GRAN CONCURSOS

DESENVOLVIMENTO DE SOFTWARE

Processos Ágeis – XP

Livro Eletrônico

CÓDIGO:

240118157928

RODRIGO GOMES

Professor. Trabalha no Ministério Público do Distrito Federal e Territórios (MPU) como analista de sistemas na função de Product Manager – PO. Principal projeto atual em que atua é o Processo Eletrônico nas partes que tangem o Ministério Público. Aprovado e nomeado nos concursos públicos: Ministério Público da União – Analistas de Sistemas / Desenvolvimento de Sistemas; Ministério Público da União – Técnico em Tecnologia da Informação; Ministério do Planejamento – Analista de Sistemas; Banco Regional de Brasília – Analistas de Sistemas / Desenvolvimento de Sistemas; Dataprev – Analista de Requisitos; CONAB – Analistas de Sistemas / Desenvolvimento de Sistemas; SERPRO – Analista de Sistemas / Área de Negócios em TI. Possui outras aprovações, porém não chegou a ser nomeado. Formação: Bacharel em Sistemas de Informação pela Universidade Católica de Brasília; especialista em Gestão de Projetos pela Universidade Católica de Brasília; especialista em Gestão de Tecnologia da Informação no Serviço Público pela faculdade Alvorada. Certificações: IBM 833 – Object Oriented Analysis and Design – Parte 1 (Analysis); IBM 834 – Object Oriented and Design – Parte 2 (Design); Scrum Foundation Professional Certificate (SFPC) – CertiProf; Remote Work and Virtual Collaboration Certificate (RWVCPC) – CertiProf.

GRAN CONCURSOS

SUMÁRIO

APRESENTAÇÃO

Olá, futuro(a) servidor(a)!

Dando continuidade aos nossos estudos sobre desenvolvimento ágil de *software* trataremos agora sobre o *Extreme Programming – XP*.

A banca escolhida para o concurso do CNU foi a Cesgranrio. A Cesgranrio é conhecida por sua abordagem metódica, apresentando enunciados de questões bastante objetivos, geralmente com extensão de duas a três linhas. De modo geral, a prova consiste em perguntas de múltipla escolha, cada uma com cinco alternativas, sendo apenas uma delas correta. Em determinados exames, as questões podem ter pontuações distintas.

Para auxiliá-lo em seus estudos, iremos colocar questões utilizadas pela banca e, na falta de mais questões, iremos incluir questões de outras bancas com o objetivo de fortalecer o entendimento, a absorção do conteúdo, o aprendizado e aumentar sua curva de conhecimento.

PROCESSOS ÁGEIS – XP

Veja, abaixo, uma revisão de alguns conceitos essenciais do Manifesto Ágil (todos devem ser mantidos em mente):

1. **Priorizamos a satisfação do cliente**: Buscamos atender ao cliente por meio da entrega contínua e antecipada de *software* de valor.

2. **Aceitamos mudanças de requisitos**: Adotamos processos ágeis que se adaptam a mudanças, permitindo ao cliente obter vantagens competitivas, mesmo no final do desenvolvimento.

3. **Entregamos *software* funcional frequentemente**: Preferimos ciclos de entrega curtos, na escala de semanas até meses.

4. **Colaboração diária entre negócios e desenvolvedores**: Promovemos a colaboração constante entre as partes ao longo do projeto.

5. **Construímos projetos em torno de indivíduos motivados**: Oferecemos ambiente e suporte adequados, confiando que realizarão seu trabalho.

6. **Comunicação cara a cara é prioritária**: Valorizamos a conversa direta como o método mais eficiente de transmitir informações dentro de uma equipe de desenvolvimento.

7. ***Software* funcional como medida de progresso**: Consideramos o *software* funcional como a principal indicação de avanço.

8. **Ambiente sustentável**: Promovemos processos ágeis que permitem a manutenção de passos constantes de maneira indefinida para patrocinadores, desenvolvedores e usuários.

9. **Atenção contínua à excelência técnica e bom *design***: Acreditamos que isso aumenta a agilidade.

10. **Simplicidade**: Buscamos maximizar a eficiência, realizando apenas o trabalho necessário.

11. **Arquiteturas e *designs* emergem de times auto-organizáveis**: Acreditamos que equipes autogerenciáveis são fundamentais para obter as melhores soluções.

12. **Reflexão e ajuste regular**: Periodicamente, o time reflete sobre sua efetividade, ajustando e aprimorando seu comportamento de acordo.

Esta imagem pode auxiliar você a compreender e memorizar:

A tabela a seguir exibe as principais diferenças entre o manifesto ágil e as metodologias tradicionais:

Fator	Tradicional	Ágil
Resultados Finais	Resultado com escopo bem elaborado e definido	Resultado em escopos dinâmicos e adaptativos
Processos	Guiado por um plano bem definido	Processo guiado por pessoas e dinâmico conforme as prioridades
Papéis	Muito bem definidos	Equipe multidisciplinar e colaborativa
Quem comanda	Normalmente, gerente de projetos.	A própria equipe do projeto
Como o cliente atua	Mais fortemente nas fases iniciais e validações do produto	Fortemente em todo o projeto
Documentação	Muita documentação e fortemente detalhada	Somente o necessário, conforme o andamento do projeto. Código é o principal artefato.
Entregas	Conforme o plano inicial do projeto. Mudanças costumam ser custosas.	Podem variar de acordo com o tamanho das entregas. Entregas costumam ser significativas.
Previsão	Depende de monitoramento e planos de controle	Melhor previsibilidade do futuro devido à fácil adaptação e entregas constantes

Fator	Tradicional	Ágil
Mudanças	Gestão formal de mudanças no projeto. Evitam-se modificações no escopo e obedecem a níveis de hierarquia formais.	Mudanças são bem-vindas. Cliente pode alterar as priorizações constantemente. Não é indicado modificar o escopo de uma *sprint* em andamento.
Tempo para entregas	Pode demorar muito até que uma entrega seja realizada para o cliente	Fixo e é conforme a definição de duração das iterações que, comumente, varia entre 2 e 4 semanas. As prioridades podem ser modificadas com o tempo.
Time do projeto	Papéis muito bem definidos	Equipe multidisciplinar, multifuncional e auto-organizada. Ela decide como fazer e atua de forma colaborativa.
Riscos	Pode exigir um grande esforço e equipe para atuar em riscos	A própria equipe atua com os riscos e pode obter apoio externo. Riscos são priorizados e revistos a cada *sprint*.
Planejamento	Costuma-se detalhar bastante o projeto já no início	Planejamento de alto nível no início. Trabalho do *backlog* da *sprint* deve ser o foco do detalhamento.
Gestão	Centralizado	Descentralizado
Medição de desempenho	Conformidade em relação ao plano	Valor no negócio
Ênfase em	Processos (mais burocracia)	Pessoas

O QUE É XP?

É uma metodologia de desenvolvimento ágil (assim como o *Scrum*) que tem como objetivo criar sistemas com alta qualidade, com base em uma interação próxima com os clientes, testagem constante e ciclos de desenvolvimento curtos. Por ser uma metodologia ágil, segue os princípios básicos já estudos do manifesto ágil (se necessário, retorne às penúltima e última aulas). É método sistêmico no qual o processo se inicia por um plano geral, sem detalhamento, neste primeiro momento. Em seguida, divide-se o problema em partes menores para dar início ao seu desenvolvimento.

A proposta é "modernizar" a sequência convencional do desenvolvimento em cascata, que segue as fases de análise, *design*, implementação e teste. Reconhecendo esse modelo como burocrático e pouco flexível em um mundo em constante transformação, a ideia é adaptar e torná-lo mais ágil.

A *Extreme Programming* (*XP*) destaca-se como uma abordagem amplamente adotada por equipes de pequeno e médio porte que lidam com o desenvolvimento de *software* baseado em

requisitos fluidos e sujeitos a rápidas mudanças. Essa metodologia prioriza a entrega rápida do projeto, buscando assegurar a satisfação do cliente e promover a aderência às estimativas.

O XP pode ser adaptado à cultura da equipe e da organização. Logo, nem todas as práticas são utilizadas 100%. Cada equipe escolhe aquelas que consideram mais úteis e válidas para sua realidade.

Decore isto, futuro servidor público:

O XP é uma metodologia ágil de desenvolvimento de *software* para equipes pequenas, coesas e multidisciplinares, cujos projetos possuem requisitos vagos e em constante mudança.

A seguir, veremos as principais características do XP.

MAIS *SOFTWARES* FUNCIONANDO, MENOS DOCUMENTAÇÃO

Assim como no *Scrum*, o XP preconiza que um *software* funcionando vale mais do que uma documentação extensa. Logo, o código fonte possui um alto valor nesta metodologia.

Portanto, o código finalizado também é uma fonte de *feedback*, assim como um produto implantado e em pleno funcionamento. *Software* funcionando agrega bastante para o cliente.

INTEGRAÇÃO CONTÍNUA

Consiste na integração do código alterado ou mesmo desenvolvido do projeto que é o principal, porém com muito mais agilidade e de forma mais constante.

O objetivo principal de utilizar a integração contínua é verificar se as alterações ou novas funcionalidades não criaram novos defeitos no projeto já existente.

Esta prática pode ser realizada por meio de processos manuais ou mesmo automatizados.

ADAPTAÇÃO ÀS MUDANÇAS

Se muitas mudanças são realizadas, todas de uma vez, não se obtém um bom resultado. No XP, as mudanças devem ser incrementais e aos poucos. Os desenvolvedores possuem a liberdade para melhorar constantemente o código do produto.

ENTREGAS PEQUENAS

Cada *release* deve ser tão pequeno quanto possível, contendo os requisitos mais importantes para o negócio. Isso possibilita ter *releases* frequentes, aumentando o *feedback* dos usuários e o time de desenvolvimento. Lembre-se: mínimo útil de forma que agregue valor ao negócio. Não adianta entregar algo que não agregue nada!

Como *releases* são constantes, a integração é CONTÍNUA (grave isto!). Quando uma funcionalidade é produzida, ela deve ser integrada brevemente. Essa prática diminui a possibilidades de erros.

HISTÓRIAS DE USUÁRIO

Consistem em artefatos com as demandas/os requisitos dos usuários. Porém, são escritos de forma mais resumida e em uma linguagem mais próxima do usuário. Representam uma pequena parte da funcionalidade a ser desenvolvida.

Detalhes mais profundos das histórias poderão ser captados com o desenvolvimento da funcionalidade, principalmente nos testes de aceitação do incremento do produto.

As histórias de usuário podem ser escritas em cartões que serão consultados durante todo o desenvolvimento. Lembrando que o cliente deve participar ativamente dessa escrita.

TAMANHO DAS HISTÓRIAS

Elas devem possuir tamanho ideal para serem desenvolvidas dentro de uma iteração, considerando a capacidade de produção da equipe de desenvolvimento. As histórias de usuário atravessam a arquitetura do produto, não focando em um subsistema específico.

Logo, uma história que não couber dentro da iteração deve ser decomposta em histórias menores. Tenha em mente que as iterações devem produzir produtos utilizáveis para o usuário.

Assim como no *Scrum*, as histórias de usuário são elaboradas obedecendo as prioridades do cliente (FOCO NO CLIENTE, sempre).

Após definidas as histórias, a equipe de desenvolvimento divide os cenários em tarefas menores construção.

As histórias de usuário possuem algumas características:

Independente – Sempre que possível, as histórias de usuário devem ser independentes umas das outras, a fim de que o Dono do Produto possa ordená-las, planejá-las e implementá-las conforme o valor que cada uma possui para o negócio.

Negociável – Conforme foi dito anteriormente, a história de usuário não contém todos os detalhes de um requisito. Sendo assim, ela deve ser descrita em um nível que permita aos Time *Scrum* e *Stakeholders* realizarem conversas a fim negociar o detalhamento desses requisitos. Uma história de usuário não é um contrato a ser seguido, e, sim, um lembrete a todos que deve ser realizada uma conversa a fim de detalhar os requisitos.

Valorosa – No artigo referente às características de um *Backlog* do Produto, dissemos que, a fim de evitar desperdício de tempo e de recursos, o Dono do Produto deve priorizar o desenvolvimento dos 20% dos itens do *Backlog* do Produto que irão agregar o maior valor para o negócio. Os 80% dos itens restantes devem ficar para depois ou, caso não sejam efetivamente necessários, devem ser descartados. Dessa forma, as histórias do usuário devem gerar valor para o negócio.

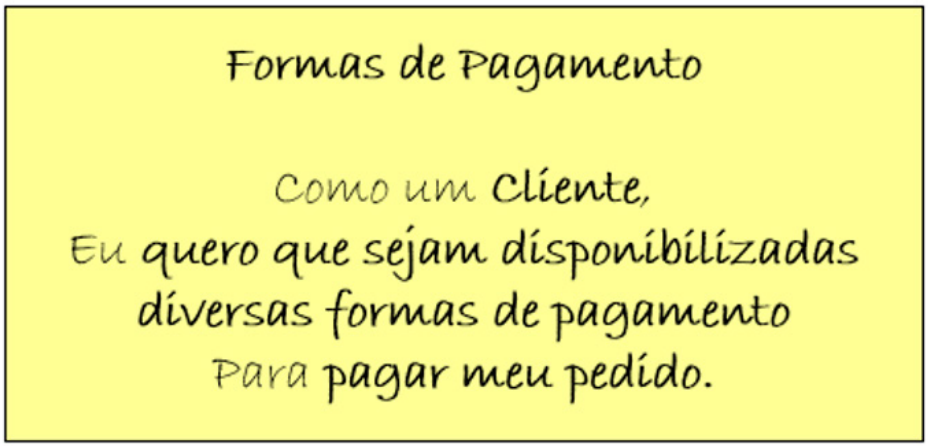
Estimável – É muito importante que uma história de usuário possa ser estimada pelo Time de Desenvolvimento, mesmo que o nível de precisão não seja alto, uma vez que a

estimativa possibilita ao Dono do Produto priorizar o *Backlog* do Produto, estimar custos, planejar *releases* etc. Quando o Time de Desenvolvimento não consegue estimar uma História de Usuário, isso pode significar que o seu tamanho é muito grande, que as informações são insuficientes, que o seu conteúdo está muito ambíguo, ou até mesmo que o Time não possui conhecimento sobre a tecnologia a ser utilizada.

"*Small*" (Pequena) – As Histórias de Usuário devem ser pequenas, ou seja, necessitam possuir um tamanho máximo que as permitam serem desenvolvidas dentro de uma única *Sprint*.

Testável – É muito importante que as Histórias de Usuário possam ser testadas, a fim de que seja possível assegurar que o produto obtido ao final do desenvolvimento esteja de acordo com o que foi definido e sem a existência de erros.

Exemplo de um cartão com história de usuário:



DESENVOLVIMENTO INCREMENTAL – *DESIGN* INCREMENTAL

A equipe de desenvolvimento deve investir no *design* da aplicação todos os dias, buscando oportunidades para remover a duplicação e fazer pequenas melhorias, alcançando o melhor código e produto possíveis. Esses incrementos são curtos.

Os requisitos são registrados em cartões de histórias e as histórias a serem incluídas em um *release* são determinadas pelo tempo disponível e sua prioridade relativa. Os desenvolvedores dividem essas histórias em tarefas.

DESENVOLVIMENTO EM PARES

Os desenvolvedores trabalham em pares para codificarem as tarefas estabelecidas para o produto. Consiste em um método de programação no qual duas pessoas trabalham juntas em um único programa.

QUESTÃO INÉDITA

001. (INÉDITA/224) Em projetos de desenvolvimento de software, a *extreme programming* (XP) é um método ágil que usa a prática de:

a) projetos com planejamento completo sem incrementos.

b) grandes *releases*.

c) grande quantidade de horas extras.

d) trabalho em pares de desenvolvedores.

e) integrações após a entrega do *software* completo.

Conforme vimos, uma das práticas do XP é a programação em pares.

Sobre os demais itens:

a) Errada. Uma das práticas do XP é a sua forma de desenvolvimento incremental constante. Tão logo o trabalho em uma tarefa é concluído, esta é integrada ao sistema.

b) Errada. Seria o contrário: pequenas *releases*. Em regra, o conjunto mínimo de funcionalidades que agregam valor ao usuário devem ser adicionadas à próxima *release*.

c) Errada. Seria o contrário da afirmação. Horas extras não são bem-vindas no XP. Devem ser evitadas ao máximo. O modelo preconiza o desenvolvimento sustentável com semanas com duração de 40 horas de trabalho.

d) Certa. Gabarito correto.

e) Errada. O código das funcionalidades implementadas pode ser integrado várias vezes ao dia. Compreende na integração contínua.

Letra d.

TESTES CONSTANTES E *TEST-FIRST*

Os testes em XP são feitos antes da programação. Existem dois tipos de teste: teste de unidade e teste funcional. Consiste em verificar o comportamento das menores unidades em sua aplicação.

Os testes funcionais são usados para verificação, junto ao cliente, do sistema como um todo e suas funcionalidades propriamente ditas.

Sempre que algo novo é codificado, novos testes e repriorizações podem ser criadas (priorização pelo usuário).

Detalhe importante: primeiro, são escritos os testes e, depois, a implementação propriamente dita. É o que chamamos de *Test-First*.

Test-First: pode-se utilizar alguma ferramenta/*framework* para escrever os testes para uma nova parte da funcionalidade antes que esta seja implementada.

Testes são EXTREMAMENTE importantes no XP!

O PULO DO GATO

Testes são EXTREMAMENTE importantes no XP!

SEMANA DE 40 HORAS DE TRABALHO (RITMO SUSTENTÁVEL DE TRABALHO)

O trabalho deve ser feito com qualidade em um ritmo sustentável, buscando o valor máximo de 40 horas semanais. Horas extras não são bem-vindas e devem ser evitadas.

DIRETO DO CONCURSO

002. (INSTITUTO AOCP/ADAF–AM/2018) Na metodologia ágil *Extreme Programming* (XP), a propriedade do código é coletiva, dessa forma, todos compartilham o mesmo orgulho e as mesmas críticas. Considerando o exposto, assinale a alternativa que apresenta uma das regras da codificação em XP.

a) No *Overtime*.
b) Eliminar gargalos de *hardware* no início.
c) O usuário não deve participar do planejamento das interfaces.
d) No *Outsourcing*.
e) No *Sprint*.

O gabarito é a letra "a".

Conforme vimos, o XP possui como prática o ritmo sustentável, dessa forma, grandes quantidades de horas extras não são bem-vindas. *Overtime* está interligado com horas extras. Logo, não se trata de uma prática do modelo.

Obs.: Os demais itens não fazem correlação com o modelo XP.

Letra a.

REFATORAÇÃO

Refatoração significa melhorar o código sem alterar sua funcionalidade e é realizado constantemente pela equipe de desenvolvimento. Logo, após a refatoração, o código continua funcionando como anteriormente (tratando-se da funcionalidade). As mudanças devem ser incluídas após este processo.

Tornar o código limpo também é um processo de refatoração (simples, direto e de fácil entendimento).

CÓDIGO COLETIVO

A propriedade do código é coletiva e todos são responsáveis. Com isso, os desenvolvedores ganham tempo, pois não precisam esperar a autorização de alguém para editar o código. Essa prática torna mais constante a identificação de oportunidades de melhoria e erros, tornando a refatoração mais presente. Qualquer um pode mudar qualquer coisa!

DIRETO DO CONCURSO

003. (CESPE/MINISTÉRIO DA ECONOMIA/2020) O *refactoring* de código não faz parte do modelo XP, visto que a expectativa é a entrega ágil, e não deve ser considerada em tempo de projeto a recriação de código para aprimoramento.

Refatoração faz parte do XP! Ela ocorre de forma constante durante o desenvolvimento do produto.

Errado.

EQUIPE DE DESENVOLVIMENTO

Formada por equipes pequenas, coesas e multidisciplinares, que trabalham de maneira colaborativa. É importante destacar que o código é de propriedade coletiva.

CIENTE SEMPRE PRESENTE NO DESENVOLVIMENTO

Envolve manter um usuário real do sistema como parte integrante da equipe, sempre disponível para responder a possíveis perguntas. Este membro deve estar presente em tempo integral, faz parte da equipe de desenvolvimento e é responsável pelos requisitos.

METÁFORA

As equipes XP utilizam metáforas para facilitar o desenvolvimento e a comunicação. Essas metáforas devem ser culturalmente aceitas pela equipe, tornando-se um contexto comum.

PADRÕES DE CODIFICAÇÃO

O modelo XP recomenda a aplicação de padrões de codificação desde as primeiras *releases* do produto, visando aumentar a produtividade e a qualidade do *software*.

REUNIÕES DIÁRIAS (DAILY) EM PÉ

Originada do *SCRUM*, esta prática envolve a realização de uma rápida reunião em pé, na qual todos discutem as atividades concluídas no dia anterior, o planejamento para o dia atual e identificam possíveis impedimentos futuros.

VALORES DO XP (MUITO COBRADO EM PROVAS!)

As regras, práticas e valores da *Extreme Programming* (XP) criam um ambiente de desenvolvimento de *software* agradável para seus seguidores, guiados por cinco valores essenciais:

- **Comunicação**: O XP promove conversas presenciais e diretas entre desenvolvedores e clientes, visando uma compreensão precisa das necessidades do cliente e garantindo que o cliente compreenda as possibilidades, estrutura e objetivos do sistema.
- **Simplicidade**: Priorizando o essencial para o projeto, o XP busca evitar desperdícios, reduzir custos e tempo, além de manter o *design* e as funcionalidades o mais fácil possível para os usuários.
- ***Feedback***: A importância do retorno constante, precoce e em ciclos curtos é destacada no XP. Isso assegura ajustes rápidos e precisos durante o processo de desenvolvimento.
- **Coragem**: A abertura a mudanças, a capacidade de enfrentar falhas, aceitar *feedback*, propor melhorias e a habilidade de dizer "não", quando necessário, são aspectos fundamentais. No contexto do XP, coragem é definida como "ação eficaz em face do medo", conforme expresso por Kent Beck em seu livro "*Extreme Programming Explained*".
- **Respeito**: O trabalho em equipe é central no XP, e, para promover isso, é essencial que os membros se respeitem, aceitem sugestões, colaborem mutuamente e cultivem um ambiente de relacionamento positivo.

Valores do XP

O PULO DO GATO

Por favor, futuro servidor, decore:

Cor Sim Com Fe Re

Cor agem

Sim plicidade

Com unicação

Fe edback

Re speito

PRINCÍPIOS DO XP

Já temos que ter em mente que os princípios básicos do XP são diferentes dos valores. Não confunda estes conceitos!

O PULO DO GATO

VALORES SÃO DIFERENTES DOS PRINCÍPIOS.

DIRETO DO CONCURSO

004. (CESPE/TCU/AUDITOR FEDERAL DE CONTROLE EXTERNO/TECNOLOGIA DA INFORMAÇÃO/ PARTE II/2010) A atividade de planejamento XP inclui a criação das denominadas histórias de usuário, nas quais devem ser descritas as características e as funcionalidades requeridas para o *software* em desenvolvimento.

Os requisitos são delineados por meio de *user stories*, uma prática comum assim como muitas equipes adotam no *Scrum*. Esse processo de especificação acontece durante a fase de planejamento do *software*.

Gabarito está CORRETO.

Certo.

Princípios do XP

Os princípios fundamentais são: *feedback* rápido, assumir simplicidade, mudança incremental, abraçando mudanças e trabalho de qualidade.

Já temos que ter em mente que os princípios básicos do XP são diferentes dos valores. Não confunda estes conceitos!

O PULO DO GATO

VALORES SÃO DIFERENTES DOS PRINCÍPIOS.

***Feedback* rápido**

É crucial manter uma comunicação colaborativa e constante entre a equipe de desenvolvimento e os clientes ao longo do projeto, proporcionando benefícios como aprendizado rápido, identificação precoce de riscos, visibilidade de mudanças e resolução eficiente de potenciais problemas.

