(45) AOCP - MPE/BA - 2014

No CMMI, existem duas diferentes representações, a contínua e a por estágios. A representação contínua é caracterizada por

(A) Níveis de Maturidade.

(B) Níveis de Procedimento.

(C) Níveis de Capacidade.

(D) Normatização Técnica.

(E) Normas de Padronização.

Java

(61) 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.

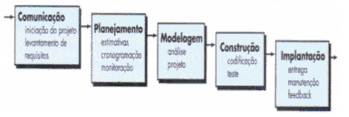
(C) programação em pares.

(D) padrões de codificação.

(E) integração contínua.

[28] BIORIO – EMGEPRON – 2014

No que diz respeito à Engenharia de Software, a figura abaixo ilustra um modelo conhecido por abordagem “top-down” e que é referência para muitos outros modelos, servindo de base para muitos projetos modernos.



Esse modelo é denominado:

(A) em cascata

(B) em eventos

(C) incremental

(D) evolucionário

[36] CESGRARIO – EPE - 2014

Caso os desenvolvedores decidissem utilizar a técnica de Análise de Pontos de Função para fazer uma estimativa do tamanho do projeto SCL, o diagrama de entidades e relacionamentos desenvolvido permitiria fazer uma contagem inicial dos arquivos lógicos internos.

Considerando a entidade Cliente como um arquivo lógico interno com apenas um tipo de registro e apenas com os campos nela descritos, ela contribuiria com quantos pontos de função não ajustados?

(A) 3

(B) 4

(C) 5

(D) 7

(E) 10

[39] CESGRARIO – EPE - 2014

Considere um programa de computador único que pode ser representado por um grafo de fluxo de controle com 9 arestas e 8 nós.

Qual o limite superior para o número de testes que devem ser projetados e executados para garantir a cobertura de todas as instruções do programa?

(A) 1

(B) 2

(C) 3

(D) 4

(E) 5

[40] CESGRARIO – EPE - 2014

No planejamento de projetos de software, e principalmente em metodologias ágeis de desenvolvimento, muitos autores defendem a técnica conhecida como “timebox”, que

(A) estima o menor e o maior tempo de desenvolvimento para cada funcionalidade a ser desenvolvida, definindo uma “caixa” de tempo em vez de um prazo fixo.

(B) parte do tempo disponível em uma fábrica de software para especificar versões consecutivas de um produto, conhecidas como “caixas”

(C) divide um produto de software em versões de complexidade crescente, conhecidas como “caixas”, especificando o tempo de desenvolvimento de cada caixa do mais rápido para o mais longo.

(D) define um tempo para cada função a ser desenvolvida e as aloca em “caixas” de igual tempo de desenvolvimento que são escolhidas pelos desenvolvedores.

(E) define o tempo a ser utilizado em um ciclo de desenvolvimento e depois define a funcionalidade que pode ser desenvolvida naquela “caixa” de tempo.

[71] CESPE – CAIXA - 2014

A estimativa final da entrega de um incremento sempre é definida pelo Product Owner.

[72] CESPE – CAIXA - 2014

De acordo com o Scrum, na entrega de um novo incremento de funcionalidade de um produto, esse incremento não precisa ser utilizável nem estar funcionando em conjunto com todos os

anteriores para que o Product Owner o libere e o adicione aos demais incrementos desse produto.

[73] CESPE – CAIXA - 2014

Os itens do backlog do produto devem conter a ordem de desenvolvimento e a estimativa de sua entrega.

[74] CESPE – CAIXA - 2014

Considere que, tendo acessado determinada loja virtual, uma pessoa, ao optar por fazer uma compra por meio do cartão de crédito, tenha tido de preencher um formulário de cadastro de

cartões de crédito, cujos dados são armazenados junto com os demais dados do cliente. Nessa situação, o cadastro de cartões de crédito é classificado como uma entrada externa.

