**1.1)** **C**

**1.2)** No TDD, o processo é simples: escreva um teste que descreva o que você quer que o código faça, execute-o (ele deve falhar), escreva o código para fazer o teste passar, execute o teste novamente (agora deve passar), refatore o código, se necessário, e, por fim, integre o código ao repositório principal. Isso ajuda a garantir a qualidade do código e a colaboração da equipe.

**2.1)**

Epic: Desenvolvimento de um Aplicativo de Compras Online

User Stories:

1. Como um cliente, eu quero poder navegar pelos produtos na loja.
2. Como um cliente, eu quero adicionar produtos ao meu carrinho de compras.
3. Como um cliente, eu quero poder visualizar meu carrinho de compras.
4. Como um cliente, eu quero poder realizar o pagamento dos produtos no meu carrinho.
5. Como um cliente, eu quero receber um e-mail de confirmação da minha compra.
6. Como um cliente, eu quero poder rastrear o status da minha entrega.

Essas histórias simplificam o desenvolvimento do aplicativo de receitas. Você pode pensar nelas como pequenos pedaços de funcionalidade, como ver receitas, marcar ingredientes e salvar suas favoritas. Cada uma delas é um passo para tornar a Epic "Fazer um App de Receitas" uma realidade.

**2.2)** Os critérios de aceitação nas User Stories são como uma "lista de verificação" no desenvolvimento de software. Os critérios de aceitação dizem exatamente o que uma funcionalidade deve fazer e como saber quando ela está pronta. É como se fosse um contrato entre a equipe de desenvolvimento e os outros envolvidos no projeto. Eles também ajudam a garantir que a funcionalidade seja realmente útil para quem a utiliza e que funcione bem. Além disso, facilitam a criação de testes, pois sabe-se o que precisa ser testado. Em resumo, os critérios de aceitação são como um GPS que mantém o projeto na direção certa e evita que as coisas fiquem confusas.

**3.1)** Teste unitário é como um exame minucioso para partes pequenas do código (ou unidades), garantindo que elas estejam funcionando direito. É como uma prevenção de problemas, ajuda a equipe a entender o código e evita confusões quando muitas pessoas trabalham nele. Resumindo, mantém o código saudável e sem surpresas ruins.

**3.2)** Em testes unitários, "mocking" é como criar substitutos simulados para partes do código que não estão diretamente envolvidas no teste, permitindo que o teste se concentre apenas na parte específica que está sendo testada, sem ser afetado por outras partes do sistema. Isso ajuda a tornar os testes mais rápidos e controlados.

**4.1)** A refatoração é a prática de melhorar a estrutura e o design do código sem alterar seu comportamento externo. No contexto do Extreme Programming (XP), a refatoração é uma técnica fundamental que se encaixa na filosofia de "melhoria contínua". Os programadores realizam refatorações regularmente para manter o código limpo, mais legível e mais fácil de manter, garantindo que o software permaneça adaptável às mudanças ao longo do tempo.

**4.2)** Programação em pares é quando dois programadores trabalham juntos no mesmo código, dividindo-se entre duas funções diferentes, sendo um para criar o código e o outro para revisá-lo, o que ajuda a encontrar erros mais rápido e produz um código de melhor qualidade. É como ter um parceiro de codificação, o que torna o processo mais eficiente e confiável.

**4.3)** No XP, o TDD significa escrever testes antes do código. Isso garante que o software seja testado desde o início, tornando-o mais confiável.

**5.1)** Para avaliar o sucesso do Scrum ou XP, métricas como velocidade da equipe, satisfação do cliente, entregas pontuais, qualidade do código e redução de defeitos após o lançamento são úteis. Essas métricas mostram o quão bem o processo está funcionando e se está agregando valor.

**5.2)** User Stories e Epics são usados para traduzir necessidades técnicas em linguagem acessível a stakeholders não técnicos. As User Stories descrevem funcionalidades de forma simples e direta, enquanto as Epics agrupam várias User Stories em objetivos maiores. Isso torna a comunicação mais clara, permitindo que os desenvolvedores compreendam as expectativas dos stakeholders e que estes entendam o que está sendo desenvolvido, promovendo uma colaboração eficaz e sem a utilização de jargões técnicos.