Curso: Engenharia de Software	Série: 2	Turma:	Turno: Noturno
Professor(a): João Choma Neto	Horário: 19:00 - 22:10		
Acadêmico (a):			RA:
Disciplina: : Teste e Qualidade de Software			Data:

Prova	Prova Prática	Atividades de estudo programadas (AEP)	Prova integrada	Nota final do bimestre
5,0	3,0	1,0	1,0	

INSTRUÇÕES PARA REALIZAÇÃO DA PROVA:

⇒ Os dados do cabeçalho deverão ser preenchidos com letra maiúscula. E as questões deverão ser respondidas com letra legível.

⇒ É vedado, durante a prova, o porte e/ou o uso de aparelhos sonoros, fonográficos, de comunicação ou de registro eletrônico ou não, tais como: notebooks, celulares, tabletes e similares.

⇒ A prova é individual e sem consulta, deverá ser respondida a caneta azul ou preta. Prova escrita a lápis não dá direito à revisão. Não é permitido o uso de corretivo.

⇒ É obrigatória a permanência do acadêmico 1 (uma) hora em sala de aula após o início da prova.

⇒ Não será permitida a entrada na sala de aula após 10 minutos do início da prova.

⇒ É obrigatória a assinatura da lista de presença impressa na qual constam RA, nome e curso.

⇒ O valor de cada questão está ao lado da mesma.

⇒ Todas as respostas devem constar no espaço destinado e autorizado pelo professor, à resposta.

⇒ Em caso de qualquer irregularidade comunicar ao Professor ou fiscal de sala.

⇒ Ao término da prova, levante o braço e aguarde atendimento do professor ou do fiscal.

⇒

1ºbim.		2ºbim.		1ªsub.		3ºbim.		4ºbim.		2ªsub.	

Orientação

Abaixo, você encontrará a tabela do Gabarito para resposta e inserção das alternativas das questões objetivas. Preencha esta tabela de acordo com a sua resposta

Gabarito das Objetivas							
Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8

Questão 01 (0,2 pontos) O teste de software desempenha um papel fundamental no ciclo de desenvolvimento de software, ajudando a identificar e corrigir erros, garantindo que um programa atenda às expectativas e funcione de maneira confiável. É importante entender a definição precisa do teste de software para garantir uma aplicação eficaz desse processo.

Das alternativas a seguir, marque a que estiver correta em relação à definição de teste de software:

a. É todo procedimento que garante que o programa não possui erros.
b. É todo e qualquer procedimento que ajuda a determinar se o programa será um sucesso de vendas.
c. É todo procedimento que garante que o programa não precisará receber manutenções.
d. É todo e qualquer procedimento que ajuda a determinar se o programa atinge as expectativas para as quais foi criado.
e. É todo procedimento que garante que o programa será popular entre os usuários.

Questão 02 (0,2 pontos) - No processo de desenvolvimento de software, é importante compreender a natureza e os objetivos dos testes de software. Os testes não podem garantir que um software esteja completamente livre de defeitos em todas as circunstâncias, mas desempenham um papel crucial na identificação e mitigação de problemas. Portanto, a definição correta do papel dos testes de software é essencial para avaliar sua eficácia.

Indique a alternativa correta que define adequadamente o papel dos testes de software:

a. Os testes não conseguem demonstrar que o software está livre de defeitos ou que vai se comportar de acordo com a sua especificação em qualquer circunstância.
b. Os testes servem somente para demonstrar que o software está livre de defeitos e que vai se comportar de acordo com a sua especificação.
c. Os testes provam que o software está livre de defeitos ou que vai se comportar de acordo com a sua especificação em cenários alternativos.
d. Os testes precisam demonstrar que o software está livre de defeitos ou senão considerados insuficientes.
e. Os testes são desnecessários no desenvolvimento de software, pois a especificação é suficiente para garantir a qualidade do programa.

Questão 03 (0,2 pontos) - A automação de testes é uma prática essencial no desenvolvimento de software moderno, que oferece várias vantagens, incluindo a eficiência na detecção de erros e a economia de tempo. No entanto, a implementação adequada da automação de testes requer um entendimento claro de seus princípios e práticas. Vamos analisar cada uma das afirmações em relação à automação de testes.

Sobre a automação de testes, analise cada uma das afirmações:

I - É recomendado utilizar um framework de automação de testes, como JUnit, para escrever e executar os testes.

