

Iniciado em Thursday, 23 Sep 2021, 10:12

Estado Finalizada

Concluída em Thursday, 23 Sep 2021, 11:09

Tempo empregado 57 minutos 11 segundos

Notas 10,33/20,00

Avaliar 5,17 de um máximo de 10,00(52%)

Questão 1

Incorreto

Atingiu 0,00 de
1,00

Qual seria o resultado da seguinte expressão:

```
Set{1,2,3,4,5,6,7}->collect(x|x.isEven()) -- isEven = “predicado é par”
```

Escolha uma opção:

- a. Sequence{false, true, false, true, false, true, false}
- b. Set{false, true, false, true, false, true, false} X Conjuntos não repetem elementos
- c. Set{true}
- d. Set{false, true}
- e. Set{2, 4, 6, 8, 10, 12, 14}

Sua resposta está incorreta.

A resposta correta é: Set{false, true}

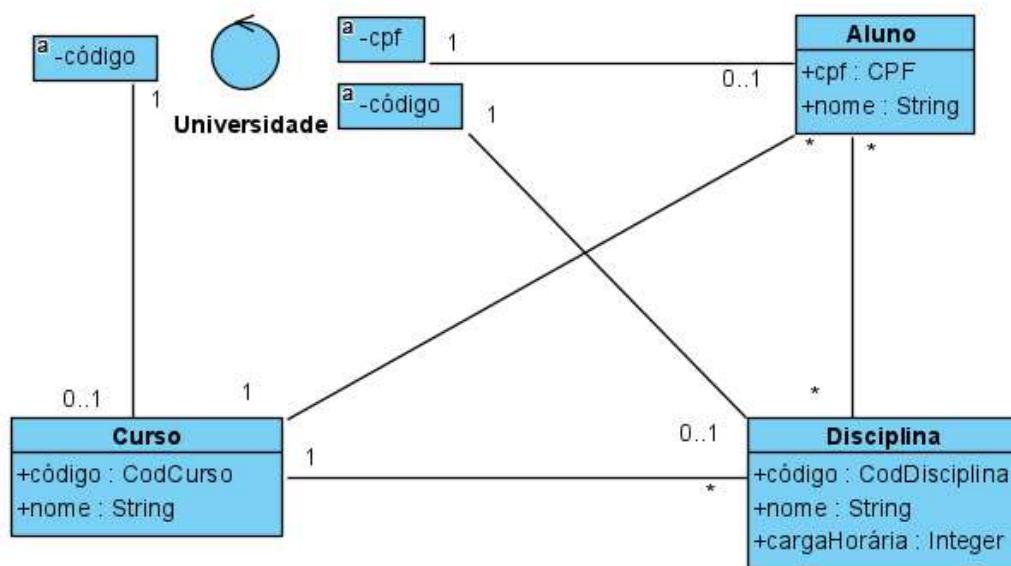
Questão 2

Correto

Atingiu 1,00 de
1,00

QUESTÃO DE MÚLTIPLA ESCOLHA

Considere o seguinte modelo conceitual:



Considere também a seguinte invariante:

```
Context Aluno
inv:
    disciplina->notEmpty() implies
        curso = disciplina.curso
```

Considerando como regras de negócio:

- alunos não podem trocar de curso
- disciplinas com alunos matriculados não podem ser deletadas

Assinale apenas os contratos bem formados e consistentes com as regras de negócio dentre os abaixo

Escolha uma ou mais:



a.

```
Context Universidade::cadastrarCurso(umCódigo:codCurso, umNome:String)
pre:
    curso[umCódigo]->isEmpty()
post:
    novoCurso.isInstanceOf(Curso) and
    novoCurso^setCódigo(umCódigo) and
    novoCurso^setNome(umNome) and
    self^addCurso(novoCurso)
```



b.

```

Context Universidade::adicionarDisciplina(umCodDisciplina:CodDisciplina,
                                         umNome:String, umaCargaHorária:Integer, umCodCurso:CodCurso)
    pre:
        disciplina[umCodDisciplina]->isEmpty() and
        curso[umCodCurso]->notEmpty()
    post:
        novaDisciplina.isInstanceOf(Disciplina) and
        novaDisciplina^setCódigo(umCodDisciplina) and
        novaDisciplina^setNome(umNome) and
        novaDisciplina^setCargaHorária(umaCargaHorária) and
        self^addDisciplina(novaDisciplina) and
        curso[umCodCurso]^addDisciplina(novaDisciplina)

```



c.

```

Context Universidade::excluirDisciplina(umCodDisciplina:CodDisciplina)
    pre:
        disciplina[umCodDisciplina]->notEmpty()
    post:
        disciplina[umCodDisciplina]->destroi()
    exception:
        disciplina[umCodDisciplina].aluno->notEmpty() implies
            throwException("essa disciplina tem alunos matriculados")

