Fundação CECIERJ - Vice Presidência de Educação Superior a Distância

Curso de Tecnologia em Sistemas de Computação
Disciplina: Engenharia de Software
AP2 1° semestre de 2016

Nome –

Assinatura –

Observações:

1. Prova sem consulta e sem uso de máquina de calcular.
2. Use caneta para preencher o seu nome e assinar nas folhas de questões e nas folhas de respostas.
3. Você pode usar lápis para responder as questões.
4. Ao final da prova devolva as folhas de questões e as de respostas.
5. Todas as respostas devem ser transcritas nas folhas de respostas. As respostas nas folhas de questões não serão corrigidas.

---

1. Quais afirmativas abaixo são corretas? A seleção de uma afirmativa incorreta elimina uma afirmativa correta. Dica: temos duas afirmativas corretas. (valor 1,0 ponto).
   a. A relação de uso entre módulos em um projeto de software é uma relação estática, independente de um cenário de execução do software.
   b. No desenvolvimento de módulos de software buscamos módulos de baixa coesão e alto acoplamento.
   c. Em um diagrama de classes, um auto-relacionamento é uma associação de uma classe com ela mesma.
   d. Na UML, um diagrama de colaboração é um diagrama estático, que não representa aspectos dinâmicos do projeto do software.

   Alternativas corretas: a e c.

2. Relacione as afirmações com os conceitos associados a elas (valor 1,0 ponto):

(a) Testes unitários	(1) Com a presença do cliente, determina se o sistema funciona de acordo com suas expectativas e com os requisitos.
(b) Testes de integração	(2) Processo que verifica se os componentes ou módulos do sistema de software funcionam corretamente em conjunto e de acordo com a especificação do projeto (design).
(c) Testes de aceitação	(3) Aplicado a uma nova versão do software para verificar se esta ainda executa as mesmas funções da mesma maneira que a versão anterior.
(d) Testes funcionais	(4) Avalia o sistema quando submetido aos seus limites por um curto período de tempo.
(e) Testes de regressão	(5) Avalia o sistema para determinar se a funcionalidade descrita na especificação de requisitos executa corretamente pelo sistema já integrado.

(f) Testes de estresse	(6) Cada componente ou módulo do software é testado de forma isolada para verificar se este funciona corretamente.

Alternativas corretas: a-6; b-2; c-1; d-5; e-3; f-4

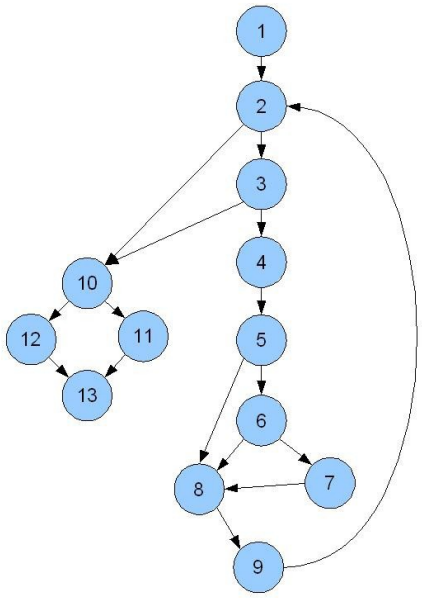
3. Explique o que é um risco em um projeto de desenvolvimento de software. Dica: o gerenciamento de riscos é uma disciplina no gerenciamento de projetos de software. (valor 1,0 ponto).

Um risco é uma incerteza que pode trazer prejuízos ou oportunidades de ganho ao processo de desenvolvimento de software. Exemplos de risco incluem a falta de experiência na equipe, incertezas sobre a maturidade de uma tecnologia, falta de capacidade do cliente em apresentar seus requisitos, entre outros.

4. Em que consiste o princípio de projeto de software da modularização e como este princípio ajuda no desenvolvimento de software? (valor: 2,0 pontos; máximo: 10 linhas).

A modularização auxilia na separação dos objetivos de um sistema por componentes, chamados *módulos*. Cada módulo contém parte do código-fonte do sistema. A modularização consiste na divisão do problema (maior) a ser resolvido pelo sistema em problemas menores, cada qual resolvido de forma independente por um módulo. Cada módulo esconde um "segredo", que é a forma como resolve o problema que lhe foi incumbido (sua implementação). Um módulo deve ser capaz de usar outro módulo sem conhecer os detalhes da sua implementação.