[75] CESPE – CAIXA - 2014

Em um projeto de melhoria do sistema de compra de passagens aéreas, as telas excluídas não deverão ser contatadas.

[76] CESPE – CAIXA - 2014

Considere que, em um sistema de locação de carros, ao se consultar o valor que deverá ser pago por um cliente, o sistema informe a quilometragem, se o cliente contratou determinado

tipo de seguro e, ainda, o valor final a ser pago. Nesse caso, essa consulta, de acordo com a APF, é considerada uma saída externa.

[78] CESPE – CAIXA - 2014

Na abordagem em espiral, todas as necessidades levantadas pelo cliente são implementadas, o que possibilita a entrega de todo o sistema de uma só vez a esse cliente.

[84] CESPE – CAIXA - 2014

Na fase de levantamento de requisitos, é necessário arrolar os fatos e se ater apenas aos dados coletados pelo cliente.

[85] CESPE – CAIXA - 2014

Na etapa de análise de requisitos, são especificadas todas as funções e o desempenho do software, determinando-se, ainda, o que deverá ou não ser feito.

[91] CESPE – CAIXA - 2014

O teste embasado em especificação ou teste caixa preta, realizado a partir das especificações do programa e requisitos funcionais, é utilizado para demonstração de que as funções do

software são operacionais, que a entrada é adequadamente aceita e que a saída é corretamente produzida conforme as especificações.

[92] CESPE – CAIXA - 2014

O processo de teste denominado teste de integração utiliza técnicas de caixa branca e concentra-se no código-fonte de cada unidade implementada, para a garantia de que as

informações fluam para dentro e fora da unidade do programa conforme a especificação levantada.

[93] CESPE – CAIXA - 2014

Todas as alterações realizadas em um produto de software devem ser documentadas e colocadas em produção e no registro de controle de defeitos. Para tanto, utilizam-se

sistemas como o (Software Configuration Management) SCM; por exemplo: (Correct Visualization System) CVS ou (Visualization Normalization System) VNS.

[94] CESPE – CAIXA - 2014

O baseline — versão de um software, sistema ou produto — é um marco de referência do estado do produto, caracterizado pela entrega em produção de um ou mais itens de configuração e pela aprovação obtida por meio de uma revisão técnica formal.

[95] CESPE – CAIXA - 2014

TDD é uma técnica de desenho e desenvolvimento embasada em práticas de uma estrutura de processo incremental e iterativo. Essa estrutura é formada por iterações de atividades

de desenvolvimento que ocorrem uma após a outra, sendo a saída de cada iteração um incremento do produto.

[107] CESPE – CAIXA - 2014

Medições e controles dos riscos de segurança da informação, além do registro e da qualificação de incidentes, são resultados esperados no nível C do MPS-BR.

[108] CESPE – CAIXA - 2014

O estabelecimento de requisitos para liberação de serviços e os acordos com as partes interessadas na liberação desses serviços são contemplados no nível F do MPS-BR.

[31] FCC - TRT3 – 2014

O modelo de ciclo de vida incremental e iterativo foi proposto como uma resposta aos problemas encontrados no modelo em cascata. Em relação a este tipo de modelo de processo, é INCORRETO afirmar que

(A) os dois padrões mais conhecidos de sistemas iterativos de desenvolvimento são: o UP (Processo Unificado) e o desenvolvimento ágil de *software*. Por isso, o desenvolvimento iterativo e incremental é também uma parte essencial do XP.

(B) um processo de desenvolvimento, segundo essa abordagem, divide o desenvolvimento de um produto de *software* em ciclos. Em cada ciclo de desenvolvimento podem ser identificadas as fases de análise, projeto, implementação e testes.

(C) cada um dos ciclos de desenvolvimento considera um subconjunto de requisitos. Os requisitos são desenvolvidos uma vez que sejam alocados a um ciclo de desenvolvimento. No próximo ciclo, um outro subconjunto dos requisitos é considerado para ser desenvolvido, o que produz um novo incremento do sistema que contém extensões e refinamentos sobre o incremento anterior.