Presunção de Simplicidade

A abordagem XP preconiza que os problemas devem ser tratados de maneira simples e eficiente. Destaca a importância de realizar um trabalho bem-feito, incluindo testes, refatoração e comunicação, para solucionar os problemas imediatos, confiando na capacidade de adicionar complexidade no futuro, quando necessário.

Consideração da Qualidade

Independentemente da evolução do projeto, mudanças ou cenários, a qualidade não deve ser comprometida. No contexto do XP, a qualidade refere-se à entrega de um sistema que atenda plenamente aos requisitos do cliente, execute 100% dos casos de teste e agregue o máximo valor possível ao negócio do cliente.

Mudanças Incrementais

Semelhante ao *Scrum*, o XP acolhe mudanças, enfatizando que elas devem ser incrementais. Os desenvolvedores têm a liberdade de realizar melhorias contínuas no código, e as mudanças nos requisitos são incorporadas de maneira incremental ao produto.

Abraçar Mudanças e Alterações de Escopo

A cultura XP incentiva a aceitação de mudanças, independentemente do tamanho ou estágio do projeto. Essa abordagem é particularmente benéfica em projetos com requisitos voláteis, nos quais os clientes podem não ter clareza sobre suas necessidades.

Trabalho de Qualidade

A qualidade é fundamental no XP, definida como ter um sistema que atenda aos requisitos do cliente, execute 100% dos casos de teste e agregue o maior valor possível ao negócio. Este princípio está alinhado com a prática de escrever testes antes de iniciar a codificação.

Segue uma tabela resumo das principais práticas do XP (essa tabela é muito importante!).

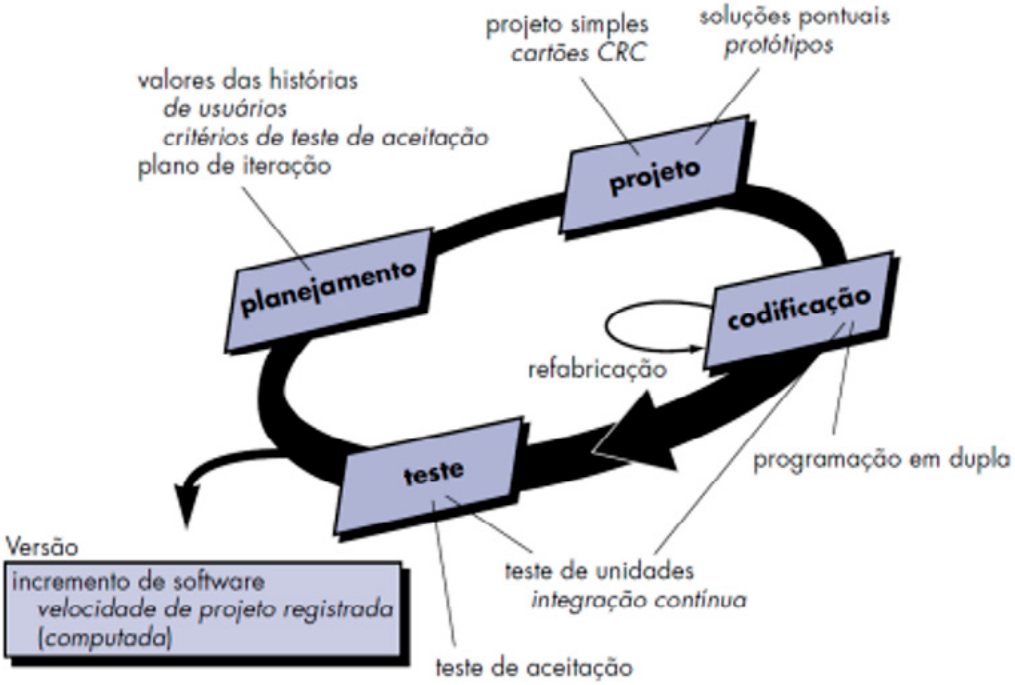
PRÁTICAS	DESCRIÇÃO
Planejamento incremental	**As demandas são documentadas em cartões de enredos, e as estórias a serem integradas em uma edição são estabelecidas com base no intervalo disponível e na sua prioridade relativa. Os desenvolvedores fragmentam essas estórias em tarefas.**
Pequenos *releases*	**O grupo mínimo essencial de funcionalidades que proporciona valor ao negócio é desenvolvido em primeiro lugar. Entregas do sistema ocorrem frequentemente, acrescentando funcionalidade de forma progressiva ao primeiro lançamento.**
Projeto Simples	**É conduzido um projeto suficientemente descomplicado para satisfazer aos requisitos vigentes e apenas isso. É crucial ter em mente que um código simples não se traduz automaticamente em código fácil (KIS – *Keep It Simple*).**
Desenvolvimento *Test-First*	**Um arcabouço automatizado de teste unitário é empregado para criar os testes de uma nova parte da funcionalidade antes de efetivar a implementação. Desse modo, inicia-se elaborando o teste para, em seguida, concluir a implementação.**
Otimização do Código – *refactoring*	**Aguarda-se que todos os desenvolvedores refinem continuamente o código sempre que identificarem oportunidades de aprimoramento. Isso resulta em um código fácil de compreender e simples de manter.**

PRÁTICAS	DESCRIÇÃO
Codificação em pares	**Os desenvolvedores colaboram em duplas, um revisando o trabalho do outro e oferecendo suporte para garantir constantemente uma execução exemplar. Eles compartilham o mesmo *mouse*, teclado e monitor.**
Propriedade Compartilhada	**Os pares de desenvolvedores atuam em todas as áreas do sistema, evitando a formação de ilhas de conhecimento, com todos os desenvolvedores compartilhando a propriedade de todo o código. Qualquer um pode alterar qualquer coisa.**
Integração Contínua	**Logo após a conclusão do trabalho em uma tarefa, ela é integrada ao sistema como um todo. Após cada integração, todos os testes unitários do sistema devem ser conduzidos.**
Ritmo sustentável	**Grandes volumes de horas extras não são considerados aceitáveis, pois, a longo prazo, resultam na diminuição da qualidade do código e da produtividade. Trabalhar por períodos prolongados torna-se contraproducente – recomenda-se 40 horas semanais.**
Metáforas	**A equipe comunica-se sobre o desenvolvimento de *software* por meio de metáforas, quando consegue identificar uma que verdadeiramente faça sentido no contexto e possa facilitar a comunicação.**
Cliente *In Loco*	**Um representante do usuário final do sistema deve estar disponível em tempo integral para apoiar a equipe. No XP, o cliente faz parte da equipe de desenvolvimento e é encarregado de trazer os requisitos do sistema.**
Reuniões em Pé	**Encontros são realizados em pé para manter o foco nos assuntos, gerando reuniões mais céleres, abordando somente as tarefas concluídas e as tarefas a serem realizadas pela equipe no futuro.**

PROCESSOS PARA O DESENVOLVIMENTO XP

Comparando-se com os demais métodos ágeis, o XP seria o que mais se aproxima de uma programação orientada a objetivos. Entretanto, fazer uso da orientação a objetos não é obrigatório. Além desta preferência, o XP preconiza o desenvolvimento dividido em 4 etapas consideradas fundamentais para a construção do produto:

- Planejamento;
- Projeto;
- Codificação;
- Testes.



Fases do XP segundo Presmann

Um projeto XP atravessa essas fases durante o seu ciclo de vida e são compostas por várias tarefas e atividades.

Vamos estudar um pouco mais cada uma destas etapas:

PLANEJAMENTO

Também conhecida como *planning game*, essa etapa inicial envolve a escuta atenta do cliente para identificar histórias, suas funcionalidades e requisitos correspondentes. O cliente detalha as histórias em termos de requisitos, permitindo que a equipe de desenvolvimento compreenda o contexto de negócios.

O processo de planejamento começa com a atividade de escuta, um levantamento de requisitos que ajuda a equipe a compreender o ambiente de negócios, proporcionando uma compreensão mais realista dos resultados desejados, principais fatores e funcionalidades.

As histórias são elaboradas pelo cliente, cada uma atribuindo um valor ao negócio, permitindo que o cliente priorize o que deve ser desenvolvido. Caso uma história seja muito extensa, sugerimos que o cliente a subdivida em histórias menores, especialmente se o tempo estimado para desenvolvimento for superior a 3 semanas.

A velocidade de desenvolvimento pode ser utilizada para estimar entregas futuras e prazos. É importante destacar que o cliente tem a flexibilidade de adicionar, alterar, excluir ou modificar histórias. Essas alterações podem exigir um novo planejamento por parte da

equipe de desenvolvimento, incluindo a redefinição de critérios de aceitação e a criação de um novo plano.

Se houver um exagero, as características das versões são ajustadas ou as datas de entrega finais são modificadas. À medida que o desenvolvimento avança, o cliente tem a possibilidade de incluir novas histórias, alterar o valor de uma história existente, dividir algumas ou removê-las. Posteriormente, a equipe XP revisita todas as versões restantes e ajusta seus planos de acordo.

PROJETO (*DESIGN*)

O projeto – ou *design* – no XP segue o valor da simplicidade. Chamamos de *Keep it Simple*: preserve a simplicidade. É preferível que se tenha um projeto simples do que uma representação muito complexa.

O PULO DO GATO

Projeto de *software* no XP: SIMPLICIDADE.

Segundo KENT, existem alguns passos básicos para definir a simplicidade no desenvolvimento:

- o sistema (código e testes) deve expressar tudo e somente aquilo que você deseja comunicar;
- o sistema não deve conter nenhum código duplicado;
- o sistema deve ter o menor número de classes possível;
- o sistema deve ter o menor número de métodos possível.

Nesta fase, temos um roteiro de implementação para a história à medida que é elaborada. A implementação seguirá rigorosamente a definição, sem excessos ou omissões. O que foi solicitado será executado conforme especificado.

O modelo incentiva a utilização de cartões CRC (classe-responsabilidade-colaborador). Esses cartões são úteis para identificar e organizar as classes orientadas a objetos no desenvolvimento do *software* em questão. Eles facilitam o incremento do código sendo codificado.

O PULO DO GATO

Os cartões CRC representam o único artefato de projeto gerado como parte do processo XP.

Em casos de compreensão complexa, protótipos podem ser desenvolvidos, proporcionando auxílio na redução de riscos e facilitando a implementação. Contudo, isso é uma abordagem pontual.

Exemplo de Cartão CRC

O projeto evolui à medida que é desenvolvido, visando controlar possíveis mudanças e propondo alterações capazes de aprimorar o projeto. A elaboração do projeto ocorre tanto antes como depois do início da codificação, sendo que a própria atividade de desenvolvimento orienta a equipe XP na melhoria contínua do projeto em construção.

Reiterando um conceito anteriormente abordado:

Refatoração envolve a modificação do sistema de *software* sem alterar seu comportamento funcional, focando na melhoria da estrutura interna. No entanto, é importante observar que a quantidade de esforço necessário para realizar a refatoração pode aumentar significativamente à medida que a aplicação cresce em tamanho.

CODIFICAÇÃO

Após a fase de projeto/*design* do produto, a equipe de desenvolvimento escreverá os testes unitários. A prática de escrever testes antes da codificação pode parecer complexa inicialmente, mas os testes unitários desempenham um papel crucial na manutenção da qualidade do projeto. Essa abordagem também pode aprimorar a eficiência do projeto, pois direciona o desenvolvedor para a implementação precisa do que é necessário.

Na etapa de implementação, é importante destacar que a programação no XP é realizada em pares. Essa técnica envolve dois programadores compartilhando uma única máquina para desenvolver o código, onde um programa e o outro observa em busca de identificar erros para correção imediata.

É recomendável integrar continuamente novas funcionalidades com a versão atual do sistema, evitando possíveis conflitos e erros no código. A integração contínua proporciona um entendimento preciso do *status* real da codificação e possibilita um *feedback* mais rápido.

TESTES

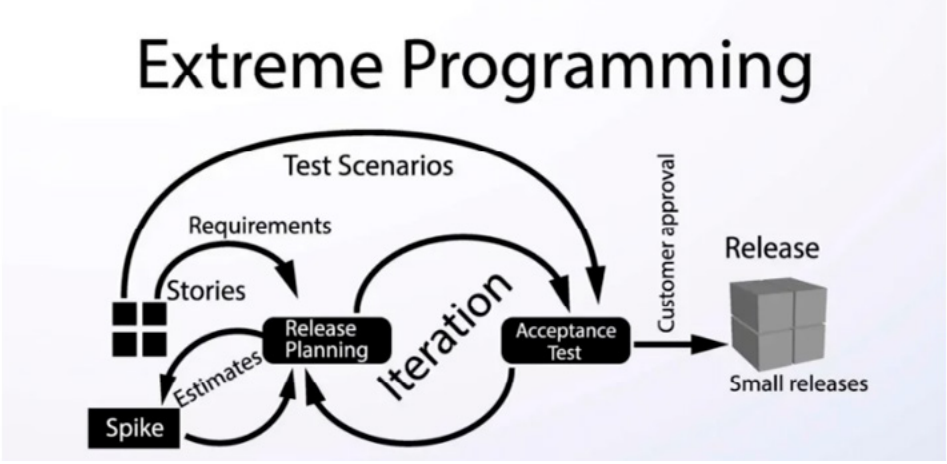
Os testes de unidade, escritos durante a fase de codificação, e os testes de aceitação, elaborados pelo cliente, são aplicados. A automação dos testes é recomendada para facilitar a execução.

Os testes de regressão são executados constantemente no XP, especialmente após a correção de defeitos ou a adição de novas funcionalidades, visando garantir que nenhum defeito seja adicionado ao sistema ao longo da evolução. Dado que várias integrações são realizadas frequentemente, os testes de regressão desempenham um papel crucial.

Os testes de unidade podem ser agrupados em conjuntos, enquanto os testes de integração e validação do sistema podem ocorrer diariamente, dependendo do ritmo da equipe.

É fundamental compreender que os testes têm o propósito de prevenir falhas futuras, evitando desperdício de recursos e tempo. Descobrir falhas no início do processo é mais eficiente do que lidar com custos mais elevados durante sua descoberta ou ajuste futuro.

Na prática diária, nem todas as organizações que adotam o XP utilizam todas as práticas. Cada organização e equipe têm a liberdade de escolher aquelas que consideram mais úteis, eficientes e viáveis para sua realidade.



Ciclo de vida no XP

A equipe tem a opção de criar um protótipo ou realizar um desenvolvimento de teste para compreender tanto o problema quanto a solução. Na linguagem XP, essa abordagem é denominada "*spike*" – um incremento no qual nenhuma programação real é executada. Além disso, podem ocorrer *spikes* relacionados ao design da arquitetura do sistema ou para desenvolver a documentação do sistema.

RESUMO

Conceito XP: método ágil que enfatiza o desenvolvimento rápido do projeto e visa garantir a satisfação do cliente, requisitos vagos, adaptativo a mudanças, além de favorecer o cumprimento das estimativas. É indicado principalmente para equipes pequenas.

Diferente das metodologias tradicionais que trabalham com a previsibilidade e controle, os modelos ágeis são orientados à construção, testes e focados em pessoas.

Características principais

- Os requisitos são especificados em *user stories* – histórias de usuário.
- Equipes pequenas, coesas e multidisciplinares.
- Baseado em requisitos vagos.
- Mais *software*, menos documentação.
- Refatoração: processo que permite a melhoria contínua da programação, o mínimo de introdução de erros e mantendo compatibilidade com o código já existente.
- Código coletivo.
- Programação em pares.
- Integração contínua.
- Uso de metáforas: utilizadas pelas equipes XP para facilitar o desenvolvimento e a comunicação.
- Ciente sempre presente no desenvolvimento (faz parte da equipe): responsável pelos requisitos.
- Integração contínua.
- Pequenas entregas (*releases*).
- *Keep it simple* (simplicidade).
- Desenvolvimento em pares: os desenvolvedores trabalham em pares para codificarem as tarefas estabelecidas para o produto.
- Adaptativo para mudanças: os requisitos, arquitetura e *design* surgem durante o desenvolvimento do projeto, que ocorre de maneira incremental (parecido com o *Scrum*).
- Ritmo sustentável (semana de 40 horas de trabalho, via de regra).
- Padrões de codificação.
- *Test-First*: primeiro se escreve o teste, depois faz-se a implementação.
- Reuniões diárias em pé (igual às reuniões do *Scrum*).

VALORES (Cor Sim Com Fe Re):

- Comunicação: para que os desenvolvedores possam compreender precisamente o que o cliente necessita e para que o próprio cliente tenha ciência das possibilidades, estrutura e objetivos do sistema, o *Extreme Programming* propõe conversas presenciais e diretas entre as partes.
- Simplicidade: para evitar desperdícios, reduzir custos e tempo e manter o *design* e as funcionalidades as mais fáceis possíveis de se utilizar, o XP procura priorizar o que é absolutamente necessário para o projeto.
- *Feedback*: os retornos e comentários constantes, precoces e em ciclos curtos sobre as práticas durante o processo são fundamentais para garantir ajustes rápidos e mais precisos.

Coragem: para estar aberto a mudanças, encarar as falhas, aceitar os *feedbacks*, propor melhorias e saber dizer não quando necessário. Significa confiar no processo. Em seu livro *Extreme Programming Explained*, o autor Kent Beck define coragem como "ação eficaz em face do medo".

Respeito: o trabalho em equipe é uma das premissas do XP e, para isso, é preciso que os membros se respeitem, aceitem sugestões, colaborem entre si e prezem por um bom relacionamento.

Princípios:

- **Trabalho de qualidade**: o produto jamais deve ter sua qualidade comprometida.
- **Abraçar mudanças**: serão realizadas de acordo com o melhor entendimento do projeto e sempre serão bem-vindas.
- **Mudanças incrementais**: o produto deve ser evoluído constantemente (em cada iteração). Desenvolvedores possuem liberdade para estas evoluções do código.
- **Presumir simplicidade**: os problemas devem ser resolvidos de forma simples e eficiente, sem perder a qualidade.
- ***Feedback* rápido:** comunicação colaborativa entre os desenvolvedores e clientes. O cliente faz parte do time e deve estar sempre presente.

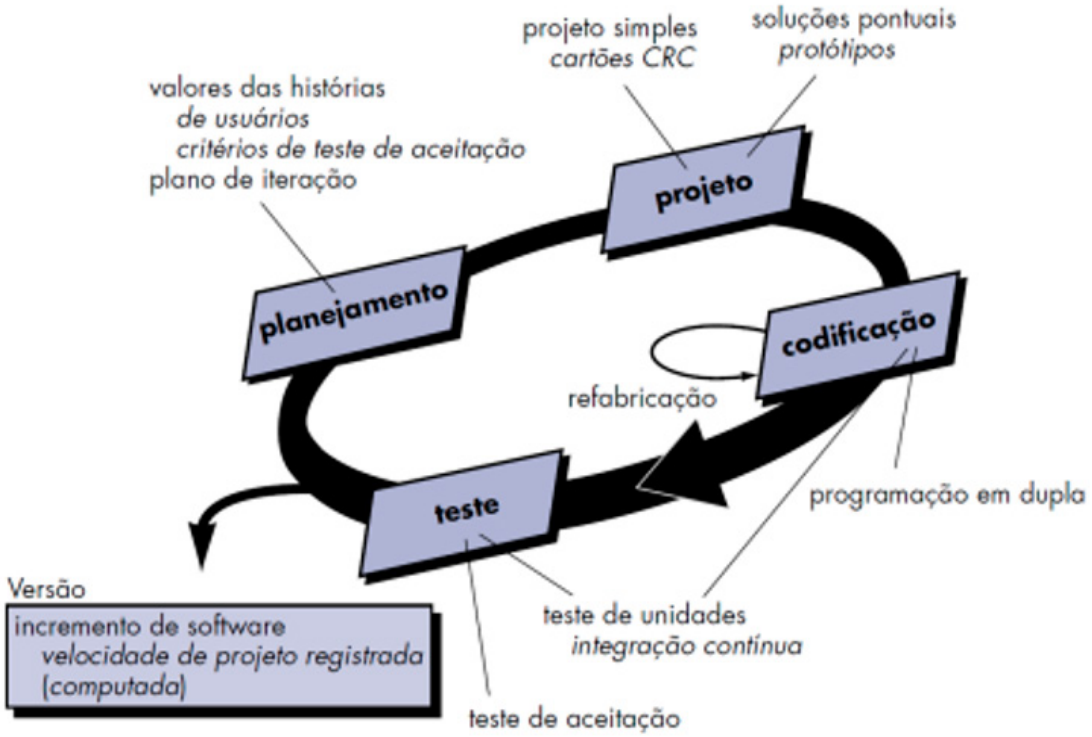
Processos/Etapas do XP:

- **Planejamento** (jogo do planejamento): inicia-se em ouvir as partes interessadas e elaborar as "histórias de usuários".
- **Projeto**: deve seguir o princípio da simplicidade, que considera mais um projeto simples do que uma representação complexa. O uso de cartões CRC (classe-responsabilidade-colaborador) é estimulado.

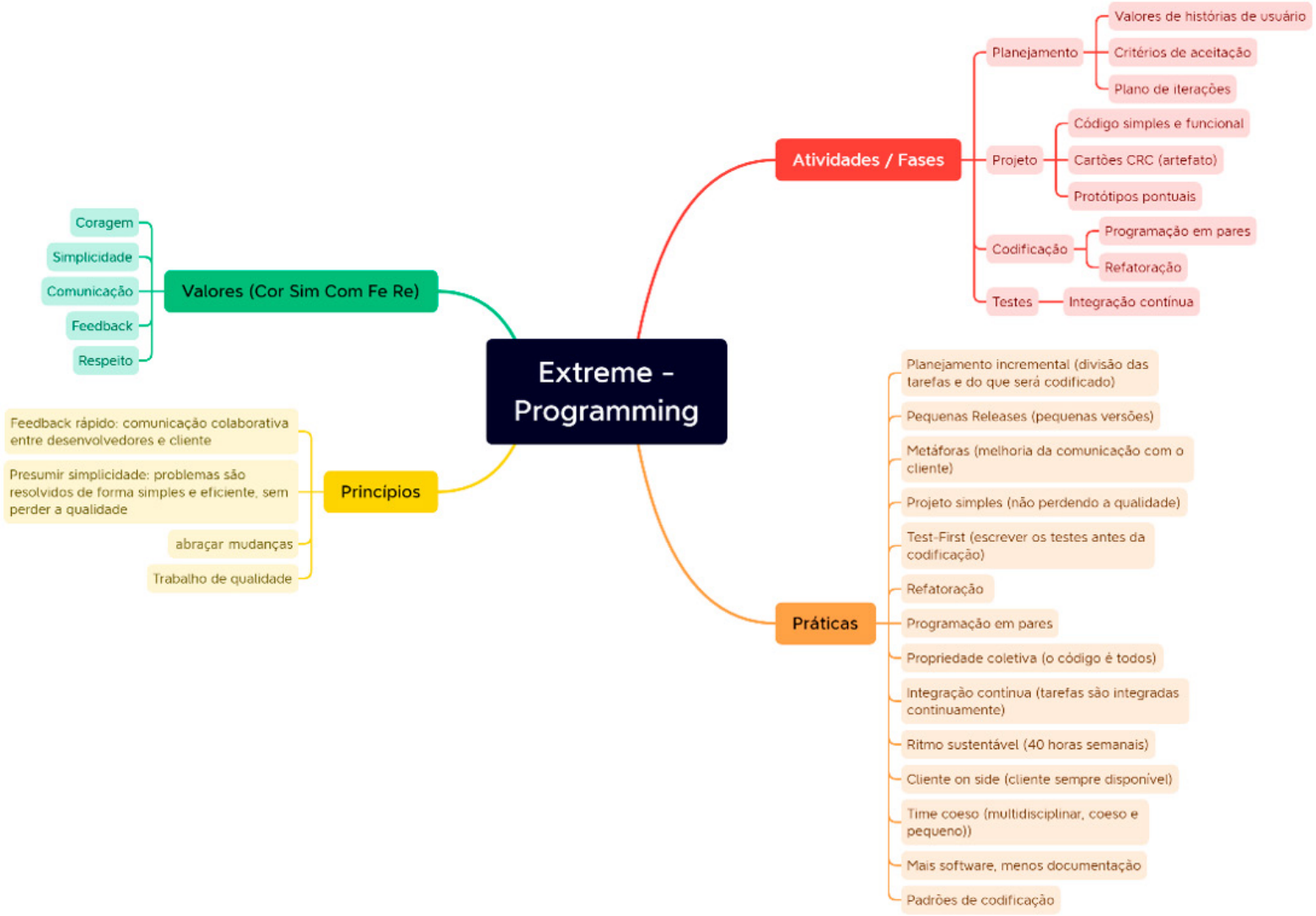
- **Codificação**: desenvolvimento de uma série de testes de unidade que serão aplicadas a cada uma das histórias de usuário. Somente após a implementação dos testes é que os desenvolvedores irão codificar de fato as histórias que irão agregar o produto (lembrando da programação em pares!).
- **Teste**: são criados e realizados os testes de aceitação, elaborados com base nas histórias de usuário. Os demais testes da aplicação também são executados para evitar possíveis falhas futuras (testes de regressão). Os testes de integração e validação do sistema podem ocorrer diariamente.

