CESGRANRIO | REVISÃO DE RETA FINAL

TI · PROGRAMAÇÃO

XML e Java

Teoria e questões comentadas
da CESGRANRIO

ROTEIRO DA AULA

01 **XML**
Bem-formado · XSD · Validação · Namespaces

02 **Classes e objetos**
Construtor

03 **Modificadores de acesso**

04 **Outros modificadores**
final · abstract · static

05 **Herança e interfaces**

06 **Enums**
values() · valueOf() · ordinal()

destrinchandoTI

Prof. Rogério Araújo
@destrinchandoti

01

MÓDULO 01

XML

Bem-formado · XSD · Validação · Namespaces

XML

EXTENSIBLE MARKUP LANGUAGE

Linguagem de marcação com **tags definidas pelo usuário**, usada para estruturar e trocar dados entre sistemas.

Programação

Anatomia de um pedido

```
<?xml version="1.0" encoding="UTF-8"?>      — Declaração opcional
<pedido>                                    — Elemento raiz
  <item codigo="A17">                       — Atributo codigo
    <nome>Teclado</nome>                    — Elemento filho
  </item>
</pedido>
```

XML bem-formado

- Um **único elemento raiz**
- Tags **fechadas** e corretamente **aninhadas**
- **Maiúsculas e minúsculas** são distintas
- Valores de atributos **entre aspas**

Validação

DTD ou **XSD** define as regras que o documento deve cumprir além da sintaxe.

Namespaces

URIs identificam vocabulários e evitam colisões entre nomes de elementos e atributos.

XSD

XML SCHEMA DEFINITION

Esquema **escrito em XML** que define a estrutura e os valores permitidos em um documento XML.

Programação

Como o XSD declara um item

```
<xs:element name="item">
  <xs:complexType>                                  — Tipo complexo
    <xs:sequence>                                   — Ordem dos filhos
      <xs:element name="nome" type="xs:string"/>
    </xs:sequence>
    <xs:attribute name="codigo" type="xs:string"
                  use="required"/>                  — Obrigatório
  </xs:complexType>
</xs:element>
```

Regras do esquema

- **Estrutura e ordem**: xs:sequence define a ordem dos filhos
- **Tipos e restrições**: xs:string aceita texto; facetas limitam valores
- **Ocorrências**: minOccurs e maxOccurs limitam elementos
- **Atributos**: use controla se são exigidos

Válido para o esquema

```
<item codigo="A17"><nome>Teclado</nome></item>
```

Bem-formado, mas inválido

```
<item><nome>Teclado</nome></item>
```

Falta o atributo obrigatório **codigo**.

Questões de concursos

[CESGRANRIO 2024 Banco da Amazônia – Técnico Científico – Tecnologia da Informação] Um grande banco estabeleceu um novo canal de comunicação com os clientes via um app de mensagens instantâneas para smartphone. Esse app irá interagir com um software de resposta automática já existente por mensagens em XML. Cada mensagem enviada ao software já existente terá o formato do seguinte exemplo:

- <MENSAGEM> <NOME>Nome do Cliente</NOME> <AGENCIA>9999-9</AGENCIA> <CONTA>9999-9</CONTA> <CPF>999.999.999-99</CPF> <TEXTO>Esse é um exemplo de mensagem</TEXTO> </MENSAGEM>

Comentários

- XML:
  - <MENSAGEM>
    - <NOME>Nome do Cliente</NOME>
    - <AGENCIA>9999-9</AGENCIA>
    - <CONTA>9999-9</CONTA>
    - <CPF>999.999.999-99</CPF>
    - <TEXTO>Esse é um exemplo de mensagem</TEXTO>
  - </MENSAGEM>

Comentários

- XSD:
  - <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    - <xsd:element name="MENSAGEM">
      - <xsd:complexType>
        - <xsd:sequence>
          - <xsd:element name="NOME" type="xsd:string"/>
          - <xsd:element name="AGENCIA" type="xsd:string"/>
          - <xsd:element name="CONTA" type="xsd:string"/>
          - <xsd:element name="CPF" type="xsd:string"/>
          - <xsd:element name="TEXTO" type="xsd:string"/>
        - </xsd:sequence>
      - </xsd:complexType>
    - </xsd:element>
  - </xsd:schema>

Questões de concursos

[CESGRANRIO 2024 Banco da Amazônia – Técnico Científico – Tecnologia da Informação] Qual esquema XSD define corretamente a estrutura dessa mensagem?

