Iniciado em	Tuesday, 18 May 2021, 10:19
Estado	Finalizada
Concluída em	Tuesday, 18 May 2021, 11:48
Tempo empregado	1 hora 29 minutos
Notas	11,00/20,00
Avaliar	**5,50** de um máximo de 10,00(**55**%)

Questão 1

Correto

Atingiu 1,00 de 1,00

Considere a classe *Control* ligada por uma associação qualificada por *cpf* 1 para 0..1 à classe *Pessoa*. *Pessoa* é ligada por uma associação 1 para * à classe *Pedido*. *Pedido* é ligado por uma associação 1 para * à classe *Item*. *Item* é ligado por uma associação * para 1 à classe *Livro*. Considere a expressão OCL:

```
Context Control::consultaLivros(umCPF:CPF):Set<Livro>
    pre:
        pessoa[umCPF]->notEmpty()
    body:
        pessoa.item.livro
```

O que há de errado com essa expressão?

Escolha uma opção:

- a. A expressão correta seria:

```
pessoa.pedido.item.livro
```

- b. No lugar de *pessoa*[*umCPF*] deveria se escrever *pessoa->select*(*umCPF=cpf*).
- c. Consultas de sistema não podem ter precondições.
- d. Não é possível em OCL escrever expressões sequenciais com ".".
- e. No lugar de *body* deveria-se usar *post.*

Sua resposta está correta.

A resposta correta é: A expressão correta seria:

```
pessoa.pedido.item.livro
```

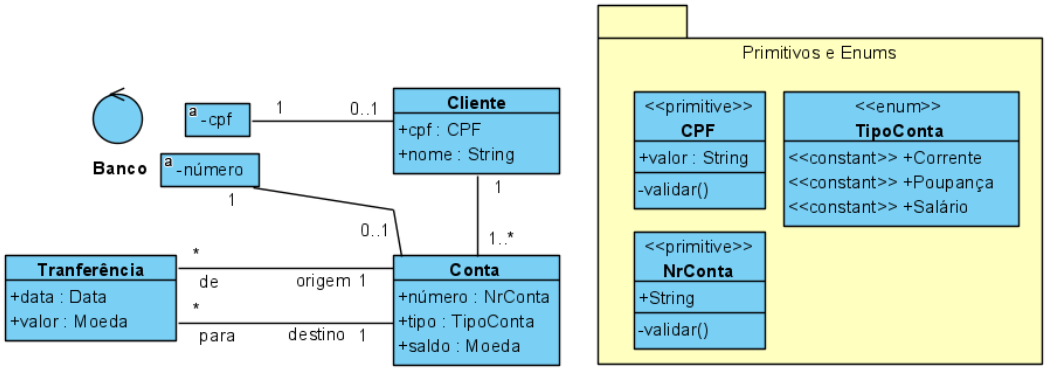
.

Questão 2

Incorreto

Atingiu 0,00 de 1,00

Considere o modelo conceitual abaixo:



Você foi solicitado a desenvolver um contrato para a consulta de sistema *consultaExtrato(cpf:CPF, nrConta:NrConta, dataInicial:Date):Tuple*. A consulta deve retornar uma tupla com dois campos:

- movimentações: uma *sequence* com as movimentações ordenadas por data, da mais antiga para a mais recente, a partir da *dataInicial*. Cada movimentação é representada por uma tupla com dois campos:
  - data: a data da transferência
  - valor: o valor positivo ou negativo transferido
- saldo: o saldo final da conta.

O número do CPF é passado como um verificador de validade para o número da conta. Assim, se o número da conta não corresponder ao CPF informado, deve ser gerada uma exceção para a consulta.

A validade do cpf e do número da conta já terão sido verificadas antes da consulta.

Não há restrições em relação à dataInicial. Se for uma data futura, a consulta retorna uma tupla com o conjunto vazio de movimentações e o saldo atual.

Assinale o contrato OCL abaixo que corretamente representa estes requisitos.

Escolha uma opção:

a.

```
Context Banco::consultaExtrato(cpf:CPF, nrConta:NrConta, dataInicial:Dat
e):Tuple
    pre:
        cliente[cpf]->notEmpty() AND
        conta[nrConta]->notEmpty()
    body:
        Tuple {
            movimentações = conta[nrConta].de.union(conta[nrConta].para)-
>select(
                 data >= dataInicial
            )->collect(
                Tuple{
                    data = data,
                    valor = valor
                }
            )
            saldo = conta[nrConta].saldo
    exception:
        not(cliente[cpf].conta->includes(conta)) implies throw('Esta cont
a não corresponde a este cliente')
```

b.

```
Context Banco::consultaExtrato(cpf:CPF, nrConta:NrConta, dataInicial:Dat
e):Tuple
    pre:
        cliente[cpf]->notEmpty() AND
        conta[nrConta]->notEmpty()
    body:
        Tuple {
            movimentações = conta[nrConta].de.union(conta[nrConta].para)-
>sortedBy(data)->select(
                 data >= dataInicial
            ).Tuple{
                data = data,
                valor = valor
            }
            saldo = conta[nrConta].saldo
    exception:
        not(cliente[cpf].conta->includes(conta)) implies throw('Esta cont
a não corresponde a este cliente')
```

c.

```
Context Banco::consultaExtrato(cpf:CPF, nrConta:NrConta, dataInicial:Dat
e):Tuple
    pre:
        cliente[cpf]->notNull() AND
        conta[nrConta]->notNull()
    body:
        Tuple {
            movimentações = conta[nrConta].de.union(conta[nrConta].para)-
>sortedBy(data)->select(
                 data >= dataInicial
            )->collect(
                Tuple{
                    data = data,
                    valor = valor
                }
            )
            saldo = conta[nrConta].saldo
    exception:
        not(cliente[cpf].conta->includes(conta)) implies throw('Esta cont
a não corresponde a este cliente')
```

d.

```
Context Banco::consultaExtrato(cpf:CPF, nrConta:NrConta, dataInicial:Dat
e):Tuple
    pre:
        cliente[cpf]->notEmpty() AND
        conta[nrConta]->notEmpty()
    body:
        Tuple {
            movimentações = de.union(para)->select(
                 data >= dataInicial
            )->sortedBy(data)->collect(
                Tuple{
                    data = data,
                    valor = valor
                }
            )
            saldo = conta[nrConta].saldo
    exception:
        not(cliente[cpf].conta->includes(conta)) implies throw('Esta cont
a não corresponde a este cliente')
```

e.

```
Context Banco::consultaExtrato(cpf:CPF, nrConta:NrConta, dataInicial:Dat
e):Tuple
    pre:
        cliente[cpf]->notEmpty() AND
        conta[nrConta]->notEmpty()
    body:
        Tuple {
            movimentações = conta[nrConta].origem.union(conta[nrConta].de
stino)->sortedBy(data)->select(
                 data >= dataInicial
            )->collect(
                Tuple{
                    data = data,
                    valor = valor
                }
            )
            saldo = conta[nrConta].saldo
    exception:
        not(cliente[cpf].conta->includes(conta)) implies throw('Esta cont
a não corresponde a este cliente')
```

✗ No contexto da classe Conta, as associações são referenciadas pelo papel no destino, ou seja "de" e "para" e não "origem" e "destino". Esses outros papeis são visíveis a partir das instâncias de Transferência.

