Universidade Federal do Paraná
Departamento de Informática
Ciência da Computação

Nome:_____Assinatura: _____GRR:_____

Atenção! Recomenda-se não marcar respostas de questão objetiva ao acaso!
A nota em cada item das questões objetivas será igual a: 1,00 ponto, caso a resposta do aluno esteja em concordância com o gabarito da prova; 0,50 ponto negativo, caso a resposta do aluno esteja em discordância com o gabarito da prova; 0,00 ponto, caso não haja marcação ou haja rasura.

[Classifique as atividades de metodologia de desenvolvimento de software a seguir em:
(G) Genéricas (podem ser utilizadas para o desenvolvimento de programas pequenos e simples, para a criação de grandes aplicações para a Internet e para a engenharia de grandes e complexos sistemas baseados em computador) **ou**
(A) de Apoio (ajudam a equipe a gerenciar, controlar o progresso, a qualidade, as mudanças e o risco)]

( ) Administração de riscos
( ) Comunicação
( ) Construção
( ) Controle e acompanhamento do projeto
( ) Garantia de qualidade de software
( ) Gerenciamento da configuração de software
( ) Gerenciamento da reusabilidade
( ) Implantação (Emprego)
( ) Medição
( ) Modelagem
( ) Planejamento
( ) Preparo e produção de artefatos de software
( ) Revisões técnicas
( ) Teste de software

[Assinale C(certo) ou E(errado)]

O modelo de processo de *software* caracterizado por iterar as atividades de especificação, desenvolvimento e validação, pode ser um...**[Assinale C(certo) ou E(errado)]**

( ) modelo de métodos formais
( ) processo unificado
( ) modelo evolucionário
( ) modelo em V
( ) modelo incremental
( ) modelo espiral
( ) modelo em cascata
( ) modelo de prototipação

[Assinale C(certo) ou E(errado)]

Segundo a IEEE Computer Society, a engenharia de *software* é a aplicação de uma abordagem sistemática, disciplinada e quantificável ao desenvolvimento, à operação e à manutenção de *software*, isto é, a aplicação da engenharia ao *software*. Acerca dos princípios da engenharia de *software*, assinale a opção correta.
[Assinale C(certo) ou E(errado)]
( ) A engenharia de requisitos de um *software*, em geral, precede a engenharia dos requisitos do sistema de informações no qual o *software* será usado.

[Assinale C(certo) ou E(errado)]
( ) A manutenção de *software* é uma atividade da engenharia de *software* que necessita do emprego de

recursos que drenam cerca de 15% do investimento total em um *software* durante todo o seu ciclo de vida.

( ) É desejável que o valor da coesão e o do acoplamento, duas importantes propriedades da arquitetura de um *software*, sejam maximizados durante a engenharia de *software.*

( ) O uso de metodologias de desenvolvimento de sistemas tem como objetivo garantir que o *software*, depois de desenvolvido, não possa sofrer alteração em sua estrutura nem em seu código para não serem modificados os requisitos.
( ) O processo unificado (PU) é um processo iterativo para a análise de projetos orientados a objetos, no qual o trabalho e as iterações são organizados em três fases principais: concepção, elaboração e construção.
( ) A metodologia *Scrum* é facilitada por um *scrum master*, que atua como um mediador entre a equipe e qualquer influência desestabilizadora, além de assegurar que a equipe esteja utilizando corretamente as práticas de Scrum, motivando e mantendo o foco na meta da *sprint.*
( ) No método de Programação Extrema (XP - *eXtreme Programming*) é encorajada a refabricação para modificar um *sofware* sem alterar o comportamento externo do código.
( ) O modelo em espiral, que descreve o processo de desenvolvimento de um *software*, apresenta uma espiral em que cada *loop* representa uma fase distinta desse processo. A ausência de risco nesse modelo o diferencia dos demais modelos de *software*.
( ) *Rapid Application Development* (RAD) é um modelo de processo de software incremental que enfatiza um ciclo de desenvolvimento longo, com o uso de uma abordagem de construção baseada no usuário. Nesse modelo, três das principais fases são abrangidas pelas modelagens: do negócio, dos dados e dos processos.
( ) O gerente geral de projetos da empresa decidiu, junto a um cliente, realizar algumas modificações nos requisitos de um produto de *software* que já se encontrava na fase de testes e comprometeu-se a incluir tais requisitos na próxima liberação do produto. Essa decisão permite inferir que o modelo de desenvolvimento de *software* empregado não é do tipo cascata.
( ) Na área de projeto de software, também conhecida como *design* de software, o software começa a ser implementado e validado pelos programadores. (CESPE-SERPRO, 2008)
( ) O levantamento de requisitos é importante, porém não é fundamental, pois recomenda-se avançar o quanto antes para as demais atividades que permitam uma visualização do software e reduzam a ansiedade do cliente em ver algo pronto. (CESPE-SERPRO, 2008)
( ) A diferença entre verificação e validação reside no fato de que a primeira se refere ao conjunto de atividades que garante que o *software* realiza corretamente uma função específica, enquanto a segunda se refere a um conjunto diferente de atividades que garante que o *software* que foi construído é rastreável às exigências do cliente.
( ) O teste de *software*, tendo em vista identificar erros de codificação, inclui, entre outros aspectos, verificar a ocorrência de erros de ortografia, o uso adequado das convenções da linguagem e se as constantes de código estão corretas.
( ) O teste de validação/aceitação tem por finalidade encontrar defeitos e inconsistências no programa com relação a sua especificação. (CESPE, TRE-ES, 2011)
( ) Nos testes de caixa branca, o código-fonte do programa é usado para identificar testes de defeitos potenciais, particularmente no processo de validação, o qual demonstra se um programa atende a sua especificação. (CESPE-ABIN,2010)
(FCC,TRT,2011) O teste de integração, principalmente: **[Assinale C(certo) ou E(errado)]**