```



d.

```

Context Universidade::matricularAlunoEmCurso(umCpf:CPF, umCodCurso: CodCurso)
    pre:
        aluno[umCpf]->notEmpty() and
        curso[umCodCurso]->notEmpty()
    post:
        curso[umCurso]^addAluno(aluno[umCpf])

```



e.

```

Context::matriculaAlunoEmDisciplina(umCpf:CPF, umCodDisciplina:CodDisciplina)
    pre:
        aluno[umCpf]->notEmpty() and
        disciplina[umCodDisciplina]->notEmpty()
    post:
        aluno[umCpf]^addDisciplina(disciplina[umCodDisciplina])

```



Este contrato não verifica se a disciplina pertence ao curso do aluno. Mas a invariante da classe aluno já trata isso como uma exceção. Assim, não há problema com o contrato já que cada condição deve ser tratada ou como précondição ou como exceção.

Sua resposta está correta.

As respostas corretas são:

```
Context Universidade::cadastrarCurso(umCódigo:codCurso, umNome:String)
pre:
    curso[umCódigo]->isEmpty()
post:
    novoCurso.isInstanceOf(Curso) and
    novoCurso^setCódigo(umCódigo) and
    novoCurso^setNome(umNome) and
    self^addCurso(novoCurso)
```

```
,
```

```
Context Universidade::adicionarDisciplina(umCodDisciplina:CodDisciplina,
umNome:String, umaCargaHorária:Integer, umCodCurso:CodCurso)
pre:
    disciplina[umCodDisciplina]->isEmpty() and
    curso[umCodCurso]->notEmpty()
post:
    novaDisciplina.isInstanceOf(Disciplina) and
    novaDisciplina^setCódigo(umCodDisciplina) and
    novaDisciplina^setNome(umNome) and
    novaDisciplina^setCargaHorária(umaCargaHorária) and
    self^addDisciplina(novaDisciplina) and
    curso[umCodCurso]^addDisciplina(novaDisciplina)
```

```
,
```

```
Context Universidade::excluirDisciplina(umCodDisciplina:CodDisciplina)
pre:
    disciplina[umCodDisciplina]->notEmpty()
post:
    disciplina[umCodDisciplina]->destroi()
exception:
    disciplina[umCodDisciplina].aluno->notEmpty() implies
        throwException("essa disciplina tem alunos matriculados")
```

```
,
```

```
Context::matriculaAlunoEmDisciplina(umCpf:CPF, umCodDisciplina:CodDisciplina)
pre:
    aluno[umCpf]->notEmpty() and
    disciplina[umCodDisciplina]->notEmpty
post:
    aluno[umCpf]^addDisciplina(disciplina[umCodDisciplina])
```

Questão 3

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 colocar uma pré-condição para verificar se já não existe uma instância alocada para a mesma variável. X Nenhuma regra exige isso.
- b. Deve-se garantir que no mesmo contrato exista uma pré-condição do tipo "existe uma instância de...".
- c. Deve-se inserir a instância em uma lista ou conjunto de instâncias da classe.
- d. Deve-se garantir que algum outro contrato terá uma pós-condição do tipo "foi destruída uma instância".
- 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.

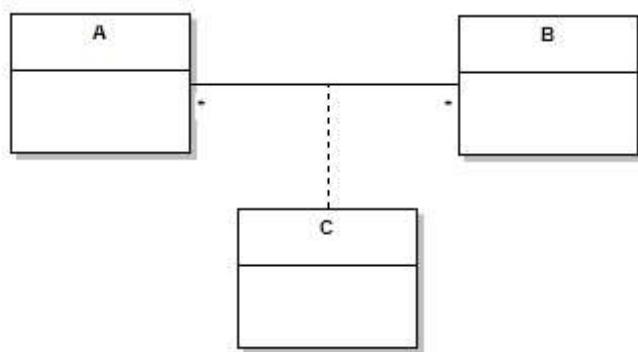
A resposta correta é: Deve-se sempre adicionar uma pós-condição do tipo "foi criada uma ligação" entre a instância recém criada e alguma outra instância que tenha um caminho de ligações até o controlador.

Questão 4

Incorreto

Atingiu 0,00 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. Para cada instância de C corresponde exatamente uma instância de A e um conjunto possivelmente vazio de instâncias de B.
- c. Não é possível criar instâncias de A e de B, pois são dependentes de C.
- d. Para cada associação entre A e B corresponde um conjunto possivelmente vazio de instâncias de C. X Corresponde exatamente uma instância de C.
- e. Sempre que a associação representada acima for criada entre A e B, necessariamente será criada uma nova instância de C.

Sua resposta está incorreta.