(D) o desenvolvimento evolui em versões, através da construção incremental e iterativa de novas funcionalidades até que o sistema completo esteja construído. Apenas uma parte dos requisitos é considerada em cada ciclo de desenvolvimento.

(E) o desenvolvimento incremental e iterativo de *software* sempre é possível, desde que a equipe de desenvolvimento divida os requisitos do sistema em partes, para que cada parte seja alocada a um ciclo de desenvolvimento. Para isso, a Norma NBR ISO/IEC 12207:2009 traz recomendações em sua seção Processos de Contexto de Sistema.

[32] FCC - TRT3 – 2014

Há diversos processos e práticas ágeis de desenvolvimento de *software*. Considere:

**I**. Seu objetivo é criar um “código limpo que funcione”. Trabalha com a estratégia *Red* − *Green - Refactor*:

Codifique o teste;

Faça-o compilar e executar. O teste não deve passar (*Red).*

Implemente o requisito e faça o teste passar (*Green*).

Refatore o código (*Refactor*).

**II**. Suas práticas, regras e valores garantem um agradável ambiente de desenvolvimento de *software* para os seus seguidores, que são conduzidos pelos princípios básicos:

Comunicação: manter o melhor relacionamento possível entre clientes e desenvolvedores, preferindo conversas pessoais a outros meios de comunicação;

Simplicidade: implementar apenas requisitos atuais, evitando adicionar funcionalidades que podem ser importantes somente no futuro;

*Feedback:* o desenvolvedor terá informações constantes do cliente e do código, em que testes constantes indicam os erros tanto individuais quanto do *software* integrado;

Coragem: encorajar as pessoas que não possuem facilidade de comunicação e bom relacionamento interpessoal,encorajar a equipe a experimentar e buscar novas soluções, além de encorajar a obtenção de *feedback* do cliente.

**III**. Objetiva capturar os critérios de aceitação para as funcionalidades em desenvolvimento. Trabalha com as seguintes etapas:

Discutir (*Discuss*): discussão colaborativa com a equipe visando elicitar os critérios de aceitação.

Refinar (*Distill*): refinamento dos critérios de aceitação em um conjunto concreto de cenários/exemplos de uso descrevendo o comportamento esperado da aplicação em uma linguagem comum a todos os membros da equipe.

Desenvolver (*Develop*): transformação dos testes de aceitação (descrevendo o comportamento esperado do *software*) em testes/especificação automatizados.

**IV**. Suas práticas incluem:

Envolver as partes interessadas no processo através de *Outside-in Development*.

Usar exemplos para descrever o comportamento de uma aplicação ou unidades de código.

Automatizar os exemplos para prover um *feedback* rápido e testes de regressão.

Usar o verbo deve (*should*) ao descrever o comportamento de *software* para ajudar a esclarecer responsabilidades e permitir que funcionalidades sejam questionadas.

Usar dublês de teste (*mocks, stubs, fakes, dummies, spies*) para auxiliar na colaboração entre módulos e códigos que ainda não foram escritos.

Os processos ágeis **I**, **II**, **III** e **IV** são, correta e respectivamente, denominados:

(A) TDD - XP - ATDD - BDD

(B) BDD - DDD - ATDD - XP

(C) TDD - BDD - DDD - XP

(D) ATDD - XP - DDD - BDD

(E) ATDD - BDD - TDD - DDD

[41] FCC - TRT3 – 2014

Ana foi contratada em uma empresa para efetuar trabalhos de desenvolvimento relacionados à área de informática. Logo no

primeiro dia foi convidada a participar de uma reunião que é efetuada diariamente, de apenas 15 minutos. Todos os participantes

ficam em pé e ela é conduzida pelos próprios desenvolvedores. Durante este pequena reunião, foram abordados o que cada

desenvolvedor conseguiu concluir desde a última reunião, o que ele pretende efetuar até a próxima e, o que Ana achou muito

importante, o que está impedindo que este desenvolvedor prossiga com seu trabalho. Ana foi informada que esta reunião

pertence ao método ágil

(A) XP, e que tanto a reunião quanto as perguntas são denominadas Interação Contínua.