- [A] <!ELEMENT MENSAGEM (NOME, AGENCIA, CONTA, CPF, TEXTO)> <!ELEMENT NOME (#PCDATA)> <!ELEMENT AGENCIA (#PCDATA)> <!ELEMENT CONTA (#PCDATA)> <!ELEMENT CPF (#PCDATA)> <!ELEMENT TEXTO (#PCDATA)>
- [B] <!DOCTYPE MENSAGEM [<!ELEMENT MENSAGEM (NOME, AGENCIA, CONTA, CPF, TEXTO)> <!ELEMENT NOME (#PCDATA)> <!ELEMENT AGENCIA (#PCDATA)> <!ELEMENT CONTA (#PCDATA)> <!ELEMENT CPF (#PCDATA)> <!ELEMENT TEXTO (#PCDATA)>]>

Questões de concursos

[CESGRANRIO 2024 Banco da Amazônia – Técnico Científico – Tecnologia da Informação] Qual esquema XSD define corretamente a estrutura dessa mensagem?

- [C] <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"> <xsd:element name="MENSAGEM"> <xsd:complexType> <xsd:sequence> <xsd:element name="NOME" type="xsd:string"/> <xsd:element name="AGENCIA" type="xsd:string"/> <xsd:element name="CONTA" type="xsd:string"/> <xsd:element name="CPF" type="xsd:string"/> <xsd:element name="TEXTO" type="xsd:string"/> </xsd:sequence> </xsd:complexType> </xsd:element> </xsd:schema>

Questões de concursos

[CESGRANRIO 2024 Banco da Amazônia – Técnico Científico – Tecnologia da Informação] Qual esquema XSD define corretamente a estrutura dessa mensagem?

- [D] <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"> <xsd:element name="MENSAGEM"> <xsd:complexType> <xsd:sequence> <xsd:element name="NOME" type="xsd:string"/> <xsd:element name="AGENCIA" type="xsd:integer"/> <xsd:element name="CONTA" type="xsd:integer"/> <xsd:element name="CPF" type="xsd:integer"/> <xsd:element name="TEXTO" type="xsd:string"/> </xsd:sequence> </xsd:complexType> </xsd:element> </xsd:schema>

Questões de concursos

[CESGRANRIO 2024 Banco da Amazônia – Técnico Científico – Tecnologia da Informação] Qual esquema XSD define corretamente a estrutura dessa mensagem?

- [E] <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"> <xsd:element name="MENSAGEM"> <xsd:sequence> <xsd:element name="NOME" type="xsd:string"/> <xsd:element name="AGENCIA" type="xsd:integer"/> <xsd:element name="CONTA" type="xsd:integer"/> <xsd:element name="CPF" type="xsd:integer"/> <xsd:element name="TEXTO" type="xsd:string"/> </xsd:sequence> </xsd:element> </xsd:schema>

Questões de concursos

[CESGRANRIO 2024 Banco da Amazônia – Técnico Científico – Tecnologia da Informação] Qual esquema XSD define corretamente a estrutura dessa mensagem?

- [C] <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"> <xsd:element name="MENSAGEM"> <xsd:complexType> <xsd:sequence> <xsd:element name="NOME" type="xsd:string"/> <xsd:element name="AGENCIA" type="xsd:string"/> <xsd:element name="CONTA" type="xsd:string"/> <xsd:element name="CPF" type="xsd:string"/> <xsd:element name="TEXTO" type="xsd:string"/> </xsd:sequence> </xsd:complexType> </xsd:element> </xsd:schema>

Questões de concursos

[CESGRANRIO 2024 Caixa – Técnico Bancário Novo – Tecnologia da Informação] Um programador foi encarregado de determinar se um determinado documento XML pode ser carregado em um sistema que integra dados recebidos de vários outros sistemas. Para isso, ele deveria verificar se o documento é válido e bem-formado, porém não tinha acesso ao esquema do documento.