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 5

Correto

Atingiu 1,00 de
1,00

Considere que uma regra de negócio estabelece que faturas terão valor devido mínimo de 100 reais, porque abaixo disso a empresa não pode atender a nenhum pedido. Neste caso, seria recomendável especificar essa condição usando qual das opções abaixo?

Escolha uma opção:

- a. Uma invariante na classe *Fatura*. ✓
- b. Um contrato com pré e pós-condições na operação que cria uma instância de *Fatura*.
- c. Um atributo com valor inicial na classe *Fatura*.
- d. Um atributo derivado na classe *Fatura*.
- e. Um contrato com pré e pós-condições na operação *setValorDevido* da classe *Fatura*.

Sua resposta está correta.

Trata-se de uma regra que vale sempre. Portanto, a estrutura correta é a invariante neste caso.

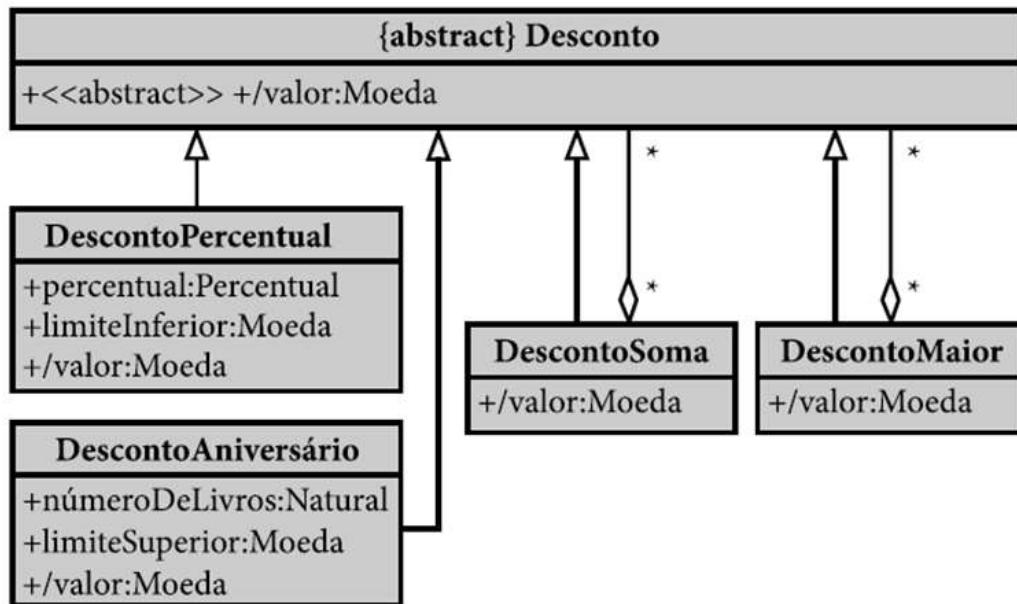
A resposta correta é: Uma invariante na classe *Fatura*.

Questão 6

Parcialmente correto

Atingiu 0,33 de 1,00

Considere o modelo conceitual abaixo:



Assinale apenas as alternativas verdadeiras:

Escolha uma ou mais:

- a. Um DescontoPercentual pode agregar um DescontoAniversário.
- b. Todas as instâncias de DescontoMaior executam a mesma versão de /valor que, por ser atributo derivado, será implementado como um método da classe.
- c. DescontoSoma e DescontoPercentual são dois diferentes tipos de Desconto, com implementações distintas para o atributo /valor.
- d. Um Desconto é composto necessariamente por instâncias de DescontoSoma ou DescontoMaior
- e. Um DescontoSoma pode agregar outro DescontoSoma ✓ Como DescontoSoma pode agregar Desconto e DescontoSoma é um tipo de Desconto, então é, sim, possível.

Sua resposta está parcialmente correta.

Você selecionou corretamente 1.

As respostas corretas são: Um DescontoSoma pode agregar outro DescontoSoma, DescontoSoma e DescontoPercentual são dois diferentes tipos de Desconto, com implementações distintas para o atributo /valor., Todas as instâncias de DescontoMaior executam a mesma versão de /valor que, por ser atributo derivado, será implementado como um método da classe.

Questão 7

Correto

Atingiu 1,00 de
1,00

Considere a classe *Contro*/ligada por uma associação qualificada por *cpf1* 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. No lugar de *body* deveria-se usar *post*.
- b. Não é possível em OCL escrever expressões sequenciais com ".".
- c. No lugar de *pessoa[umCPF]* deveria se escrever *pessoa->select(umCPF=cpf)*.
- d. Consultas de sistema não podem ter precondições.
- e. A expressão correta seria:

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



Sua resposta está correta.

A resposta correta é: A expressão correta seria:

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

Questão 8

Incorreto

Atingiu 0,00 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()>=idadeMinima)`

b.