(B) *Jerkins*, e que o nome dado a esta reunião é *Sprint*.

(C) *Kanban*, e as questões efetuadas são chamadas de artefatos.

(D) *Scrum*, e que o nome dado a esta reunião é *Daily Scrum*.

(E) *Sprint*, e que as questões efetuadas são chamadas de *Backlog*.

[56] FCC - TRT3 – 2014

O nível de maturidade B do MPS.BR (Gerenciado Quantitativamente), é composto pelos processos dos níveis de maturidade anteriores (G ao C). Este nível de maturidade

(A) possui, dentre seus processos específicos, Gerência de Requisitos e Gerência de Projetos.

(B) deve satisfazer integralmente os atributos de processo AP 4.1, AP 4.2, AP 5.1 e AP 5.2.

(C) possui, dentre seus processos específicos, Gerenciamento de Decisões e Gerência de Riscos.

(D) não possui processos específicos.

(E) possui, dentre seus processos específicos, Desenvolvimento de Requisitos e Validação e Verificação.

[56] FGV - DPE – 2014

Uma organização está interessada em definir um processo para orientar a sua equipe de desenvolvimento a executar as atividades necessárias para a criação e disponibilização de novas versões do produto de software que é o carro-chefe da empresa. Esse processo precisa conter explicitamente as etapas comuns de um desenvolvimento de software (por exemplo, levantamento, análise, projeto, construção e testes) e, como o produto de software em questão tem um forte requisito de qualidade, é necessário que as atividades de garantia da qualidade sejam bem explícitas em relação às etapas e/ou documentos relacionados sendo avaliados.

Dentre as opções de modelos de ciclo de vida abaixo, o mais adequado a essa necessidade é

(A) RAD.

(B) modelo em V.

(C) cascata.

(D) prototipação.

(E) XP.

[57] FGV - DPE – 2014

Uma das características da metodologia ágil Scrum é

(A) focar nas práticas de engenharia.

(B) focar na documentação formal do software.

(C) ser um método iterativo e incremental.

(D) exigir o planejamento do projeto, de acordo com as práticas do PMBOK.

(E) não exigir interação com o cliente.

[58] FGV - DPE – 2014

Um mecanismo comum de garantia da qualidade adotado por muitas organizações é o de revisões de software, que podem ser aplicadas ao longo do ciclo de vida de desenvolvimento de software. Sobre revisões de software é correto afirmar que

(A) podem ser aplicadas apenas nas etapas iniciais do ciclo de desenvolvimento visando identificar falhas não percebidas pelo gerente no planejamento do projeto.

(B) as revisões são capazes de detectar e eliminar grande percentual de erros, reduzindo substancialmente o custo das atividades seguintes do ciclo de desenvolvimento.

(C) tem um custo muito elevado, o que, em geral, pode não compensar o investimento realizado.

(D) devem ser realizadas após o software ter sido construído, porque o software pronto é o produto mais importante de um projeto de desenvolvimento.

(E) podem ser aplicadas em diferentes estágios do ciclo de desenvolvimento, porém com benefício menor quanto mais cedo se estiver neste ciclo de desenvolvimento.

[59] FGV - DPE – 2014

Considere uma organização desenvolvedora de software que possua um processo de software composto das fases de Levantamento de Requisitos, Análise de Software, Projeto de Software, Codificação, Testes e Entrega do Software e tenha a cultura de estimar seus projetos utilizando Análise de Pontos por Função. Caso essa organização esteja interessada na criação de uma base de estimativas de seus projetos, a contagem de pontos por função seria mais indicada

(A) na etapa de Análise do Software, pois, assim, o gerente de projetos será capaz de estimar o tempo restante do projeto com maior precisão.