MAPAS MENTAIS

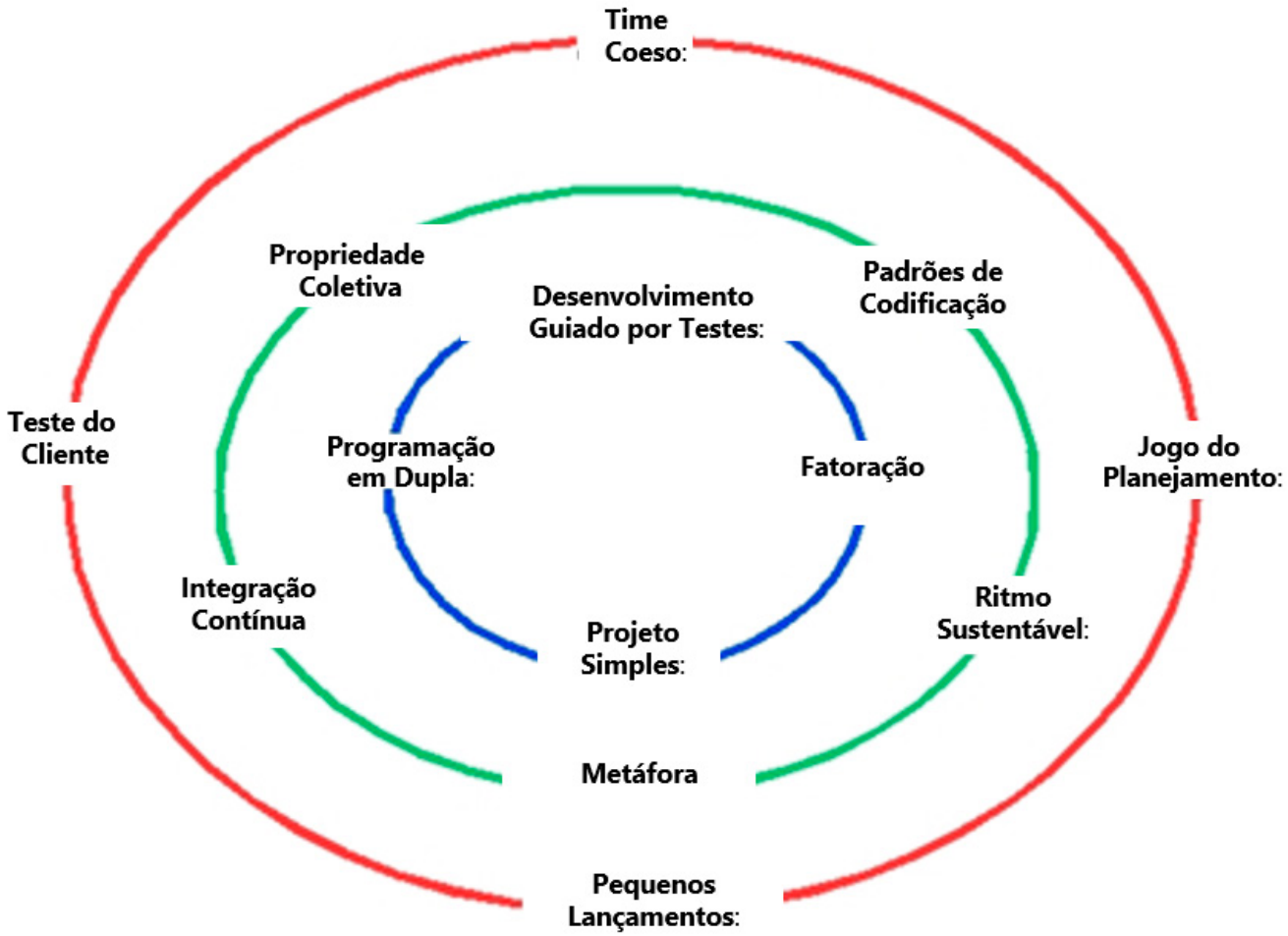
Valores do XP



Fases do XP segundo Presmann



Principais características do XP



Práticas do XP

QUESTÕES DE CONCURSO

001. (CESGRANRIO/BANCO DA AMAZÔNIA/TÉCNICO CIENTÍFICO/ANALISE DE SISTEMAS/2014) Uma prática que NÃO é adotada por *Extreme Programming* (XP) é:
a) usar duas pessoas trabalhando juntas em um único computador para produzir todo o código que será enviado para a produção.
b) criar os testes antes do código que será testado.
c) refatorar frequentemente, e ao longo de todo o projeto, o código produzido pelos desenvolvedores.
d) integrar continuamente o código recém-produzido com o código existente no repositório.
e) variar a duração de cada iteração durante todo o projeto para acomodar eventuais mudanças de prioridade dos requisitos, definidas pelo cliente.

002. (CESPE/CEBRASPE/PGE–RJ/ANALISTA DE SISTEMAS E MÉTODOS/2022) Julgue o próximo item, relativo a metodologias ágeis para a gestão de projetos e o desenvolvimento de *software*. A metodologia XP define que a programação seja feita em pares: dois desenvolvedores codificam o mesmo programa, ao mesmo tempo e no mesmo computador.

003. (CESGRANRIO/IBGE/ANALISTA DE SISTEMAS/ÁREA DESENVOLVIMENTO DE APLICAÇÕES/2010) O XP (*Extreme Programming*) usa uma abordagem orientada a objetos como seu paradigma de desenvolvimento predileto. Nessa perspectiva, analise as afirmativas abaixo.
I – A atividade de Codificação começa com a criação de um conjunto de histórias que descreve as características e as funcionalidades requeridas para o *software* a ser construído.
II – O XP encoraja o uso de cartões CRC (*Class-Responsibility-Colaborator*) como um mecanismo efetivo para raciocinar sobre o *software* no contexto orientado a objetos.
III – O XP emprega a técnica de *refectoring* na codificação, mas desaconselha a utilização da programação por pares.
IV – A criação de testes unitários antes da codificação começar é uma prática do XP.
V – Se um difícil problema de projeto é encontrado como parte do projeto de uma história, o XP recomenda a criação imediata de um protótipo operacional daquela parte do projeto.
Estão corretas APENAS as afirmativas:
a) I, II e IV.
b) I, III e IV.
c) I, IV e V.
d) II, III e V.
e) II, IV e V.

004. (CESGRANRIO/AERONÁUTICA/TÉCNICO DE DEFESA AÉREA E CONTROLE DE TRÁFEGO AÉREO/ÁREA ANÁLISE DE SISTEMAS/2006) Assinale a metodologia de desenvolvimento de *software* que tem como prática a programação em pares.
a) MSF
b) XP
c) RUP
d) PMBOK
e) CMMI

005. (CESGRANRIO/PETROBRAS PROFISSIONAL JÚNIOR/ANÁLISE DE SISTEMAS/2012) Dentre as metodologias de desenvolvimento ágil, a *eXtreme Programming* (XP) é uma das mais conhecidas. Nessa metodologia, são usados os conceitos de teste de aceitação (acceptance test) e de história do usuário (user story). Com relação às práticas recomendadas pela XP, analise as afirmações a seguir.
I – Uma história do usuário deve refletir corretamente as necessidades do cliente com relação a certa funcionalidade do sistema esperada por esse cliente.
II – É uma prática recomendada pela XP que um desenvolvedor especifique os cenários para os testes de aceitação de cada história de usuário que ele implemente.
III – Uma história de usuário é exatamente igual a um caso de uso, devendo ser utilizada em conjunto com um documento de requisitos e ter apenas um teste de aceitação escrito para ela.
É correto APENAS o que se afirma em:
a) I
b) II
c) III
d) I e II
e) II e III

006. CESGRANRIO/BANCO DA AMAZÔNIA TÉCNICO CIENTÍFICO/ANÁLISE DE SISTEMAS/2014) Uma prática que NÃO é adotada por *Extreme Programming* (XP) é:
a) usar duas pessoas trabalhando juntas em um único computador para produzir todo o código que será enviado para a produção.
b) criar os testes antes do código que será testado.
c) refatorar frequentemente, e ao longo de todo o projeto, o código produzido pelos desenvolvedores.
d) integrar continuamente o código recém-produzido com o código existente no repositório.
e) variar a duração de cada iteração durante todo o projeto para acomodar eventuais mudanças de prioridade dos requisitos, definidas pelo cliente.

007. (CESGRANRIO/BNDES/PROFISSIONAL BÁSICO/ANÁLISE DE SISTEMAS/DESENVOLVIMENTO/2013) Sendo atualmente conhecida por just-in-time, a produção enxuta contém princípios que compõem a base dos processos ágeis de desenvolvimento de software, como o *Extremme Programming* (XP).
Um dos princípios básicos do XP, a eliminação de desperdícios, busca:
a) evitar o efeito negativo que uma definição de risco, na fase inicial do projeto, possa causar na performance do *software* como um todo, tendo, como saída, informações não relevantes para o processo.
b) produzir requisitos bem definidos e completos de forma a abranger todos os processos e rotinas administrativas, funcionais e produtivas almejadas pelos *Stakeholders* envolvidos no projeto.
c) reduzir, o máximo possível, o volume de trabalho executado e os subprodutos envolvidos nesse trabalho, concentrando os esforços apenas no que pode produzir um resultado objetivo e palpável ao cliente final.
d) descrever os processos que garantam a inclusão, no projeto, de todo o serviço necessário, e somente o serviço necessário, para que esse projeto seja finalizado com sucesso.
e) descrever os processos envolvidos no planejamento, no monitoramento e na garantia de que o projeto será realizado dentro dos prazos definidos no escopo, mantendo a qualidade definida e o enxugamento dos custos inicialmente programados.

008. (CESGRANRIO/FINEP/ANALISTA/DESENVOLVIMENTO DE SISTEMAS/2011) São práticas recomendadas pelo processo ágil de desenvolvimento de software *Extreme Programming* (XP), EXCETO a:
a) Programação em Pares.
b) Integração Contínua.
c) Documentação Abundante e Detalhada.
d) Refatoração Frequente.
e) Padronização de Código.

009. (CESGRANRIO/ELETROBRAS/ANALISTA DE SISTEMAS/SUPORTE BASIS SAP R3/2010) No âmbito de desenvolvimento de sistemas, o XP tem como característica a programação em par, na qual o(a):
a) programador codifica em companhia do analista de requisitos.
b) testador realiza os casos de teste em companhia do programador.
c) código é escrito em par, reduzindo a inserção de *bugs*.
d) programador codifica em companhia do gestor do sistema.
e) disseminação do conhecimento não é priorizada, mas, sim, a individualidade.

010. (CESGRANRIO/BNDES/PROFISSIONAL BÁSICO/ANÁLISE DE SISTEMAS/DESENVOLVIMENTO/2009) Determinado projeto de software utiliza XP (*eXtreme Programming*) como metodologia de desenvolvimento. A esse respeito, é INCORRETO afirmar que:
a) o cliente participa ativamente e acompanha os passos dos desenvolvedores diariamente.
b) os integrantes da equipe se reúnem rapidamente no início do dia, de preferência em pé.
c) a equipe de desenvolvimento concentra esforços naquilo que gera maior valor para o cliente.
d) a programação em pares dispensa o desenvolvimento orientado a testes no projeto.
e) as funcionalidades do *software* são descritas em histórias, da forma mais simples possível.

011. (CESGRANRIO/BNDES/PROFISSIONAL BÁSICO/ANÁLISE DE SISTEMAS/DESENVOLVIMENTO/2008) Que situação favorece a escolha do uso de XP para um projeto de desenvolvimento de *software*, em oposição à escolha do RUP ou do modelo Cascata?
a) Equipe do projeto localizada em diferentes cidades e com poucos recursos de colaboração.
b) Equipe do projeto formada por pessoas com alto grau de competitividade.
c) Cliente do projeto trabalhando em parceria com a equipe do projeto e sempre disponível para retirar dúvidas.
d) Requisitos do *software* com pequena probabilidade de mudanças.
e) Presença de um processo organizacional que exige a elaboração de vários documentos específicos para cada projeto.

012. (CESGRANRIO/REFAP SA/ANALISTA DE SISTEMAS JÚNIOR/2007) NÃO é uma característica da *Extreme Programming* (XP):
a) simplicidade.
b) agilidade.
c) desenvolvimento orientado a testes.
d) programação em par.
e) documentação extensa e abundante em artefatos.

013. (CESPE/BNB/ESPECIALISTA TÉCNICO/ANALISTA DE SISTEMA/2018) Acerca dos métodos ágeis, julgue o próximo item.
Em XP, a técnica de *planning game* é utilizada pelo cliente para identificar as prioridades do que deve ser construído em um *software*, sem a participação dos desenvolvedores.

014. (CESPE/STJ/TÉCNICO JUDICIÁRIO/DESENVOLVIMENTO DE SISTEMAS/2018) Julgue o seguinte item, relativo à gestão ágil de projetos com XP (*extreme programming*).
A integração contínua descrita na XP pode fazer parte do processo de desenvolvimento de *software* que utiliza o *Scrum*, pois, à medida que as entregas sejam realizadas na *Scrum*, pode-se validá-las por meio de testes automatizados.

015. (CESGRANRIO/UNIRIO/2019) Uma das principais práticas de XP (*Extreme Programming*) é o *Iteration Planning Game*. Entre as atividades realizadas em uma sessão de *Iteration Planning*, está a:
a) definição, pelos programadores, de quais *story cards* serão implementados em uma iteração.
b) estimação do esforço que será necessário para implementar cada *story card*.
c) estimação da data de entrega de um release baseado na estimativa de esforço de cada *story card*.
d) estimação, feita por cada programador, do tempo que será necessário para realizar cada tarefa sob sua responsabilidade.
e) designação, por parte do *coach*, dos programadores que irão realizar as tarefas contidas na lista de tarefas.

016. (CESPE/ABIN/OFICIAL TÉCNICO DE INTELIGÊNCIA/ÁREA 9/2018) A respeito da metodologia XP (*Extreme Programming*), julgue o item que segue.
Para apoiar a equipe de desenvolvimento, é uma prática o uso do cliente *on-site* em tempo integral.

017. (FUNDATEC/IPE SAÚDE/ANALISTA DE GESTÃO EM SAÚDE/ENGENHARIA DA COMPUTAÇÃO/2022) O processo de desenvolvimento de software especificado pela Programação Extrema (*eXtreme Programming* – XP) começa com uma fase de planejamento, na qual são levantados e descritos requisitos para o *software* na forma de _____.
O projeto e desenvolvimento dos requisitos busca focar nas necessidades imediatas. Necessidades de melhoria no projeto são realizadas através de processos de _____.
Além disso, se recomenda que a atividade de codificação ocorra em _____ e seja guiada por _____.
Assinale a alternativa que preenche, correta e respectivamente, as lacunas do trecho acima.
a) histórias de usuários – refatoração – quartetos – testes
b) histórias de usuários – testes – pares – casos de uso
c) histórias de usuários – refatoração – pares – testes
d) modelos de domínio – refatoração – pares – testes
e) modelos de domínio – testes – quartetos – casos de uso

018. (FAURGS/TJ–RS/2018) Considere as seguintes afirmações sobre princípios ou práticas da XP (*Extreme Programming*).

I – Um representante do usuário final do sistema (cliente) deve estar disponível todo o tempo à equipe de XP. Em um processo de *Extreme Programming*, o cliente é um membro da equipe de desenvolvimento e é responsável por levar ao grupo os requisitos de sistema para implementação.
II – Todos os desenvolvedores devem refatorar o código continuamente, assim que encontrarem oportunidades de melhorias de código.
III – Os desenvolvedores trabalham em todas as áreas do sistema, de modo que não se desenvolvam ilhas de expertise. Todos os desenvolvedores têm responsabilidade em relação ao código; qualquer um pode mudar qualquer coisa.
Quais estão corretas?
a) Apenas I.
b) Apenas I e II.
c) Apenas I e III.
d) Apenas II e III.
e) I, II e III.

019. (CESPE/ABIN/OFICIAL TÉCNICO DE INTELIGÊNCIA/ÁREA 9/2018) A respeito da metodologia XP (*Extreme Programming*), julgue o item que se segue.
O ritmo ágil de desenvolvimento de *softwares* é uma prática usada para favorecer a entrega das *releases* quando grandes volumes de horas extras são tolerados.

020. (INSTITUTO AOCP/ADAF–AM/2018) Na metodologia ágil *Extreme Programming* (XP), a propriedade do código é coletiva, dessa forma, todos compartilham o mesmo orgulho e as mesmas críticas. Considerando o exposto, assinale a alternativa que apresenta uma das regras da codificação em XP.
a) No *Overtime*.
b) Eliminar gargalos de *hardware* no início.
c) O usuário não deve participar do planejamento das interfaces.
d) No *Outsourcing*.
e) No *Sprint*.

021. (CESPE/STM/ANALISTA JUDICIÁRIO/ANÁLISE DE SISTEMAS/2018) Julgue o próximo item, referente à metodologia de desenvolvimento de *software*.
Na XP (*Extreme Programming*), programadores trabalham em pares, e requisitos são expressos como cenários, denominados histórias de usuários, os quais são implementados como uma série de tarefas.

022. (CESPE/TRE–BA/TÉCNICO JUDICIÁRIO/PROGRAMAÇÃO DE SISTEMAS/2017) Considerando uma situação hipotética com o uso da XP (*eXtreme Programming*) concomitante com *Scrum* em um projeto de desenvolvimento de *software* em uma organização, julgue os seguintes itens.

I – É viável a utilização do TDD (*Test Driven Development*) na fase de *sprint*, de modo que se escreva o teste automático antes da codificação.

II – O princípio da integração contínua da XP deve ser utilizado especificamente na retrospectiva da *sprint* com vistas a integrar a equipe *Scrum*.

III – Integrantes da equipe *Scrum* podem realizar a programação do código em pares, o que proporciona, entre outras vantagens, o nivelamento de conhecimento da equipe.

IV – O conceito de requisito "pronto" continuaria válido, contudo, inviabilizaria o *refactoring*, pois é proibitivo inserir o mesmo item (requisito) em várias *sprints*.

Estão certos apenas os itens:

Alternativas

a) I e II.

b) I e III.

c) II e IV.

d) I, III e IV.

e) II, III e IV.

023. (CESPE/TRE–TO/TÉCNICO JUDICIÁRIO/PROGRAMAÇÃO DE SISTEMAS/2017) Em projetos de desenvolvimento de software, a *extreme programming* (XP) é um método ágil que usa a prática de:

a) projetos com planejamento completo sem incrementos.

b) grandes *releases*.

c) grande quantidade de horas extras.

d) trabalho em pares de desenvolvedores.

e) integrações após a entrega do *software* completo.

024. (CESPE/TRE–PE/TÉCNICO JUDICIÁRIO/PROGRAMAÇÃO DE SISTEMAS/2016) Com relação às metodologias ágeis XP, *Scrum* e UP e à metodologia RUP, assinale a opção correta.

a) A padronização da arquitetura de código-fonte é o foco principal da metodologia RUP.

b) As metodologias ágeis são focadas no produto, sendo caracterizadas pela ausência de modelo de dados, de diagramas de classes e de documentação de código-fonte.

c) A metodologia RUP, fundamentada em um modelo preditivo com foco no planejamento futuro, realiza entregas de *software* executável na mesma dinâmica das metodologias ágeis.

d) Os métodos ágeis podem ser classificados como métodos adaptativos à necessidade de desenvolvimento de *software*, pois mudam conforme a necessidade do projeto, diferentemente das metodologias tradicionais de desenvolvimento de *software*.
e) Uma metodologia clássica de engenharia de *software* deve ser aplicada em projetos com requisitos do sistema instáveis.

025. (CESPE/MPU/ANALISTA DE INFORMÁTICA/DESENVOLVIMENTO DE SISTEMAS/2010) *Extreme programming* (XP) é embasado em requisitos conhecidos, definidos de antemão, que não sofram muitas alterações, devendo ser usado por equipes de pequeno porte, formadas por representantes de todos os *Stakeholders*.

026. (CESPE/STF/ANALISTA JUDICIÁRIO/ANÁLISE DE SISTEMAS DE INFORMAÇÃO/2013) XP (*Extreme Programming*) é uma metodologia ágil voltada para equipes pequenas e médias que desenvolvam *software* baseado em requisitos vagos e se caracteriza por possibilitar modificações rápidas.

027. (IF–RS/2018) Sobre as práticas encontradas na metodologia ágil de desenvolvimento de software, conhecida por Programação Extrema (XP Programming), de acordo com Dooley (2017) no livro Software Development, Design and Coding, classifique cada uma das afirmativas abaixo como verdadeira (V) ou falsa (F) e assinale a alternativa que apresenta a sequência CORRETA, de cima para baixo:

( ) Participação intensa do representante do cliente no desenvolvimento do projeto.
( ) Testes são realizados continuamente. Quando todos os testes forem aprovados, o módulo foi concluído.
( ) Programação em par: enquanto um escreve o código, o outro monitora falhas, realiza testes, faz sugestões e planeja próximas ações.
( ) Lançamentos frequentes de novas versões.

a) F – V – V – F
b) V – F – F – V
c) V – V – F – V
d) V – V – V – V
e) F – F – V – V

028. (INSTITUTO AOCP/UFPB/2019) Um dos principais métodos ágeis de desenvolvimento de *software* foi concebido para impulsionar práticas reconhecidas como boas, por exemplo, o desenvolvimento iterativo a nível extremo, em que novas versões de um determinado sistema podem ser implementadas, integradas e, até mesmo, testadas em um único

dia por programadores diferentes. Essa é uma das características de qual método de desenvolvimento ágil de *software*?
a) *Scrum*.
b) *Adaptative Software Development*.
c) *Extreme Programming*.
d) *Pramatic Programming*.
e) *Test Driven Development*.

029. (FCM/PREFEITURA DE CARANAÍBA–MG/2019) De acordo com Pressman e Maxim (2016), a Programação Extrema (*Extreme Programming* – XP) é uma abordagem amplamente utilizada do desenvolvimento ágil de *software* que consiste das atividades:
a) Planejamento (*Planning*) / Projeto (*Designing*) / Codificação (*Coding*) / Teste (*Test*).
b) Colaboração (*Collaboration*) / Projeto (*Designing*) / Codificação (*Coding*) / Teste (*Test*).
c) Colaboração (*Collaboration*) / Projeto (*Designing*) / Codificação (*Coding*) / Teste (*Test*) / Adaptação (*Adaptation*).
d) Colaboração (*Collaboration*) / Projeto (*Designing*) / Codificação (*Coding*) / Teste (Test) / Adaptação (*Adaptation*) / Melhoria (*Improvement*)

030. (VUNESP/CÂMARA DE PIRACICABA–SP/2019) Um dos processos ágeis de desenvolvimento de software é a programação extrema (*extreme programming* – XP), cuja fase ou atividade inicial é composta pela descrição dos cenários (características e funcionalidades) requisitadas para o *software* a ser desenvolvido. Essa atividade recebe a denominação de:
a) métodos práticos.
b) histórias de usuário.
c) estruturas de apoio.
d) classes de projeto.
e) artefatos de usuário

031. (CESPE/TJ–AM/2019) No XP (*Extreme Programming*), o valor de uma história de usuário é atribuído pelos membros da equipe e é medido em termos de semanas estimadas para o desenvolvimento.

