UNIVERSIDADE FEDERAL DE RORAIMA
PROFESSOR (A) : Herbert Oliveira Rocha .
DISCIPLINA DE INTRODUÇÃO AO SISTEMAS EMBARCADOS.
ALUNO : Ibukun Chife didier Adjitche

Resolução da lista 2 ISE

1, 2 e 3.) A resolução dessas questões estão na pasta do Projeto

4) Defina o que é teste de software e descreva o microciclo do TDD.

R4:

O teste do software é a investigação do software a fim de fornecer informações sobre sua qualidade em relação ao contexto em que ele deve operar. Isso inclui o processo de utilizar o produto para encontrar seus defeitos.

O Ciclo do TDD (Test-Driven Development) é simples: criamos um teste -> Fazemos a codificação para passar no teste -> Refatoramos nosso código, conforme figura a seguir:

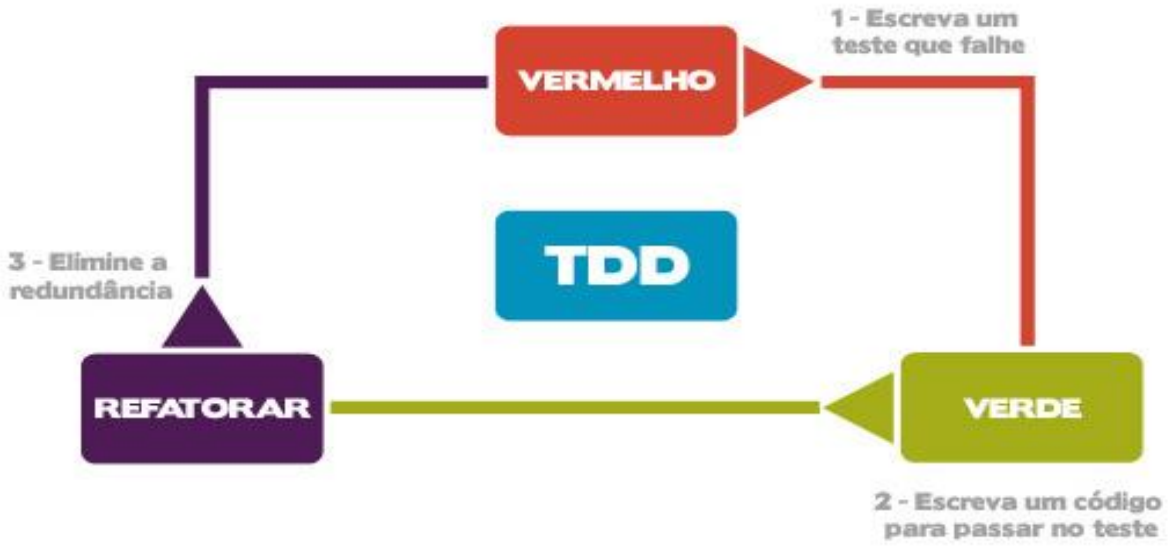

Figura 1: Ciclo do TDD

Notemos aqui que o teste visa auxiliar a codificação, reduzindo consideravelmente os problemas na fase de desenvolvimento. No TDD é indicado que o projeto de teste unitário ocorra antes da fase de codificação/implementação.Note que ao criar o teste antes de implementar a unidade, são reduzidos problemas como mal entendimento de requisitos ou interfaces.

6. Faça uma pesquisa sobre três frameworks para teste de unidade e apresente as vantagens e desvantagens de cada um, bem como, uma tabela descrevendo os itens (assertivas, análise de memory leak e outros) suportados por cada um deles.

R:

UNIVERSIDADE FEDERAL DE RORAIMA
PROFESSOR (A) : Herbert Oliveira Rocha .
DISCIPLINA DE INTRODUÇÃO AO SISTEMAS EMBARCADOS.
ALUNO : Ibukun Chife didier Adjitche

Jasmine :

Jasmine é um framework de desenvolvimento orientado por comportamento (BDD) para testes de Javascript. Ele não depende de nenhum outro framework de Javascript, nem mesmo requer um DOM (Document Object Model). E possui uma sintaxe óbvia e limpa para escrever testes facilmente.

Junit:

O JUnit é um framework open-source, que se assemelha ao raio de testes software java, criado por Erich Gamma e Kent Beck, com suporte à criação de testes automatizados na linguagem de programação Java.

Esse framework facilita a criação de código para a automação de testes com apresentação dos resultados. Com ele, pode ser verificado se cada método de uma classe funciona da forma esperada, exibindo possíveis erros ou falhas podendo ser utilizado tanto para a execução de baterias de testes como para extensão.

Com JUnit, o programador tem a possibilidade de usar esta ferramenta para criar um modelo padrão de testes, muitas vezes de forma automatizada.

O JUnit permite a realização de testes de unidades, conhecidos como "caixa branca", facilitando assim a correção de métodos e objetos.

Algumas vantagens de se utilizar JUnit:

1. Permite a criação rápida de código de teste enquanto possibilita um aumento na qualidade do sistema sendo desenvolvido e testado;
2. Não é necessário escrever o próprio framework;
3. Amplamente utilizado pelos desenvolvedores da comunidade código-aberto, possuindo um grande número de exemplos;
4. Uma vez escritos, os testes são executados rapidamente sem que, para isso, seja interrompido o processo de desenvolvimento;
5. JUnit checa os resultados dos testes e fornece uma resposta imediata;
6. Pode-se criar uma hierarquia de testes que permitirá testar apenas uma parte do sistema ou todo ele;
7. Escrever testes com JUnit permite que o programador perca menos tempo depurando seu código;
8. JUnit é LIVRE.

A experiência adquirida com o JUnit tem sido importante na consolidação do Test Driven Development (desenvolvimento direcionado a testes). Além disso, ele foi adaptado a outras linguagens, tais como C♯ (NUnit), Python, Fortran, e C++.

Mocha:

O Mocha é uma estrutura de teste de JavaScript rica em recursos executada no Node.js e no navegador, tornando o teste assíncrono *simples* e *divertido* . Os testes de Mocha são executados em série, permitindo relatórios flexíveis e precisos, enquanto mapeiam exceções não capturadas para os casos de teste corretos.