Sua resposta está incorreta.

A resposta correta é:

```
Context Banco::consultaExtrato(cpf:CPF, nrConta:NrConta, dataInicial:Date):T
uple
    pre:
        cliente[cpf]->notEmpty() AND
        conta[nrConta]->notEmpty()
    body:
        Tuple {
            movimentações = de.union(para)->select(
                 data >= dataInicial
            )->sortedBy(data)->collect(
                Tuple{
                    data = data,
                    valor = valor
                }
            )
            saldo = conta[nrConta].saldo
    exception:
        not(cliente[cpf].conta->includes(conta)) implies throw('Esta conta n
ão corresponde a este cliente')
```

.

Questão 3

Incorreto

Atingiu 0,00 de 1,00

Considere a classe *Control*, com uma associação 1 para * com a classe *Pedido*. *Pedido*, por sua vez tem um atributo *id* e uma associação 1 para * com a classe *Item*. Considere a seguinte expressão OCL:

```
Context Control::deletaPedido(umId:IDPedido)
     pre:
         pedido->select(id=umId)->notEmpty()
     post:
         pedido.destroy() AND
         pedido.item.destroy()
```

O que se pode afirmar sobre esta expressão?

Escolha uma opção:

- a. Ela correta, mas como a associação entre *Control* e *Pedido* não é qualificada seria melhor que a pré condição fosse escrita como *pedido*[*umId*]->*notEmpty*(). ✗ Pelo contrário, ela só poderia ser escrita assim *se* fosse qualificada.
- b. Ela define corretamente que quando a operação *deletaPedido* for executada o pedido identificado é destruído, bem como todos os seus itens.
- c. Ela é falha porque linguagens de programação modernas implementam o *Garbage Collector*, com o qual não é necessário mais destruir objetos explicitamente, bastando para isso remover todas as associações fortes para ele.
- d. Ela é falha porque define que o pedido é destruído antes dos seus itens. Mas se o pedido já foi destruído, como ter acesso aos itens para destruí-los?
- e. Ela é falha porque dentro do *select* na precondição há um contexto ambíguo (*Control* e *Pedido*) e, portanto, o uso de *self* ali seria obrigatório.

Sua resposta está incorreta.

A resposta correta é: Ela define corretamente que quando a operação *deletaPedido* for executada o pedido identificado é destruído, bem como todos os seus itens..

Questão 4

Correto

Atingiu 1,00 de 1,00

Sobre contratos de operação de sistema, pode-se afirmar que:

Escolha uma opção:

- a. São feitos para cada operação de sistema. Contém a especificação do algoritmo que realiza a operação (usualmente um fluxograma).
- b. É feito um contrato para cada caso de uso, indicando o que ele produz como resultado para os atores.
- c. São celebrados entre o desenvolvedor e seus clientes para definir o cronograma do desenvolvimento do software e seus custos.
- d. Podem ser feitos para cada consulta e comando de sistema. Podem conter pré-condições e contêm necessariamente pós-condições ou resultados dependendo do tipo. ✓
- e. Cada contrato define o que deve ser verdadeiro antes da operação ser executada através de pré-condições, portanto, um contrato de operação de sistema não pode prever exceções.

Sua resposta está correta.

A resposta correta é: Podem ser feitos para cada consulta e comando de sistema. Podem conter pré-condições e contêm necessariamente pós-condições ou resultados dependendo do tipo..

Questão 5

Correto

Atingiu 1,00 de 1,00

Qual a interpretação correta para a expressão OCL abaixo?

```
Context Pedido::data
    init: Date.getCurrent()
```

Escolha uma opção:

- a. *data é* um atributo derivado, portanto, não importa o dia em que estivermos, ele sempre terá um valor igual à data do sistema.
- b. A expressão define uma invariante que estabelece que a data de um pedido deve ser sempre igual à data do sistema.
- c. Depois de definido o valor inicial da data de um pedido como a data do sistema ele não pode mais ser mudado, ou seja, é imutável.
- d. A expressão não faz sentido.
- e. Toda vez que uma instância de pedido for criada, o atributo *data* já estará inicializado com a data do sistema. ✓

Sua resposta está correta.

A resposta correta é: Toda vez que uma instância de pedido for criada, o atributo *data* já estará inicializado com a data do sistema..

Questão 6

Correto

Atingiu 1,00 de 1,00

Qual o efeito da seguinte expressão OCL?

```
{1,2,3,4,5,6,7,8} -> select(x | x*2 > 6)
```

Escolha uma opção:

- a. Ela retorna o conjunto {1,2,3}.
- b. Ela retira os elementos {1,2,3} do conjunto original.
- c. Ela retira os elementos {4,5,6,7,8} do conjunto original.
- d. Ela retorna o somatório de todos os elementos do conjunto.
- e. Ela retorna o conjunto {4,5,6,7,8}. ✓

Sua resposta está correta.

A expressão seleciona e retorna apenas os elementos cujo dobro é maior do que seis, ou seja, 4, 5, 6, 7 e 8. Ela não modifica o conjunto original.

A resposta correta é: Ela retorna o conjunto {4,5,6,7,8}..

Questão 7

Correto

Atingiu 1,00 de 1,00

O conceito de alta coesão, utilizado no processo de modelagem de software, é um princípio essencia de

Escolha uma opção:

- a. Generalidade
- b. Abstração
- c. Modularidade ✓
- d. Incrementação
- e. Separação de interesses

Sua resposta está correta.

A resposta correta é: Modularidade.

Questão 8

Correto

Atingiu 1,00 de 1,00

Considere o seguinte modelo conceitual:

Regras de negócio:

- A cada venda corresponde um certo número de parcelas (no mínimo 1 e, no máximo, tipicamente 12).
- Cada parcela será associada a um pagamento quando este ocorrer., o que implica na quitação da parcela.

Assinale abaixo a opção de multiplicidades que melhor representa estes requisitos respectivamente: p1, p2, p3 e p4.

Escolha uma opção:

- a. 1 / * / 1 / 0..1
- b. 1 / 1..* / 0..1 / 0..1
- c. 1 / 1..* / 1 / 0..1 ✓
- d. 1 / 1..12 / 1 / 0..1
- e. 1 / 1..* / 1 / 1

Sua resposta está correta.

A resposta correta é: 1 / 1..* / 1 / 0..1.

Questão 9

Incorreto

Atingiu 0,00 de 1,00

O *design pattern* "coesão alta" implica que:

Escolha uma opção:

- a. Deve-se minimizar as ligações de visibilidade entre as classes.
- b. Deve-se aumentar o número médio de ligações de visibilidade entre as classes.
- c. Deve-se minimizar a complexidade interna das classes.
- d. Deve-se minimizar o número de atributos das classes. ✗ O padrão não tem a ver com o número de atributos, mas a forma como eles se relacionam uns com os outros.
- e. Deve-se aumentar a complexidade interna das classes.