Questões de concursos

[CESGRANRIO 2024 Caixa – Técnico Bancário Novo – Tecnologia da Informação] Considerando-se esse contexto, o programador

- [A] poderá ler, validar e verificar se o documento é bem-formado.
- [B] poderá ler o documento e poderá validá-lo, mas não poderá verificar se o documento é bem-formado.
- [C] poderá ler o documento e verificar se o documento é bem-formado, mas não poderá validá-lo.

Questões de concursos

[CESGRANRIO 2024 Caixa – Técnico Bancário Novo – Tecnologia da Informação] Considerando-se esse contexto, o programador

- [D] poderá ler o documento, mas não poderá validá-lo nem verificar se o documento é bem-formado.
- [E] não poderá ler o documento XML sem o esquema fornecer a autorização necessária.

Questões de concursos

[CESGRANRIO 2024 Caixa – Técnico Bancário Novo – Tecnologia da Informação] Considerando-se esse contexto, o programador

- [C] poderá ler o documento e verificar se o documento é bem-formado, mas não poderá validá-lo.

02

MÓDULO 02

Java

Classes | Acesso | final, abstract, static | Herança e interfaces | Enums

Classes

CLASSES E OBJETOS EM JAVA

A classe é o **molde**; o objeto é a **instância** criada a partir dele. Exemplo de toda a revisão: a classe **Conta**.

Java

Anatomia da classe Conta

```
public class Conta {                                    — Classe
    private double saldo;                               — Atributo
    public Conta(double saldoInicial) {                 — Construtor
        this.saldo = saldoInicial;
    }
    public double getSaldo() {                          — Método
        return saldo;
    }
}
```

Construtor

- Tem o **mesmo nome** da classe
- **Não declara** tipo de retorno
- Com **void**, vira um método comum
- **Inicializa** o estado do objeto

Classe

Tipo que descreve o **estado** (atributos) e o **comportamento** (métodos) de seus objetos.

Objeto

Uma **instância** da classe, criada em tempo de execução.

Exemplo

AbsoluteAndCanonicalPathExample.java

```java
package com.journaldev.examples;

import java.io.File;

public class AbsoluteAndCanonicalPathExample {

    public static void main(String[] args) throws IOException {
        File file = new File("/Users/pankaj/source.txt");
        File file1 = new File("/Users/pankaj/temp/../source.txt");

        System.out.println("Absolute Path : " + file.getAbsolutePath());
        System.out.println("Canonical Path : " + file.getCanonicalPath());

        System.out.println("Absolute Path : " + file1.getAbsolutePath());
        System.out.println("Canonical Path : " + file1.getCanonicalPath());

    }

}
```

Problems Javadoc Declaration Search Console Progress Call Hiera

```
<terminated> AbsoluteAndCanonicalPathExample (1) [Java Application] /Library/Java/JavaVirtualMachin
Absolute Path : /Users/pankaj/source.txt
Canonical Path : /Users/pankaj/source.txt
Absolute Path : /Users/pankaj/temp/../source.txt
Canonical Path : /Users/pankaj/source.txt
```

package

import

class

Acesso

MODIFICADORES DE ACESSO

Definem de onde um membro da classe pode ser acessado, do **mais aberto** (public) ao **mais restrito** (private).

Java

Visibilidade de membros de classe

Modificador	Na própria classe	No mesmo pacote	Em subclasse de outro pacote	Em classe não relacionada de outro pacote
`public`	✓ Sim	✓ Sim	✓ Sim	✓ Sim
`protected`	✓ Sim	✓ Sim	✓ Sim*	× Não
`pacote` (sem modificador)	✓ Sim	✓ Sim	× Não	× Não
`private`	✓ Sim	× Não	× Não	× Não

* **Nota sobre protected:** em outro pacote, a subclasse acessa membros de instância por **this**, **super** ou por referência cujo tipo declarado seja a própria subclasse ou um subtipo dela.