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

c.

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

d.

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

 A invariante, conforme escrita deve estar na classe *Pessoa* e não na classe *Video*.

e.

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

Sua resposta está incorreta.

A resposta correta é:

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

Questão 9

Correto

Atingiu 1,00 de
1,00

Considerando os quatro tipos principais de coleção disponíveis em UML, OCL e na maioria das linguagens de programação. Associe cada situação do mundo real com a estrutura mais adequada:

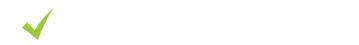
Lista de disciplinas de um curso (sem considerar pré requisitos)

Conjunto



Lista de sócios em uma empresa.

Conjunto



Lista de tarefas a fazer organizadas por prioridade (ex. consertar telhado, dar comida ao gato, cortar a grama, etc).

Conjunto ordenado



Lista dos fatores primos de um número N

Bag



Lista de pacientes na sala de espera

Conjunto ordenado



Lista de compras em um supermercado.

Conjunto



Lista de aprovados no vestibular em um determinado curso.

Conjunto ordenado



Sequência de números de Fibonacci

Sequence



Lista de senhas para atendimento em um banco

Conjunto ordenado



Sua resposta está correta.

A resposta correta é: Lista de disciplinas de um curso (sem considerar pré requisitos) → Conjunto, Lista de sócios em uma empresa. → Conjunto, Lista de tarefas a fazer organizadas por prioridade (ex. consertar telhado, dar comida ao gato, cortar a grama, etc). → Conjunto ordenado, Lista dos fatores primos de um número N → Bag, Lista de pacientes na sala de espera → Conjunto ordenado, Lista de compras em um supermercado. → Conjunto, Lista de aprovados no vestibular em um determinado curso. → Conjunto ordenado, Sequência de números de Fibonacci → Sequence, Lista de senhas para atendimento em um banco → Conjunto ordenado.

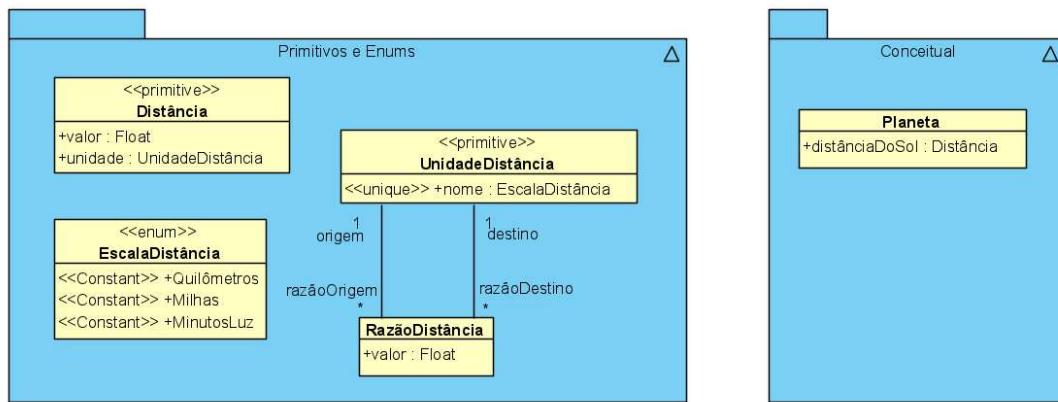
Comentário:

Questão 10

Correto

Atingiu 1,00 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: (assinala 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.
Deletar a instância de UnidadeDistância cujo nome é EscalaDistância::Milhas.
- b. 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.
- c. 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.

- 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.nome atual deve ser substituído por EscalaDistância::Quilômetros.
- 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 atual deve ser substituído pela instância de UnidadeDistância cujo nome é EscalaDistância::Quilômetros. ✓

Sua resposta está correta.

