**Pedro Henrique Mequelim da Silva**

Aula 01 – Atividades de Apoio

1. Qual atividade de apoio da Engenharia de Software é responsável por assegurar que os processos estão sendo seguidos corretamente e que os produtos estão sendo desenvolvidos de acordo com os padrões e procedimentos estabelecidos?
2. Auditoria.
3. Gerenciamento de Configuração.
4. Revisão Conjunta.
5. Documentação.
6. Garantia de Qualidade de Software.

**Resposta: A**

**Justificativa da questão:**

A atividade de apoio da Engenharia de Software responsável por assegurar que os processos estão sendo seguidos corretamente e que os produtos estão sendo desenvolvidos de acordo com os padrões e procedimentos estabelecidos é a Auditoria.

A Auditoria é responsável por avaliar e verificar se os processos e produtos estão em conformidade com as normas e diretrizes estabelecidas, identificando possíveis não conformidades e recomendando ações corretivas quando necessário.

Conforme destacado por Pressman, em “*Engenharia de Software: Uma Abordagem Profissional*” (2014), a auditoria é uma atividade fundamental para garantir a qualidade e a conformidade dos produtos e processos de software.

Aula 02 – Fundamentos da Qualidade de Software

1. Qual das seguintes afirmações sobre requisitos de software e qualidade é correta?
2. Os requisitos explícitos são menos importantes que os requisitos implícitos para avaliar a qualidade do software.
3. O software só precisa atender aos requisitos explícitos para ter qualidade.
4. Os requisitos explícitos são os únicos requisitos relevantes para avaliar a qualidade do software.
5. O software deve atender tanto aos requisitos explícitos quanto aos implícitos para ter qualidade.
6. Os requisitos implícitos são opcionais e não impactam na qualidade do software.

**Resposta: D**

**Justificativa da questão:**

De acordo com o livro “*Introduction to Software Testing*”, de Paul Ammann e Jeff Offutt, a qualidade de um software é determinada pela sua capacidade de atender aos requisitos explícitos e implícitos.

Os requisitos explícitos são aqueles que estão documentados e são diretamente comunicados pelos stakeholders, enquanto os requisitos implícitos são aqueles que *não* são expressamente declarados, mas são esperados pelos usuários ou derivados do contexto do sistema.

O atendimento apenas aos requisitos explícitos pode levar a um software funcional, mas não necessariamente satisfatório para os usuários ou adequado ao contexto em que será utilizado. Portanto, para garantir a qualidade do software de forma abrangente, é fundamental considerar e atender tanto aos requisitos explícitos quanto aos implícitos.

Aula 03 – Normalização

1. Qual das seguintes afirmações sobre a Normalização no contexto da disciplina de Testes e Qualidade de Software está correta?
2. A Normalização não possui relevância para a gestão da qualidade de software, pois é focada apenas em processos industriais gerais.
3. A Normalização é um processo exclusivo de documentação e não influencia diretamente na qualidade do software.
4. A Normalização é fundamental para a gestão da qualidade de software, pois oferece diretrizes e requisitos mínimos, como os definidos na família de normas ISO 9000.
5. A Normalização é aplicável apenas em grandes empresas e não tem impacto em empresas de pequeno porte.
6. A Normalização é um conceito ultrapassado e não se alinha com as práticas modernas de desenvolvimento de software.

**Resposta: C**

**Justificativa da questão:**

A ISO 9000 é uma família de normas que oferecem um conjunto de diretrizes e requisitos mínimos para gestão da qualidade e garantia de qualidade.

Segundo o livro “*Quality Software Management: Systems Thinking*”, de Gerald M. Weinberg, a normalização é essencial para estabelecer padrões e processos consistentes na gestão da qualidade de software. Ela fornece uma estrutura para organizações implementarem práticas eficazes de gerenciamento de qualidade, ajudando a garantir que os produtos e serviços atendam aos requisitos e expectativas dos clientes.

