

**Quizz 6**

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

## **Modelagem Funcional com Contratos e Object Constraint Language (OCL)**

### **1. Conceitos Fundamentais e Responsabilidades do Contrato**

Analise as afirmações a seguir sobre os princípios do OCL e Contratos por Projeto:

- I. OCL é uma linguagem puramente declarativa usada para especificar restrições e consultas, mas não permite a criação de novas operações no modelo.
- II. As pré-condições de uma operação definem a responsabilidade do cliente (quem chama) em garantir que o sistema esteja em um estado válido para a chamada.
- III. As pós-condições garantem que o servidor (a operação) entregou o estado prometido e/ou o valor de retorno esperado.
- IV. Restrições de invariante devem ser verdadeiras em todos os momentos (incluindo durante a execução da operação) para que o objeto seja considerado válido.

Assinale a alternativa que indica as afirmações CORRETAS.

- A) Somente I, II e III estão corretas.
- B) Somente I e IV estão corretas.
- C) Somente II e III estão corretas.
- D) Somente I, III e IV estão corretas.
- E) Todas as afirmações estão corretas.

### **2. OCL Navigation e Palavras-chave de Contexto**

Sobre a sintaxe e a navegação em OCL, considere:

- I. A palavra-chave self sempre se refere à instância da classe no contexto, permitindo o acesso direto a seus atributos e a navegação de associações.

II. O operador de ponto (.) deve ser utilizado para navegar associações com multiplicidade de coleção (e.g., 0..\* ou 1..\*).

III. Para chamar uma operação da biblioteca padrão do OCL em uma coleção (e.g., size(), select), deve-se usar a notação de seta (->).

IV. A cláusula context é obrigatória e pode ser aplicada a uma classe ou a uma operação específica, mas não a um atributo.

Assinale a alternativa que indica as afirmações CORRETAS.

- A) Somente I, II e III estão corretas.
- B) Somente I e II estão corretas.
- C) Somente II, III e IV estão corretas.
- D) Somente I, III e IV estão corretas.
- E) Todas as afirmações estão corretas.

### 3. Tipos e Propriedades de Coleção

Sobre os tipos de coleções OCL (Set, Bag e Sequence), avalie:

I. Set (Conjunto) é a coleção ideal quando a ordem de inserção é importante, mas os elementos não podem ser duplicados.

II. Bag (Multiconjunto) é o único tipo de coleção que permite elementos duplicados e não possui ordem definida.

III. Sequence (Sequência) é usado quando a ordem de inserção dos elementos deve ser preservada e eles podem ser acessados por índices ou posições.

IV. A operação size() é universal e pode ser aplicada a qualquer coleção OCL para contar seus elementos.

Assinale a alternativa que indica as afirmações CORRETAS.

- A) Somente I e II estão corretas.
- B) Somente I, II e III estão corretas.
- C) Somente III e IV estão corretas.
- D) Somente I, III e IV estão corretas.
- E) Todas as afirmações estão corretas.

#### 4. Operações de Iteração e Filtro

Considere as operações de iteração e filtro essenciais na biblioteca padrão do OCL:

- I. A operação `collect()` retorna uma nova coleção, aplicando uma expressão a cada elemento da coleção original.
- II. A operação `forAll()` retorna `true` se a condição booleana for satisfeita por pelo menos um elemento da coleção.
- III. A operação `reject()` é o oposto de `select()` e retorna os elementos que não satisfazem uma determinada condição.
- IV. Para verificar se a coleção `pedidos` não está vazia, a expressão correta é `pedidos->notEmpty()`.

Assinale a alternativa que indica as afirmações CORRETAS.

- A) Somente I e II estão corretas.
- B) Somente II, III e IV estão corretas.
- C) Somente I, III e IV estão corretas.
- D) Somente I, II e IV estão corretas.
- E) Todas as afirmações estão corretas.

#### 5. Pós-Condições, `@pre` e `result`

Em relação à especificação do estado resultante de uma operação:

- I. O sufixo `@pre` pode ser usado em uma pós-condição para obter o valor de um atributo, uma expressão de navegação ou uma coleção inteira antes da execução da operação.
- II. Se a operação tiver um valor de retorno, a pós-condição deve fazer referência a ele usando a palavra-chave `result` para validar se o valor retornado está correto.
- III. Para verificar se um objeto `novItem` foi criado pela operação, a expressão correta na pós-condição é `novItem.oclIsNew()`.
- IV. O modificador `@pre` pode ser usado em pré-condições para garantir que uma operação só seja executada se uma propriedade tiver um valor inicial específico.

Assinale a alternativa que indica as afirmações CORRETAS.

- A) Somente I, II e III estão corretas.
- B) Somente I, II e IV estão corretas.
- C) Somente II, III e IV estão corretas.
- D) Todas as afirmações estão corretas.
- E) Somente I e III estão corretas.

#### 6. Invariantes e Restrições de Unicidade

Sobre a especificação de restrições globais e de unicidade:

- I. Invariantes de classe são a maneira correta em OCL de modelar restrições de unicidade em todo o sistema, como garantir que o CPF seja único para todos os Clientes.
- II. A expressão `A.allInstances()` retorna todas as instâncias existentes da classe A no sistema.
- III. A restrição `context Cliente inv: self.idade >= 18` é uma invariante válida para garantir que a idade mínima de um cliente seja 18 anos.
- IV. A operação `isUnique(expression)` garante que a expressão avaliada seja única para todos os objetos em uma coleção.

Assinale a alternativa que indica as afirmações CORRETAS.