(B) no início do projeto, na fase de Levantamento de Requisitos, pois a equipe já terá elementos suficientes para determinar o prazo de entrega do projeto.

(C) durante etapa de Projeto de Software, pois já haverá um grande entendimento dos requisitos e, como o software será construído logo em seguida, o valor da contagem obtido pode ser utilizado tanto para estimar o projeto quanto para calibrar a taxa média de produtividade da organização.

(D) no início do projeto para poder estimar a duração do projeto e elaborar um cronograma inicial, depois do detalhamento dos requisitos para refinar o cronograma e após o software ter sido concluído para, com base no tamanho real, recalcular a taxa média de produtividade da organização.

(E) durante a etapa de Codificação, pois os programadores, por conhecerem o produto e a complexidade de implementar os requisitos, são os mais indicados, dentre os membros da equipe de desenvolvimento, para indicar o prazo para término do projeto e de conclusão das atividades.

[60] FGV - DPE – 2014

A engenharia de requisitos estabelece uma base sólida para o projeto e para a construção de software. Sem ela, o software resultante tem grande probabilidade de não atender às necessidades do cliente. No entanto, sabe-se que é comum, ao longo de um projeto, que os clientes e/ou fornecedores de requisitos mudem de opinião e sugiram mudanças no que foi acordado inicialmente. Neste cenário, é uma boa prática

(A) impedir que os requisitos sejam alterados durante o projeto de desenvolvimento, se necessário, com restrições previstas em contrato.

(B) implementar as solicitações de mudança de requisitos dos clientes o mais rápido possível, mesmo que o custo do projeto aumente, para que ele continue motivado e satisfeito com os prazos acordados inicialmente.

(C) apresentar a formalização dos requisitos ao cliente no início do projeto e entregar o produto final pronto o mais rápido possível para evitar maiores problemas.

(D) adotar procedimentos de gerência de requisitos que possibilitem à equipe de projeto identificar, controlar e acompanhar as necessidades e suas mudanças a qualquer momento, mesmo que isso possa alterar a data de entrega inicial, se o cliente estiver de acordo.

(E) adotar o modelo de ciclo de vida cascata e prever a criação de casos de uso na fase de levantamento de requisitos.

[74] FGV - DPE – 2014

Considere a estrutura de níveis de maturidade e capacidade do modelo de maturidade CMMI versão 1.2. O nível 2 de maturidade é composto pelas áreas de processo: Planejamento do Projeto (PP - Project Planning), Monitoração e Controle do Projeto (PMC – Project Monitoring and Control), Gerência de Requisitos (REQM – Requirements Management), Garantia da Qualidade do Processo e do Produto (PPQA – Process and Product Quality Assurance), Medição e Análise (MA – Measurement and Analysis), Gerência de Configuração (CM – Configuration Management), Gerência de Acordo com Fornecedores (SAM – Supplier Agreement Management).

Se uma organização teve seus processos avaliados como aderentes às áreas de processo de nível 2, exceto PPQA, para a qual falhou em apresentar evidências para as práticas genéricas referentes ao Objetivo Genérico 2 (GG2), é correto afirmar que essa organização

(A) está aderente ao nível 2 de maturidade do CMMI.

(B) está aderente ao nível 3 de maturidade do CMMI.

(C) está aderente ao nível 4 de maturidade do CMMI.

(D) tem as áreas de processo PP, PMC, REQM, MA, CM e SAM aderentes ao nível 2 de capacidade do CMMI e PPQA ao nível 1.

(E) tem as áreas de processo PP, PMC, REQM, PPQA, MA, CM e SAM aderentes ao nível 2 de capacidade do CMMI.

[75] FGV - DPE – 2014

Considere o cenário onde uma organização possui várias demandas para construção e manutenção de softwares e precisa contratar uma fábrica de software. Considerando um conjunto de fábricas de software concorrentes, a fábrica de software a seguir mais indicada de acordo com a forma como desenvolve software para atender a esse cenário é

(A) uma fábrica de software avaliada com sucesso no nível 2 de maturidade do CMMI há 5 anos.