Exemplo da Conta

`c1.getSaldo()` compila: o método é **public**. `c1.saldo` dá erro: o campo é **private**.

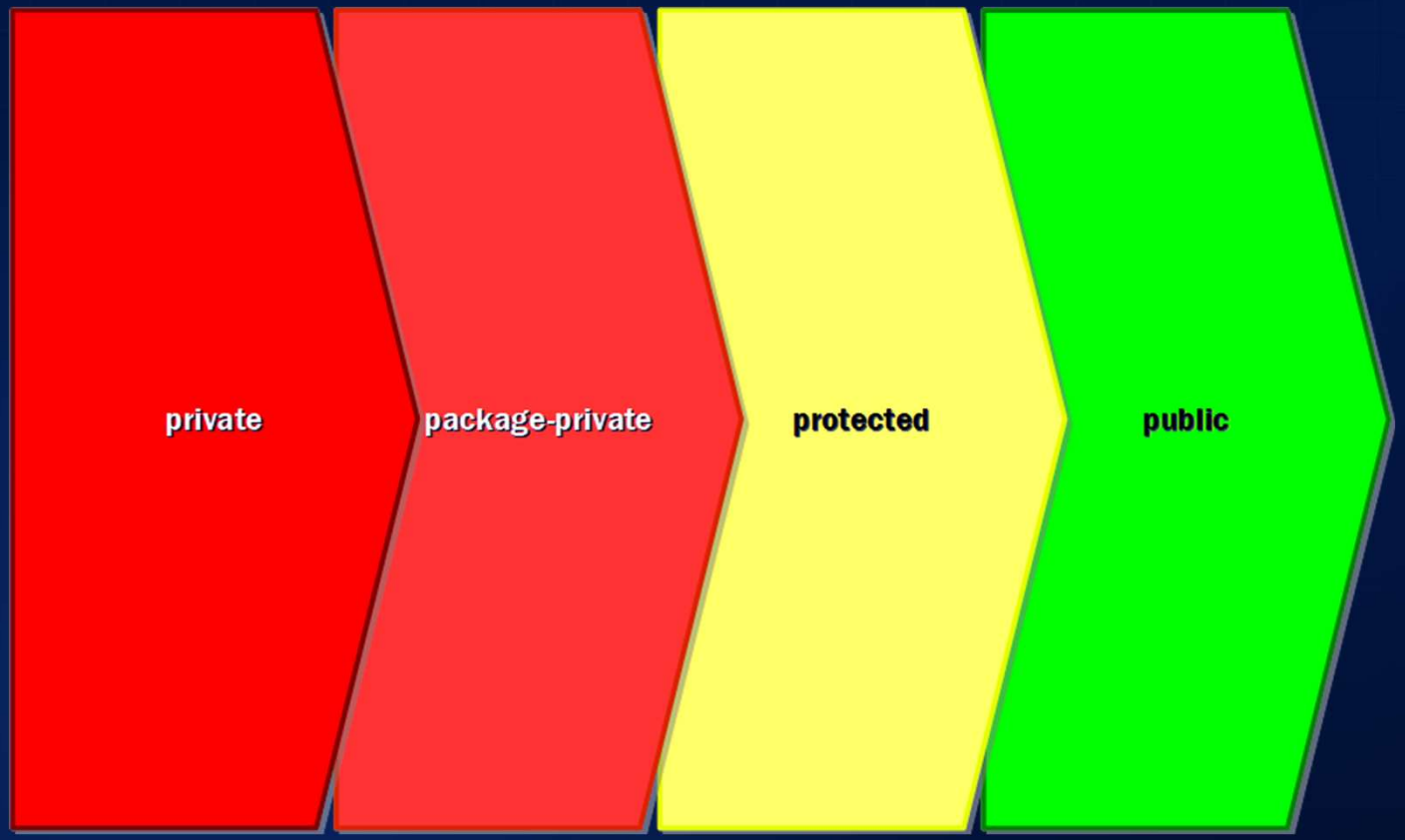
Observação de prova

Uma classe de **nível superior** só pode ser **public** ou ter **acesso de pacote**.