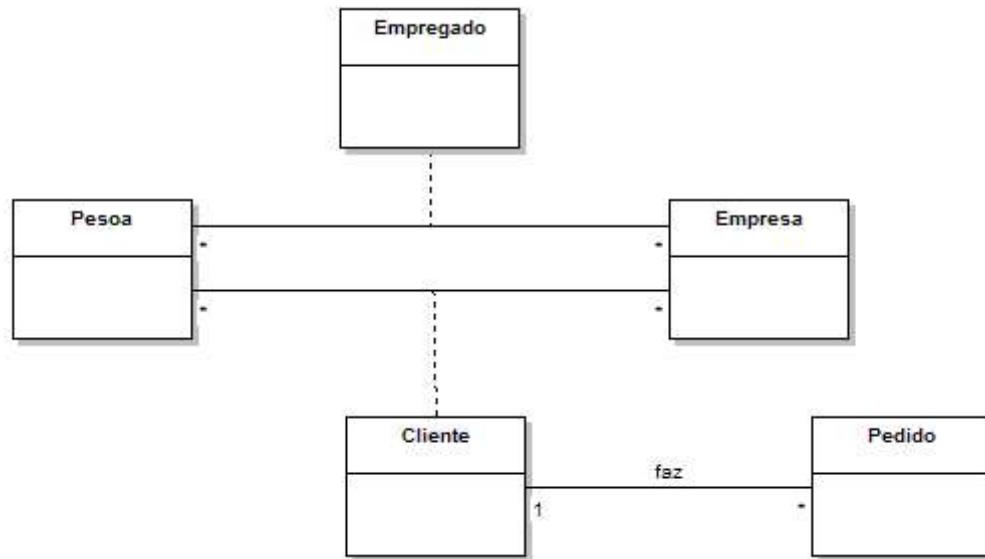
A resposta correta é: No contexto da classe Planeta, pegar o conjunto self.distânciaDoSol.unidade razãoOrigem e encontrar nele a instância "umaRazão" de RazãoDistância cujo destino.nome é EscalaDistância::Quilômetros.
O atributo self.distânciaDoSol.valor deve passar a valer self.distânciaDoSol.valor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

Questão 11

Correto

Atingiu 1,00 de
1,00

Considerando o modelo conceitual abaixo, é possível afirmar que:



Escolha uma opção:

- a. Um Pedido está associado a n empresas.
- b. A classe Empresa deve possuir apenas uma única instância.
- c. Dado um Pedido não é possível saber a qual Empresa ele se associa pela falta de uma associação entre estes elementos.
- d. Para cada instância de Cliente que for criada deve ser também criada uma instância de Empregado, pois estas classes têm uma correspondência de um para um através da classe Pessoa.
- e. É possível que uma mesma Pessoa seja Empregado e Cliente na mesma Empresa e/ou em empresas diferentes. ✓

Sua resposta está correta.

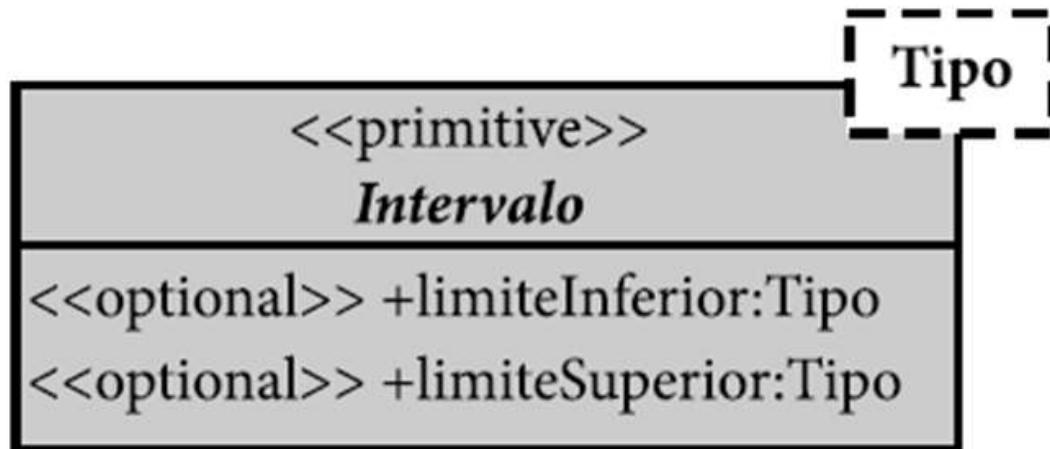
A resposta correta é: É possível que uma mesma Pessoa seja Empregado e Cliente na mesma Empresa e/ou em empresas diferentes.

Questão 12

Correto

Atingiu 1,00 de
1,00

Considerando o padrão Intervalo, representado pela figura abaixo, assinale apenas as razões corretas para usar este padrão.



Escolha uma ou mais:

- a. A interface com usuário fica mais amigável.
- b. Permite implementar o código específico de operações sobre intervalos em uma única classe ao invés de estar espalhado em várias classes. ✓
- c. O código gerado ganha em eficiência de tempo de execução
- d. Aumenta a coesão da classe à qual o atributo pertence. ✓
- e. O padrão é suportado por todos os bancos de dados comerciais.

Sua resposta está correta.

As respostas corretas são: Aumenta a coesão da classe à qual o atributo pertence., Permite implementar o código específico de operações sobre intervalos em uma única classe ao invés de estar espalhado em várias classes.

Questão 13

Correto

Atingiu 1,00 de
1,00

Qual dos grupos de classes abaixo é tipicamente mais bem modelado com o uso de herança (generalização/especialização)?

