a Objetos:** Teste de classes, objetos e suas interações.
8. **Planejamento de Testes:**
   * **Documento de Requisitos:** Essencial para entender as funções e criar casos de teste adequados.
   * **Escolha de Casos de Teste:** Baseado em uso real e condições de usabilidade.
   * **Participantes dos Testes:** Inclui desenvolvedores e outros stakeholders para garantir um entendimento profundo do sistema e dos requisitos.

Esses pontos fornecem uma visão abrangente das práticas e técnicas envolvidas na gestão e execução de testes de software.