5. Explique o que é teste estrutural. Apresente três critérios de teste que podem ser utilizados para projetar casos de teste para este tipo de teste e exemplifique um destes critérios mostrando alguns casos de teste para o código a seguir. (modelos extraídos do material preparado pelo Prof. Andrey Ricardo Pimentel, UFPR, www.inf.ufpr.br/andrey/ci221/apresentacaoTesteEstrutural.pdf, *11/10/2015)* (valor: 3,0 pontos)




```
Procedimento media
INTERFACE ACEITA valor, min, max
INTERFACE RETORNA media, entradas, validas

var
   valor[1..100] vetor de real
   media, entradas, validas, min, max, soma: real
   i : inteiro
inicio
   i = 1
1  totalEntradas = 0
   totalValidas = 0
   soma = 0                    2              3
   enquanto valor[i]<>-999 e entradas<100 faça
   4  entradas=entradas+1
                5                  6
      se valor[i]>=min e valor[i]<=max então
   7     validas=validas+1
         soma=soma+valor[i]
      senão pule
   8  fimse
      i=i+1
   9 fimenquanto
     se validas>0 então  10
  11    media=soma/validas
  12 senão
        media=-999
  13 fimse
fim
```

Teste estrutural, ou teste caixa aberta, utiliza a informação sobre a estrutura do software para o planejamento dos testes. Neste sentido, a organização utilizada para a construção do software pode ser servir para apoiar os diferentes critérios de teste. Por exemplo, o teste pode ser inspirado no fluxo de controle do software ou no fluxo de dados. Assim, podemos usar como critérios percorrer todos os nós, arcos, caminhos, uso de dados, e demais possibilidades.

No modelo apresentado, um critério possível seria todas-as-decisões. Para isso, seis casos de teste são necessários para testar a aplicação:
Caminho 1: Nós { 1, 2, 3, 4, 5, 6, 8, 9, 2, 10, 12, 13}
Caminho 2: Nós { 1, 2, 10, 12, 13}
Caminho 3: Nós { 1, 2, 3, 10, 11, 13}
Caminho 4: Nós { 1, 2, 3, 4, 5, 8, 9, 10, 11, 13 }
Caminho 5: Nós { 1, 2, 3, 4, 5, 6, 7, 8, 9, 2, 3, 11, 13}
Caminho 6: Nós { 1, 2, 10, 11, 13}

Para percorrer os caminhos, os valores de entrada de Valor[i], min e Max devem ser ajustados para satisfazer as condições de desvio do código:
Caminho 1: Valor[i] <> -999, min >= Valor[i]; Max <= Valor[i]
Caminho 2: Valor [1] = -999; min <= qualquer; Max >= qualquer
Caminho 3: Intangível. Como fazer entradas > 100?
Caminho 4: Valor[i] <> -999, min <= Valor[i]; Max >= Valor[i]
Caminho 5: Valor[i] <> -999, um Valor[i] <= min; Max <= Valor[i]

Caminho 6: Intangível. Como fazer entradas > 100 antes da entrada do Loop?

6. O que representa a métrica Complexidade Ciclomática? Calcule o valor da métrica (número ciclomático) para o grafo de fluxo de controle apresentado na questão 4 (derivado do programa apresentado). O que este valor (ou a variação dele) pode representar nas perspectivas de teste e manutenção? (valor 2,0 pontos)

Esta métrica fornece uma medida quantitativa da complexidade lógica de um programa. No contexto do teste estrutural, seu valor define o número de caminhos independentes e nos fornece o número máximo de casos de teste que garantem que todos os comandos tenham sido executados pelo menos uma vez, o que fornece uma expectativa de esforço de teste e manutenção.

Seu calculo se dá através da identificação do número de regiões no grafo de fluxo do programa, através da fórmula:

V(G)=E-N+2
E: número de arcos
N: número de nós
ou,

V(G) = P +1
P: número de nós predicados (decisões)

Neste caso, o valor da Métrica é 6 (17 arcos, 13 Nós) ou (5 Nós Predicados: 2, 3, 5, 6 e 10)

Boa Prova !