Escolha uma opção:

- a. Pessoa, PessoaFisica, PessoaJuridica. ✓
- b. Pai, Mãe e Filho.
- c. Pessoa, Funcionário e Cliente.
- d. País, Estado e Município.
- e. Duplicata, DuplicataEmitida e DuplicataPaga.

Sua resposta está correta.

A resposta correta é: Pessoa, PessoaFisica, PessoaJuridica.

Questão 14

Incorreto

Atingiu 0,00 de
1,00

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

Escolha uma opção:

- a. É feito um contrato para cada caso de uso, indicando o que ele produz como resultado para os atores. ✗
- b. Podem ser feitos para cada consulta e comando de sistema. Podem conter pré-condições e contêm necessariamente pós-condições ou resultados dependendo do tipo.
- c. São feitos para cada operação de sistema. Contém a especificação do algoritmo que realiza a operação (usualmente um fluxograma).
- d. São celebrados entre o desenvolvedor e seus clientes para definir o cronograma do desenvolvimento do software e seus custos.
- 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 incorreta.

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 15

Correto

Atingiu 1,00 de
1,00

Uma classe de especificação:

Escolha uma opção:

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

Sua resposta está correta.

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

Questão 16

Incorreto

Atingiu 0,00 de
1,00

Qual das expressões OCL abaixo melhor representa a seguinte pós-condição: "o salário de todos os funcionários que ganham menos de 1000 foi reajustado em 10%"?

Escolha uma opção:

a.

```
self.funcionários->select (f | f.salário@pre < 1000 implies f.setSalário(f.salário * 1.1))
```

b.

```
self.funcionários->collect (f | f.salário@pre < 1000 implies f.setSalário(f.salário@pre * 1.1))
```

c.

```
self.funcionários->collect (f | f.salário < 1000 implies f.setSalário(f.salário@pre * 1.1))
```

d.

```
self.funcionários->forAll (f | f.salário@pre < 1000 implies f.setSalário(f.salário@pre * 1.1))
```

e.

```
self.funcionários->forAll (f | f.salário@pre < 1000 implies f.setSalário(f.salário * 1.1))
```

 faltou usar @pre dentro do parênteses mais interno.

Sua resposta está incorreta.

A resposta correta é:

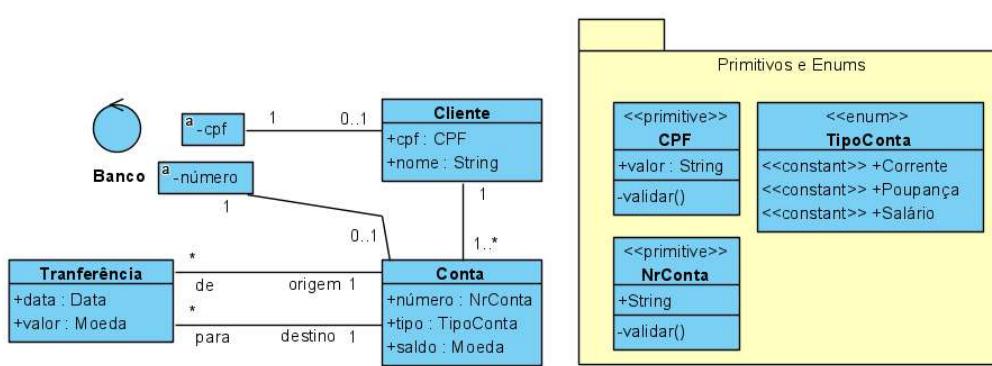
```
self.funcionários->forAll (f | f.salário@pre < 1000 implies f.setSalário(f.salário@pre * 1.1))
```

Questão 17

Correto

Atingiu 1,00 de
1,00

Considere o modelo conceitual abaixo:



Você foi solicitado a desenvolver um contrato para a consulta de sistema *consultaExtrato(cpfcCPF, nrConta:NrConta, dataIncial: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 *dataIncial*. 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 *dataIncial*. 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:D
ate):Tuple
    pre:
        cliente[cpf]->notEmpty() AND
        conta[nrConta]->notEmpty()
    body:
        Tuple {
            movimentações = conta[nrConta].de.union(conta[nrConta].par
a)->sortedBy(data)->select(
                data >= dataInicial
            ).Tuple{
                data = data,
                valor = valor
            }
            saldo = conta[nrConta].saldo
        }
    exception:
        not(cliente[cpf].conta->includes(conta)) implies throw('Esta c
onta não corresponde a este cliente')

```

b.

```

Context Banco::consultaExtrato(cpf:CPF, nrConta:NrConta, dataInicial:D
ate):Tuple
    pre:
        cliente[cpf]->notEmpty() AND
        conta[nrConta]->notEmpty()
    body:
        Tuple {
            movimentações = conta[nrConta].origem.union(conta[nrCont
a].destino)->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 c
onta não corresponde a este cliente')

```

c.