032. (QUADRIX/COBRA TECNOLOGIA SA (BB)/ANALISTA DE OPERAÇÕES/NEGÓCIOS/2014) O modelo ágil *Extreme Programming* (XP) segue uma série de práticas que dizem respeito ao relacionamento com o cliente, a gerência do projeto, a programação e aos testes. NAO é uma dessas práticas:
a) jogo do planejamento.
b) programação em pares.

c) desenvolvimento orientado a testes.
d) estabelecer e seguir padrões de codificação.
e) listar requisitos no *product backlog*.

033. (CESPE/ANATEL/ANALISTA ADMINISTRATIVO/TECNOLOGIA DA INFORMAÇÃO E COMUNICAÇÃO/2014) Acerca dos processos de desenvolvimento de *software*, julgue o item subsequente.
A etapa de planejamento do *Extreme Programming* (XP) inicia-se com a escrita de *User Stories* (história do usuário). Por meio dessa ferramenta, aqueles que conhecem a técnica de construção de uma solução poderão guiar quem necessita dessa solução no exercício de descrevê-la de forma simples e concisa.

034. (FCC/DPE–AM/2018) Considere a definição de algumas práticas da *eXtreme Programming* – XP.
I – Todo o código desenvolvido pelo time é incorporado em um repositório comum várias vezes ao dia. Isso garante que qualquer problema de integração ao longo do projeto possa ser notado e corrigido rapidamente. II. Qualquer programador do time pode alterar qualquer seção do código, se necessário. Por mais que esta prática pareça perigosa, ela aumenta a velocidade do desenvolvimento e problemas em potencial podem ser detectados pelos testes de unidade.
III – Traz a ideia de que qualquer pessoa do time seja capaz de verificar o código sendo desenvolvido em alto nível e ter uma compreensão clara de qual funcionalidade do sistema está sendo trabalhada.
IV – Permite aplicar melhorias ao código sem mudar sua funcionalidade, visando sua simplificação. Se o cliente deseja alterar alguma coisa no produto final, o time pode fazer os ajustes rapidamente, e esta prática contribui para alcançar este objetivo.
As práticas de I a IV são, correta e respectivamente,
a) *pair programming – test-driven development – system metaphor – continuous integration*.
b) *planning game – pair programming – system simplicity – continuous integration*.
c) *planning game – test-driven development – system simplicity – refactoring*.
d) *continuous integration – pair programming – feedback – planning game*.
e) *continuous integration – collective code ownership – system metaphor – refactoring*.

035. (INSTITUTO AOCP/PRODEB/2018) O método de desenvolvimento ágil denominado de XP (*Extreme Programming*) tem sua estrutura baseada em algumas prerrogativas, dentre as quais, é correto citar como princípios do XP:

a) Princípio da Legalidade, Princípio da Agilidade e Princípio do *Feedback*. b) Princípio da Coragem, Princípio da Agilidade e Princípio da Simplicidade.
c) Princípio da Comunicação, Princípio da Simplicidade, Princípio do *Feedback* e Princípio da Coragem.
d) Princípio da Agilidade, Princípio da Qualidade, Princípio do *Feedback* e Princípio da Coragem.
e) Princípio da Simplicidade, Princípio do Desenvolvimento e Princípio de Governança.

036. (INSTITUTO AOCP/UFOB/2018) Uma das práticas do *Extreme Programming* é o uso do código coletivo, na qual todos os desenvolvedores têm acesso ao código.

037. (INSTITUTO AOCP/PRODEB/2018) *Extreme Programming* (XP) é um método de desenvolvimento ágil amplamente utilizado pelas *software houses*. Com base neste método, qual alternativa a seguir possui uma prática que NÃO faz parte do XP?
a) *Small Releases* (Pequenas Entregas).
b) *Pair Programming* (Programação em Pares).
c) *Sprint Review* (Reunião de revisão da *Sprint*).
d) *Sustainable Pace* (Ritmo Saudável).
e) *Collective Owership* (Posse Coletiva).

038. (FCC/TCE–AL/2008) Originalmente, o único produto da atividade de Projeto que é realizado como parte do processo XP (*Extreme Programming*):
a) é a definição do caso de uso de contexto.
b) são os cartões CRC.
c) são os diagramas de objetos.
d) são os diagramas de sequência.
e) é a codificação, feita em pares.

039. (INSTITUTO AOCP/EBSERH/2017) O *Extreme Programming* (XP) surgiu em 1999, a partir de uma publicação sobre o assunto, mas suas bases se conectam a princípios da década de 80 e ao manifesto ágil. O XP é baseado em 4 atividades de arcabouços. Assinale a alternativa que contém 3 desses arcabouços:
a) Levantamento de requisitos, análise de negócio, planejamento e documentação.
b) Levantamento de requisitos, planejamento, documentação e implantação.
c) Levantamento de requisitos, documentação, codificação e teste.
d) Planejamento, projeto, documentação e implantação.
e) Governança, planejamento, codificação e teste.

040. (FCC/TRE–SE/2007) Na XP (*eXtreme Programming*):
a) deve-se usar o modelo em cascata para o desenvolvimento do *software*.
b) os programadores desenvolvem o *software* criando primeiramente os testes.
c) deve ser evitada a comunicação pessoal entre clientes e desenvolvedores, sempre dando preferência a outros meios de comunicação mais formais.
d) os programadores desenvolvem o *software* fazendo todos os testes possíveis no término do desenvolvimento.
e) deve-se projetar todas as funções possíveis com a máxima previsão do que ocorrerá no futuro, antes do desenvolvimento do *software*, a fim de evitar alterações desnecessárias.

041. (CESPE/PRODEST/2008) Projetar detalhadamente todo o *software* antes de iniciar a sua implementação é uma prática recomendada pelo XP. O *software* deve ser projetado para atender tanto aos requisitos atuais quanto aos potenciais requisitos futuros. Para atingir esse objetivo, são analisados os possíveis cenários de evolução futura e são empregados padrões de projeto para facilitar a manutenção.

042. (CESPE/ABIN/2010) Na *Extreme Programming*, os requisitos são expressos como cenários e implementados diretamente como uma série de tarefas. O representante do cliente faz parte do desenvolvimento e é responsável pela definição de testes de aceitação do sistema.

043. (CESPE/PRODEST/2008) Constituem práticas recomendadas pelo XP a colocação rápida de uma versão simples em produção, a liberação das novas versões em curtos intervalos de tempo, a programação em duplas, a refatoração (refactor) dos códigos produzidos, a adoção de padrões para a codificação; a integração e o teste contínuos de códigos; a limitação em 40 horas da carga de trabalho semanal.

044. (SERPRO/ANALISTA/DESENVOLVIMENTO DE SISTEMAS/2010) Cada sequência de etapas de um ciclo de vida aderente ao modelo em cascata é equivalente a uma iteração em um processo embasado no XP.

045. (SERPRO/ANALISTA/DESENVOLVIMENTO DE SISTEMAS/2010) Metodologias ágeis, como a XP, enfatizam a documentação de software no próprio código, que deve ser escrito por meio de ferramenta CASE voltada ao desenvolvimento rápido de aplicações (RAD Tools).

046. (TRE9–BA/ANALISTA JUDICIÁRIO/ANÁLISE DE SISTEMAS/2009) A metodologia XP prevê valores e princípios básicos para serem considerados durante o desenvolvimento de *software*. *Feedback*, coragem e respeito são exemplos de valores; mudanças incrementais, abraçar mudanças e trabalho de qualidade são exemplos de princípios básicos.

047. (TCU/AUDITOR FEDERAL DE CONTROLE EXTERNO/TECNOLOGIA DA INFORMAÇÃO/2010) O processo XP (*extreme programming*) envolve a realização das atividades de planejamento, de projeto, de codificação e de teste.

048. (TCU/AUDITOR FEDERAL DE CONTROLE EXTERNO/TECNOLOGIA DA INFORMAÇÃO/2010) A atividade de planejamento XP inclui a criação das denominadas histórias de usuário, nas quais devem ser descritas as características e as funcionalidades requeridas para o *software* em desenvolvimento.

049. (TCU/AUDITOR FEDERAL DE CONTROLE EXTERNO/TECNOLOGIA DA INFORMAÇÃO/2010) A atividade de projeto é uma desvantagem do processo XP, pelo fato de requerer uma quantidade de produtos de trabalho considerada excessiva pela comunidade de desenvolvimento de *software*.

050. (ANTAQ/ANALISTA ADMINISTRATIVO/INFORMÁTICA/2009) O *extreme programming* (XP) constitui método ágil de desenvolvimento de *software*. Uma das práticas que se enquadram nos princípios dos métodos ágeis é a programação em pares, que promove o compartilhamento da autoria do código do sistema. Além dessa vantagem, a programação em pares atua como processo informal de revisão porque cada linha de código é vista por pelo menos duas pessoas.

051. (IBFC/EBSERH/2017) Dentro das práticas do XP (*eXtreme Programming*) existe uma fundamental que é o Jogo de Planejamento (Planning Game). Para serem realizadas adequadamente essas reuniões com os usuários, deve(m) ter sido feito(s) antecipadamente:
a) o *Sustainable Pace*
b) as *Small Releases*
c) os *Customer Tests*
d) as *User stories*
e) um *Simple Design*

052. (CESPE/TJ–DFT/ANALISTA JUDICIÁRIO/ANALISTA DE SISTEMAS/2015) Na metodologia XP (*extreme programming*), em que todos os requisitos são expressos como cenários, deve-se aguardar, após a conclusão das tarefas, ciclos de cento e oitenta dias para a publicação de grandes *releases* do *software*.

053. (IBFC/TRE–PA/2020) Para aplicar valores e princípios do XP (*Extreme Programming*), durante os processos e práticas ágeis de desenvolvimento de *software*, se propõe uma série

específica de práticas. Assinale a alternativa que apresenta algumas dessas "boas práticas" utilizadas tradicionalmente em projetos, usando XP.
a) *Reformation - Pair Programming - PlayStation Game*
b) *Refactoring - Pair Programming - Planning Game*
c) *Reformation - Pair Production - Planning Game*
d) *Refactoring - Pair Production - PlayStation Game*

054. (IDECAN/UNIVASF/2019) *Extreme Programming* (XP), em sua essência, possui um conjunto de regras que devem ser seguidas em projetos ágeis que queiram utilizá-la em sua completude. Sobre as regras do XP, assinale a alternativa correta.
a) Apenas as operações mais críticas devem possuir testes unitários.
b) Todo o código-fonte de produção deve ser implementado em programação em pares.
c) A integração de código deve ser feita nos computadores dos desenvolvedores.
d) É importante estabelecer o papel de um XP Master, que será responsável pela implementação da metodologia.
e) A velocidade do projeto deve ser medida com o objetivo de informar ao cliente o tempo médio de correção de falhas.

055. (FCC/SANASA–CAMPINAS/2019) Em um projeto de *software* baseado na metodologia ágil XP, um Analista de TI deve:
a) consultar o cliente quando uma história exigir, por estimativa, menos do que 3 semanas de desenvolvimento, para que o cliente a complemente com mais tarefas.
b) ouvir o cliente, durante o levantamento de requisitos, para que este crie as histórias de usuários. Após essa importante etapa nenhuma história nova deve ser criada para não comprometer o cronograma do projeto.
c) evitar que o projeto caia na armadilha de seguir o princípio KISS de forma a estimular que o projeto de uma funcionalidade extra, que poderá ser necessária no futuro, faça parte do modelo do *software*.
d) realizar os testes de unidade de forma manual, evitando que sejam usadas baterias de testes automatizados, pois estes impedem a realização de testes de regressão.
e) estimular o uso de cartões CRC como um mecanismo eficaz para pensar o software em um contexto orientado a objetos.

056. (INSTITUTO AOCP/MPE–BA/2014) O processo ágil XP possui doze práticas que são os princípios fundamentais do processo. A prática que encoraja a equipe inteira a trabalhar mais unida em busca de qualidade no código fazendo melhorias e refatoramentos em qualquer parte do código a qualquer tempo é conhecida como:

a) propriedade coletiva do código.
b) semana de quarenta horas.
c) programação em pares.
d) padrões de codificação.
e) integração contínua.

057. (FGV/CÂMARA MUNICIPAL DO RECIFE–PE/2014) Uma das práticas do método ágil XP (*eXtreme Programming*) é:
a) documentação extensiva;
b) prototipação;
c) ciclos longos de desenvolvimento;
d) desenvolvimento orientado a testes (TDD);
e) utilização de todos os artefatos do RUP.

058. (CESPE/ANTT/2013) São práticas ou princípios recomendados no modelo de desenvolvimento de software XP (*eXtreme Programming*) proposto por Kent Beck: programação em pares; semana de trabalho de 40 horas; refatoração sem piedade; desenvolvimento orientado a testes TDD (Test Driven Development); e desenvolvimento de metáforas arquiteturais.

059. (CESPE/MPU/2013) XP é um método de desenvolvimento de *software* em que os requisitos são especificados em *user stories*; requisitos, arquitetura e *design* surgem durante o curso do projeto; e o desenvolvimento ocorre de maneira incremental.

060. (CESPE/TCE–RO/2019) No que diz respeito a processos e práticas ágeis, o desenvolvimento incremental:
a) é, assim como o *test-driven development*, uma prática da XP (*Extreme Programming*) que exige teste automatizado, *domain-driven design*, *refactoring* e integração contínua.
b) é, na XP (*Extreme Programming*), sustentado por meio de pequenos e frequentes *releases* do sistema, e os clientes estão intimamente envolvidos na especificação e na priorização dos requisitos do sistema.
c) enfoca, assim como o *acceptance test-driven development*, a qualidade do código desenvolvido quanto a recursividade, declaração das variáveis e *clean code*, de modo a torná-lo de fácil entendimento, modificação e testagem.

d) pressupõe o uso do *behavior driven development*, que considera a linguagem de programação a ser usada, da 4ª geração em diante, com foco, principalmente, no comportamento visual, interativo e cognitivo do sistema.
e) enfoca a integração contínua como uma prática de desenvolvimento de *software*, incompatível com a XP (*Extreme Programming*) e o *Scrum*, que permite aos desenvolvedores agregarem alterações de código e realizarem testes.

061. (INSTITUTO CONSULPLAN/TJM–MG/ANALISTA JUDICIÁRIO/TECNOLOGIA DA INFORMAÇÃO/2021) O XP (*Extreme Programming*), uma metodologia ágil de desenvolvimento, foi empregado, pela primeira vez, em 1996, em um projeto da Chrysler, chamado de C3 (Chrysler Comprehensive Compensation). Considerando que são apresentados cinco principais valores, assinale, a seguir, dois desses valores.
a) Revisão e Respeito
b) Coragem e Respeito.
c) Simplicidade e Revisão.
d) *Feedback* e Informação.

062. (CESPE/TCU/AUDITOR FEDERAL DE CONTROLE EXTERNO/TECNOLOGIA DA INFORMAÇÃO/PARTE II/2010) A atividade de planejamento XP inclui a criação das denominadas histórias de usuário, nas quais devem ser descritas as características e as funcionalidades requeridas para o *software* em desenvolvimento.

063. (CESPE/ANAC/ANALISTA ADMINISTRATIVO/TECNOLOGIA DA INFORMAÇÃO/2009) *Extreme Programming* é um modelo de processo de desenvolvimento de *software* para equipes com grande número de pessoas, que desenvolvem *software* com base em requisitos vagos e que são modificados rapidamente.

064. (CESPE/MINISTÉRIO DA ECONOMIA/2020) O XP possui planejamento incremental com requisitos registrados em histórias.

065. (CESPE/MINISTÉRIO DA ECONOMIA/2020) O *refactoring* de código não faz parte do modelo XP, visto que a expectativa é a entrega ágil, e não deve ser considerada em tempo de projeto a recriação de código para aprimoramento.

066. (UFCG/UFCG/2019) Marque a alternativa INCORRETA com relação a *Extreme Programming* (XP).
a) Comunicação, coragem e respeito são valores dessa metodologia.
b) Em XP, uma das regras é codificar os testes de unidade primeiro.

c) Refatoramento é uma prática recomendada nesse processo.
d) O código a ser enviado para produção é criado por duas pessoas trabalhando juntas em um único computador.
e) O autor, Don Wells, exige que o processo seja seguido à risca, de forma que todas suas regras devem ser respeitadas e, nenhum projeto pode ser realizado sem adaptações e/ou remoção dessas regras.

067. (IBFC/EBSERH/ANALISTA DE TECNOLOGIA DA INFORMAÇÃO/PROCESSOS (HU-FURG)/2016) Para aplicar os valores e princípios durante o desenvolvimento de software, a Programação Extrema (*eXtreme Programming* – XP) propõe uma série de práticas. Selecione a única alternativa que NÃO seja uma dessas práticas:
a) Time Coeso (*Whole Team*).
b) Design Complexo (*Complex Design*).
c) Programação Pareada (*Pair Programming*).
d) Semana de 40 horas (*Sustainable Pace*).
e) Refatoração (*Refactoring*).

068. (IBFC/TRE–PA/ANALISTA JUDICIÁRIO/ANÁLISE DE SISTEMAS/2020) Para aplicar valores e princípios do XP (*Extreme Programming*), durante os processos e práticas ágeis de desenvolvimento de *software*, se propõe uma série específica de práticas. Assinale a alternativa que apresenta algumas dessas "boas práticas" utilizadas tradicionalmente em projetos, usando XP.
a) *Reformation – Pair Programming – PlayStation Game*
b) *Refactoring – Pair Programming – Planning Game*
c) *Reformation – Pair Production – Planning Game*
d) *Refactoring – Pair Production – PlayStation Game*

069. (IBFC/EBSERH/ANALISTA DE TECNOLOGIA DA INFORMAÇÃO/PROCESSOS (HUAP-UFF)/2016) Equipes XP (*eXtreme Programming*) planejam utilizando histórias escritas em pequenos cartões. Essas histórias devem ter como objetivo:
a) a modelagem de dados.
b) as métricas de *software*.
c) os requisitos não funcionais.
d) os requisitos funcionais.
e) tanto os requisitos funcionais como os requisitos não funcionais.

070. (IBFC/TJ–PE/TÉCNICO JUDICIÁRIO/PROGRAMADOR DE COMPUTADOR/2017) Está sendo implementado o XP (*eXtreme Programming*) em uma equipe de TI. Para tanto, está sendo colocada a seguinte série de práticas específicas da metodologia XP em análise:
I – Programação Pareada (*Pair Programming*).
II – Fases pequenas (*Small Releases*).
III – Refatoração (*Refactoring*).
IV – Jogo de Planejamento (*Planning Game*).
Com base no seu conhecimento sobre a metodologia citada acima, suas práticas específicas estão corretamente relacionadas nos itens:
a) I, II e III, apenas.
b) I, II e IV, apenas.
c) II, III e IV, apenas.
d) I, III e IV, apenas.
e) I, II, III e IV.

071. (CESPE/CEBRASPE/TCE–RJ/ANALISTA DE CONTROLE EXTERNO/2022) Acerca de RUP (rational unified process) e XP (*extreme programming*), julgue o seguinte item.
Na XP, as histórias dos usuários (casos de uso) devem descrever os detalhes dos requisitos da solução, tais como a tecnologia a ser utilizada e a modelagem do banco de dados; isso irá permitir planejar melhor a interface do usuário na *release planning* e, consequentemente, o desenvolvimento da solução.

072. (IBFC/EBSERH/ANALISTA DE TECNOLOGIA DA INFORMAÇÃO/PROCESSOS (HUGG–UNIRIO)/2017) Dentro das práticas do XP (*eXtreme Programming*) existe uma fundamental que é o Jogo de Planejamento (Planning Game). Para serem realizadas adequadamente essas reuniões com os usuários, deve(m) ter sido feito(s) antecipadamente:
a) o *Sustainable Pace*
b) as *Small Releases*
c) os *Customer TestsD*
d) as *User stories*
e) um *Simple Design*

073. (CESPE/CEBRASPE/TCE–RJ/ANALISTA DE CONTROLE EXTERNO/2022) Acerca de RUP (rational unified process) e XP (*extreme programming*), julgue o seguinte item.
Uma das práticas da XP é a integração contínua que visa aperfeiçoar o projeto de codificação do sistema de *software*, tal que a estrutura interna se aprimora sem que seu comportamento se altere.

074. (VUNESP/CÂMARA DE PIRACICABA–SP/2019) Um dos processos ágeis de desenvolvimento de software é a programação extrema (*extreme programming* – XP), cuja fase ou atividade inicial é composta pela descrição dos cenários (características e funcionalidades) requisitadas para o *software* a ser desenvolvido. Essa atividade recebe a denominação de:
a) métodos práticos.
b) histórias de usuário.
c) estruturas de apoio.
d) classes de projeto.
e) artefatos de usuário

075. (CESPE/MINISTÉRIO DA ECONOMIA/2020) Grandes quantidades de horas extras são aceitáveis em médio e longo prazo, para agilizar a entrega de requisitos.

076. (CESPE/CEBRASPE/SERPRO/ANALISTA/ESPECIALIZAÇÃO: DESENVOLVIMENTO DE SISTEMAS/2021) Acerca de metodologias ágeis de desenvolvimento, julgue o item seguinte.
Em XP, a estruturação do valor *feedback* pode ser alcançada de forma rápida por meio de testes automatizados de *software*, que validam ou não um código produzido ou alterado.

077. (CESPE/CEBRASPE/MINISTÉRIO DA ECONOMIA/TECNOLOGIA DA INFORMAÇÃO/ DESENVOLVIMENTO DE SOFTWARE/2020) Julgue o item seguinte, a respeito de programação ágil com XP (*extreme programming*).
Como forma de agilizar as implantações de novas releases nesse modelo, são acumulados grandes grupos de funcionalidades e implantadas grandes *releases*.

078. (CESPE/CEBRASPE/EBC/2011) Uma metodologia de desenvolvimento de software pode ser classificada como uma metodologia ágil quando efetua o desenvolvimento do software de forma incremental (libera pequenas versões, em iterações de curta duração) e é colaborativa (cliente e desenvolvedores trabalham juntos, em constante comunicação), direta (o método em si é simples de aprender e modificar) e adaptativa (capaz de responder eficientemente às mudanças).
Considerando a definição acima, de Abrahamsson, julgue o item a seguir, a respeito das metodologias ágeis de desenvolvimento de *software*.
O que os métodos ágeis buscam é como evitar as mudanças desde o início do projeto e não a melhor maneira de tratar essas mudanças.

079. (CESPE/CEBRASPE/TJ–AM/ANALISTA JUDICIÁRIO/ANALISTA DE SISTEMAS/2019) Julgue o item seguinte, a respeito das metodologias de desenvolvimento de *software*.

No XP (*Extreme Programming*), o valor de uma história de usuário é atribuído pelos membros da equipe e é medido em termos de semanas estimadas para o desenvolvimento.

080. (IBFC/EBSERH/ANALISTA DE TECNOLOGIA DA INFORMAÇÃO/PROCESSOS (HU-FURG)/2016) Para aplicar os valores e princípios durante o desenvolvimento de software, a Programação Extrema (*eXtreme Programming* – XP) propõe uma série de práticas. Selecione a única alternativa que NÃO seja uma dessas práticas:
a) Time Coeso (*Whole Team*).
b) Design Complexo (*Complex Design*).
c) Programação Pareada (*Pair Programming*).
d) Semana de 40 horas (*Sustainable Pace*).
e) Refatoração (*Refactoring*).