(B) uma fábrica de software recentemente adquirida por uma multinacional que, em seu país de origem, foi avaliada com sucesso no nível 4 de maturidade do CMMI.

(C) uma fábrica de software que está iniciando a implantação do CMMI.

(D) uma fábrica de software que não possui nem segue um processo de software definido.

(E) uma fábrica de software recém avaliada no nível 2 de maturidade do CMMI.

[76] FGV - DPE – 2014

Considere, no contexto de um processo de desenvolvimento de software, o conjunto de atividades relacionadas a (1) garantir que determinados produtos de trabalho atendam os requisitos especificados e (2) demonstrar que um produto ou componente de produto é capaz de atender o uso pretendido quando colocado no ambiente em que deveria funcionar. Essas atividades, em relação ao CMMI, referem-se respectivamente às áreas de processo

(A) Verificação (VER - Verification) e Validação (VAL - Validation).

(B) Gerência de Requisitos (REQM - Requeriments Management) e Desenvolvimento de Requisitos (RD - Requirements Development).

(C) Planejamento do projeto (PP – Project Planning) e Medição e Análise (MA – Measurement and Analysis).

(D) Desenvolvimento de Requisitos (RD - Requirements Development) e Integração do Produto (PI - Product Integration).

(E) Verificação (VER - Verification) e Solução Técnica (TS - Technical Solution).

[77] FGV - DPE – 2014

A organização em que você trabalha deseja contratar uma fábrica de software e você é o responsável por elaborar um contrato de aquisição de software. Como você não tem experiência no assunto, decide recorrer ao livro texto do CMMI como fonte de informação. Além da área de processo Gerência de Acordo com Fornecedores (SAM – Supplier Agreement Management), a outra área de processo cujo conhecimento e boas práticas pode auxiliar nessa tarefa é

(A) Treinamento Organizacional (OT - Organizational Training), pois apresenta boas práticas de contratação de pessoal que devem ser executadas pela organização fornecedora.

(B) Gerência de Riscos (RSKM - Risk Management), por conter boas práticas relacionadas à gerência de requisitos dos projetos de desenvolvimento do fornecedor.

(C) Monitoração e Controle do Projeto (PMC - Project Monitoring and Control), pois apresenta boas práticas relacionadas à monitoração de projetos e à tomada de ações corretivas, que podem ser aplicadas ao seu projeto de aquisição de software.

(D) Definição do Processo Organizacional (OPD - Organizational Process Definition), por conter boas práticas relacionadas à definição de processos que podem ser utilizadas pela organização fornecedora ao apresentar uma proposta de fornecimento de software.

(E) Validação (VAL - Validação), por apresentar critérios úteis para avaliar o contrato antes de ele ser apresentado ao fornecedor e assinado.

[25] DOM CINTRA - FIOCRUZ - 2014

Observe as afirmativas a seguir, em relação à Análise de Ponto de Função (APF).

I. Tem como objetivo, medir as funcionalidades que o software fornece ao usuário, independente da tecnologia usada para implementação do mesmo.

II. É uma ferramenta para determinar o tamanho de pacote comprado.

III. O Arquivo de Interface Externa (AIE) é obrigatoriamente um Arquivo Lógico Interno (ALI) de outra aplicação.

Dessas afirmativas:

(A) apenas I está correta.

(B) apenas I e II estão corretas.

(C) apenas II e III estão corretas.

(D) apenas I e III estão corretas.

(E) todas estão corretas.

[26] DOM CINTRA - FIOCRUZ - 2014

De acordo com a Análise de Ponto de Função, um exemplo de entrada externa (EE) são:

(A) dados transacionais que são usados para manter um Arquivo Lógico Interno.

(B) respostas a mensagens de confirmação para executar uma função.