Sua resposta está incorreta.

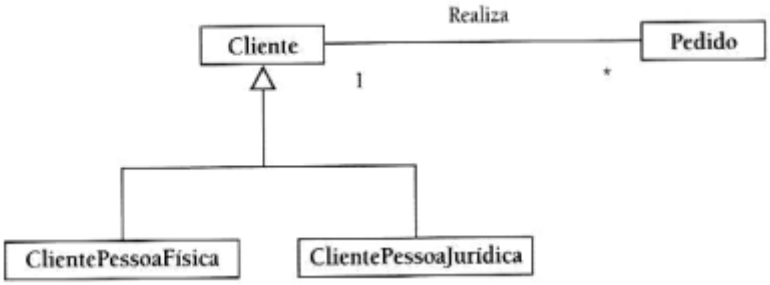
A resposta correta é: Deve-se minimizar a complexidade interna das classes..

Questão 10

Correto

Atingiu 1,00 de 1,00

Considerando-se o diagrama de classes apresentado a seguir, é correto afirmar que



Escolha uma opção:

- a. as classes ClientePessoaFísica e ClientePessoaJurídica possuem um relacionamento de associação com a classe Pedido, já que subclasses herdam as associações da superclasse. ✓
- b. a classe Cliente é uma especialização das classes ClientePessoaFísica e ClientePessoaJurídica (herança múltipla), já que, além de herdar as propriedades de ambas, adiciona um relacionamento com a classe Pedido.
- c. a classe Cliente mantém uma relação do tipo todo-parte com as classes ClientePessoaFísica e ClientePessoaJurídica, e uma relação de associação um-para-muitos com a classe Pedido.
- d. as classes ClientePessoaFísica e ClientePessoaJurídica não possuem um relacionamento de associação com a classe Pedido, apenas a classe Cliente possui este relacionamento.

Sua resposta está correta.

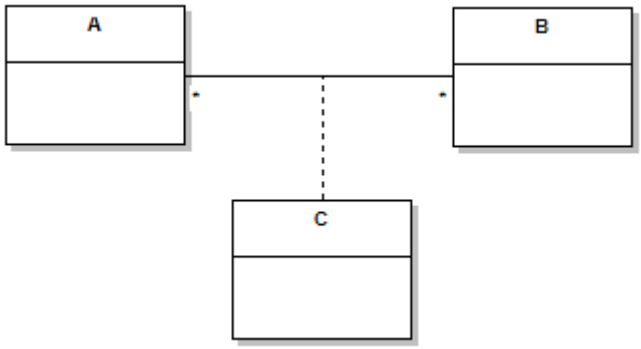
A resposta correta é: as classes ClientePessoaFísica e ClientePessoaJurídica possuem um relacionamento de associação com a classe Pedido, já que subclasses herdam as associações da superclasse..

Questão 11

Correto

Atingiu 1,00 de 1,00

Sobre o modelo conceitual abaixo, representado em UML, pode-se afirmar que:



Escolha uma opção:

- a. Para cada instância de A corresponde exatamente uma instância de C.
- b. Sempre que a associação representada acima for criada entre A e B, necessariamente será criada uma nova instância de C. ✓
- c. Para cada instância de C corresponde exatamente uma instância de A e um conjunto possivelmente vazio de instâncias de B.
- d. Para cada associação entre A e B corresponde um conjunto possivelmente vazio de instâncias de C.
- e. Não é possível criar instâncias de A e de B, pois são dependentes de C.

Sua resposta está correta.

A resposta correta é: Sempre que a associação representada acima for criada entre A e B, necessariamente será criada uma nova instância de C..

Questão 12

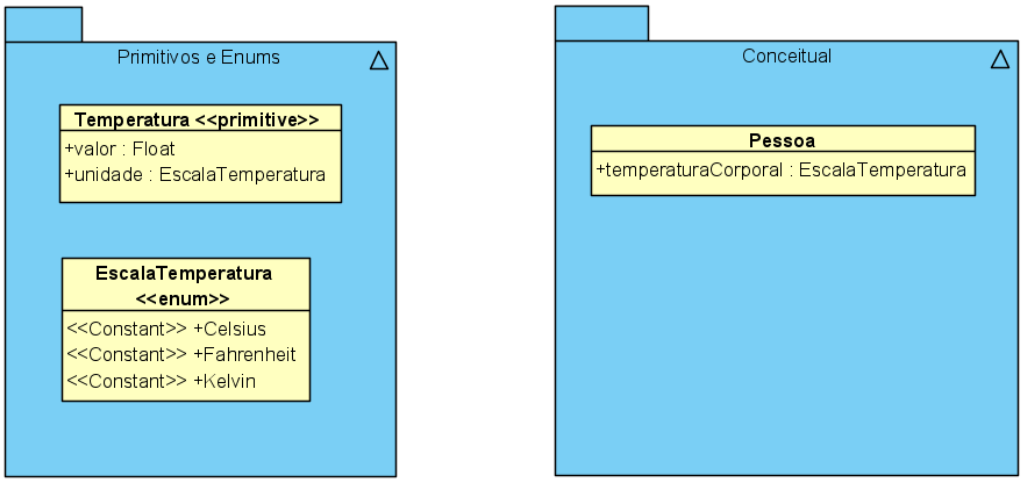
Incorreto

Atingiu 0,00 de 1,00

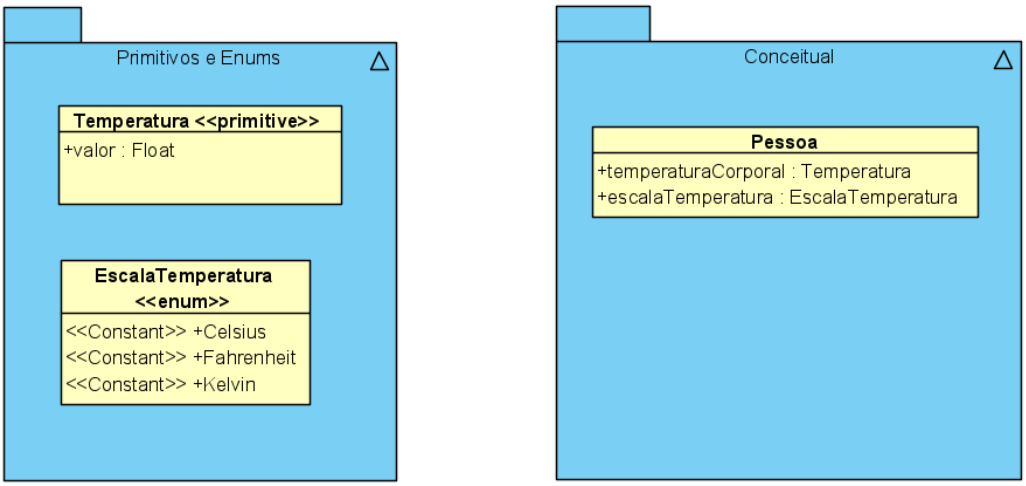
Dentre as opções abaixo assinale aquela que apresenta a correta aplicação do padrão "Quantidade".

