PUC Minas

Quizz 1

INFORMAÇÕES DOCENTE						
CURSO: ENGENHARIA DE SOFTWARE	**DISCIPLINA:** PROJETO DE SOFTWARE	**TURNO**	MANHÃ	TARDE	NOITE	**PERÍODO/SALA:** 4º
			x		x	
PROFESSOR (A): João Paulo Carneiro Aramuni						

1) [SANEAGO, 2018] Dentro dos bons princípios de projeto e construção de software, a Lei de Démeter diz que "um método deve enviar mensagens somente para objetos a que ele tem acesso direto". Essa lei tem como objetivo:

a) facilitar a criação de dependência entre as classes.
b) aumentar a coesão.
c) aumentar a quantidade de casos de teste.
d) diminuir o acoplamento.

2) [SEPLAD, 2021] Em engenharia de software, coesão e acoplamento são princípios que se deve levar em consideração na busca pela qualidade e facilidade de manutenção e evolução dos sistemas. O que o Engenheiro deve buscar é um conjunto que leve o software para uma situação de:

a) Baixa coesão e forte acoplamento.
b) Alta coesão e fraco acoplamento.
c) Baixa coesão e fraco acoplamento.
d) Alta coesão e forte acoplamento.

3) [IF-CE, 2021] No que diz respeito à manutenção e reengenharia de software, um termo define o processo de alterar o código-fonte, de modo que não altere o comportamento externo e ainda melhore a sua estrutura interna. É uma técnica disciplinada de limpar e organizar o código, e por consequência, minimizar a chance de introduzir novos bugs. Esse termo é conhecido como:

a) recodificação.
b) elicitação.
c) replicação.
d) refatoração.

4)[UFRN, 2018] Considere a situação em que uma classe A é superclasse das classes B e C e que, tanto B quanto C possuem um método M com a mesma assinatura e código. Nessa situação, a operação de refatoração (refactoring) de código mais apropriada a ser aplicada é:

a) Extract method.
b) Extract module.
c) Pull up method.
d) Inline method.

Gabarito: D - B - D - C

Link útil: https://engsoftmoderna.info/cap7.html#anti-padrões-arquiteturais

Big ball of mud (ou grande bola de lama) é um anti-padrão arquitetural proposto por Brian Foote e Joseph Yoder:

"Esse anti-padrão (...) descreve sistemas nos quais qualquer módulo comunica-se com praticamente qualquer outro módulo, como mostra a próxima figura. Ou seja, um big ball of mud não possui uma arquitetura definida. Em vez disso, o que existe é uma explosão no número de dependências, que dá origem a um espaguete de código. Consequentemente, a manutenção do sistema torna-se muito difícil e arriscada. (...) *Em um artigo publicado em 2009 na revista IEEE Software, Santonu Sarkar e mais cinco colegas — na época consultores da empresa indiana InfoSys — descrevem uma experiência de modularização de um grande sistema bancário. O sistema nasceu no final da década de 90 e desde então aumentou seu tamanho em 10 vezes: passou de 2.5 milhões para mais de 25 milhões de linhas de código! Segundo os autores, os times de desenvolvimento do sistema contavam com várias centenas de engenheiros. Apesar de não usarem o termo, o artigo caracteriza a arquitetura desse sistema bancário como uma big ball of mud. Por exemplo, os autores mencionam que apenas um diretório, chamado sources, possuía quase 15 mil arquivos"*

Marco Tulio Valente, Engenharia de Software Moderna. Capítulo 7.8 Anti-padrões Arquiteturais.

Discuta *quais são os problemas para manter esse sistema.*