II - A automação executa todos os testes criados pelo desenvolvedor, embora isso não consiga atender aos critérios da automatização.

III - Vários testes podem ser executados em série, e um relatório indica quais regiões do código necessitam da atenção do desenvolvedor/testador.

Agora, selecione a alternativa que corresponde às afirmações corretas:
a) I e II, somente
b) I, somente
c) I e III, somente
d) Todas as afirmações
e) III, somente

Questão 04 (0,2 pontos) – Em um projeto de software que implementa as operações básicas de uma calculadora como soma, subtração, multiplicação e divisão, são implementados algumas classes que utilizam métodos e atributos. Analise o código a seguir e marque a opção que explica de maneira completa o resultado produzido por ele.

```
public class Calculadora Teste {
        private Calculadora calculadora;
…
```

a) Cria uma classe chamada "CalculadoraTeste".
b) Cria um objeto chamado "CalculadoraTeste".
c) Cria uma classe chamada "calculadora".
d) Cria uma classe chamada "CalculadoraTeste"
e) Criar um teste chamdo CalculadoraTeste

Questão 05 (0,3 pontos) – A questão envolve a avaliação de um trecho de código que utiliza um método "somar" para adicionar dois números e, em seguida, compara o resultado com um valor esperado. É importante entender como o código funciona e qual será o resultado da comparação.
Sabendo que o método "somar" realiza a soma de dois números, quando o trecho de código a seguir for executado, qual será o resultado obtido:

```
int resultadoEsperado = 10;
float resultado = calculadora.somar(5.2, 5.2);

if (resultado == resultadoEsperado) {
    System.out.println("Passou.");
} else {
```

```
    System.out.println("Falhou.");
}
```

Selecione a alternativa que representa o resultado da execução do código:

a) Impressão da frase "Teste falhou".
b) O programa não será executado por erro na criação das variáveis.
c) Impressão da frase "Teste OK!".
d) O programa não será executado por erro na forma de invocar o método.
e) Impressão da frase "Passou."

Questão 06 (0,3 pontos) – A abordagem incremental de teste é uma prática comum no desenvolvimento de software, onde os testes são realizados em diferentes estágios do ciclo de vida do desenvolvimento. É importante entender a sequência correta de testes nessa abordagem.

Uma estratégia de teste que é preferida pela maioria das equipes de software assume uma visão incremental do teste. Sobre essa visão, marque a opção com a sequência correta de testes. Indique a alternativa CORRETA.

a) Testes de integração, teste das unidades e testes que usam o sistema concluído.

b) Testes que usam o sistema concluído, teste das unidades e testes de integração.

c) Teste das unidades, testes de integração e testes que usam o sistema concluído.

d) Teste das unidades, testes que usam o sistema concluído e testes de integração.

e) A ordem dos testes na visão incremental pode variar dependendo das necessidades do projeto, não seguindo uma sequência fixa.

Questão 07 (0,3 pontos) – Os testes funcionais desempenham um papel crucial na garantia da qualidade de um software. Eles são projetados para identificar defeitos nas funcionalidades de um sistema e garantir que ele funcione conforme o esperado. Ao considerar um exemplo específico, como testar uma interface de login, os testes funcionais ajudam a verificar se todos os elementos e processos relacionados à autenticação do usuário estão operando corretamente.

Com base na contextualização acima e na análise das afirmações relacionadas aos testes funcionais, assinale a alternativa que indica corretamente os tipos de erros que podem ser identificados por esses testes:

i. Erros de comportamento ou de desempenho.

ii. Erros de interface, erros em estruturas de dados ou acesso a bases de dados externas.

iii. Funções incorretas ou faltando.

iv. Erros típicos de linguagem de programação.

v. Erros de segurança no código-fonte

a) I, II e III, somente
b) II e IV, somente
c) I e V, somente
d) Todas as afirmações
e) III, somente

Questão 08 (0,3 pontos) – Os testes de software desempenham um papel crucial no desenvolvimento de programas de computador confiáveis e eficazes. Entre os diferentes tipos de testes, os testes estruturais de caixa branca são particularmente importantes. Eles se concentram na análise da lógica interna do software, o que ajuda a garantir que o código-fonte seja robusto e que todas as partes do sistema trabalhem harmoniosamente.