( ) verifica se o sistema atinge a carga projetada, quando processados todos os módulos em conjunto
( ) verifica se as interfaces dos módulos se comportam de acordo com sua especificação
( ) verifica se os módulos do sistema estão de acordo com os requisitos não funcionais
( ) é um processo de teste de caixa-preta no qual os testes são derivados da especificação do sistema, cujo comportamento pode ser determinado por meio do estudo das entradas e saídas do sistema

[Assinale C(certo) ou E(errado)]

( ) Uma abordagem para o projeto de casos de teste consiste em identificar as partições de equivalência. Uma partição de equivalência de entrada contém conjuntos de dados que são processados de modo equivalente.
( ) No teste estrutural usa-se o conhecimento da estrutura do programa. O teste de caminho é um teste estrutural no qual se procura exercitar os caminhos percorridos ao se executar o programa.
(CESGRANRIO, Petrobrás, 2010) Testar é uma disciplina de suma importância para a engenharia de software. A

literatura divide os tipos de testes em duas grandes categorias: teste de caixa preta e teste de caixa branca. Sobre esta classificação, pode-se afirmar que: **[Assinale C(certo) ou E(errado)]**
( ) testes de interfaces são classificados como de caixa branca;
( ) testes de caixa preta são também chamados de teste comportamental ou funcional;
( ) testes de caixa preta são complementares aos testes de caixa branca, uma vez que contemplam diferentes classes de erros.

[Assinale C(certo) ou E(errado)]

( ) Projetar o processo de teste criando casos de teste, rotinas de teste e, eventualmente, desenvolvendo programas que fazem o teste de forma automática, é uma das principais atividades do processo de teste em um ciclo de vida de um projeto qualquer.
( ) Corrigir erros e produzir o manual de especificação do uso do sistema que é utilizado para ensinar o usuário a manipular o produto final do software, é uma das principais atividades do processo de teste em um ciclo de vida de um projeto qualquer.
( ) Testar as unidades de software na fase de operação e manutenção do sistema e utilizar os resultados como métricas para eventuais ajustes em projetos anteriores, é uma das principais atividades do processo de teste em um ciclo de vida de um projeto qualquer.
( ) A atividade de teste é o processo de executar um programa com a intenção de descobrir um erro.
( ) A atividade de teste pode comprovar a ausência de erros.
( ) Um bom caso de teste é aquele que tem uma elevada probabilidade de revelar um erro ainda não descoberto.
( ) Um teste bem-sucedido é aquele que revela um erro não descoberto.

[Escolha a única opção correta] O teste que força o software a falhar de diversos modos e verifica se o retorno do processamento está dentro de limites de tempo aceitáveis é um tipo de: **[Escolha a única opção correta]**

a) Teste de Integração.
b) Teste de Estresse.
c) Teste de Recuperação.
d) Teste de Desempenho.
e) Teste de Segurança.

[Escolha a única opção correta] Um novo sistema de informação interno de uma empresa está sendo testado por um grupo restrito de usuários, fora de um ambiente controlado pelos desenvolvedores. Isso caracteriza o teste: **[Escolha a única opção correta]**
a) de unidade.
b) de usabilidade.
c) alfa.
d) beta.
e) de estresse.

[Escolha a única opção correta] Três projetos de software (X, Y e Z) de uma empresa entraram em produção no último mês. Considere a fase em que ocorreu a maior quantidade de erros, descobertos em produção, de cada projeto: **[Escolha a única opção correta]**
X: implementação
Y: requisitos
Z: instalação
Considerando-se que os projetos são extremamente similares, conclui-se que, de maneira geral, os erros de

a) Z foram provocados pela ausência de testes unitários.
b) Z são influenciados, fortemente, pela qualidade técnica dos testadores.
c) X apresentam o maior custo de correção.
d) Y apresentam o menor custo de correção.
e) Y seriam mais baratos se fossem detectados no início.

NA PRÓXIMA PÁGINA!
[Seja conciso e direto. Você deve ser capaz de responder a essas perguntas no espaço dado. A resposta será anulada caso seja o espaço ocupado pela mesma seja maior que o espaço dado.]

[Seja conciso e direto. Você deve ser capaz de responder a essas perguntas no espaço dado. <u>A resposta será anulada caso seja o espaço ocupado pela mesma seja maior que o espaço dado</u> pelas linhas**.**]

É possível combinar modelos de processo? Justifique sua resposta e dê um exemplo.

Muitas aplicações modernas mudam com frequência – antes de serem apresentadas ao usuário final e muitas vezes antes de serem colocadas em uso. Sugira algumas maneiras de construir software para impedir a deterioração decorrente de mudanças.