Escolha uma opção:

a.

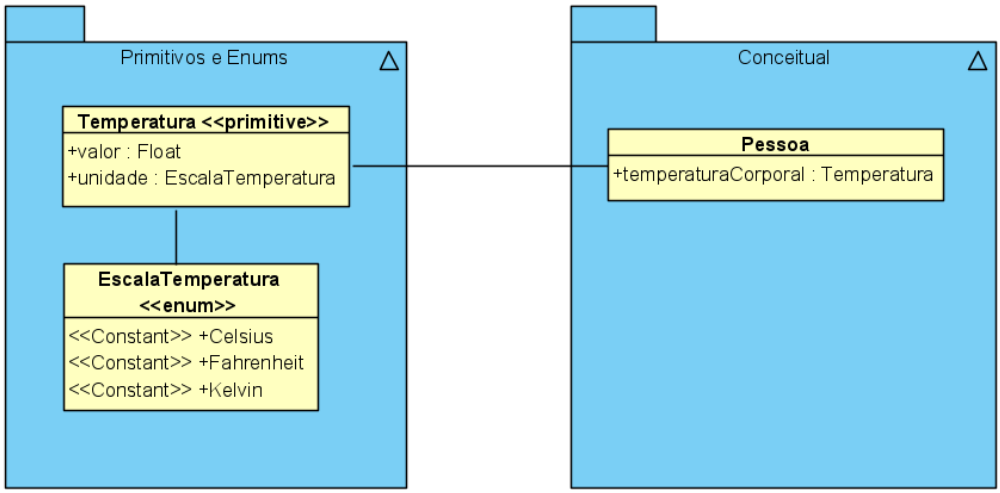


b.

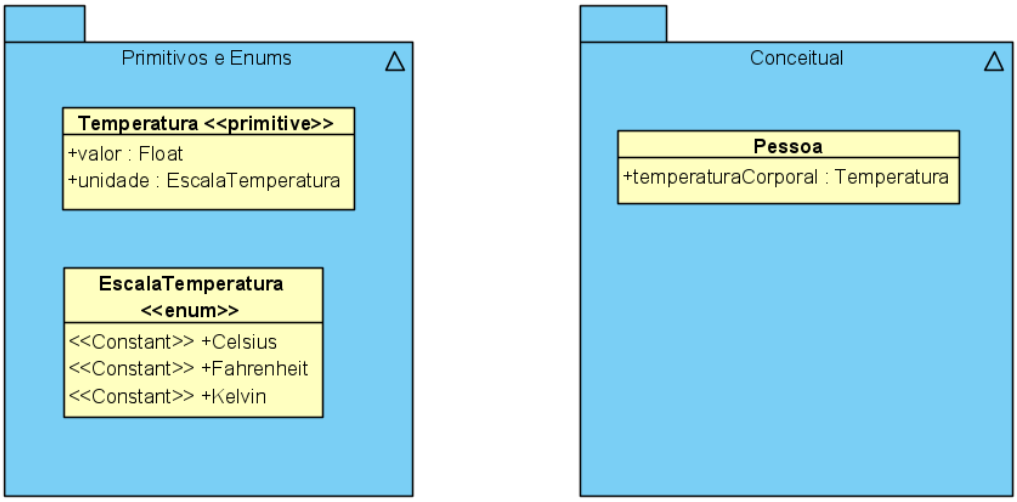


✗ Esta aplicação incorreta do padrão Quantidade ainda fere o padrão Coesão Alta, por ter dois atributos fortemente relacionados em uma classe com semântica mais geral.

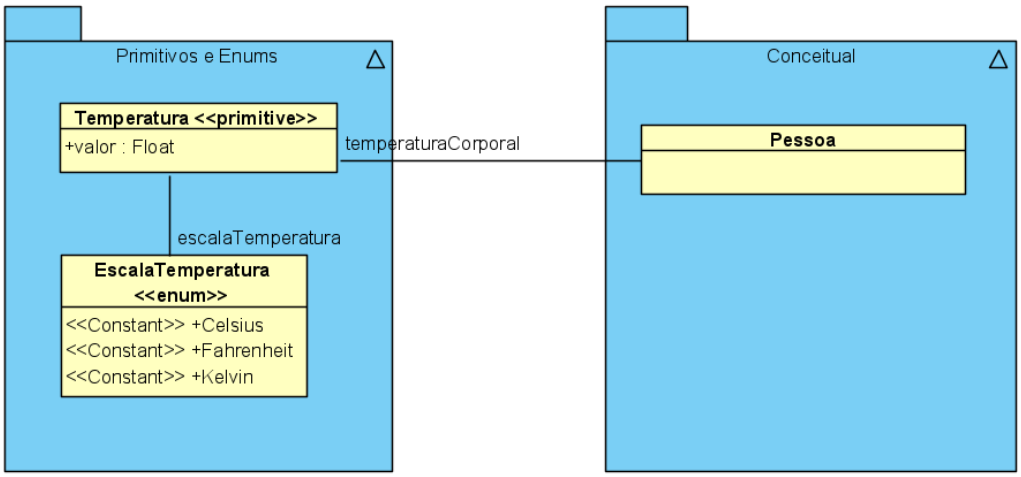
c.



d.

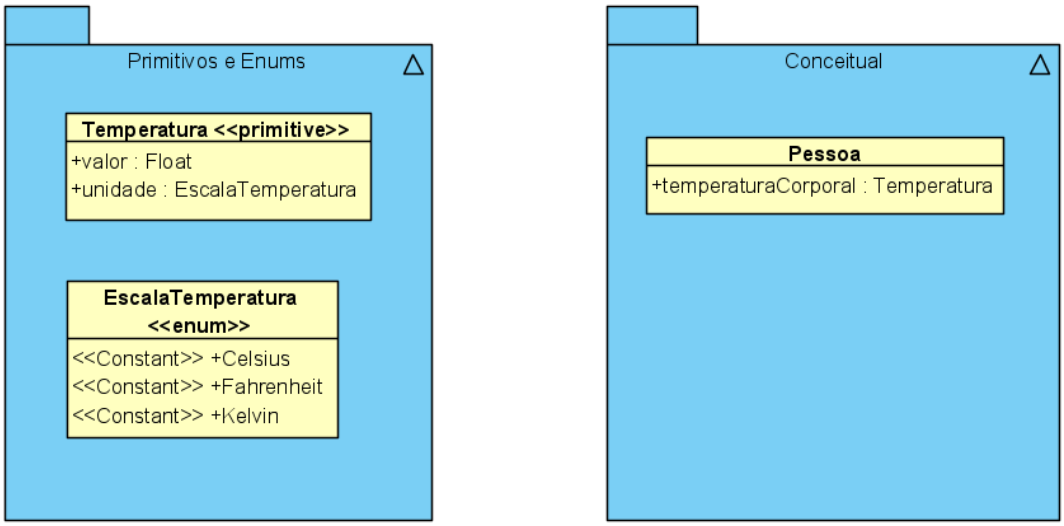


e.



