Test-Driven Development (TDD)

O **Test-Driven Development (TDD)** é uma técnica de desenvolvimento de software onde os testes são escritos **antes** do código. Segue um ciclo iterativo conhecido como **Red-Green-Refactor:**

1. **Red (Vermelho):** Escrever um teste que falha, pois a funcionalidade ainda não foi implementada.
2. **Green (Verde):** Escrever o código mínimo necessário para fazer o teste passar.
3. **Refactor (Refatoração):** Melhorar o código, garantindo que os testes ainda passem.

Benefícios do TDD

- Como os testes são criados antes, o código tende a ser mais bem projetado.
- **Menos bugs** – Erros são detectados logo no início do desenvolvimento.
- **Facilidade na manutenção** – A base de testes ajuda a evitar regressões ao modificar o código.
- **Melhor entendimento dos requisitos** – O processo força o desenvolvedor a pensar nos casos de uso antes da implementação.

Desafios do TDD

- **Curva de aprendizagem** – Desenvolvedores sem experiência podem ter dificuldades para adotar a abordagem.
- **Mais tempo inicialmente** – Criar testes antes do código pode tornar o desenvolvimento inicial mais lento.
- **Cobertura de testes nem sempre é ideal** – TDD não garante que todos os cenários sejam testados; ainda é necessário um bom planeamento.

Possíveis Perguntas sobre TDD

1. Qual a diferença entre TDD e desenvolvimento tradicional?

No **desenvolvimento tradicional**, o código é escrito primeiro e os testes são feitos depois, mas sem uma estrutura rígida. Isto pode resultar em testes menos abrangentes e na descoberta tardia de falhas.

Já no **TDD (Test-Driven Development)**, os testes são escritos **antes** do código funcional. O desenvolvimento segue um ciclo iterativo (Red-Green-Refactor), garantindo que cada nova funcionalidade seja testável desde o início.

➡️ **Resumo da diferença:**

- **TDD:** Testes antes do código, desenvolvimento guiado pelos testes.
- **Desenvolvimento tradicional:** Código antes dos testes.

---

2. Como é que o TDD se relaciona com metodologias ágeis, como Scrum?

TDD está fortemente ligado às **metodologias ágeis**, pois incentiva ciclos curtos de desenvolvimento e feedback contínuo.

➡️ TDD combina com desenvolvimento ágil, pois melhora a qualidade do software e facilita entregas frequentes e incrementais.

---

3. Que tipos de testes são usados no TDD?

No TDD, o foco principal está em **testes automatizados**, sendo os mais comuns:

1. **Testes Unitários** – Testam pequenas partes do código (funções, classes, métodos). São a base do TDD.
2. **Testes de Integração** – Garantem que diferentes módulos ou componentes funcionam corretamente juntos.
3. **Testes de Aceitação (ATDD - Acceptance TDD)** – Definem requisitos do sistema em forma de testes, ajudando a validar se a funcionalidade atende às necessidades do usuário.

➡️ **Resumo:** TDD baseia-se principalmente em **testes unitários**, mas pode ser complementado com testes de integração e aceitação.