081. (IBFC/TRE–PA/ANALISTA JUDICIÁRIO/ANÁLISE DE SISTEMAS/2020) Para aplicar valores e princípios do XP (*Extreme Programming*), durante os processos e práticas ágeis de desenvolvimento de *software*, se propõe uma série específica de práticas. Assinale a alternativa que apresenta algumas dessas "boas práticas" utilizadas tradicionalmente em projetos, usando XP.
a) *Reformation – Pair Programming – PlayStation Game*
b) *Refactoring – Pair Programming – Planning Game*
c) *Reformation – Pair Production – Planning Game*
d) *Refactoring – Pair Production – PlayStation Game*

082. (IBFC/TJ–PE/TÉCNICO JUDICIÁRIO/PROGRAMADOR DE COMPUTADOR/2017) Está sendo implementado o XP (*eXtreme Programming*) em uma equipe de TI. Para tanto, está sendo colocada a seguinte série de práticas específicas da metodologia XP em análise:
I – Programação Pareada (*Pair Programming*).
II – Fases pequenas (*Small Releases*).
III – Refatoração (*Refactoring*).
IV – Jogo de Planejamento (*Planning Game*).
Com base no seu conhecimento sobre a metodologia citada acima, suas práticas específicas estão corretamente relacionadas nos itens:
a) I, II e III, apenas.
b) I, II e IV, apenas.
c) II, III e IV, apenas.
d) I, III e IV, apenas.
e) I, II, III e IV.

083. (IBFC/EBSERH/ANALISTA DE TECNOLOGIA DA INFORMAÇÃO/PROCESSOS (HUGG–UNIRIO)/2017) Dentro das práticas do XP (*eXtreme Programming*) existe uma fundamental que é o Jogo de Planejamento (Planning Game). Para serem realizadas adequadamente essas reuniões com os usuários, deve(m) ter sido feito(s) antecipadamente:

a) o *Sustainable Pace*

b) as *Small Releases*

c) os *Customer TestsD*

d) as *User stories*

e) um *Simple Design*

084. (IBFC/EBSERH/ANALISTA DE TECNOLOGIA DA INFORMAÇÃO/PROCESSOS (HUAP-UFF)/2016) Equipes XP (*eXtreme Programming*) planejam utilizando histórias escritas em pequenos cartões. Essas histórias devem ter como objetivo:

a) a modelagem de dados.

b) as métricas de *software.*

c) os requisitos não funcionais.

d) os requisitos funcionais.

e) tanto os requisitos funcionais como os requisitos não funcionais.

GABARITO

1. e
2. C
3. e
4. b
5. a
6. e
7. c
8. c
9. c
10. d
11. c
12. e
13. E
14. C
15. d
16. C
17. c
18. e
19. E
20. a
21. C
22. b
23. d
24. d
25. E
26. C
27. d
28. c
29. a
30. b
31. E
32. e
33. C
34. e
35. c
36. C
37. c
38. b
39. d
40. b
41. E
42. C
43. C
44. E
45. E
46. C
47. C
48. C
49. E
50. C
51. d
52. E
53. b
54. b
55. e
56. a
57. d
58. C
59. C
60. b
61. b
62. C
63. E
64. C
65. E
66. b
67. b
68. b
69. d
70. e
71. E
72. d
73. E
74. b
75. E
76. C
77. E
78. E
79. E
80. b
81. b
82. e
83. d
84. d

GABARITO COMENTADO

001. (CESGRANRIO/BANCO DA AMAZÔNIA/TÉCNICO CIENTÍFICO/ANALISE DE SISTEMAS/2014) Uma prática que NÃO é adotada por *Extreme Programming* (XP) é:

a) usar duas pessoas trabalhando juntas em um único computador para produzir todo o código que será enviado para a produção.

b) criar os testes antes do código que será testado.

c) refatorar frequentemente, e ao longo de todo o projeto, o código produzido pelos desenvolvedores.

d) integrar continuamente o código recém-produzido com o código existente no repositório.

e) variar a duração de cada iteração durante todo o projeto para acomodar eventuais mudanças de prioridade dos requisitos, definidas pelo cliente.

A duração das iterações no XP varia entre 1 a 3 semanas e não é alterada. A quantidade de iterações pode ser variada.

Não precisamos ser muito criteriosos com relação ao item "a". Ele informa que a dupla da programação em pares produzirá todo o código que será desenvolvido por eles (a demanda destes programadores) para produção.

Letra e.

---

002. (CESPE/CEBRASPE/PGE–RJ/ANALISTA DE SISTEMAS E MÉTODOS/2022) Julgue o próximo item, relativo a metodologias ágeis para a gestão de projetos e o desenvolvimento de *software*.

A metodologia XP define que a programação seja feita em pares: dois desenvolvedores codificam o mesmo programa, ao mesmo tempo e no mesmo computador.

O time de desenvolvimento trabalha em pares, um vistoriando o trabalho do outro e fornecendo apoio ao outro. Eles utilizam a mesma máquina para trabalhar.

Certo.

---

003. (CESGRANRIO/IBGE/ANALISTA DE SISTEMAS/ÁREA DESENVOLVIMENTO DE APLICAÇÕES/2010) O XP (*Extreme Programming*) usa uma abordagem orientada a objetos como seu paradigma de desenvolvimento predileto. Nessa perspectiva, analise as afirmativas abaixo.

I – A atividade de Codificação começa com a criação de um conjunto de histórias que descreve as características e as funcionalidades requeridas para o *software* a ser construído.

II – O XP encoraja o uso de cartões CRC (*Class-Responsibility-Colaborator*) como um mecanismo efetivo para raciocinar sobre o *software* no contexto orientado a objetos.
III – O XP emprega a técnica de *refectoring* na codificação, mas desaconselha a utilização da programação por pares.
IV – A criação de testes unitários antes da codificação começar é uma prática do XP.
V – Se um difícil problema de projeto é encontrado como parte do projeto de uma história, o XP recomenda a criação imediata de um protótipo operacional daquela parte do projeto.
Estão corretas APENAS as afirmativas:
a) I, II e IV.
b) I, III e IV.
c) I, IV e V.
d) II, III e V.
e) II, IV e V.

I – Incorreto. XP engloba um conjunto constante de regras e práticas distribuídas em quatro atividades metodológicas: Planejamento, Projeto, Codificação e Testes. Durante a atividade de Planejamento, conhecida como o jogo do planejamento, são concebidas as histórias de usuários que descrevem os resultados, características e funcionalidades do *software* a ser construído.
II – Correto. O XP incentiva a aplicação de cartões CRC como uma ferramenta eficaz para a análise do software em um contexto orientado a objetos. Esses cartões identificam e organizam as classes relevantes para o incremento atual do software, sendo o único artefato de projeto produzido no processo XP.
III – Incorreto. Um conceito central na atividade de codificação (e um dos aspectos mais debatidos do XP) é a programação em pares. O XP recomenda que duas pessoas colaborem em uma única estação de trabalho para desenvolver o código relacionado a uma história.
IV – Correto. A elaboração de testes unitários antes de iniciar a codificação é um dos pilares do XP.
V – Correto. Se um problema de projeto complexo é identificado durante o desenvolvimento de uma história, o XP recomenda a criação imediata de um protótipo operacional para essa parte do projeto, conhecido como solução pontual. Esse protótipo é implementado e avaliado para reduzir o risco antes do início da implementação real, validando as estimativas originais para a história que inclui o problema de projeto.
Letra e.

---

004. (CESGRANRIO/AERONÁUTICA/TÉCNICO DE DEFESA AÉREA E CONTROLE DE TRÁFEGO AÉREO/ÁREA ANÁLISE DE SISTEMAS/2006) Assinale a metodologia de desenvolvimento de *software* que tem como prática a programação em pares.

a) MSF
b) XP
c) RUP
d) PMBOK
e) CMMI

Questão sem muita dificuldade. XP prega a programação em pares.

Letra b.

---

005. (CESGRANRIO/PETROBRAS PROFISSIONAL JÚNIOR/ANÁLISE DE SISTEMAS/2012) Dentre as metodologias de desenvolvimento ágil, a *eXtreme Programming* (XP) é uma das mais conhecidas. Nessa metodologia, são usados os conceitos de teste de aceitação (acceptance test) e de história do usuário (user story). Com relação às práticas recomendadas pela XP, analise as afirmações a seguir.

I – Uma história do usuário deve refletir corretamente as necessidades do cliente com relação a certa funcionalidade do sistema esperada por esse cliente.

II – É uma prática recomendada pela XP que um desenvolvedor especifique os cenários para os testes de aceitação de cada história de usuário que ele implemente.

III – Uma história de usuário é exatamente igual a um caso de uso, devendo ser utilizada em conjunto com um documento de requisitos e ter apenas um teste de aceitação escrito para ela.

É correto APENAS o que se afirma em:

a) I
b) II
c) III
d) I e II
e) II e III

II – Os testes de aceitação do XP, também chamados de testes do cliente, são especificados pelo cliente...", indicando que os testes de aceitação são determinados pelo cliente.

III – Idealmente, as histórias de usuário são escritas pelo cliente e são o seu principal método de influência no desenvolvimento dos sistemas. Os testes de aceitação fazem parte do aspecto de confirmação e dão origem aos casos de teste do sistema.

Letra a.

---

006. CESGRANRIO/BANCO DA AMAZÔNIA TÉCNICO CIENTÍFICO/ANÁLISE DE SISTEMAS/2014) Uma prática que NÃO é adotada por *Extreme Programming* (XP) é:

a) usar duas pessoas trabalhando juntas em um único computador para produzir todo o código que será enviado para a produção.

b) criar os testes antes do código que será testado.

c) refatorar frequentemente, e ao longo de todo o projeto, o código produzido pelos desenvolvedores.

d) integrar continuamente o código recém-produzido com o código existente no repositório.

e) variar a duração de cada iteração durante todo o projeto para acomodar eventuais mudanças de prioridade dos requisitos, definidas pelo cliente.

Questão capciosa.

a) Certa. "produzir todo o código que será enviado para a produção" – atente-se que ele informa sobre o código que será enviado para produção e não todo o código do sistema.

e) Errada. O desenvolvimento utilizando o XP é feito em iterações. Uma iteração é um período curto de tempo (1 ou 2 semanas) quando a equipe desenvolve um conjunto de funcionalidades. Sendo assim, no início da semana, desenvolvedores e clientes se reúnem para priorizar as funcionalidades. A duração das iterações não altera, somente a quantidade de vezes que elas são executadas.

Letra e.

---

007. (CESGRANRIO/BNDES/PROFISSIONAL BÁSICO/ANÁLISE DE SISTEMAS/DESENVOLVIMENTO/2013) Sendo atualmente conhecida por just-in-time, a produção enxuta contém princípios que compõem a base dos processos ágeis de desenvolvimento de software, como o Extremme Programming (XP).

Um dos princípios básicos do XP, a eliminação de desperdícios, busca:

a) evitar o efeito negativo que uma definição de risco, na fase inicial do projeto, possa causar na performance do *software* como um todo, tendo, como saída, informações não relevantes para o processo.

b) produzir requisitos bem definidos e completos de forma a abranger todos os processos e rotinas administrativas, funcionais e produtivas almejadas pelos *Stakeholders* envolvidos no projeto.

c) reduzir, o máximo possível, o volume de trabalho executado e os subprodutos envolvidos nesse trabalho, concentrando os esforços apenas no que pode produzir um resultado objetivo e palpável ao cliente final.

d) descrever os processos que garantam a inclusão, no projeto, de todo o serviço necessário, e somente o serviço necessário, para que esse projeto seja finalizado com sucesso.
e) descrever os processos envolvidos no planejamento, no monitoramento e na garantia de que o projeto será realizado dentro dos prazos definidos no escopo, mantendo a qualidade definida e o enxugamento dos custos inicialmente programados.

XP prega que a equipe deve se concentrar em executar o esforço necessário para alcançar os resultados esperados. Esse conceito está ligado a uma das bases do XP chamado simplicidade que busca eliminar possíveis desperdícios de recursos.
Letra c.

---

008. (CESGRANRIO/FINEP/ANALISTA/DESENVOLVIMENTO DE SISTEMAS/2011) São práticas recomendadas pelo processo ágil de desenvolvimento de software *Extreme Programming* (XP), EXCETO a:
a) Programação em Pares.
b) Integração Contínua.
c) Documentação Abundante e Detalhada.
d) Refatoração Frequente.
e) Padronização de Código.

O foco do XP está na entrega contínua de *software* funcional e na resposta ágil às mudanças nos requisitos do cliente. Isso não significa que o XP não valorize a documentação, mas sim que prefere formas mais leves e práticas, dando prioridade à comunicação cara a cara e à colaboração direta.
O XP encoraja a criação de documentação mínima, apenas o suficiente para atender às necessidades imediatas da equipe.
Os princípios do XP priorizam o código fonte como a forma mais atualizada e precisa de documentação do sistema. Essa abordagem favorece a flexibilidade e a adaptabilidade, permitindo que a equipe XP responda rapidamente às mudanças nos requisitos e mantenha um ciclo de desenvolvimento ágil.
Letra c.

---

009. (CESGRANRIO/ELETROBRAS/ANALISTA DE SISTEMAS/SUPORTE BASIS SAP R3/2010) No âmbito de desenvolvimento de sistemas, o XP tem como característica a programação em par, na qual o(a):

a) programador codifica em companhia do analista de requisitos.
b) testador realiza os casos de teste em companhia do programador.
c) código é escrito em par, reduzindo a inserção de *bugs*.
d) programador codifica em companhia do gestor do sistema.
e) disseminação do conhecimento não é priorizada, mas, sim, a individualidade.

Na programação em pares, dois programadores trabalham juntos em um único computador para realizar uma tarefa de programação. Um dos programadores assume o papel de "piloto", que está no controle do teclado e do mouse, enquanto o outro assume o papel de "navegador", revisando cada linha de código conforme ela é escrita.
Benefícios: melhoria na qualidade do código, compartilhamento de conhecimento, tomada de decisões colaborativa e redução de erros e *bugs*.
Letra c.

---

010. (CESGRANRIO/BNDES/PROFISSIONAL BÁSICO/ANÁLISE DE SISTEMAS/DESENVOLVIMENTO/2009) Determinado projeto de software utiliza XP (*eXtreme Programming*) como metodologia de desenvolvimento. A esse respeito, é INCORRETO afirmar que:
a) o cliente participa ativamente e acompanha os passos dos desenvolvedores diariamente.
b) os integrantes da equipe se reúnem rapidamente no início do dia, de preferência em pé.
c) a equipe de desenvolvimento concentra esforços naquilo que gera maior valor para o cliente.
d) a programação em pares dispensa o desenvolvimento orientado a testes no projeto.
e) as funcionalidades do *software* são descritas em histórias, da forma mais simples possível.

Conforme já estudamos, a programação em pares é uma prática central no *Extreme Programming* (XP)
Letra d.

---

011. (CESGRANRIO/BNDES/PROFISSIONAL BÁSICO/ANÁLISE DE SISTEMAS/DESENVOLVIMENTO/2008) Que situação favorece a escolha do uso de XP para um projeto de desenvolvimento de *software*, em oposição à escolha do RUP ou do modelo Cascata?
a) Equipe do projeto localizada em diferentes cidades e com poucos recursos de colaboração.
b) Equipe do projeto formada por pessoas com alto grau de competitividade.
c) Cliente do projeto trabalhando em parceria com a equipe do projeto e sempre disponível para retirar dúvidas.

d) Requisitos do *software* com pequena probabilidade de mudanças.
e) Presença de um processo organizacional que exige a elaboração de vários documentos específicos para cada projeto.

O cliente não é uma entidade externa, mas, sim, um membro integrado à equipe. Sua disponibilidade contínua é essencial, estando sempre pronto para esclarecer dúvidas dos desenvolvedores. Isso facilita os questionamentos e fortalece bastante a comunicação de toda a equipe.
Letra c.

---

012. (CESGRANRIO/REFAP SA/ANALISTA DE SISTEMAS JÚNIOR/2007) NÃO é uma característica da *Extreme Programming* (XP):
a) simplicidade.
b) agilidade.
c) desenvolvimento orientado a testes.
d) programação em par.
e) documentação extensa e abundante em artefatos.

O *Extreme Programming* (XP) geralmente não enfatiza a produção extensiva de documentação detalhada. O foco do XP está na entrega contínua de *software* funcional e na resposta ágil às mudanças nos requisitos do cliente. Isso não significa que o XP não valorize a documentação, mas, sim, que prefere formas mais leves e práticas, dando prioridade à comunicação cara a cara e à colaboração direta.
O XP encoraja a criação de documentação mínima, apenas o suficiente para atender às necessidades imediatas da equipe. Os princípios do XP priorizam o código fonte como a forma mais atualizada e precisa de documentação do sistema. Essa abordagem favorece a flexibilidade e a adaptabilidade, permitindo que a equipe XP responda rapidamente às mudanças nos requisitos e mantenha um ciclo de desenvolvimento ágil.
Letra e.

---

013. (CESPE/BNB/ESPECIALISTA TÉCNICO/ANALISTA DE SISTEMA/2018) Acerca dos métodos ágeis, julgue o próximo item.
Em XP, a técnica de *planning game* é utilizada pelo cliente para identificar as prioridades do que deve ser construído em um *software*, sem a participação dos desenvolvedores.

Jogo do Planejamento consiste em determinar o escopo do próximo incremento, combinando prioridades do negócio e a parte técnica. Logo, no XP o planejamento ocorre constantemente. Com um cliente presente de forma assídua no projeto, ele define o escopo para a próxima entrega. Dessa forma, define-se as tarefas, prioridades e o escopo da *release*.
A técnica de *planning game* é utilizada pelo cliente para identificar as prioridades do que deve ser construído em um *software*, SEMPRE com a participação da equipe de desenvolvimento de *software*.
Errado.

---

014. (CESPE/STJ/TÉCNICO JUDICIÁRIO/DESENVOLVIMENTO DE SISTEMAS/2018) Julgue o seguinte item, relativo à gestão ágil de projetos com XP (*extreme programming*).
A integração contínua descrita na XP pode fazer parte do processo de desenvolvimento de *software* que utiliza o *Scrum*, pois, à medida que as entregas sejam realizadas na *Scrum*, pode-se validá-las por meio de testes automatizados.

Questão muito boa. Integra o XP com *Scrum*. Os dois são participantes do manifesto ágil, logo, a integração contínua faz parte do processo de desenvolvimento do produto.
O código das funcionalidades implementadas pode ser integrado várias vezes ao longo do desenvolvimento do *software*. Lembrando que toda a equipe é responsável pela qualidade do produto.
Certo.

---

015. (CESGRANRIO/UNIRIO/2019) Uma das principais práticas de XP (*Extreme Programming*) é o *Iteration Planning Game*. Entre as atividades realizadas em uma sessão de *Iteration Planning*, está a:
a) definição, pelos programadores, de quais *story cards* serão implementados em uma iteração.
b) estimação do esforço que será necessário para implementar cada *story card*.
c) estimação da data de entrega de um release baseado na estimativa de esforço de cada *story card*.
d) estimação, feita por cada programador, do tempo que será necessário para realizar cada tarefa sob sua responsabilidade.
e) designação, por parte do *coach*, dos programadores que irão realizar as tarefas contidas na lista de tarefas.

a) Errada. Esta definição é realizada pelo cliente.
b) Errada. Neste momento planejamento do tempo.
c) Errada. Não há do que se falar de estimativa do esforço.
e) Errada. O papel do *coach* não existe no XP.
O *planning game* consiste em uma prática para elaborar a estratégia das interações, que é a forma como se trabalha o "cronograma" de um projeto do XP, no qual se define um tempo padrão para as interações e especifica-se quais e quantas estórias podem ser implementadas em uma interação. Logo, o *planning game* é uma reunião de planejamento para discutir o que será feito.
Letra d.

---

016. (CESPE/ABIN/OFICIAL TÉCNICO DE INTELIGÊNCIA/ÁREA 9/2018) A respeito da metodologia XP (*Extreme Programming*), julgue o item que segue.
Para apoiar a equipe de desenvolvimento, é uma prática o uso do cliente *on-site* em tempo integral.

Uma das práticas do XP é O cliente sempre disponível. Ele deve estar sempre disponível para colaborar em dúvidas, alterações, e prioridades em um escopo, ou seja, dando um dinamismo ativo ao projeto
Certo.

---

017. (FUNDATEC/IPE SAÚDE/ANALISTA DE GESTÃO EM SAÚDE/ENGENHARIA DA COMPUTAÇÃO/2022) O processo de desenvolvimento de *software* especificado pela Programação Extrema (*eXtreme Programming* – XP) começa com uma fase de planejamento, na qual são levantados e descritos requisitos para o *software* na forma de _____.
O projeto e desenvolvimento dos requisitos busca focar nas necessidades imediatas. Necessidades de melhoria no projeto são realizadas através de processos de _____.
Além disso, se recomenda que a atividade de codificação ocorra em _____ e seja guiada por _____.
Assinale a alternativa que preenche, correta e respectivamente, as lacunas do trecho acima.
a) histórias de usuários – refatoração – quartetos – testes
b) histórias de usuários – testes – pares – casos de uso
c) histórias de usuários – refatoração – pares – testes
d) modelos de domínio – refatoração – pares – testes
e) modelos de domínio – testes – quartetos – casos de uso

No *eXtreme Programming*, todos os requisitos são expressos como cenários (também chamados histórias do usuário), que são implementados diretamente como uma série de tarefas.
Refatoração – uma técnica de construção que também é um método para otimização de projetos. Refatoração é o processo de alteração de um sistema de *software* de tal forma que não se altere o comportamento externo do código, mas se aprimore a estrutura interna.
Os desenvolvedores trabalham em pares, um verificando o trabalho do outro.
No XP, primeiro se escreve o teste depois a implementação. Testes são EXTREMAMENTE importantes para o modelo.
Letra c.

---

018. (FAURGS/TJ–RS/2018) Considere as seguintes afirmações sobre princípios ou práticas da XP (*Extreme Programming*).
I – Um representante do usuário final do sistema (cliente) deve estar disponível todo o tempo à equipe de XP. Em um processo de *Extreme Programming*, o cliente é um membro da equipe de desenvolvimento e é responsável por levar ao grupo os requisitos de sistema para implementação.
II – Todos os desenvolvedores devem refatorar o código continuamente, assim que encontrarem oportunidades de melhorias de código.
III – Os desenvolvedores trabalham em todas as áreas do sistema, de modo que não se desenvolvam ilhas de expertise. Todos os desenvolvedores têm responsabilidade em relação ao código; qualquer um pode mudar qualquer coisa.
Quais estão corretas?
a) Apenas I.
b) Apenas I e II.
c) Apenas I e III.
d) Apenas II e III.
e) I, II e III.

I – Ter o cliente sempre disponível é uma prática essencial do XP. Ele deve estar acessível para esclarecer dúvidas, discutir alterações e definir prioridades no escopo, promovendo uma dinâmica ativa no projeto.
II – Como discutimos anteriormente, a refatoração é uma técnica construtiva que também serve como método de otimização de projetos. Envolve a alteração do sistema de *software* de maneira que não impacte o comportamento externo do código, mas aprimore sua estrutura interna.

III – A equipe de desenvolvimento no XP é composta por profissionais engajados e multidisciplinares, colaborando de maneira integrada. Possuem uma variedade de habilidades essenciais para o sucesso do projeto.

Letra e.

---

019. (CESPE/ABIN/OFICIAL TÉCNICO DE INTELIGÊNCIA/ÁREA 9/2018) A respeito da metodologia XP (*Extreme Programming*), julgue o item que se segue.

O ritmo ágil de desenvolvimento de *softwares* é uma prática usada para favorecer a entrega das *releases* quando grandes volumes de horas extras são tolerados.