Os testes estruturais de caixa branca são essenciais para a identificação de problemas na lógica interna do software. Eles permitem que os desenvolvedores compreendam como o software funciona em detalhes, garantindo que cada parte do sistema contribua para um todo coeso e eficaz. Qual das seguintes alternativas descreve melhor o teste estrutural de caixa branca em relação à atividade de teste de software?

A. Teste que foca na funcionalidade externa do software sem considerar sua implementação interna.
B. Teste que verifica a conformidade do software com requisitos funcionais específicos.
C. Teste que examina a lógica interna, estruturas de controle e fluxo de dados do software.
D. Teste que avalia a usabilidade e a experiência do usuário durante a interação com o software.
E. Teste que verifica a segurança do software contra ameaças externas.

Questão 09 (0,3 pontos) - Para criar um teste unitário eficaz é muito importante fornecer um nível de cobertura adequado aos seus propósitos. Por exemplo, podemos testar todas as linhas de código, todas as classes ou todos os métodos. Além disso, na orientação a objeto ainda é necessário testar um objeto em todos os seus estados possíveis. **RELATE** como um teste unitário deve ser implementado para que tenha sucesso no seu objetivo.

Questão 10 (0,3 pontos) - Um programa é executado sempre com uso de dados reais, e os resultados são conferidos em busca de erros, anomalias ou informações sobre os atributos não funcionais do programa. Com base nisso, **EXPLIQUE** o que é um caso de teste.

Questão 11 (0,3 pontos) - Imagine que você está prestes a comprar um novo smartphone, mas antes de tomar a decisão final, deseja saber se ele atende às suas expectativas em termos de desempenho, recursos e qualidade da câmera. Uma maneira de fazer isso é solicitar a opinião de amigos que já possuem o mesmo modelo de smartphone, ou você pode pesquisar na internet por avaliações e análises detalhadas. Isso é essencialmente um tipo de "teste de caixa preta" que você realiza para obter informações sobre o produto sem conhecer todos os detalhes internos do dispositivo. **EXPLIQUE** o teste caixa preta.

Questão 12 (0,3 pontos) - Imagine que você é um engenheiro de segurança encarregado de verificar a resistência de um cofre de alta segurança. Sua tarefa é determinar se o cofre é realmente à prova de arrombamento, de modo que nenhum invasor possa acessar seu conteúdo protegido. Para fazer isso, você não apenas deseja verificar a resistência da porta do cofre, mas também entender o funcionamento interno do mecanismo de travamento para identificar quaisquer pontos fracos que um invasor poderia explorar. Esse processo de avaliação minuciosa interna é semelhante ao que acontece no teste caixa branca. EXPLIQUE o teste caixa branca.

Questão 13 (0,3 pontos) - O teste estrutural é composto por critérios de teste. **EXPLIQUE** o que é um critério de teste. Inclua exemplos.

Questão 14 (0,5 pontos) – **DESENHE** um grafo de fluxo de controle para o seguinte código:

```
public class BubbleSort {
    public static void main(String[] args) {
        int[] arr = {5, 3, 1, 6, 2, 8, 4};

        boolean swapped;
        do {
            swapped = false;
            for (int i = 0; i < arr.length - 1; i++) {
                if (arr[i] > arr[i + 1]) {
                    // Troca os elementos de posição
                    int temp = arr[i];
                    arr[i] = arr[i + 1];
                    arr[i + 1] = temp;
                    swapped = true; // Indica que houve uma troca
                }
            }
        } while (swapped);

        // Imprime o array ordenado
        for (int num : arr) {
            System.out.print(num + " ");
        }
    }
}
```

Questão 15 (0,5 pontos) – **CRIE** casos de teste para testar o seguinte código. CORRIJA o código caso encontre algum defeito:

```
public static void main(String[] args) {
        int x = 5;
        int y = 10;
        int z = 0;

        if (x > y) {
            z = x + y;
        } else if (x < y) {
            for (int i = 0; i < 3; i++) {
                z += i;
            }
        } else {
            z = x * y;
        }

        System.out.println("O valor de z é: " + z);
    }
}
```

Questão 16 (0,5 pontos) – **CRIE** casos de teste para testar o seguinte código. CORRIJA o código caso encontre algum defeito:

```
public class CalculadoraMedia {
    public double calcularMedia(double[] valores) {
        return sum(valores) / valores.length-1;
    }

    public double sum(double[] valores) {
        double soma = 0;
        for (double valor : valores) {
            soma += valor;
        }
        return soma;
    }
}
```