- A) Somente I e II estão corretas.
- B) Somente II, III e IV estão corretas.
- C) Somente I, III e IV estão corretas.
- D) Todas as afirmações estão corretas.
- E) Somente III e IV estão corretas.

#### 7. Modelagem Funcional e Limitações do OCL

Sobre o papel e as limitações do OCL na especificação funcional:

- I. OCL é uma linguagem puramente de especificação e não pode ser usada para definir a lógica algorítmica de como uma operação deve ser executada (e.g., o passo a passo de um cálculo).

II. O principal benefício do uso de OCL em Modelagem Funcional é a remoção da ambiguidade e da subjetividade na especificação de requisitos.

III. OCL suporta o uso de if-then-else apenas como uma expressão condicional (que retorna um valor), não como uma instrução de controle de fluxo de execução.

IV. Contratos OCL são valiosos apenas para documentação e não podem ser utilizados por ferramentas de validação de modelos.

Assinale a alternativa que indica as afirmações CORRETAS.

- A) Somente I e II estão corretas.
- B) Somente I, II e III estão corretas.
- C) Somente II, III e IV estão corretas.
- D) Somente I, III e IV estão corretas.
- E) Todas as afirmações estão corretas.

#### 8. Navegação de Associações e Coleções (Exemplo)

Contexto: Classe Pedido (Order) associada a ItemPedido (OrderItem) por meio da role itens.

I. Para verificar o número de itens em um pedido, a expressão correta é `self.itens->size()`.

II. Para garantir que todo pedido tenha um total de itens maior que 5, a invariante seria `self.itens->size() > 5`.

III. Para verificar se todos os itens de um pedido têm preço unitário maior que 100, a expressão correta é `self.itens->forAll(i | i.preco > 100)`.

IV. Se a associação itens fosse 1..1, a expressão `self.itens.preco > 100` seria válida.

Assinale a alternativa que indica as afirmações CORRETAS.

- A) Somente I e II estão corretas.
- B) Somente I, II e III estão corretas.
- C) Somente II, III e IV estão corretas.
- D) Todas as afirmações estão corretas.
- E) Somente III e IV estão corretas.

## 9. OCL e Comparação de Estados (Pós-Condição)

Contexto: Operação adicionarItem(i: ItemPedido) na classe Pedido.

- I. A pós-condição `self.itens->includes(i)` garante que o item `i` foi adicionado à coleção de itens do pedido.
- II. A pós-condição `self.itens->size() = self.itens->size()@pre + 1` verifica se exatamente um item novo foi adicionado à coleção.
- III. A pós-condição `self.valorTotal = self.valorTotal@pre + i.preco` é válida para expressar que o valor total foi atualizado corretamente (assumindo que `valorTotal` é um atributo/operação).
- IV. Na pós-condição, é impossível verificar se a lista de itens anteriores (`self.itens@pre`) está contida na lista atual (`self.itens`), pois isso requer lógica de execução.

Assinale a alternativa que indica as afirmações CORRETAS.

- A) Somente I, II e III estão corretas.
- B) Somente I, II e IV estão corretas.
- C) Somente II, III e IV estão corretas.
- D) Todas as afirmações estão corretas.
- E) Somente III e IV estão corretas.

## 10. OCL Collections: Advanced Operations

- I. A operação `exists()` é a contraparte de `forAll()` e retorna `true` se houver pelo menos um elemento satisfazendo a condição.
- II. Para verificar se não há elementos negativos em uma coleção valores, a expressão `valores->forAll(v | v >= 0)` é a mais correta.
- III. A operação `asSet()` pode ser usada para remover duplicatas de um Bag ou Sequence e retornar os elementos como um Set.
- IV. A operação `sortedBy(expression)` pode ser aplicada a qualquer coleção (Set, Bag, Sequence) para garantir que ela seja ordenada de acordo com a expressão.

Assinale a alternativa que indica as afirmações CORRETAS.

- A) Somente I, II e III estão corretas.
- B) Somente I, II e IV estão corretas.
- C) Somente II, III e IV estão corretas.
- D) Todas as afirmações estão corretas.
- E) Somente III e IV estão corretas.

Gabarito:

1. Resposta Correta: C) Somente II e III estão corretas. (I é incorreta: OCL é usado para consultas. IV é incorreta: Invariantes podem ser violadas durante a execução da operação, mas devem ser restauradas no final).
2. Resposta Correta: D) Somente I, III e IV estão corretas. (II é incorreta: a navegação de coleções exige o operador de seta ->).
3. Resposta Correta: C) Somente III e IV estão corretas. (I é incorreta: Set não tem ordem. II é incorreta: Sequence também permite duplicatas).
4. Resposta Correta: C) Somente I, III e IV estão corretas. (II é incorreta: a descrição corresponde à operação exists(), não forAll()).
5. Resposta Correta: A) Somente I, II e III estão corretas. (IV é incorreta: @pre só pode ser usado em pós-condições para referenciar o estado antes da operação, não em pré-condições).
6. Resposta Correta: D) Todas as afirmações estão corretas.
7. Resposta Correta: B) Somente I, II e III estão corretas. (IV é incorreta: Contratos OCL são extremamente úteis para ferramentas de validação e verificação).
8. Resposta Correta: D) Todas as afirmações estão corretas.
9. Resposta Correta: A) Somente I, II e III estão corretas. (IV é incorreta: OCL possui operações de conjuntos como includesAll que permitem essa comparação de estado).
10. Resposta Correta: A) Somente I, II e III estão corretas. (IV é incorreta: Set e Bag não têm semântica de ordem; o resultado de sortedBy é sempre uma Sequence).