Sua resposta está incorreta.

A resposta correta é:



.

Questão 13

Incorreto

Atingiu 0,00 de 1,00

Considerando o modelo conceitual, que cuidado deve-se tomar quando um contrato possui uma pós-condição do tipo “foi criada uma instância”?

Escolha uma opção:

- a. Deve-se garantir que algum outro contrato terá uma pós-condição do tipo “foi destruída uma instância”.
- b. Deve-se garantir que no mesmo contrato exista uma pré-condição do tipo “existe uma instância de...”. ✗ Não há necessariamente relação entre criação de instância e a existência de outra instância.
- c. Deve-se colocar uma pré-condição para verificar se já não existe uma instância alocada para a mesma variável.
- d. Deve-se inserir a instância em uma lista ou conjunto de instâncias da classe.
- e. Deve-se sempre adicionar uma pós-condição do tipo “foi criada uma ligação” entre a instância recém criada e alguma outra instância que tenha um caminho de ligações até o controlador.

Sua resposta está incorreta.

A resposta correta é: Deve-se sempre adicionar uma pós-condição do tipo “foi criada uma ligação” entre a instância recém criada e alguma outra instância que tenha um caminho de ligações até o controlador..

Questão 14

Correto

Atingiu 1,00 de 1,00

Em que situação pode ser necessário usar a expressão OCL *@pre*?

Escolha uma opção:

- a. Invariantes.
- b. Atributos ou associações derivados.
- c. Precondições.
- d. Exceções.
- e. Pós condições. ✓

Sua resposta está correta.

A expressão @pre só faz sentido em pós-condições porque ela remete ao valor de um atributo ou associação antes de a operação ser executada. por default pós-condições referenciam valores existentes após a operação ser executada.

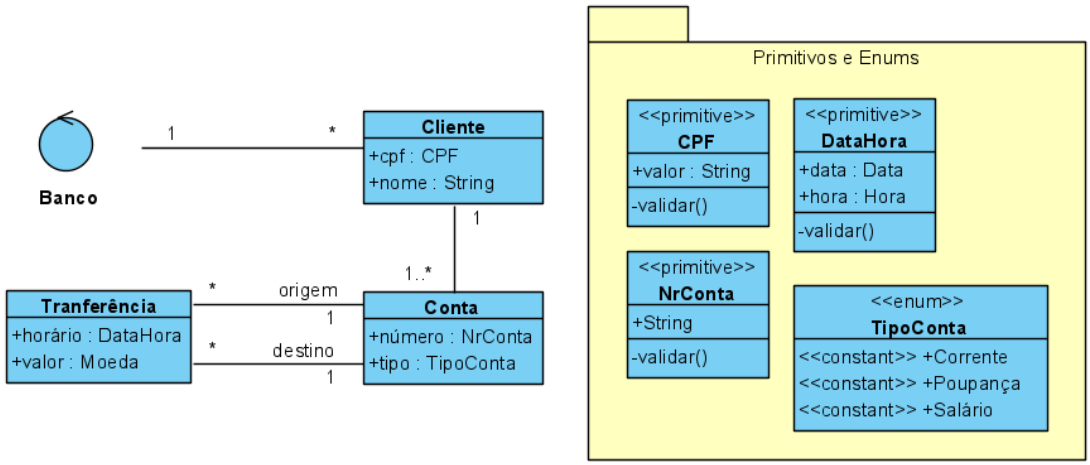
A resposta correta é: Pós condições..

Questão 15

Correto

Atingiu 1,00 de 1,00

Considere o seguinte modelo conceitual:



Falta neste modelo uma invariante que garanta que cada transferência é feita entre contas diferentes e que este valor tem que ser superior a zero. Assinale a expressão OCL que melhor representa esta invariante.

Escolha uma opção:

a.

```
Context Transferência
    inv:
        origem <> destino AND
        valor <> 0
```

b.

```
Context Transferência::valor
    inv:
        origem <> destino AND
        valor <> 0
```

c.

```
Context Transferência
    pre:
        origem <> destino AND
        valor <> 0
```

d.

```
Context Transferência
    inv:
        origem <> destino AND
        valor > 0
```

e.

```
Context Transferência::valor
    inv:
        origem <> destino AND
        valor > 0
```

Sua resposta está correta.

A resposta correta é:

```
Context Transferência
    inv:
        origem <> destino AND
        valor > 0
```

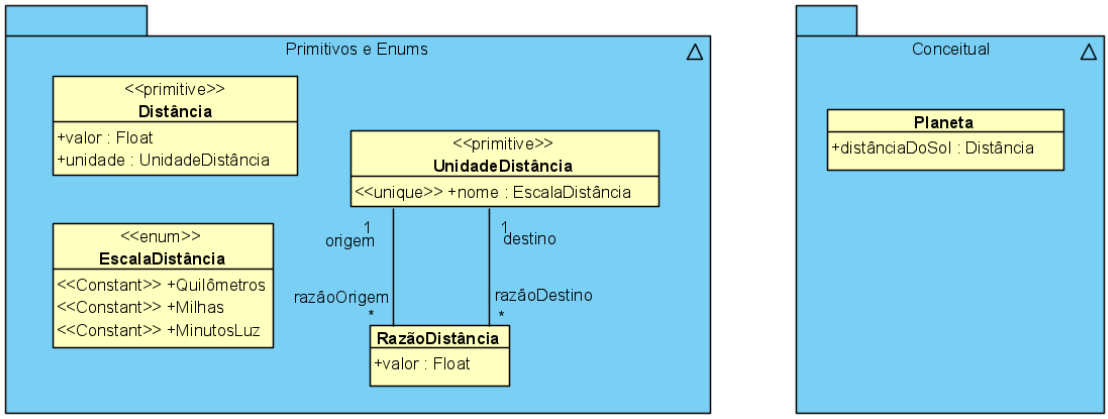
.

Questão 16

Incorreto

Atingiu 0,00 de 1,00

Considere o modelo abaixo que apresenta um exemplo do padrão Quantidade com razão de conversão.