As equipes em projetos XP têm uma carga horária semanal de 40 horas. A fadiga e o descontentamento causados por horas extras podem levar a uma diminuição na qualidade do código, prejudicando o processo de desenvolvimento. Portanto, a realização de grandes quantidades de horas extras não é considerada aceitável.

Errado.

---

020. (INSTITUTO AOCP/ADAF–AM/2018) Na metodologia ágil *Extreme Programming* (XP), a propriedade do código é coletiva, dessa forma, todos compartilham o mesmo orgulho e as mesmas críticas. Considerando o exposto, assinale a alternativa que apresenta uma das regras da codificação em XP.

a) No *Overtime*.

b) Eliminar gargalos de *hardware* no início.

c) O usuário não deve participar do planejamento das interfaces.

d) No *Outsourcing*.

e) No *Sprint*.

Como discutimos, o XP adota a prática do ritmo sustentável, o que significa que a realização de grandes quantidades de horas extras não é encorajada. O *Overtime* está relacionado às horas extras, mas não é uma prática adotada por esse modelo. Vale observar que os demais elementos não têm conexão com o XP.

Letra a.

---

021. (CESPE/STM/ANALISTA JUDICIÁRIO/ANÁLISE DE SISTEMAS/2018) Julgue o próximo item, referente à metodologia de desenvolvimento de *software*.

Na XP (*Extreme Programming*), programadores trabalham em pares, e requisitos são expressos como cenários, denominados histórias de usuários, os quais são implementados como uma série de tarefas.

Os requisitos do *software* a ser desenvolvido são apresentados como cenários (ou histórias do usuário), os quais são detalhados como uma sequência de tarefas. Esses cenários são descritos e classificados em ordem de prioridade pelo cliente.
Os desenvolvedores colaboram na codificação do *software* em duplas, garantindo que cada um revise o trabalho do outro para assegurar a qualidade do projeto.
Certo.

---

022. (CESPE/TRE–BA/TÉCNICO JUDICIÁRIO/PROGRAMAÇÃO DE SISTEMAS/2017) Considerando uma situação hipotética com o uso da XP (*eXtreme Programming*) concomitante com *Scrum* em um projeto de desenvolvimento de *software* em uma organização, julgue os seguintes itens.
I – É viável a utilização do TDD (*Test Driven Development*) na fase de *sprint*, de modo que se escreva o teste automático antes da codificação.
II – O princípio da integração contínua da XP deve ser utilizado especificamente na retrospectiva da *sprint* com vistas a integrar a equipe *Scrum*.
III – Integrantes da equipe *Scrum* podem realizar a programação do código em pares, o que proporciona, entre outras vantagens, o nivelamento de conhecimento da equipe.
IV – O conceito de requisito "pronto" continuaria válido, contudo, inviabilizaria o *refactoring*, pois é proibitivo inserir o mesmo item (requisito) em várias *sprints*.
Estão certos apenas os itens:
Alternativas
a) I e II.
b) I e III.
c) II e IV.
d) I, III e IV.
e) II, III e IV.

Os dois modelos ágeis para desenvolvimento podem compartilhar inúmeras características, não tornando inviável nem invalidando seus respectivos usos.
I – Como uma prática do XP, os testes devem ser escritos antes da codificação propriamente dita. Eles são executados a partir de testes unitários do código produzido e são preparados com base nos critérios de aceitação definidos previamente pelo cliente.

II – Não faz sentido afirmar que a integração contínua deve ser aplicada somente na retrospectiva da *Sprint*. Essa prática é contínua. O código das funcionalidades implementadas pode ser integrado várias vezes ao dia.
III – Embora seja uma prática do XP, isso não significa que não possa ser adotada por outros métodos e modelos, como o *Scrum*.
O código é produzido por um par de programadores que têm funções distintas, trabalhando lado a lado em um mesmo computador.
IV – Considerada uma das práticas do XP, a refatoração consiste em aplicar uma série de passos para aprimorar o design do código existente, tornando-o mais simples e melhor estruturado, sem alterar sua funcionalidade. Deve ser aplicada constantemente no projeto e pode ser realizada por qualquer um dos desenvolvedores, pois o código é de propriedade coletiva.
Letra b.

---

023. (CESPE/TRE–TO/TÉCNICO JUDICIÁRIO/PROGRAMAÇÃO DE SISTEMAS/2017) Em projetos de desenvolvimento de *software*, a *extreme programming* (XP) é um método ágil que usa a prática de:
a) projetos com planejamento completo sem incrementos.
b) grandes *releases*.
c) grande quantidade de horas extras.
d) trabalho em pares de desenvolvedores.
e) integrações após a entrega do *software* completo.

Conforme vimos, uma das práticas do XP é a programação em pares.
a) Errada. Uma das práticas do XP é a sua forma de desenvolvimento incremental constante. Tão logo o trabalho em uma tarefa é concluído, esta é integrada ao sistema.
b) Errada. Seria o contrário: pequenas *releases*. Em regra, o conjunto mínimo de funcionalidades que agregam valor ao usuário devem ser adicionadas à próxima *release*.
c) Errada. Seria o contrário da afirmação. Horas extras não são bem-vindas no XP. Devem ser evitadas ao máximo. O modelo preconiza o desenvolvimento sustentável com semanas com duração de 40 horas de trabalho.
d) Certa.
e) Errada. O código das funcionalidades implementadas pode ser integrado várias vezes ao dia. Compreende na integração contínua.
Letra d.

---

024. (CESPE/TRE–PE/TÉCNICO JUDICIÁRIO/PROGRAMAÇÃO DE SISTEMAS/2016) Com relação às metodologias ágeis XP, *Scrum* e UP e à metodologia RUP, assinale a opção correta.
a) A padronização da arquitetura de código-fonte é o foco principal da metodologia RUP.
b) As metodologias ágeis são focadas no produto, sendo caracterizadas pela ausência de modelo de dados, de diagramas de classes e de documentação de código-fonte.
c) A metodologia RUP, fundamentada em um modelo preditivo com foco no planejamento futuro, realiza entregas de *software* executável na mesma dinâmica das metodologias ágeis.
d) Os métodos ágeis podem ser classificados como métodos adaptativos à necessidade de desenvolvimento de *software*, pois mudam conforme a necessidade do projeto, diferentemente das metodologias tradicionais de desenvolvimento de *software*.
e) Uma metodologia clássica de engenharia de *software* deve ser aplicada em projetos com requisitos do sistema instáveis.

a) Errada. Ainda não exploramos o RUP, mas podemos antecipar que ele não foi concebido especificamente para resolver a padronização da arquitetura do código-fonte.
b) Errada. Metodologias ágeis concentram-se no produto, mas não proíbem o uso de modelos e documentação. Vale ressaltar que um código funcional tem mais valor do que uma documentação extensa e desnecessária.
c) Errada. O RUP é um modelo adaptativo.
e) Errada. Os modelos clássicos assumem uma realidade estática e bem conhecida dos requisitos desde as fases iniciais. Portanto, requisitos instáveis podem ocasionar grandes problemas para esses modelos.
Letra d.

---

025. (CESPE/MPU/ANALISTA DE INFORMÁTICA/DESENVOLVIMENTO DE SISTEMAS/2010) *Extreme programming* (XP) é embasado em requisitos conhecidos, definidos de antemão, que não sofram muitas alterações, devendo ser usado por equipes de pequeno porte, formadas por representantes de todos os *Stakeholders*.

Requisitos bem conhecidos são melhor aplicados para metodologias tradicionais, como o modelo cascata. O XP (e demais métodos ágeis) estão preparados para receber mudanças frequentes em seus requisitos
A equipe de desenvolvimento realmente quase sempre é pequena, mas existir representantes de todas as partes interessadas não faz sentido e iria onerar bastante o cliente. Precisamos de participação ativa do cliente durante o desenvolvimento, mas não se faz necessária a presença de todos.
Errado.

---

026. (CESPE/STF/ANALISTA JUDICIÁRIO/ANÁLISE DE SISTEMAS DE INFORMAÇÃO/2013) XP (*Extreme Programming*) é uma metodologia ágil voltada para equipes pequenas e médias que desenvolvam *software* baseado em requisitos vagos e se caracteriza por possibilitar modificações rápidas.

XP é uma metodologia para desenvolvimento ágil que preconiza equipes pequenas e médias para implementarem requisitos vagos e que podem estar em constantes mudanças.
Lembre-se, futuro servidor: equipe pequenas/médias e mudanças nos requisitos são bem-vindas.
Certo.

---

027. (IF–RS/2018) Sobre as práticas encontradas na metodologia ágil de desenvolvimento de software, conhecida por Programação Extrema (XP Programming), de acordo com Dooley (2017) no livro Software Development, Design and Coding, classifique cada uma das afirmativas abaixo como verdadeira (V) ou falsa (F) e assinale a alternativa que apresenta a sequência CORRETA, de cima para baixo:

( ) Participação intensa do representante do cliente no desenvolvimento do projeto.
( ) Testes são realizados continuamente. Quando todos os testes forem aprovados, o módulo foi concluído.
( ) Programação em par: enquanto um escreve o código, o outro monitora falhas, realiza testes, faz sugestões e planeja próximas ações.
( ) Lançamentos frequentes de novas versões.

a) F – V – V – F
b) V – F – F – V
c) V – V – F – V
d) V – V – V – V
e) F – F – V – V

Esta é uma excelente questão para reforçarmos alguns conceitos do XP, futuro servidor.
Cliente no local: É fundamental incluir na equipe uma pessoa representando o cliente, que utilizará o sistema. Essa pessoa trabalhará em colaboração com os outros membros da equipe, respondendo perguntas e esclarecendo dúvidas.
Testes constantes: No XP, os testes são realizados antes da programação. Existem dois tipos principais de teste: teste de unidade e teste funcional. Esses testes determinam se

uma versão pode ou não ser liberada. Antes da implementação, é crucial garantir que os testes estejam em conformidade.
Programação em pares: Todo o código gerado no XP é elaborado por um par de programadores, promovendo colaboração para assegurar a qualidade do código.
Releases pequenos: Cada lançamento deve ser o menor possível, incluindo os requisitos mais críticos para o negócio. Essa abordagem possibilita lançamentos frequentes, gerando mais *feedback* para clientes e programadores, o que facilita a aprendizagem e a correção de falhas no sistema.
Letra d.

---

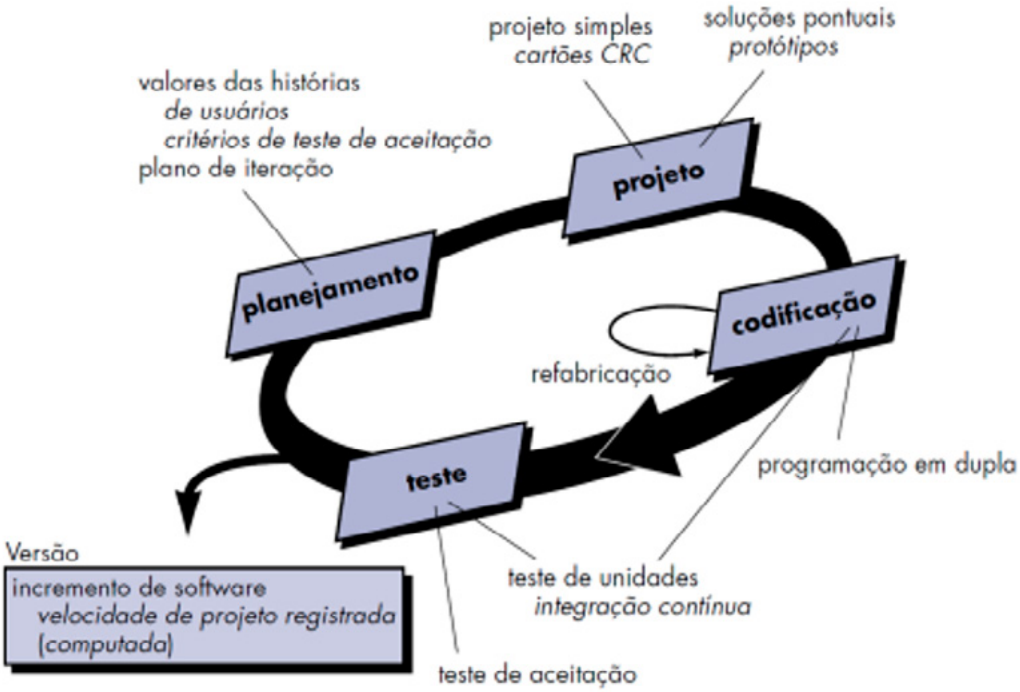
028. (INSTITUTO AOCP/UFPB/2019) Um dos principais métodos ágeis de desenvolvimento de *software* foi concebido para impulsionar práticas reconhecidas como boas, por exemplo, o desenvolvimento iterativo a nível extremo, em que novas versões de um determinado sistema podem ser implementadas, integradas e, até mesmo, testadas em um único dia por programadores diferentes. Essa é uma das características de qual método de desenvolvimento ágil de *software*?
a) *Scrum*.
b) *Adaptative Software Development*.
c) *Extreme Programming*.
d) *Pramatic Programming*.
e) *Test Driven Development*.

No XP, o código das funcionalidades implementadas pode ser integrado várias vezes ao dia, cada vez que uma nova tarefa for concluída. Na integração, as funcionalidades só podem ser integradas se não houver erros, caso contrário os erros devem ser corrigidos.
Letra c.

---

029. (FCM/PREFEITURA DE CARANAÍBA–MG/2019) De acordo com Pressman e Maxim (2016), a Programação Extrema (*Extreme Programming* – XP) é uma abordagem amplamente utilizada do desenvolvimento ágil de *software* que consiste das atividades:
a) Planejamento (*Planning*) / Projeto (*Designing*) / Codificação (*Coding*) / Teste (*Test*).
b) Colaboração (*Collaboration*) / Projeto (*Designing*) / Codificação (*Coding*) / Teste (*Test*).
c) Colaboração (*Collaboration*) / Projeto (*Designing*) / Codificação (*Coding*) / Teste (*Test*) / Adaptação (*Adaptation*).
d) Colaboração (*Collaboration*) / Projeto (*Designing*) / Codificação (*Coding*) / Teste (Test) / Adaptação (*Adaptation*) / Melhoria (*Improvement*)

Conforme estudamos, as fases / etapas / atividades são planejamento, projeto, codificação e testes.
A imagem a seguir, retirada do autor Presmann, pode auxiliar no entendimento (grave estas fases!).



Letra a.

---

030. (VUNESP/CÂMARA DE PIRACICABA–SP/2019) Um dos processos ágeis de desenvolvimento de software é a programação extrema (*extreme programming* – XP), cuja fase ou atividade inicial é composta pela descrição dos cenários (características e funcionalidades) requisitadas para o *software* a ser desenvolvido. Essa atividade recebe a denominação de:
a) métodos práticos.
b) histórias de usuário.
c) estruturas de apoio.
d) classes de projeto.
e) artefatos de usuário

O planejamento de um *release* e das iterações são feitos com base nas histórias de usuário (baseado em cenários) e conta com a colaboração de toda a equipe de desenvolvimento, incluindo (claro) o cliente.

A primeira etapa para se iniciar uma nova *release* é a escolha das histórias com base na priorização do cliente. Lembre-se de que o cliente faz parte da equipe e deve estar sempre presente.

Letra b.

---

031. (CESPE/TJ–AM/2019) No XP (*Extreme Programming*), o valor de uma história de usuário é atribuído pelos membros da equipe e é medido em termos de semanas estimadas para o desenvolvimento.

O cliente é encarregado de criar cada história, registrando-a em uma ficha. Nesse documento, cada história é avaliada com base em seu valor, determinando sua prioridade conforme a contribuição esperada para o produto. Os desenvolvedores, por sua vez, atribuem a cada história um custo estimado, considerando o tempo em semanas necessário para a implementação. Em resumo, é responsabilidade do cliente atribuir um valor a cada história, influenciando a ordem de prioridade no *backlog*.

Errado.

---

032. (QUADRIX/COBRA TECNOLOGIA SA (BB)/ANALISTA DE OPERAÇÕES/NEGÓCIOS/2014) O modelo ágil *Extreme Programming* (XP) segue uma série de práticas que dizem respeito ao relacionamento com o cliente, a gerência do projeto, a programação e aos testes. NAO é uma dessas práticas:

a) jogo do planejamento.
b) programação em pares.
c) desenvolvimento orientado a testes.
d) estabelecer e seguir padrões de codificação.
e) listar requisitos no *product backlog*.

Product Backlog é um conceito aplicado ao *Scrum*. O *Backlog* do Produto é uma lista priorizada de itens sobre os quais o Time de Desenvolvimento trabalhará no decorrer do projeto. Trata-se da lista de funcionalidades e requisitos que deverão ser entregues ao cliente ao longo das *Sprints*.

Letra e.

---

033. (CESPE/ANATEL/ANALISTA ADMINISTRATIVO/TECNOLOGIA DA INFORMAÇÃO E COMUNICAÇÃO/2014) Acerca dos processos de desenvolvimento de *software*, julgue o item subsequente.

A etapa de planejamento do *Extreme Programming* (XP) inicia-se com a escrita de *User Stories* (história do usuário). Por meio dessa ferramenta, aqueles que conhecem a técnica de construção de uma solução poderão guiar quem necessita dessa solução no exercício de descrevê-la de forma simples e concisa.

As Histórias de Usuários desempenham um papel fundamental no desenvolvimento para equipes que adotam XP. O uso de metáforas orienta o processo, simplificando e compartilhando as histórias de usuário com toda a equipe.

Essas histórias possibilitam que aqueles familiarizados com a construção de *software*, como ferramentas e linguagens de programação, guiem eficientemente aqueles que necessitam da solução. A abordagem visa a elaboração de histórias de usuário de maneira simples, consistente e concisa, conforme mencionado na questão.

Certo.

---

034. (FCC/DPE–AM/2018) Considere a definição de algumas práticas da *eXtreme Programming* – XP.

I – Todo o código desenvolvido pelo time é incorporado em um repositório comum várias vezes ao dia. Isso garante que qualquer problema de integração ao longo do projeto possa ser notado e corrigido rapidamente. II. Qualquer programador do time pode alterar qualquer seção do código, se necessário. Por mais que esta prática pareça perigosa, ela aumenta a velocidade do desenvolvimento e problemas em potencial podem ser detectados pelos testes de unidade.

III – Traz a ideia de que qualquer pessoa do time seja capaz de verificar o código sendo desenvolvido em alto nível e ter uma compreensão clara de qual funcionalidade do sistema está sendo trabalhada.

IV – Permite aplicar melhorias ao código sem mudar sua funcionalidade, visando sua simplificação. Se o cliente deseja alterar alguma coisa no produto final, o time pode fazer os ajustes rapidamente, e esta prática contribui para alcançar este objetivo.

As práticas de I a IV são, correta e respectivamente,

a) *pair programming – test-driven development – system metaphor – continuous integration*.

b) *planning game – pair programming – system simplicity – continuous integration*.

c) *planning game – test-driven development – system simplicity – refactoring*.

d) *continuous integration – pair programming – feedback – planning game*.

e) *continuous integration – collective code ownership – system metaphor – refactoring*.

I – Diz respeito à prática de integração contínua, que consiste na atualização frequente e integração das versões do sistema várias vezes ao dia, sempre que uma nova tarefa é concluída.
II – Código de propriedade coletiva refere-se à ausência de um proprietário único para o código. Pelo contrário, a propriedade do código pertence a toda a equipe envolvida no projeto, permitindo que qualquer membro faça modificações em qualquer parte do código a qualquer momento.
III – A utilização de metáforas destina-se a orientar o desenvolvimento por meio de histórias de usuário simplificadas e compartilhadas com todos os membros da equipe.
IV – Refatoração envolve reestruturar o sistema sem alterar seu comportamento, eliminando duplicações sempre que possível, melhorando e simplificando o que já existe e tornando-o mais flexível, conciso e seguro.
Letra e.

---

035. (INSTITUTO AOCP/PRODEB/2018) O método de desenvolvimento ágil denominado de XP (*Extreme Programming*) tem sua estrutura baseada em algumas prerrogativas, dentre as quais, é correto citar como princípios do XP:
a) Princípio da Legalidade, Princípio da Agilidade e Princípio do *Feedback*. b) Princípio da Coragem, Princípio da Agilidade e Princípio da Simplicidade.
c) Princípio da Comunicação, Princípio da Simplicidade, Princípio do *Feedback* e Princípio da Coragem.
d) Princípio da Agilidade, Princípio da Qualidade, Princípio do *Feedback* e Princípio da Coragem.
e) Princípio da Simplicidade, Princípio do Desenvolvimento e Princípio de Governança.

O macete para lembrar os valores (princípios na questão) é: **Cor Sim Com Fe Re**.
Coragem: Fomentar um ambiente encorajador onde a equipe possa implementar suas ideias sem receios.
Simplicidade: Priorizar a simplificação e evitar complexidades, otimizando o processo de codificação.
Comunicação: Assegurar clareza e precisão na comunicação entre a equipe e o cliente. O XP favorece projetos simples, metáforas comuns, colaboração entre usuários, programadores e outros *Stakeholders*, comunicação verbal frequente e *feedback*.
Feedback: Buscar retornos constantes em relação ao time e ao produto. Os testes unitários e de integração auxiliam nos *feedbacks*, juntamente com a presença constante do cliente.
Respeito: Fomentar um ambiente respeitoso entre os membros da equipe e suas decisões. Cada membro deve respeitar seu próprio trabalho, sempre buscando oferecer alta qualidade.
Letra c.

---

036. (INSTITUTO AOCP/UFOB/2018) Uma das práticas do *Extreme Programming* é o uso do código coletivo, na qual todos os desenvolvedores têm acesso ao código.

O código não possui um dono ou responsável exclusivo. Toda a equipe pode trabalhar no código como um todo, sem necessidade de solicitar permissão. Lembrando que padrões para auxiliar o desenvolvimento são bem-vindos no XP, aumentando assim a qualidade a eficiência de produção.

Certo.

---

037. (INSTITUTO AOCP/PRODEB/2018) *Extreme Programming* (XP) é um método de desenvolvimento ágil amplamente utilizado pelas *software houses*. Com base neste método, qual alternativa a seguir possui uma prática que NÃO faz parte do XP?

a) *Small Releases* (Pequenas Entregas).
b) *Pair Programming* (Programação em Pares).
c) *Sprint Review* (Reunião de revisão da *Sprint*).
d) *Sustainable Pace* (Ritmo Saudável).
e) *Collective Owership* (Posse Coletiva).

Pequenas entregas: pequenos incrementos do *software* são constantemente entregues de forma funcional.
Programação em pares: dois desenvolvedores codificam juntos no mesmo computador.
Ritmo sustentável: semanas de trabalho de 40 horas de trabalho, evitando-se horas extras.
Propriedade coletiva: o código é de propriedade da equipe.
A alternativa "c" – *Sprint Review* – é um conceito do *Scrum*: A *Sprint Review* é um evento dentro do *Scrum*, onde o *Product Owner*, *Scrum Master*, *Development Team* e *Stakeholders* se reúnem para receber *feedback* do Incremento produzido no *Sprint* corrente.

Letra c.

---

038. (FCC/TCE–AL/2008) Originalmente, o único produto da atividade de Projeto que é realizado como parte do processo XP (*Extreme Programming*):

a) é a definição do caso de uso de contexto.
b) são os cartões CRC.
c) são os diagramas de objetos.
d) são os diagramas de sequência.
e) é a codificação, feita em pares.

A metodologia XP praticamente não possui artefatos gerados em seu desenvolvimento. Entretanto, os cartões CRC são gerados e são considerados artefatos. Esses cartões permitem identificar e organizar as classes orientadas a objeto para o incremento que está sendo desenvolvido pela equipe.

Lembre-se: cartão CRC é considerado o único artefato do XP!

Letra b.