(C) telas de menu usadas para navegação ou seleção.

(D) dados passados entre on-line e batch dentro da mesma aplicação.

(E) dados passados entre cliente e servidor dentro da mesma aplicação.

[27] DOM CINTRA - FIOCRUZ - 2014

Um relatório solicitado pelo usuário foi considerado muito trabalhoso pelo desenvolvedor. Para facilitar a solução do problema e a geração do relatório, o desenvolvedor criou uma tabela temporária no sistema para guardar dados transitórios para a conclusão do processo. A classificação dessa tabela em

pontos de função é:

(A) arquivo de interface externo.

(B) saída externa.

(C) arquivo lógico interno.

(D) consulta externa.

(E) o arquivo não é contado.

[28] DOM CINTRA - FIOCRUZ - 2014

Um relatório de compras apresenta os seguintes dados, todos recuperados do arquivo lógico de compras: vendedor, data da compra, cliente, valor total da compra, unidade. Esse

relatório é apresentado ordenado pela data da compra. Outro relatório apresenta os mesmos dados, mas ordenado pelo cliente. Em relação ao ponto de função, a contagem é de:

(A) uma saída externa.

(B) duas saídas externas.

(C) uma consulta externa.

(D) duas consultas externas.

(E) cinco saídas externas.

[29] DOM CINTRA - FIOCRUZ - 2014

Um projeto de melhoria adiciona 52 pontos de função não ajustados para uma aplicação de 350 pontos de função ajustados. Além disso, foram adicionado mais 15 pontos de função não ajustados para funcionalidades de conversão. A contagem de ponto de função do projeto de melhoria é:

(A) 417.

(B) 52.

(C) 365.

(D) 67.

(E) 15.

[30] DOM CINTRA - FIOCRUZ - 2014

Uma aplicação tem 620 pontos de função não ajustados.

Um projeto de manutenção incluiu 4 ALI (arquivo lógico interno) de complexidade baixa, 3 EE (entrada externa) de complexidade media, 2 SE (saída externa) de complexidade alta. O tamanho, em ponto de função, da aplicação após a manutenção é:

(A) 670.

(B) 680.

(C) 686.

(D) 674.

(E) 671.

[34] DOM CINTRA - FIOCRUZ - 2014

Dentre as alternativas a seguir, marque aquela que NÃO representa um dos 12 princípios do desenvolvimento ágil de software.

(A) software funcionando é a principal medida do progresso.

(B) as melhores ideias surgem de equipes auto-organizadas.

(C) documentos precisos e detalhados formam a base para o

sucesso do desenvolvimento.

(D) modificações nos requisitos, mesmo que tardias, são bemvindas.

(E) responder a mudanças mais do que seguir um plano.

[35] DOM CINTRA - FIOCRUZ - 2014

A Implantação da Função de Qualidade (IFQ) identifica três tipos de requisitos. Dentre os requisitos abaixo, o que está relacionado como “requisitos que estão implícitos no sistema, embora possam ser fundamentais para o cliente” é o:

(A) normal.

(B) excitante.

(C) reverso.

(D) esperado.

(E) funcional.

[37] DOM CINTRA - FIOCRUZ - 2014

A prototipagem de sistemas é utilizada, principalmente, para demonstrar os requisitos de um sistema. Em relação à prototipagem, é correto afirmar que:

(A) na engenharia de requisitos, a prototipagem é empregada para gerar protótipos de classes.

(B) a prototipagem facilita a compreensão, por parte dos *stakeholders*, do sistema de software a ser desenvolvido, no levantamento e validação de requisitos.

(C) a prototipagem reduz a ambiguidade, consistência e falta de compreensão, atuando na escolha de requisitos.

(D) a prototipagem modifica, de uma forma organizada, especificações e implementações não realizadas.

(E) os protótipos podem ser criados para explorar múltiplas questões após o desenvolvimento do software.