Se a distância do planeta X ao Sol está expressa em milhas, para convertê-la em quilômetros devemos: (assinale a opção correta)

Escolha uma opção:

a. No contexto da classe Planeta, pegar o conjunto self.distânciaDoSol.unidade.razãoOrigem e encontrar nele a instância "umaRazão" de RazãoDistância cujo destino.nome é EscalaDistância::Quilômetros.
O atributo self.distânciaDoSol.valor deve passar a valer self.distânciaDoSol.valor / umaRazão.valor.
self.distânciaDoSol.unidade atual deve ser substituído pela instância de UnidadeDistância cujo nome é EscalaDistância::Quilômetros.

b. No contexto da classe Planeta, pegar o conjunto self.distânciaDoSol.unidade.razãoOrigem.destino.nome que seja igual a EscalaDistância::Quilômetros.
O atributo self.distânciaDoSol.valor deve passar a valer self.distânciaDoSol.valor / umaRazão.valor.
self.distânciaDoSol.unidade atual deve ser substituído pela instância de UnidadeDistância cujo nome é EscalaDistância::Quilômetros.

c. No contexto da classe Distância, pegar o conjunto self.unidade.razãoOrigem e encontrar nele a instância "umaRazão" de RazãoDistância cujo destino.nome é EscalaDistância::Quilômetros.
O atributo self.valor deve passar a valer self.valor / umaRazão.valor.

self.unidade atual deve ser substituído pela instância de UnidadeDistância cujo nome é EscalaDistância::Quilômetros.

d. No contexto da classe Planeta, pegar o conjunto self.distânciaDoSol.unidade.razãoOrigem e encontrar nele a instância "umaRazão" de RazãoDistância cujo destino.nome é EscalaDistância::Quilômetros.
O atributo self.distânciaDoSol.valor deve passar a valer self.distânciaDoSol.valor / umaRazão.valor.
self.distânciaDoSol.unidade atual deve ser substituído pela instância de UnidadeDistância cujo nome é EscalaDistância::Quilômetros.
Deletar a instância de UnidadeDistância cujo nome é EscalaDistância::Milhas.

e. No contexto da classe Planeta, pegar o conjunto self.distânciaDoSol.unidade.razãoOrigem e encontrar nele a instância "umaRazão" de RazãoDistância cujo destino.nome é EscalaDistância::Quilômetros.
O atributo self.distânciaDoSol.valor deve passar a valer self.distânciaDoSol.valor / umaRazão.valor.
self.distânciaDoSol.unidade.nome atual deve ser substituído por EscalaDistância::Quilômetros. ✗ o atributo nome é imutável

Sua resposta está incorreta.

A resposta correta é: No contexto da classe Planeta, pegar o conjunto self.distânciaDoSol.unidade.razãoOrigem e encontrar nele a instância "umaRazão" de RazãoDistância cujo destino.nome é EscalaDistância::Quilômetros.
O atributo self.distânciaDoSol.valor deve passar a valer self.distânciaDoSol.valor / umaRazão.valor.
self.distânciaDoSol.unidade atual deve ser substituído pela instância de UnidadeDistância cujo nome é EscalaDistância::Quilômetros..

Questão 17

Incorreto

Atingiu 0,00 de 1,00

Uma *classe de especificação*:

Escolha uma opção:

- a. É uma classe usada para especificar outra, com a qual ela mantém sempre uma associação de 1 para 1.
- b. É uma classe abstrata que tem classes concretas como subclasses.
- c. Representa um conceito cujos atributos têm valores que seriam repetidos por grupos de instâncias de uma outra classe.
- d. Representa um conceito cujos atributos têm valores que seriam só repetidos por instâncias de várias outras classes diferentes. ✗ Eles poderiam ser repetidos na mesma classe.
- e. É uma classe concreta que tem classes abstratas como subclasses.

Sua resposta está incorreta.

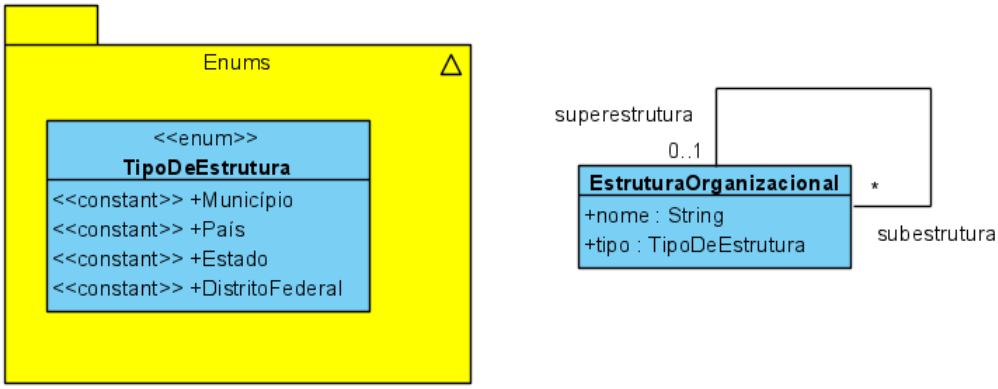
A resposta correta é: Representa um conceito cujos atributos têm valores que seriam repetidos por grupos de instâncias de uma outra classe..

Questão 18

Incorreto

Atingiu 0,00 de 1,00

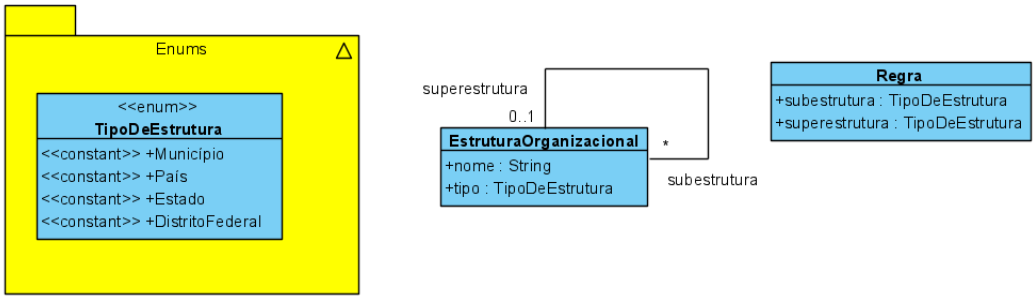
Considere o modelo conceitual abaixo que corresponde ao padrão de hierarquia organizacional:



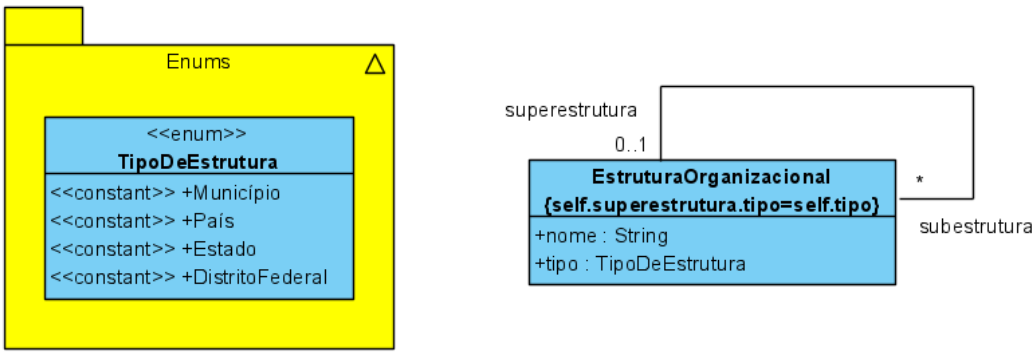
Supondo que se deseja criar um conjunto de regras que definem quais tipos de estruturas podem estar diretamente subordinadas a quais outros tipos, assinale dentre as opções abaixo aquela que melhor permite representar este tipo de requisito.