Esquema 1

Modificador	Modificador	Classe	Pacote	Subclasse	Todos
public	Público	✓	✓	✓	✓
protected	Protegido	✓	✓	✓	✗
package-private	Pacote	✓	✓	✗	✗
private	Privado	✓	✗	✗	✗

Esquema 2



Questões de concursos

[CESGRANRIO 2024 UNEMAT – Analista de Sistemas] Considere um tipo de visibilidade que pode ser usado em um método de objeto de uma classe P para permitir acesso nas seguintes situações:

- [i] quando uma classe Q estender da classe P; e
- [ii] quando uma classe R faz parte do mesmo pacote da classe P.

Questões de concursos

[CESGRANRIO 2024 UNEMAT – Analista de Sistemas] Assim, tanto objetos do tipo da classe Q como da classe R poderão acessar o método que está na classe P. Na situação descrita, o tipo de visibilidade a ser usado é o

- [A] static
- [B] public
- [C] private
- [D] package
- [E] protected

Questões de concursos

[CESGRANRIO 2024 UNEMAT – Analista de Sistemas] Assim, tanto objetos do tipo da classe Q como da classe R poderão acessar o método que está na classe P. Na situação descrita, o tipo de visibilidade a ser usado é o

- [A] static
- [B] public
- [C] private
- [D] package
- [E] protected

Outros modificadores

Para classes

- final
- abstract

Para atributos

- final
- static

Para métodos

- final
- abstract
- static

Para classes

final	abstract
Torna uma classe	Indica uma classe abstrata que
Impossível de ser estendida	Pode ter ou não métodos abstratos
	Não pode ser instanciada, mas pode ter subclasses

Para atributos

final

Torna um atributo

Imutável

static

Faz com que o atributo pertença à classe

Não a uma instância específica

Para métodos

final

- Impede que o método
- Seja sobrescrito em subclasses

abstract

- Declara um método sem corpo
- Que deve ser implementado pelas subclasses

static

- Permite que o método seja chamado
- Sem criar uma instância da classe

Exemplos de classes

Classe package-private

- class Classe {
  - ...
- }

Classe que é subclasse de outra

- public class Classe extends SuperClasse {
  - ...
- }

Exemplos de classes

Classe que implementa uma interface

- class Classe implements Interface {
  - …
- }

Classe que subclasse de outra e implementa mais de uma interface

- public class Classe extends SuperClasse implements IF1, IF2 {
  - …
- }

Exemplos de classes

Classe final que não pode ser estendida

- public final class Classe {
  - ...
- }

Classe abstrata que não pode ser instanciada

- public abstract class Classe {
  - ...
- }

Exemplo de herança

- Código da superclasse:
  - public class Superclasse {
    - public void metodo(String x) {
      - System.out.println(x);
    - }
  - }
- Código da subclasse:
  - public class Subclasse extends Superclasse {
    - @Override
    - public void metodo(String x) {
      - System.out.println("Valor: " + x);
    - }
  - }

Questões de concursos

[CESGRANRIO 2024 UNEMAT – Analista de Sistemas] Na linguagem Java, para impedir que um método criado em uma classe J não possa ser sobrescrito por alguma outra classe K que herde de J, deve-se usar a palavra

- [A] extends
- [B] final
- [C] implements
- [D] super
- [E] void

Questões de concursos

[CESGRANRIO 2024 UNEMAT – Analista de Sistemas] Na linguagem Java, para impedir que um método criado em uma classe J não possa ser sobrescrito por alguma outra classe K que herde de J, deve-se usar a palavra