[39] DOM CINTRA - FIOCRUZ - 2014

São funções de um testador de software, EXCETO:

(A) ter atitude de questionar todos os aspectos relacionados ao software.

(B) possuir um bom entendimento do processo de desenvolvimento e do produto a ser gerado, além da

habilidade de indicar possíveis erros e falhas.

(C) gerenciar os projetos de software.

(D) investigar um software a fim de fornecer informações sobre sua qualidade.

(E) observar o produto a ser desenvolvido e questionar sua validade.

[40] DOM CINTRA - FIOCRUZ - 2014

A técnica de teste de software que seleciona caminhos de teste com base na definição e o uso das variáveis de um programa é:

(A) o teste de ciclo.

(B) o teste de fluxo de dados.

(C) a análise do valor-limite.

(D) o particionamento de equivalência.

(E) o teste de condição.

[45] DOM CINTRA - FIOCRUZ - 2014

O modelo de processo de desenvolvimento de software que foi concebido com o objetivo de combinar as atividades de desenvolvimento e gerenciamento de riscos, de modo a minimizar e controlar os riscos é o:

(A) em espiral.

(B) em V.

(C) em cascata.

(D) transformacional.

(E) orientado a objetos.

[50] DOM CINTRA - FIOCRUZ - 2014

Em relação aos tipos de requisitos de software, avalie se são verdadeiras (V) ou falsas (F) as afirmações a seguir:

I) Requisitos de usuário definem, detalhadamente, as funções, os serviços, especificando o comportamento interno e restrições operacionais do sistema.

II) Requisitos funcionais são declarações de serviços que o sistema deve fornecer, como o sistema deve reagir a entradas específicas e como deve se comportar em determinadas situações. As funções requeridas pelo usuário devem estar definidas.

III) Requisitos de domínio são provenientes do domínio da aplicação do sistema e refletem as características e as restrições desse domínio.

IV) Requisitos de sistema são declarações, em uma linguagem natural com diagramas, de quais serviços são esperados do sistema.

As afirmativas I, II, III e IV são respectivamente:

(A) F, V, V e F.

(B) V, V, F e F.

(C) V, F, V e V.

(D) V, V, F e V.

(E) F, F, V e F.

[51] DOM CINTRA - FIOCRUZ - 2014

Um exemplo de requisito funcional é:

(A) a interface com o usuário deve ser organizada em abas e menus.

(B) o tempo de resposta máximo do sistema a qualquer ação do usuário deve ser de 5s.

(C) o sistema deve fornecer dois idiomas de apresentação de mensagens e textos.

(D) caso o produto possua uma quantidade máxima permitida por compra, esse limite deve ser imposto pelo sistema durante uma compra.

(E) o sistema deve ser desenvolvido usando ferramentas de software livre.

[53] DOM CINTRA - FIOCRUZ - 2014

São papéis do facilitador na elicitação de requisitos, EXCETO:

(A) estabelecer e reforçar as regras do encontro.

(B) manter a agenda elaborada.

(C) estabelecer o tom profissional e objetivo do encontro.

(D) controla comportamentos de ruptura ou contraproducentes.

(E) facilitar o processo de decisão, inserindo mudanças pontuais na documentação.

**Gabarito**

**AOCP**

45 – C

61 – A

**BIORIO**

28 – A

**CESGRANRIO**

36 – D

39 – C

40 – E

**CESPE**

71 – E

72 – E

73 – C

74 – C

75 – E

76 – E

78 – E

84 – E

85 – C

91 – C

92 – E

93 – E

94 – C

95 – E

107 – C

108 – E

**FCC**

31 – E

32 – A

41 – D

56 – D

**FGV**

56 - D

57 – C

58 – B

59 - D

60 – D

74 – D

75 – E

76 – A

77 – C

**DOM CINTRA**

25 – E

26 – A

27 – E

28 – C

29 – D

30 – D

34 – C

35 – D

37 – B

39 – C

40 – B

45 – A

50 – A

51 – D

53 - E