```

Context Banco::consultaExtrato(cpf:CPF, nrConta:NrConta, dataInicial:D
ate):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 c
onta não corresponde a este cliente')

```

- ✓ Esta implementação, além de ser a única correta é também mais elegante, pois aplica o filtro no conjunto de movimentações antes de aplicar a ordenação. As outras opções invertem esta ordem.

d.

```

Context Banco::consultaExtrato(cpf:CPF, nrConta:NrConta, dataInicial:D
ate):Tuple
    pre:
        cliente[cpf]->notNull() AND
        conta[nrConta]->notNull()
    body:
        Tuple {
            movimentações = conta[nrConta].de.union(conta[nrConta].par
a)->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 c
onta não corresponde a este cliente')

```

e.

```

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

```

Sua resposta está correta.

A resposta correta é:

```

Context Banco::consultaExtrato(cpf:CPF, nrConta:NrConta, dataInicial:D
ate):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')

```

Questão 18

Incorreto

Atingiu 0,00 de
1,00

Em relação a contratos de operação de sistema como $operação(x:Tipo)$ é correto afirmar que:

Escolha uma opção:

- a. Mesmo que x possa assumir valores inválidos para a operação, não é necessário definir nem precondição nem exceção no contrato.
- b. Se x pode assumir valores inválidos para a operação então deve ser definida obrigatoriamente uma precondição.  Também poderia ser definida uma exceção.
- c. Se x pode assumir valores inválidos para a operação então deve ser definida obrigatoriamente uma precondição e uma exceção
- d. Se x pode assumir valores inválidos para a operação então deve ser definida obrigatoriamente uma exceção.