Escolha uma opção:

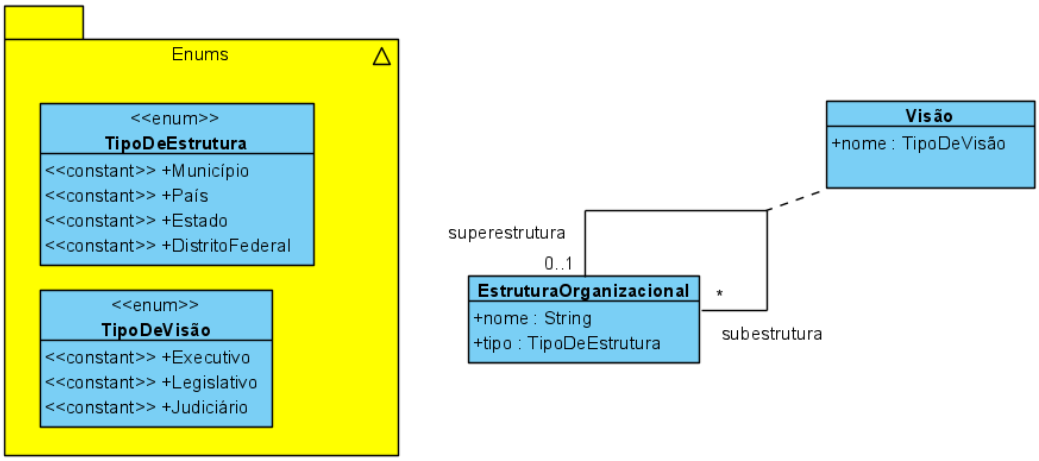
a.



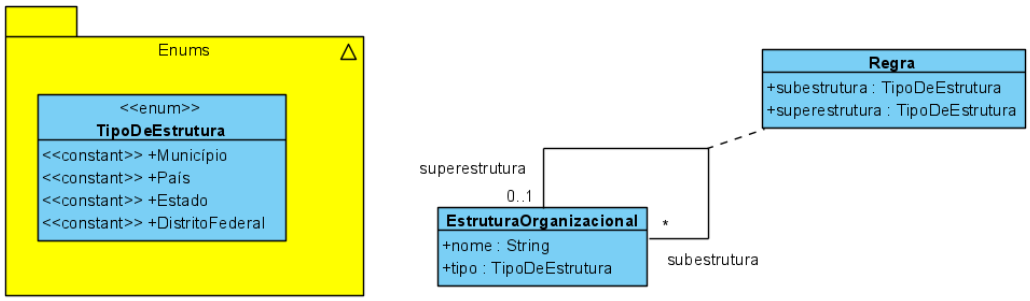
b.



c.

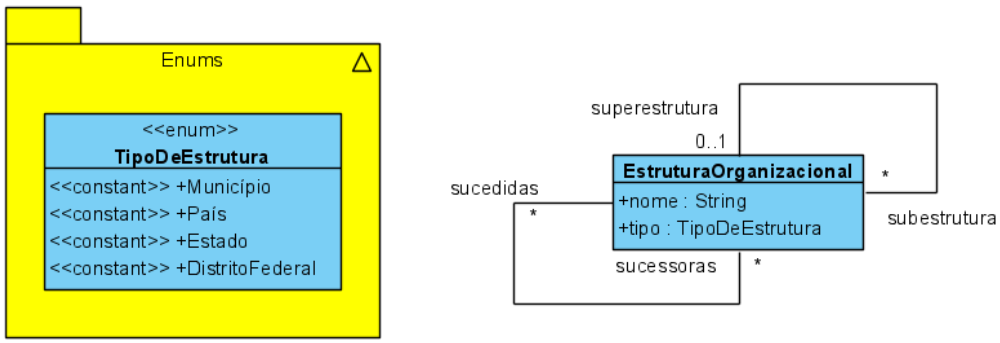


d.



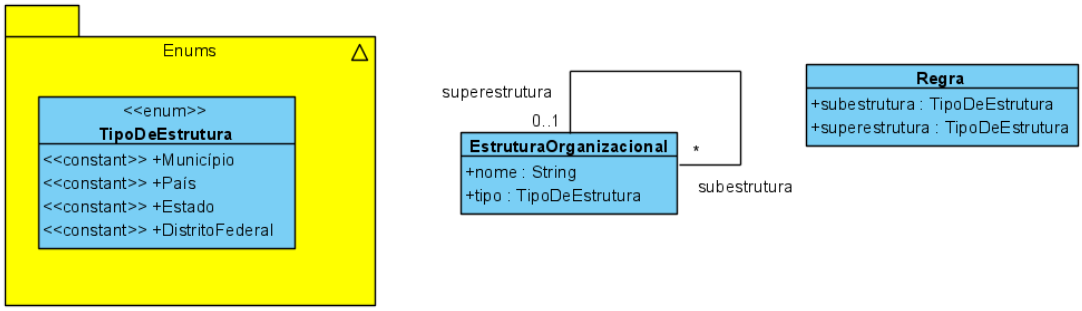
✗ Neste caso, cada vez que uma ligação super/subestrutura for criada, uma nova regra também é criada. Isso não corresponde ao requisito.

e.



Sua resposta está incorreta.

A resposta correta é:



.

Questão 19

Correto

Atingiu 1,00 de 1,00

Considere que a classe *Pessoa* tem atributos *nome* e *dataNascimento*. Considere que a classe *Video* tem os atributos *titulo* e *idadeMinima* (para assistir). Considere também que as duas classes têm uma associação * para * cujo nome de papel do lado da classe *Video* é *assistido.* Qual das expressões abaixo representa uma invariante que estabelece que uma pessoa só pode assistir a um vídeo se tiver idade suficiente para isso?

Escolha uma opção:

a.

```
Context Pessoa inv:
    assistido->exists((Date.getCurrent()-dataNascimento).getYears()>=idad
eMinima)
```

b.

```
Context Pessoa inv:
    assistido->forAll((Date.getCurrent()-dataNascimento).getYears()>=idad
eMinima)
```

c.

```
Context Pessoa inv:
    assistido->forAll((Date.getCurrent()-dataNascimento).getYears()<idade
Minima)
```

d.

```
Context Video inv:
    pessoa->exists((Date.getCurrent()-dataNascimento).getYears()>=idadeMi
nima
```

e.

```
Context Video inv:
    assistido->forAll((Date.getCurrent()-dataNascimento).getYears()>=idad
eMinima)
```

Sua resposta está correta.