Além disso, o livro “*Software Testing and Quality Assurance: Theory and Practice*”, de Kshirasagar Naik e Priyadarshi Tripathy, também destaca a importância da normalização para garantir a conformidade com padrões reconhecidos internacionalmente, como os da ISO 9000, que são amplamente reconhecidos e adotados por organizações em todo o mundo.

1. Qual das seguintes afirmações sobre a Qualidade do Processo está correta no contexto de Testes e Qualidade de Software?
2. O CMMI (Capability Maturity Model Integration) é um modelo de qualidade de processo específico para o setor automotivo, não sendo aplicável em outros contextos.
3. ISO 12207 é uma norma internacional que define um ciclo de vida genérico para o desenvolvimento de software, mas não aborda aspectos relacionados à qualidade do processo.
4. A ISO 12207 é uma norma voltada exclusivamente para a gestão de projetos de software, não abordando aspectos relacionados à qualidade do processo.
5. O SPICE (Software Process Improvement and Capability Determination) é um modelo que oferece diretrizes para melhorar a capacidade dos processos de desenvolvimento de software e está alinhado com o CMMI.
6. O CMMI é um modelo de referência que se concentra principalmente na qualidade do produto final, negligenciando a qualidade dos processos internos.

**Resposta: D**

**Justificativa da questão:**

De acordo com o livro “Software Engineering: A Practitioner's Approach”, de Roger S. Pressman, o SPICE (*Software Process Improvement and Capability Determination*) é um modelo de referência que visa melhorar a capacidade dos processos de desenvolvimento de software. Ele fornece diretrizes e práticas para avaliar e aprimorar a qualidade dos processos, ajudando as organizações a alcançar níveis mais elevados de maturidade.

Além disso, o CMMI (*Capability Maturity Model Integration*), conforme descrito no livro “*CMMI for Development: Guidelines for Process Integration and Product Improvement*”, de Mary Beth Chrissis, Mike Konrad e Sandy Shrum, é um modelo de referência amplamente reconhecido para avaliar e aprimorar a maturidade dos processos de desenvolvimento de software.

O CMMI e o SPICE estão alinhados em termos de objetivos e práticas, sendo que o SPICE oferece uma abordagem mais detalhada para a melhoria dos processos.

Aula 04 – Níveis de otimização

1. Qual das seguintes afirmações sobre Maturidade de Software está correta?
2. A maturidade de software é uma medida estática que avalia apenas a capacidade de definição dos processos de desenvolvimento.
3. A maturidade de software não está relacionada ao gerenciamento, medição, controle ou execução dos processos de forma eficaz.
4. A maturidade de software reflete a capacidade de um processo ser definido, gerenciado, medido, controlado e executado de maneira eficaz.
5. A maturidade de software é uma medida subjetiva que varia de acordo com a experiência dos desenvolvedores envolvidos.
6. A maturidade de software é uma medida exclusiva para avaliar a qualidade do produto final de um software.

**Resposta: C**

**Justificativa da questão:**

Conforme descrito no livro “*Software Engineering: A Practitioner's Approach*”, de Roger S. Pressman, a maturidade de software é uma medida que reflete o quanto um processo pode ser definido, gerenciado, medido, controlado e executado de forma eficaz. Isso significa que quanto mais maduro um processo, melhor ele é capaz de atender às necessidades do negócio, reduzir riscos e entregar produtos de alta qualidade de forma consistente.

Além disso, o livro “*Managing the Software Process*”, de Watts S. Humphrey, aborda a importância da maturidade de software como uma medida objetiva e quantificável da capacidade dos processos de software de uma organização. Modelos como o CMMI (Capability Maturity Model Integration) são exemplos de estruturas que ajudam a avaliar e melhorar a maturidade dos processos de desenvolvimento de software.

Portanto, a maturidade de software não se limita apenas à capacidade de definição dos processos, mas abrange aspectos como gerenciamento, medição, controle e execução eficaz dos processos, sendo uma medida fundamental para avaliar a excelência e a eficiência operacional das práticas de desenvolvimento de software em uma organização.