---

039. (INSTITUTO AOCP/EBSERH/2017) O *Extreme Programming* (XP) surgiu em 1999, a partir de uma publicação sobre o assunto, mas suas bases se conectam a princípios da década de 80 e ao manifesto ágil. O XP é baseado em 4 atividades de arcabouços. Assinale a alternativa que contém 3 desses arcabouços:

a) Levantamento de requisitos, análise de negócio, planejamento e documentação.

b) Levantamento de requisitos, planejamento, documentação e implantação.

c) Levantamento de requisitos, documentação, codificação e teste.

d) Planejamento, projeto, documentação e implantação.

e) Governança, planejamento, codificação e teste.

As fases, etapas e atividades do XP são:

- **Planejamento (Jogo do Planejamento)**: Inicia-se com a escuta das partes interessadas e a elaboração das "histórias de usuários".
- **Projeto**: Deve seguir o princípio da simplicidade, priorizando um projeto simples em detrimento de uma representação complexa. Estimula-se o uso de cartões CRC (classe-responsabilidade-colaborador).
- **Codificação**: Envolve o desenvolvimento de uma série de testes de unidade que serão aplicados a cada uma das histórias de usuário. Somente após a implementação dos testes, os desenvolvedores codificam as histórias, seguindo o princípio da programação em pares.
- **Teste**: São criados e executados os testes de aceitação, elaborados com base nas histórias de usuário. Além disso, os demais testes da aplicação são realizados para prevenir possíveis falhas futuras, incluindo os testes de regressão. Os testes de integração e validação do sistema podem ocorrer diariamente.

Letra d.

---

040. (FCC/TRE–SE/2007) Na XP (*eXtreme Programming*):

a) deve-se usar o modelo em cascata para o desenvolvimento do *software*.

b) os programadores desenvolvem o *software* criando primeiramente os testes.

c) deve ser evitada a comunicação pessoal entre clientes e desenvolvedores, sempre dando preferência a outros meios de comunicação mais formais.

d) os programadores desenvolvem o *software* fazendo todos os testes possíveis no término do desenvolvimento.

e) deve-se projetar todas as funções possíveis com a máxima previsão do que ocorrerá no futuro, antes do desenvolvimento do *software*, a fim de evitar alterações desnecessárias.

a) Errada. Não faz sentido ter que utilizar o modelo tradicional cascata no XP.

b) Certa. Programadores XP escrevem testes primeiro, desenvolvem código e rodam testes para validar o código escrito.

c) Errada. Um dos valores do XP é a comunicação. Ela deve ser estimulada constantemente entre a equipe e cliente.

d) Errada. Os testes são feitos antes da implementação (*Test First Design* – Primeiro os testes) e são revisados a todo tempo.

e) Errada. O XP é receptivo a mudanças.

Letra b.

---

041. (CESPE/PRODEST/2008) Projetar detalhadamente todo o *software* antes de iniciar a sua implementação é uma prática recomendada pelo XP. O *software* deve ser projetado para atender tanto aos requisitos atuais quanto aos potenciais requisitos futuros. Para atingir esse objetivo, são analisados os possíveis cenários de evolução futura e são empregados padrões de projeto para facilitar a manutenção.

Mudanças devem ser bem-vindas e ocorrerão de acordo com o melhor entendimento do projeto. Responder a mudanças é mais valorizado do que seguir um plano específico.

Errado.

---

042. (CESPE/ABIN/2010) Na *Extreme Programming*, os requisitos são expressos como cenários e implementados diretamente como uma série de tarefas. O representante do cliente faz parte do desenvolvimento e é responsável pela definição de testes de aceitação do sistema.

A banca forneceu definições apropriadas.

Os requisitos do produto são formulados como cenários (histórias do usuário), sendo implementados diretamente como uma série de tarefas pela equipe de desenvolvimento. É essencial ter um cliente real no local para responder perguntas, priorizar histórias de usuários e colaborar com testes de aceitação.

Certo.

---

043. (CESPE/PRODEST/2008) Constituem práticas recomendadas pelo XP a colocação rápida de uma versão simples em produção, a liberação das novas versões em curtos intervalos de tempo, a programação em duplas, a refatoração (refactor) dos códigos produzidos, a adoção de padrões para a codificação; a integração e o teste contínuos de códigos; a limitação em 40 horas da carga de trabalho semanal.

Entrega de pequenas *releases*, programação em pares, refatoração do código, projetos para codificação, integração contínua e ritmo sustentável de trabalho (40 horas semanais) são práticas do XP.

Certo.

---

044. (SERPRO/ANALISTA/DESENVOLVIMENTO DE SISTEMAS/2010) Cada sequência de etapas de um ciclo de vida aderente ao modelo em cascata é equivalente a uma iteração em um processo embasado no XP.

O examinador tentou confundir conceitos. O modelo cascata prioriza as fases do desenvolvimento em requisitos, análise, projeto, implementação, testes e implantação do *software*. Cada iteração no XP passa pelas fases de planejamento, projeto, codificação e testes.

Errado.

---

045. (SERPRO/ANALISTA/DESENVOLVIMENTO DE SISTEMAS/2010) Metodologias ágeis, como a XP, enfatizam a documentação de software no próprio código, que deve ser escrito por meio de ferramenta CASE voltada ao desenvolvimento rápido de aplicações (RAD Tools).

Métodos ágeis, como o XP, simplificam o uso de documentação, dando prioridade ao código-fonte como o principal meio de documentar um *software*. O modelo não impõe restrições quanto ao uso de ferramentas CASE; no entanto, se a equipe optar por utilizá-las, não há problema.

Errado.

---

046. (TRE9–BA/ANALISTA JUDICIÁRIO/ANÁLISE DE SISTEMAS/2009) A metodologia XP prevê valores e princípios básicos para serem considerados durante o desenvolvimento de *software*. *Feedback*, coragem e respeito são exemplos de valores; mudanças incrementais, abraçar mudanças e trabalho de qualidade são exemplos de princípios básicos.

Valores do XP (**Cor Sim Com Fe Re**):
Comunicação: A equipe deve se comunicar de maneira clara.
Simplicidade: O projeto de *software* deve buscar a simplicidade.
Coragem: A equipe deve ter coragem para alterar o código visando a melhoria do *software* em desenvolvimento.
Feedback: Cada teste deve fornecer *feedback* para contribuir com a evolução do projeto.
Respeito: Uma premissa fundamental do XP é que todos devem se preocupar com seu trabalho. Nenhuma quantidade de excelência técnica pode salvar um projeto se não houver cuidado e respeito.
Abraçar as mudanças, desenvolvimento incremental e produção de um produto de qualidade são parte dos princípios do XP.
Certo.

---

047. (TCU/AUDITOR FEDERAL DE CONTROLE EXTERNO/TECNOLOGIA DA INFORMAÇÃO/2010) O processo XP (*extreme programming*) envolve a realização das atividades de planejamento, de projeto, de codificação e de teste.

Aa quatro atividades/fases do XP são: planejamento, projeto, codificação e testes.
Certo.

---

048. (TCU/AUDITOR FEDERAL DE CONTROLE EXTERNO/TECNOLOGIA DA INFORMAÇÃO/2010) A atividade de planejamento XP inclui a criação das denominadas histórias de usuário, nas quais devem ser descritas as características e as funcionalidades requeridas para o *software* em desenvolvimento.

A fase/atividade de planejamento do XP se inicia com a criação das histórias de usuário, contendo características das funcionalidades do *software*. O responsável por descrever as histórias é o usuário.
Certo.

---

049. (TCU/AUDITOR FEDERAL DE CONTROLE EXTERNO/TECNOLOGIA DA INFORMAÇÃO/2010) A atividade de projeto é uma desvantagem do processo XP, pelo fato de requerer uma quantidade de produtos de trabalho considerada excessiva pela comunidade de desenvolvimento de *software*.

A atividade do XP denominada 'projeto' trabalha com base no conceito da simplicidade. Isso significa desenvolver um *software* funcional da forma mais simples possível.
Errado.

---

050. (ANTAQ/ANALISTA ADMINISTRATIVO/INFORMÁTICA/2009) O *extreme programming* (XP) constitui método ágil de desenvolvimento de *software*. Uma das práticas que se enquadram nos princípios dos métodos ágeis é a programação em pares, que promove o compartilhamento da autoria do código do sistema. Além dessa vantagem, a programação em pares atua como processo informal de revisão porque cada linha de código é vista por pelo menos duas pessoas.

O examinador colocou perfeitamente alguns conceitos do XP. Enquanto uma pessoa trabalha em uma máquina, outra a acompanha para dar sugestões ou fazer críticas. Assim, irão colaborar com o aprendizado, a comunicação, a produtividade e a qualidade das entregas.
Certo.

---

051. (IBFC/EBSERH/2017) Dentro das práticas do XP (*eXtreme Programming*) existe uma fundamental que é o Jogo de Planejamento (Planning Game). Para serem realizadas adequadamente essas reuniões com os usuários, deve(m) ter sido feito(s) antecipadamente:
a) o *Sustainable Pace*
b) as *Small Releases*
c) os *Customer Tests*
d) as *User stories*
e) um *Simple Design*

As histórias de usuários devem estar descritas antes do jogo de planejamento. Nesta reunião, serão discutidos detalhes da história após suas devidas priorizações.
Letra d.

---

052. (CESPE/TJ–DFT/ANALISTA JUDICIÁRIO/ANALISTA DE SISTEMAS/2015) Na metodologia XP (*extreme programming*), em que todos os requisitos são expressos como cenários, deve-se aguardar, após a conclusão das tarefas, ciclos de cento e oitenta dias para a publicação de grandes *releases* do *software*.

O método de desenvolvimento XP não é engessado ao ponto de determinar prazo para liberação de versão do produto. A metodologia preconiza a liberação de versões ao usuário de forma rápida, contínua e integrada.

Errado.

---

053. (IBFC/TRE–PA/2020) Para aplicar valores e princípios do XP (*Extreme Programming*), durante os processos e práticas ágeis de desenvolvimento de *software*, se propõe uma série específica de práticas. Assinale a alternativa que apresenta algumas dessas "boas práticas" utilizadas tradicionalmente em projetos, usando XP.

a) *Reformation - Pair Programming - PlayStation Game*
b) *Refactoring - Pair Programming - Planning Game*
c) *Reformation - Pair Production - Planning Game*
d) *Refactoring - Pair Production - PlayStation Game*

A refatoração significa aprimorar o código sem alterar sua funcionalidade.

A programação em pares, conhecida como *pair programming*, é um estilo de programação no qual dois programadores trabalham lado a lado em um computador, colaborando continuamente no mesmo projeto, algoritmo, código e teste.

O jogo do planejamento – *Planning Game* – tem como objetivo determinar rapidamente o escopo da próxima *Release*, combinando as prioridades do negócio com as estimativas técnicas.

Letra b.

---

054. (IDECAN/UNIVASF/2019) *Extreme Programming* (XP), em sua essência, possui um conjunto de regras que devem ser seguidas em projetos ágeis que queiram utilizá-la em sua completude. Sobre as regras do XP, assinale a alternativa correta.

a) Apenas as operações mais críticas devem possuir testes unitários.
b) Todo o código-fonte de produção deve ser implementado em programação em pares.
c) A integração de código deve ser feita nos computadores dos desenvolvedores.
d) É importante estabelecer o papel de um XP Master, que será responsável pela implementação da metodologia.
e) A velocidade do projeto deve ser medida com o objetivo de informar ao cliente o tempo médio de correção de falhas.

a) Errada. Todo o código e funcionalidade devem possuir testes.
b) Certa. Conceito que faz parte das práticas do XP. Correto.
c) Errada. A integração de um sistema geralmente ocorre em um sistema robusto, minimizando as chances de erros e mitigando os riscos.
d) Errada. Não existe este papel no XP.
e) Errada. Como o cliente faz parte da equipe de desenvolvimento e está sempre presente, o cronograma do produto pode ser periodicamente reajustado. Com o cliente presente, é possível definir o escopo para a próxima *release*. Dessa forma, define-se as features, prioridades e o escopo da *release*.
Letra b.

---

055. (FCC/SANASA–CAMPINAS/2019) Em um projeto de *software* baseado na metodologia ágil XP, um Analista de TI deve:
a) consultar o cliente quando uma história exigir, por estimativa, menos do que 3 semanas de desenvolvimento, para que o cliente a complemente com mais tarefas.
b) ouvir o cliente, durante o levantamento de requisitos, para que este crie as histórias de usuários. Após essa importante etapa nenhuma história nova deve ser criada para não comprometer o cronograma do projeto.
c) evitar que o projeto caia na armadilha de seguir o princípio KISS de forma a estimular que o projeto de uma funcionalidade extra, que poderá ser necessária no futuro, faça parte do modelo do *software*.
d) realizar os testes de unidade de forma manual, evitando que sejam usadas baterias de testes automatizados, pois estes impedem a realização de testes de regressão.
e) estimular o uso de cartões CRC como um mecanismo eficaz para pensar o software em um contexto orientado a objetos.

a) Errada. Clientes devem estar presentes para escrevem testes de aceitação, definirem prioridades e histórias para as futuras iterações.
b) Errada. O XP é eficiente para projetos que precisam de flexibilidade para lidar com as mudanças (mudanças são bem-vindas).
c) Errada. Com o objetivo de maximizar o valor produzido a partir dos investimentos realizados em cada etapa, as equipes desenvolvem o necessário (funcional e com qualidade) buscando sempre a simplicidade.
d) Errada. Os testes são automatizados e contínuos.

e) Certa. O *Extreme Programming* preconiza o uso de cartões CRC (Classe – Responsabilidade – Colaborador) com o objetivo de desenvolver o *software* em um contexto orientado a objetos.
Letra e.

---

056. (INSTITUTO AOCP/MPE–BA/2014) O processo ágil XP possui doze práticas que são os princípios fundamentais do processo. A prática que encoraja a equipe inteira a trabalhar mais unida em busca de qualidade no código fazendo melhorias e refatoramentos em qualquer parte do código a qualquer tempo é conhecida como:
a) propriedade coletiva do código.
b) semana de quarenta horas.
c) programação em pares.
d) padrões de codificação.
e) integração contínua.

Todos os integrantes da equipe podem editar o código a qualquer hora. Códigos não são propriedades de uma pessoa. Esta prática diminui a especialização chamada ilha de conhecimento. O objetivo é que todos possam ter a liberdade de adaptar, modificar, melhorar e evoluir o código, além de prosperar o conhecimento do produto para todos.
Letra a.

---

057. (FGV/CÂMARA MUNICIPAL DO RECIFE–PE/2014) Uma das práticas do método ágil XP (*eXtreme Programming*) é:
a) documentação extensiva;
b) prototipação;
c) ciclos longos de desenvolvimento;
d) desenvolvimento orientado a testes (TDD);
e) utilização de todos os artefatos do RUP.

No XP, os testes são redigidos antes das funcionalidades, alternando as funções de testar e codificar. O TDD (Desenvolvimento Orientado por Testes) também exige que a equipe saiba o que deve ser verdadeiro no programa e o que não deve ser para garantir seu correto funcionamento. Vale ressaltar que é necessário buscar sempre a simplicidade sem comprometer a qualidade.
A prototipação pode ser utilizada no XP de forma PONTUAL para aprimorar a compreensão de uma determinada funcionalidade. No entanto, não é considerada uma prática fundamental do modelo.
Letra d.

---

058. (CESPE/ANTT/2013) São práticas ou princípios recomendados no modelo de desenvolvimento de software XP (*eXtreme Programming*) proposto por Kent Beck: programação em pares; semana de trabalho de 40 horas; refatoração sem piedade; desenvolvimento orientado a testes TDD (Test Driven Development); e desenvolvimento de metáforas arquiteturais.

Programação em pares, desenvolvimento sustentável com semanas de trabalho de 40 horas, desenvolvimento orientado a testes e metáforas arquiteturais estão bem claros que são práticas do XP.
Refatoração sem piedade: refatorar é umas das práticas do XP. Entretanto, as palavras 'sem piedade' podem nos gerar certa confusão. Neste caso, o examinador não levou a questão ao pé da letra – com um forte preciosismo.
Certo.

---

059. (CESPE/MPU/2013) XP é um método de desenvolvimento de *software* em que os requisitos são especificados em *user stories*; requisitos, arquitetura e *design* surgem durante o curso do projeto; e o desenvolvimento ocorre de maneira incremental.

O XP utiliza histórias de usuário fortemente em seu desenvolvimento; os requisitos, arquitetura e o *design* do produto irão se tornar mais claros com o desenvolvimento, de forma que não são definidos previamente. *Releases* do sistema são frequentes e são adicionadas incrementalmente ao produto.
Certo.

---

060. (CESPE/TCE–RO/2019) No que diz respeito a processos e práticas ágeis, o desenvolvimento incremental:
a) é, assim como o *test-driven development*, uma prática da XP (*Extreme Programming*) que exige teste automatizado, *domain-driven design*, *refactoring* e integração contínua.
b) é, na XP (*Extreme Programming*), sustentado por meio de pequenos e frequentes *releases* do sistema, e os clientes estão intimamente envolvidos na especificação e na priorização dos requisitos do sistema.
c) enfoca, assim como o *acceptance test-driven development*, a qualidade do código desenvolvido quanto a recursividade, declaração das variáveis e *clean code*, de modo a torná-lo de fácil entendimento, modificação e testagem.

d) pressupõe o uso do *behavior driven development*, que considera a linguagem de programação a ser usada, da 4ª geração em diante, com foco, principalmente, no comportamento visual, interativo e cognitivo do sistema.
e) enfoca a integração contínua como uma prática de desenvolvimento de *software*, incompatível com a XP (*Extreme Programming*) e o *Scrum*, que permite aos desenvolvedores agregarem alterações de código e realizarem testes.

a) Errada. A prática de desenvolvimento incremental não preconiza o *domain-driven*.
b) Certa. Cliente *on side* e pequenas e frequentes *releases* são práticas do XP. *Release* é algo que é implantado no cliente, ou seja, está pronto para ser utilizado. Funcionalidades prioritárias são desenvolvidas mais cedo para serem entregues mais rapidamente ao cliente, pois prioriza-se o que ele mais precisa em um determinado momento de forma a agregar mais valor ao produto.
c) Errada. Esta prática diz respeito ao TDD, no qual os testes são escritos antes da funcionalidade.
d) Errada. XP não preconiza o uso do *behavior driven development*.
e) Errada. A integração contínua é uma prática compatível com o XP e o *SCRUM*. Todo código deve ser integrado diariamente e todos os testes devem ser executados antes e depois da integração. Se algum problema é encontrado ele deve ser corrigido imediatamente.
Letra b.

---

061. (INSTITUTO CONSULPLAN/TJM–MG/ANALISTA JUDICIÁRIO/TECNOLOGIA DA INFORMAÇÃO/2021) O XP (*Extreme Programming*), uma metodologia ágil de desenvolvimento, foi empregado, pela primeira vez, em 1996, em um projeto da Chrysler, chamado de C3 (*Chrysler Comprehensive Compensation*). Considerando que são apresentados cinco principais valores, assinale, a seguir, dois desses valores.
a) Revisão e Respeito
b) Coragem e Respeito.
c) Simplicidade e Revisão.
d) *Feedback* e Informação.

Os valores do XP são: coragem, simplicidade, comunicação, *feedback* e respeito. Lembre-se do macete para decorar os valores (**Cor Sim Com Fe Re**).
Letra b.

---

062. (CESPE/TCU/AUDITOR FEDERAL DE CONTROLE EXTERNO/TECNOLOGIA DA INFORMAÇÃO/ PARTE II/2010) A atividade de planejamento XP inclui a criação das denominadas histórias de usuário, nas quais devem ser descritas as características e as funcionalidades requeridas para o *software* em desenvolvimento.

Os requisitos são especificados em *user stories* (assim como muitas equipes fazem no *Scrum*). A especificação ocorre na fase de planejamento do *software*.

Certo.

---

063. (CESPE/ANAC/ANALISTA ADMINISTRATIVO/TECNOLOGIA DA INFORMAÇÃO/2009) *Extreme Programming* é um modelo de processo de desenvolvimento de *software* para equipes com grande número de pessoas, que desenvolvem *software* com base em requisitos vagos e que são modificados rapidamente.

O único equívoco na questão está em afirmar que o XP é um modelo para times grandes. O XP é uma metodologia ágil destinada a equipes de tamanho pequeno a médio, que desenvolvem *software* diante de requisitos vagos e que se modificam rapidamente.

Errado.

---

064. (CESPE/MINISTÉRIO DA ECONOMIA/2020) O XP possui planejamento incremental com requisitos registrados em histórias.

Como estudamos, o processo de desenvolvimento do XP é incremental e os requisitos são descritos em histórias de usuário.

Certo.

---

065. (CESPE/MINISTÉRIO DA ECONOMIA/2020) O *refactoring* de código não faz parte do modelo XP, visto que a expectativa é a entrega ágil, e não deve ser considerada em tempo de projeto a recriação de código para aprimoramento.

Refatoração faz parte do XP! Ela ocorre de forma constante durante o desenvolvimento do produto/projeto. Guarde isso.

Errado.

---

066. (UFCG/UFCG/2019) Marque a alternativa INCORRETA com relação a *Extreme Programming* (XP).

a) Comunicação, coragem e respeito são valores dessa metodologia.

b) Em XP, uma das regras é codificar os testes de unidade primeiro.

c) Refatoramento é uma prática recomendada nesse processo.

d) O código a ser enviado para produção é criado por duas pessoas trabalhando juntas em um único computador.

e) O autor, Don Wells, exige que o processo seja seguido à risca, de forma que todas suas regras devem ser respeitadas e, nenhum projeto pode ser realizado sem adaptações e/ou remoção dessas regras.

a) Certa. Estes são os valores do XP (Lembre-se: **Cor SIM Com Fe Re**).

b) Errada. Exatamente. Primeiro são escritos os testes e depois a implementação propriamente dita.

c) Certa. Refatoração faz parte da 'espinha' do XP.

d) Certa. É o que chamamos de programação em pares – uma prática comum do XP.

e) Certa. O processo é adaptativo. Logo, não precisa ser seguido à risca.

Letra b.

---

067. (IBFC/EBSERH/ANALISTA DE TECNOLOGIA DA INFORMAÇÃO/PROCESSOS (HU-FURG)/2016) Para aplicar os valores e princípios durante o desenvolvimento de *software*, a Programação Extrema (*eXtreme Programming* – XP) propõe uma série de práticas. Selecione a única alternativa que NÃO seja uma dessas práticas:

a) Time Coeso (*Whole Team*).

b) Design Complexo (*Complex Design*).

c) Programação Pareada (*Pair Programming*).

d) Semana de 40 horas (*Sustainable Pace*).

e) Refatoração (*Refactoring*).

O XP não preconiza o uso de *design* complexo. Um dos princípios do XP é a utilização de *design* simples. Essa prática se encaixa no princípio da simplicidade e, basicamente, consiste em seguir o que o usuário está pedindo. Por conta disso, o *design* do *software* acaba sendo mais simplista. Além disso, o *software* fica mais fácil de ser alterado. Com essa metodologia, é possível fazer modificações no código quando necessário, sem comprometer a qualidade.

Letra b.

---

068. (IBFC/TRE–PA/ANALISTA JUDICIÁRIO/ANÁLISE DE SISTEMAS/2020) Para aplicar valores e princípios do XP (*Extreme Programming*), durante os processos e práticas ágeis de desenvolvimento de *software*, se propõe uma série específica de práticas. Assinale a alternativa que apresenta algumas dessas "boas práticas" utilizadas tradicionalmente em projetos, usando XP.

a) *Reformation – Pair Programming – PlayStation Game*

b) *Refactoring – Pair Programming – Planning Game*

c) *Reformation – Pair Production – Planning Game*

d) *Refactoring – Pair Production – PlayStation Game*

Planning Game: No *planning game*, o *software* é planejado de forma eficaz pela equipe de desenvolvimento e pela equipe do cliente. Nessa atividade, cada equipe tem um papel claro e definido: a equipe do cliente é responsável por definir escopo e prioridades, enquanto a equipe de desenvolvimento fornece estimativas.