A resposta correta é:

```
Context Pessoa inv:
    assistido->forAll((Date.getCurrent()-dataNascimento).getYears()>=idadeMi
nima)
```

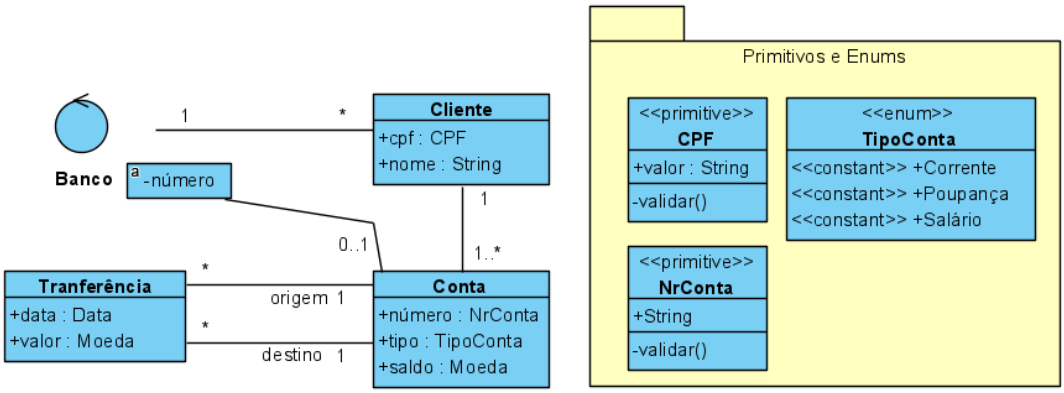
.

Questão 20

Incorreto

Atingiu 0,00 de 1,00

Considere o seguinte modelo conceitual:



Foi solicitado que você especificasse o contrato do comando de sistema *Banco::efetuarTransferência(nrContaFonte:NrConta, nrContaDestino:NrConta, valor:Moeda)*.

Regras de negócio:

- O valor transferido deve ser positivo e debitado da conta origem e creditado na conta destino.
- A data da transferência deve ser preenchida com a data de hoje.
- Os números das contas de origem e destino já foram validados antes de chamar a operação.
- Se a conta de origem ficar negativa, deve ser gerada uma exceção.

Selecione abaixo o contratos que corretamente especifica este comando.

Escolha uma opção:

a.

```
Banco::efetuarTransferência(nrContaFonte:NrConta, nrContaDestino:NrConta,
valor:Moeda)
    def:
        contaFonte = conta[nrContaFonte]
        contaDestino = conta[nrContaDestino]
    pre:
        contaFonte->notEmpty() AND
        contaDestino->notEmpty() AND
        valor > 0
    post:
        novaTransferência.isNewInstanceOf(Transferência) AND
        novaTransferência^setData(Date.today()) AND
        novaTranferência^setValor(valor) AND
        contaFonte^setSaldo(contaFonte.saldo@pre - valor) AND
        contaDestino^setSaldo(contaDestino@pre + valor)
    exception:
        contaFonte < 0 implies throw("saldo insuficiente")
```

✗ Todas as clausulas, exceto a "post" referem-se ao estado dos objetos ANTES da execução do comando. Então, a exceção, neste caso, estará verificando o saldo

da conta de origem antes da execução da operação. Ela deveria verificar o saldo posterior ao comando, caso este fosse executado.

b.

```
Banco::efetuarTransferência(nrContaFonte:NrConta, nrContaDestino:NrConta,
valor:Moeda)
    def:
        contaFonte = conta[nrContaFonte]
        contaDestino = conta[nrContaDestino]
    pre:
        contaFonte->notNull() AND
        contaDestino->notNull() AND
        valor > 0
    post:
        novaTransferência.isNewInstanceOf(Transferência) AND
        novaTransferência^setData(Date.today()) AND
        novaTranferência^setValor(valor) AND
        contaFonte^setSaldo(contaFonte.saldo@pre - valor) AND
        contaDestino^setSaldo(contaDestino@pre + valor)
    exception:
        contaFonte - valor < 0 implies throw("saldo insuficiente")
```

c.

```
Banco::efetuarTransferência(nrContaFonte:NrConta, nrContaDestino:NrConta,
valor:Moeda)
    def:
        contaFonte = conta[nrContaFonte]
        contaDestino = conta[nrContaDestino]
    pre:
        contaFonte->notEmpty() AND
        contaDestino->notEmpty() AND
        valor > 0
    post:
        novaTransferência = isNewInstanceOf(Transferência) AND
        novaTransferência^setData(Date.today()) AND
        novaTranferência^setValor(valor) AND
        contaFonte^setSaldo(contaFonte.saldo@pre - valor) AND
        contaDestino^setSaldo(contaDestino@pre + valor)
    exception:
        contaFonte - valor < 0 implies throw("saldo insuficiente")
```

d.

```
Banco::efetuarTransferência(nrContaFonte:NrConta, nrContaDestino:NrConta,
valor:Moeda)
    def:
        contaFonte = conta[nrContaFonte]
        contaDestino = conta[nrContaDestino]
    pre:
        contaFonte->notEmpty()
        contaDestino->notEmpty()
        valor > 0
    post:
        novaTransferência.isNewInstanceOf(Transferência)
        novaTransferência^setData(Date.today())
        novaTranferência^setValor(valor)
        contaFonte^setSaldo(contaFonte.saldo@pre - valor)
        contaDestino^setSaldo(contaDestino@pre + valor)
    exception:
        contaFonte - valor < 0 implies throw("saldo insuficiente")
```

e.

```
Banco::efetuarTransferência(nrContaFonte:NrConta, nrContaDestino:NrConta,
valor:Moeda)
    def:
        contaFonte = conta[nrContaFonte]
        contaDestino = conta[nrContaDestino]
    pre:
        contaFonte->notEmpty() AND
        contaDestino->notEmpty() AND
        valor > 0
    post:
        novaTransferência.isNewInstanceOf(Transferência) AND
        novaTransferência^setData(Date.today()) AND
        novaTranferência^setValor(valor) AND
        contaFonte^setSaldo(contaFonte.saldo@pre - valor) AND
        contaDestino^setSaldo(contaDestino@pre + valor)
    exception:
        contaFonte - valor < 0 implies throw("saldo insuficiente")
```

Sua resposta está incorreta.

A resposta correta é:

```
Banco::efetuarTransferência(nrContaFonte:NrConta, nrContaDestino:NrConta, va
lor:Moeda)
    def:
        contaFonte = conta[nrContaFonte]
        contaDestino = conta[nrContaDestino]
    pre:
        contaFonte->notEmpty() AND
        contaDestino->notEmpty() AND
        valor > 0
    post:
        novaTransferência.isNewInstanceOf(Transferência) AND
        novaTransferência^setData(Date.today()) AND
        novaTranferência^setValor(valor) AND
        contaFonte^setSaldo(contaFonte.saldo@pre - valor) AND
        contaDestino^setSaldo(contaDestino@pre + valor)
    exception:
        contaFonte - valor < 0 implies throw("saldo insuficiente")
```

.