- e. Se x pode assumir valores inválidos para a operação então deve ser definida obrigatoriamente uma precondição ou exceção.

Sua resposta está incorreta.

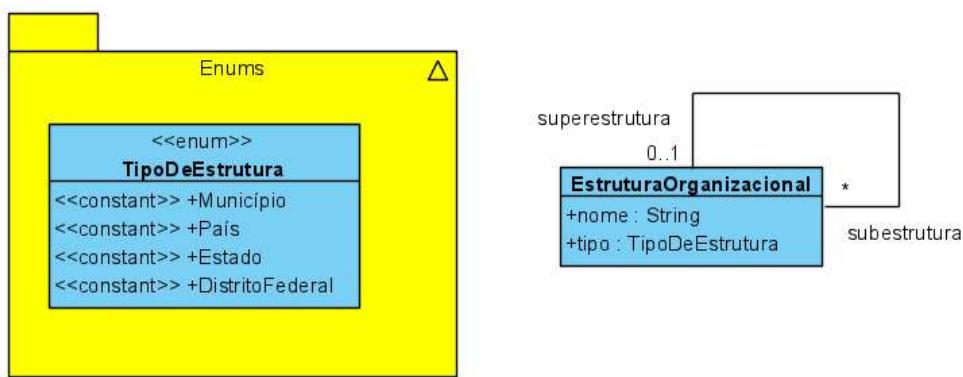
A resposta correta é: Se x pode assumir valores inválidos para a operação então deve ser definida obrigatoriamente uma precondição ou exceção.

Questão 19

Incorreto

Atingiu 0,00 de
1,00

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



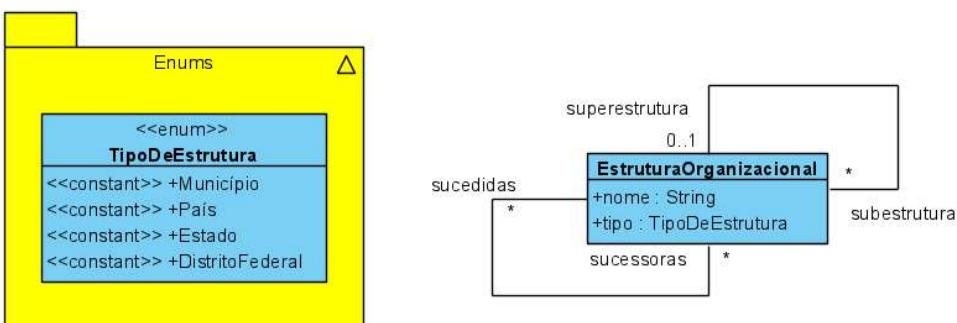
Supondo que se deseja criar um conjunto de regras que definem quais tipos de estruturas podem estar diretamente subordinadas a quais outros tipos, assinale dentre as opções abaixo aquela que melhor permite representar este tipo de requisito.

Escolha uma opção:

a.



b.

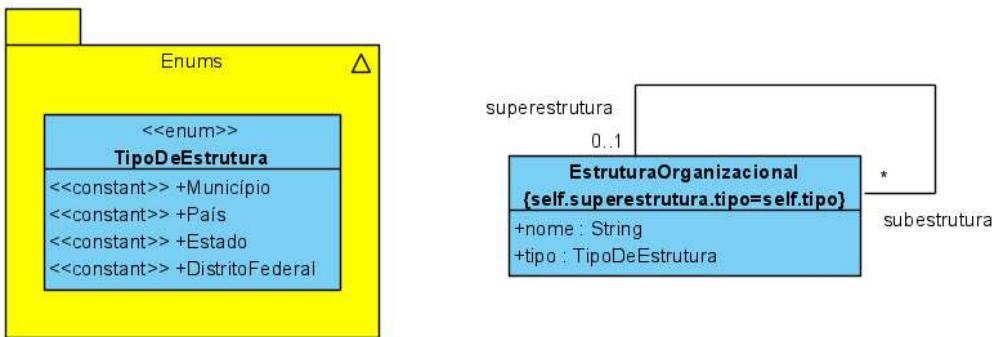


X Este modelo resolve a questão de sucessão temporal das estruturas, mas não a questão das regras de subordinação.

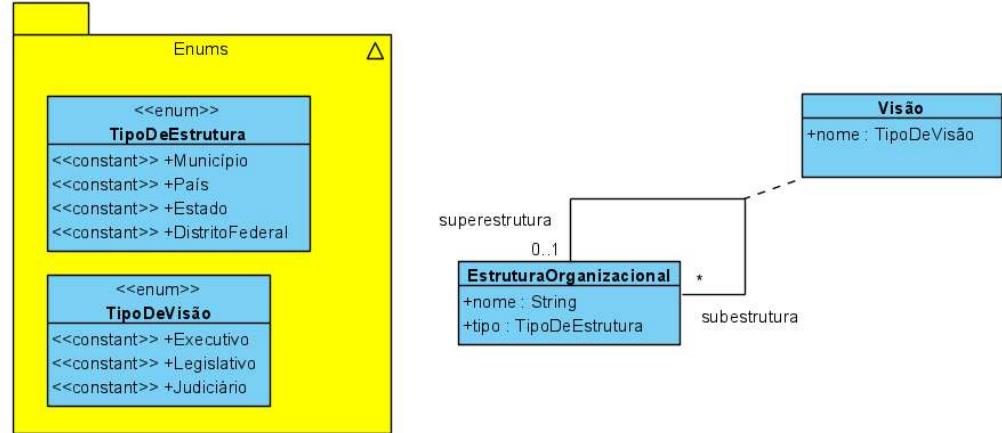
C.



d.



e.



Sua resposta está incorreta.

A resposta correta é:

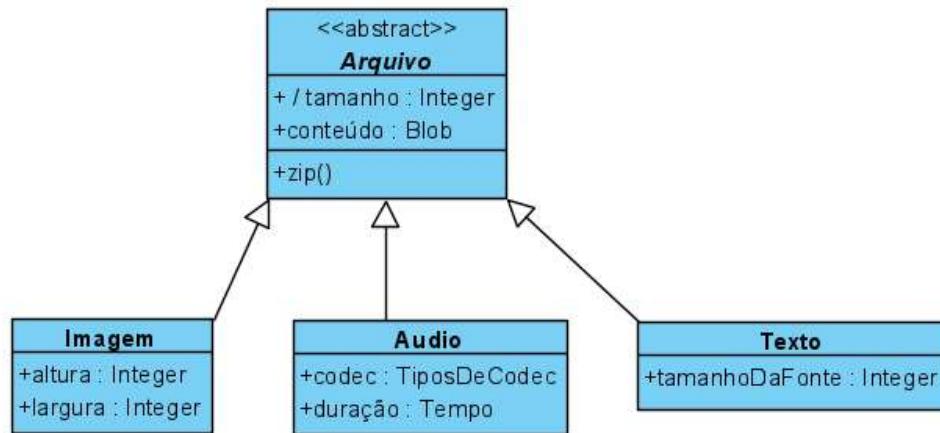


Questão 20

Incorreto

Atingiu 0,00 de
1,00

Ao desenvolver um sistema de gerenciamento de arquivos, um desenvolvedor precisou implementar um módulo para compactação de arquivos, os quais podem ser arquivos de texto, áudio ou imagens. Sua solução é apresentada abaixo:



Supondo que existam vários algoritmos de compactação e que o usuário pode escolher qualquer um deles para compactar seus arquivos, observa-se que a solução acima não é totalmente adequada. Qual padrão de análise deveria ser usado no modelo acima para melhor atender aos requisitos?

Resposta: Classes de especificação X

A resposta correta é: Estratégia

◀ CHAPTER 8

Seguir para...

Lâminas 6-8 ▶