Programação em par: É uma prática que envolve dois programadores trabalhando em conjunto em um único computador. Como há uma revisão constante do *software* por duas pessoas, a probabilidade de falhas é reduzida. Busca-se continuamente a evolução da equipe, melhorando a qualidade do código-fonte. Essa prática é fundamental no XP, pois a colaboração de dois programadores contribui significativamente para o trabalho em equipe.

Refatoração: É um processo que permite a melhoria contínua da programação, minimizando a introdução de erros e mantendo compatibilidade com o código já existente. Refatorar melhora a clareza e a legibilidade do código, facilitando a manutenção. Além disso, o código torna-se mais coeso, evitando duplicação no código-fonte e proporcionando um melhor aproveitamento.

Letra b.

---

069. (IBFC/EBSERH/ANALISTA DE TECNOLOGIA DA INFORMAÇÃO/PROCESSOS (HUAP-UFF)/2016) Equipes XP (*eXtreme Programming*) planejam utilizando histórias escritas em pequenos cartões. Essas histórias devem ter como objetivo:

a) a modelagem de dados.

b) as métricas de *software*.

c) os requisitos não funcionais.

d) os requisitos funcionais.

e) tanto os requisitos funcionais como os requisitos não funcionais.

- **Testável**: É essencial que as Histórias de Usuário sejam testáveis, garantindo que o produto final esteja conforme o definido e livre de erros ao término do desenvolvimento.
- **"*Small*" (Pequena)**: As histórias de usuário devem ser pequenas o suficiente para serem desenvolvidas em uma única *Sprint*, evitando a complexidade desnecessária.
- **Estimável**: As histórias de usuário devem ser passíveis de estimativa pelo Time de Desenvolvimento. Embora a precisão possa variar, a estimativa é crucial para a priorização do *Backlog* do Produto, o planejamento de custos e *releases*. Dificuldades em estimar podem indicar tamanho excessivo, informações insuficientes ou ambiguidade.
- **Valorosa**: Para evitar desperdício de tempo e recursos, o Dono do Produto deve priorizar as histórias que proporcionam o maior valor ao negócio. As histórias de usuário devem gerar valor significativo.
- **Negociável**: As histórias de usuário não detalham todos os requisitos, facilitando a realização de conversas entre a equipe *Scrum* e os *Stakeholders* para negociar os detalhes. Elas servem como lembretes de que uma conversa detalhada é necessária, não sendo contratos rígidos.
- **Independente**: Idealmente, as histórias de usuário devem ser independentes umas das outras. Isso permite que o Dono do Produto as ordene, planeje e as implemente com base no valor que cada uma agrega ao negócio.

Letra d.

---

070. (IBFC/TJ–PE/TÉCNICO JUDICIÁRIO/PROGRAMADOR DE COMPUTADOR/2017) Está sendo implementado o XP (*eXtreme Programming*) em uma equipe de TI. Para tanto, está sendo colocada a seguinte série de práticas específicas da metodologia XP em análise:
I – Programação Pareada (*Pair Programming*).
II – Fases pequenas (*Small Releases*).
III – Refatoração (*Refactoring*).
IV – Jogo de Planejamento (*Planning Game*).
Com base no seu conhecimento sobre a metodologia citada acima, suas práticas específicas estão corretamente relacionadas nos itens:
a) I, II e III, apenas.
b) I, II e IV, apenas.
c) II, III e IV, apenas.
d) I, III e IV, apenas.
e) I, II, III e IV.

Todos os itens estão corretos! São práticas constantes no XP:

RELEMBRANDO

- **Planning Game**: Durante o *Planning Game*, a equipe de desenvolvimento e a equipe do cliente participam efetivamente do planejamento do *software*. Cada equipe tem funções claras e definidas: a equipe do cliente determina escopo e prioridades, enquanto a equipe de desenvolvimento fornece estimativas.
- **Refatoração**: Um processo contínuo que aprimora a programação, minimizando a introdução de erros e mantendo a compatibilidade com o código existente. A refatoração aprimora a clareza e a leitura do código, facilitando a manutenção. Além disso, torna o código mais coeso, evitando duplicações.
- ***Releases* Curtos**: Liberações de pequenas versões funcionais do projeto facilitam a aceitação pelo cliente, permitindo que este teste partes do sistema. Essas versões são ainda menores do que as produzidas em outras metodologias incrementais, como o RUP. Entregar pequenos pedaços do produto ao cliente antes do término total proporciona uma visualização antecipada.
- **Programação em Par**: Realizada em um único computador, a programação em par (dupla) envolve dois programadores que revisam continuamente o *software*, reduzindo a possibilidade de falhas. Essa prática é fundamental no XP, promovendo a evolução da equipe e melhorando a qualidade do código-fonte.

Letra e.

---

071. (CESPE/CEBRASPE/TCE–RJ/ANALISTA DE CONTROLE EXTERNO/2022) Acerca de RUP (*rational unified process*) e XP (*extreme programming*), julgue o seguinte item.

Na XP, as histórias dos usuários (casos de uso) devem descrever os detalhes dos requisitos da solução, tais como a tecnologia a ser utilizada e a modelagem do banco de dados; isso irá permitir planejar melhor a interface do usuário na *release planning* e, consequentemente, o desenvolvimento da solução.

Histórias de Usuário, também conhecidas como *User stories*, são artefatos amplamente utilizados em metodologias ágeis. Elas consistem em descrições concisas de funcionalidades fornecidas pelo usuário, representando pequenas partes do sistema a ser construído. Essas histórias não se configuram como especificações detalhadas de funcionalidades, evitando abordar aspectos técnicos ou formas específicas de desenvolvimento.

Pelo contrário, adotam uma linguagem mais acessível e centrada no ponto de vista do usuário. Essa abordagem permite uma compreensão intuitiva das necessidades do cliente, sem entrar em minúcias técnicas durante a fase de definição de requisitos.
Errado.

---

072. (IBFC/EBSERH/ANALISTA DE TECNOLOGIA DA INFORMAÇÃO/PROCESSOS (HUGG–UNIRIO)/2017) Dentro das práticas do XP (*eXtreme Programming*) existe uma fundamental que é o Jogo de Planejamento (*Planning Game*). Para serem realizadas adequadamente essas reuniões com os usuários, deve(m) ter sido feito(s) antecipadamente:
a) o *Sustainable Pace*
b) as *Small Releases*
c) os *Customer TestsD*
d) as *User stories*
e) um *Simple Design*

Na reunião de planejamento, o cliente apresenta as funcionalidades, também conhecidas como histórias, indicando a prioridade de desenvolvimento. Os desenvolvedores, por sua vez, têm o direito de estimar e fornecer suas considerações técnicas. O Jogo do Planejamento visa criar um plano de trabalho que maximize o valor para o negócio do cliente por meio das funcionalidades.
Alguns detalhes adicionais: caso uma história seja demasiadamente extensa, é recomendável dividi-la em tarefas com duração máxima de alguns dias. A redação das histórias deve ser realizada pelo cliente, permitindo-lhe uma reflexão mais aprofundada sobre cada funcionalidade. O planejamento desempenha um papel crucial ao garantir que as prioridades estejam alinhadas com as ações mais cruciais a serem realizadas.
Letra d.

---

073. (CESPE/CEBRASPE/TCE–RJ/ANALISTA DE CONTROLE EXTERNO/2022) Acerca de RUP (*rational unified process*) e XP (*extreme programming*), julgue o seguinte item.
Uma das práticas da XP é a integração contínua que visa aperfeiçoar o projeto de codificação do sistema de *software*, tal que a estrutura interna se aprimora sem que seu comportamento se altere.

Este é o conceito de refatoração: aprimoração do código de forma contínua e constante, tornando-o simples de entender e fácil de manter.
Errado.

---

074. (VUNESP/CÂMARA DE PIRACICABA–SP/2019) Um dos processos ágeis de desenvolvimento de software é a programação extrema (*extreme programming* – XP), cuja fase ou atividade inicial é composta pela descrição dos cenários (características e funcionalidades) requisitadas para o *software* a ser desenvolvido. Essa atividade recebe a denominação de:
a) métodos práticos.
b) histórias de usuário.
c) estruturas de apoio.
d) classes de projeto.
e) artefatos de usuário

A primeira atividade de uma *release* é selecionar as histórias do usuário. Sem as histórias, não há como desenvolver o código. Não se prenda ao nome 'atividade' apresentado na questão.
Letra b.

---

075. (CESPE/MINISTÉRIO DA ECONOMIA/2020) Grandes quantidades de horas extras são aceitáveis em médio e longo prazo, para agilizar a entrega de requisitos.

O XP enfatiza a prática do desenvolvimento sustentável, preconizando uma carga de trabalho de 40 horas semanais. Horas extras não são encorajadas, visto que podem indicar possíveis falhas no processo de desenvolvimento.
Errado.

---

076. (CESPE/CEBRASPE/SERPRO/ANALISTA/ESPECIALIZAÇÃO: DESENVOLVIMENTO DE SISTEMAS/2021) Acerca de metodologias ágeis de desenvolvimento, julgue o item seguinte. Em XP, a estruturação do valor *feedback* pode ser alcançada de forma rápida por meio de testes automatizados de *software*, que validam ou não um código produzido ou alterado.

Uma das formas de alcançar a qualidade é através dos testes automatizados que permitem *feedback* rápido. Os testes automatizados respondem de forma imediata se aquilo que foi introduzido ainda está funcionando e de acordo com o que foi estabelecido. Lembrando que o cliente participa da etapa de testes e podem fornecer *feedback´s* de forma rápida.
Certo.

---

077. (CESPE/CEBRASPE/MINISTÉRIO DA ECONOMIA/TECNOLOGIA DA INFORMAÇÃO/DESENVOLVIMENTO DE SOFTWARE/2020) Julgue o item seguinte, a respeito de programação ágil com XP (*extreme programming*).

Como forma de agilizar as implantações de novas releases nesse modelo, são acumulados grandes grupos de funcionalidades e implantadas grandes *releases*.

Cada *release* deve ser tão pequeno quanto possível, contendo os requisitos mais importantes para o negócio. Isso possibilita ter *releases* entregues em um espaço de tempo menor o que resulta em maior *feedback* para clientes e programadores, facilitando o aprendizado e a correção dos defeitos do sistema.

XP = entregas em pequenas *releases*.

Errado.

---

078. (CESPE/CEBRASPE/EBC/2011) Uma metodologia de desenvolvimento de *software* pode ser classificada como uma metodologia ágil quando efetua o desenvolvimento do *software* de forma incremental (libera pequenas versões, em iterações de curta duração) e é colaborativa (cliente e desenvolvedores trabalham juntos, em constante comunicação), direta (o método em si é simples de aprender e modificar) e adaptativa (capaz de responder eficientemente às mudanças).

Considerando a definição acima, de Abrahamsson, julgue o item a seguir, a respeito das metodologias ágeis de desenvolvimento de *software*.

O que os métodos ágeis buscam é como evitar as mudanças desde o início do projeto e não a melhor maneira de tratar essas mudanças.

Uma das características do métodos ágeis (incluindo o XP) é *Abraçar as mudanças*. Elas serão realizadas de acordo com o melhor entendimento do projeto e sempre serão bem-vindas.

Errado.

---

079. (CESPE/CEBRASPE/TJ–AM/ANALISTA JUDICIÁRIO/ANALISTA DE SISTEMAS/2019) Julgue o item seguinte, a respeito das metodologias de desenvolvimento de *software*.

No XP (*Extreme Programming*), o valor de uma história de usuário é atribuído pelos membros da equipe e é medido em termos de semanas estimadas para o desenvolvimento.

O cliente estabelece a prioridade das histórias a serem implementadas, atribuindo a cada uma um valor baseado no impacto de negócios para o produto. Simultaneamente, os

membros da equipe avaliam cada história, atribuindo um custo de desenvolvimento medido em semanas. Assim, o usuário contribui com o valor, enquanto a equipe contribui com o custo para o desenvolvimento. A dinâmica dessa questão pode exigir uma análise cuidadosa.
Errado.

---

080. (IBFC/EBSERH/ANALISTA DE TECNOLOGIA DA INFORMAÇÃO/PROCESSOS (HU-FURG)/2016) Para aplicar os valores e princípios durante o desenvolvimento de *software*, a Programação Extrema (*eXtreme Programming* – XP) propõe uma série de práticas. Selecione a única alternativa que NÃO seja uma dessas práticas:
a) Time Coeso (*Whole Team*).
b) Design Complexo (*Complex Design*).
c) Programação Pareada (*Pair Programming*).
d) Semana de 40 horas (*Sustainable Pace*).
e) Refatoração (*Refactoring*).

O XP não preconiza o uso de *design* complexo. Um dos princípios do XP é a utilização de *design* simples. Essa prática se encaixa no princípio da simplicidade e é basicamente seguir o que o usuário está pedindo, por conta disso o *design* do *software* acaba sendo mais simplista. Além disso, o *software* acaba ficando mais fácil de ser alterado. Com essa metodologia você consegue alterar o código quando precisar, sem comprometer a qualidade.
Letra b.

---

081. (IBFC/TRE–PA/ANALISTA JUDICIÁRIO/ANÁLISE DE SISTEMAS/2020) Para aplicar valores e princípios do XP (*Extreme Programming*), durante os processos e práticas ágeis de desenvolvimento de *software*, se propõe uma série específica de práticas. Assinale a alternativa que apresenta algumas dessas "boas práticas" utilizadas tradicionalmente em projetos, usando XP.
a) *Reformation – Pair Programming – PlayStation Game*
b) *Refactoring – Pair Programming – Planning Game*
c) *Reformation – Pair Production – Planning Game*
d) *Refactoring – Pair Production – PlayStation Game*

Refatoração: É um processo que permite a melhoria contínua da programação, o mínimo de introdução de erros e mantendo compatibilidade com o código já existente. Refatorar

melhora a clareza, leitura do código e facilita a manutenção. Além disso, o código fica mais coeso e você tem um melhor aproveitamento, evitando duplicação no código fonte.
Programação em par: É uma programação em par (dupla) em um único computador. Como é apenas um computador, o *software* sempre é revisto por duas pessoas diminuindo assim a possibilidade de falhas. Busca-se sempre a evolução da equipe melhorando a qualidade do código fonte. Ela é uma das práticas primordiais do XP, pois dois programadores fazendo o trabalho juntos acaba agregando muito para o trabalho em equipe.
Planning Game: No *planning game* o *software* é efetivamente planejado pela equipe de desenvolvimento e pela equipe do cliente. Nesta atividade, cada equipe tem o seu papel bem claro e definido: a equipe do cliente é responsável por definir escopo e prioridades; e a equipe de desenvolvimento por fornecer estimativas.
Letra b.

---

082. (IBFC/TJ–PE/TÉCNICO JUDICIÁRIO/PROGRAMADOR DE COMPUTADOR/2017) Está sendo implementado o XP (*eXtreme Programming*) em uma equipe de TI. Para tanto, está sendo colocada a seguinte série de práticas específicas da metodologia XP em análise:
I – Programação Pareada (*Pair Programming*).
II – Fases pequenas (*Small Releases*).
III – Refatoração (*Refactoring*).
IV – Jogo de Planejamento (*Planning Game*).
Com base no seu conhecimento sobre a metodologia citada acima, suas práticas específicas estão corretamente relacionadas nos itens:
a) I, II e III, apenas.
b) I, II e IV, apenas.
c) II, III e IV, apenas.
d) I, III e IV, apenas.
e) I, II, III e IV.

Todos os itens estão corretos! São práticas constantes no XP:

RELEMBRANDO

Programação em par: É uma programação em par (dupla) em um único computador. Como é apenas um computador, o *software* sempre é revisto por duas pessoas diminuindo assim a possibilidade de falhas. Busca-se sempre a evolução da equipe melhorando a qualidade

do código fonte. Ela é uma das práticas primordiais do XP, pois dois programadores fazendo o trabalho juntos acaba agregando muito para o trabalho em equipe.
Releases curtos: As liberações de pequenas versões funcionais do projeto auxiliam muito no processo de aceitação por parte do cliente que já pode testar uma parte do sistema. As versões chegam ainda ser menores que as produzidas em outras metodologias incrementais, como o RUP. Os *releases* são pequenos pedaços do produto que são entregues ao cliente antes do tempo, assim o cliente não precisa esperar o produto todo ficar pronto para ver.
Refatoração: É um processo que permite a melhoria contínua da programação, o mínimo de introdução de erros e mantendo compatibilidade com o código já existente. Refatorar melhora a clareza, leitura do código e facilita a manutenção. Além disso, o código fica mais coeso e você tem um melhor aproveitamento, evitando duplicação no código fonte.
Planning Game: No *planning game* o *software* é efetivamente planejado pela equipe de desenvolvimento e pela equipe do cliente. Nesta atividade, cada equipe tem o seu papel bem claro e definido: a equipe do cliente é responsável por definir escopo e prioridades; e a equipe de desenvolvimento por fornecer estimativas.
Letra e.

---

083. (IBFC/EBSERH/ANALISTA DE TECNOLOGIA DA INFORMAÇÃO/PROCESSOS (HUGG–UNIRIO)/2017) Dentro das práticas do XP (*eXtreme Programming*) existe uma fundamental que é o Jogo de Planejamento (*Planning Game*). Para serem realizadas adequadamente essas reuniões com os usuários, deve(m) ter sido feito(s) antecipadamente:
a) o *Sustainable Pace*
b) as *Small Releases*
c) os *Customer TestsD*
d) as *User stories*
e) um *Simple Design*

Na reunião de planejamento, o cliente informará as funcionalidades, também chamadas de histórias, e indicar a prioridade de desenvolvimento das mesmas. Os desenvolvedores, por sua vez, têm o direito de estimar e apresentar suas considerações técnicas. O objetivo do Jogo do Planejamento é criar um plano de trabalho, que seja capaz de gerar (através de funcionalidades) o máximo de valor possível para o negócio do cliente.
Mais alguns detalhes: Se uma história for muito grande, ela deve ser dividida em tarefas com duração máxima de alguns dias. Elas devem ser escritas pelo cliente, pois, assim, ele consegue pensar melhor em cada funcionalidade.

O planejamento é importante para que você sempre faça a coisa mais importante a ser feita.
Letra d.

---

084. (IBFC/EBSERH/ANALISTA DE TECNOLOGIA DA INFORMAÇÃO/PROCESSOS (HUAP-UFF)/2016) Equipes XP (*eXtreme Programming*) planejam utilizando histórias escritas em pequenos cartões. Essas histórias devem ter como objetivo:
a) a modelagem de dados.
b) as métricas de *software*.
c) os requisitos não funcionais.
d) os requisitos funcionais.
e) tanto os requisitos funcionais como os requisitos não funcionais.

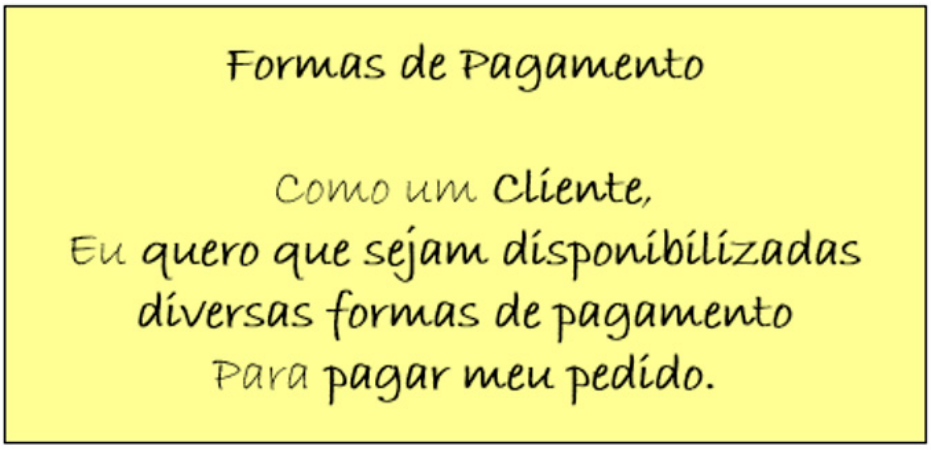
Histórias de usuário são uma descrição resumida de alguma funcionalidade fornecida pelo sistema do ponto de vista de um usuário desse sistema. Elas podem ser escritas em cartões. Tenha em mente que as histórias descrevem os requisitos funcionais do produto a ser desenvolvido e SEMPRE devem ter a participação ativa do usuário. Esses cartões contendo os requisitos serão utilizados pela equipe de desenvolvimento durante todo o processo de desenvolvimento.
As histórias de usuário possuem algumas características:
Independente – Sempre que possível, as histórias de usuário devem ser independentes umas das outras, a fim de que o Dono do Produto possa ordená-las, planejá-las e implementá-las conforme o valor que cada uma possui para o negócio.
Negociável – Conforme foi dito anteriormente, a história de usuário não contém todos os detalhes de um requisito. Sendo assim, ela deve ser descrita em um nível que permita ao Time *Scrum* e *Stakeholders* realizarem conversas a fim negociar o detalhamento desses requisitos. Uma história de usuário não é um contrato a ser seguido, e, sim, um lembrete a todos que deve ser realizada uma conversa a fim de detalhar os requisitos.
Valorosa – No artigo referente às características de um Backlog do Produto, dissemos que, a fim de evitar desperdício de tempo e de recursos, o Dono do Produto deve priorizar o desenvolvimento dos 20% dos itens do *Backlog* do Produto que irão agregar o maior valor para o negócio. Os 80% dos itens restantes devem ficar para depois ou, caso não sejam efetivamente necessários, devem ser descartados. Dessa forma, as histórias do usuário devem gerar valor para o negócio.
Estimável – É muito importante que uma história de usuário possa ser estimada pelo Time de Desenvolvimento, mesmo que o nível de precisão não seja alto, uma vez que a estimativa possibilita ao Dono do Produto priorizar o *Backlog* do Produto, estimar custos, planejar

releases etc. Quando o Time de Desenvolvimento não consegue estimar uma História de Usuário, isso pode significar que o seu tamanho é muito grande, que as informações são insuficientes, que o seu conteúdo está muito ambíguo, ou até mesmo que o Time não possui conhecimento sobre a tecnologia a ser utilizada.

"*Small*" (Pequena) – As Histórias de Usuário devem ser pequenas, ou seja, necessitam possuir um tamanho máximo que as permitam serem desenvolvidas dentro de uma única *Sprint*.

Testável – É muito importante que as Histórias de Usuário possam ser testadas, a fim de que seja possível assegurar que o produto obtido ao final do desenvolvimento esteja de acordo com o que foi definido e sem a existência de erros.

Exemplo de um cartão com história de usuário:



Letra d.

REFERÊNCIAS

ASTEL, Dave; MILLER, Granville. **Extreme Programming – Guia Prático**. São Paulo, 2002.

BECK, Kent. **Programação extrema explicada: acolha as mudanças**. Porto Alegre: Bookman, 2010.

BECK, Kent; ANDRES, Cynthia. **Extreme Programming Explained: Embrace Change**. Addison-Wesley, 2. edition, 2004.

http://www.eripi.com.br/2017/images/anais/minicursos/3.pdf. Acessado em Janeiro de 2024.

https://agilemanifesto.org/iso/ptbr/manifesto.html. Acessado em Janeiro de 2024.

https://manifestoagil.com.br/.Acessado em Janeiro de 2024.

https://www.desenvolvimentoagil.com.br/xp/dissertacaoXP.pdf. Acessado em Janeiro de 2024.

PRESSMAN, Roger S. **Engenharia de Sw**. São Paulo: Campus, 1995.

SOMMERVILLE, I. **Engenharia de Sw**. 9. ed. São Paulo: Pearson Addison-Wesley, 2011.

Abra

caminhos

crie

futuros

gran.com.br