- [A] extends
- [B] final
- [C] implements
- [D] super
- [E] void

Questões de concursos

[CESGRANRIO 2024 Banco da Amazônia – Técnico Científico – Tecnologia da Informação] Considere um sistema bancário em Java que possui a classe Cliente e suas subclasses, ClientePessoaFisica e ClientePessoaJuridica, onde Cliente é uma classe abstrata. Nesse sistema, um método getDesconto(valor) deve fornecer o cálculo do desconto para um tipo de cliente, de forma que os clientes do tipo pessoa física e os clientes do tipo pessoa jurídica tenham descontos diferenciados. Suponha que, utilizando corretamente os mecanismos associados à herança e ao polimorfismo, se deseje implementar essas classes de modo que o método getDesconto possa ser aplicado indistintamente a qualquer instância que tenha sido declarada como da classe Cliente.

Questões de concursos

[CESGRANRIO 2024 Banco da Amazônia – Técnico Científico – Tecnologia da Informação] Para atender a essa condição, a implementação dessas classes deve possuir o método getDesconto

- [A] apenas na classe Cliente, identificando a qual subclasse o cliente pertence e calculando o desconto por meio de uma estrutura se-então dentro da implementação.
- [B] apenas nas classes ClientePessoaFisica e ClientePessoaJuridica, pois não há instâncias da classe Cliente no sistema, já que é uma classe abstrata.

Questões de concursos

[CESGRANRIO 2024 Banco da Amazônia – Técnico Científico – Tecnologia da Informação] Para atender a essa condição, a implementação dessas classes deve possuir o método getDesconto

- [C] em todas as três classes, sendo possível, nesse caso, que a função getDesconto da classe Cliente seja abstrata.
- [D] fora das três classes, dado que uma estrutura do tipo se-então deve ser usada para descobrir qual é a classe adequada.

Questões de concursos

[CESGRANRIO 2024 Banco da Amazônia – Técnico Científico – Tecnologia da Informação] Para atender a essa condição, a implementação dessas classes deve possuir o método getDesconto

- [E] implementada totalmente em todas as classes, sendo que a função getDesconto da classe Cliente chama a função getDesconto das classes ClientePessoaFisica e ClientePessoaJuridica de acordo com a necessidade.

Questões de concursos

[CESGRANRIO 2024 Banco da Amazônia – Técnico Científico – Tecnologia da Informação] Para atender a essa condição, a implementação dessas classes deve possuir o método getDesconto

- [C] em todas as três classes, sendo possível, nesse caso, que a função getDesconto da classe Cliente seja abstrata.

Interface Animal

- public interface Animal {
  - void emitirSom();
- }

Classe Cachorro

```
public class Cachorro implements Animal {
    @Override
    public void emitirSom() {
        System.out.println("Cachorro latindo");
    }
}
```

Classe Gato

```
public class Gato implements Animal {
    @Override
    public void emitirSom() {
        System.out.println("Gato miando");
    }
}
```

Exemplo geral

- Código:
  - public class Main {
    - public static void main(String[] args) {
      - Animal animal;
      - animal = new Cachorro();
      - animal.emitirSom();
      - animal = new Gato();
      - animal.emitirSom();
    - }
  - }

- Resultado da execução:
  - Cachorro latindo
  - Gato miando

Questões de concursos

[CESGRANRIO 2024 UNEMAT – Analista de Sistemas] É uma característica da linguagem Java conseguir representar a

- [A] codificação de métodos completos em interfaces
- [B] herança múltipla de interface
- [C] herança múltipla de classe
- [D] instanciação de classes abstratas
- [E] instanciação de interfaces

Questões de concursos

[CESGRANRIO 2024 UNEMAT – Analista de Sistemas] É uma característica da linguagem Java conseguir representar a

- [A] codificação de métodos completos em interfaces
- [B] herança múltipla de interface
- [C] herança múltipla de classe
- [D] instanciação de classes abstratas
- [E] instanciação de interfaces

Questões de concursos

[CESGRANRIO 2023 Banco do Brasil – Agente de Tecnologia] Um programador foi instruído pelo seu gerente a implementar, em Java, uma classe MemoriaCalculoVenda que implementasse a interface MemoriaCalculo, já criada pela organização e que representa as exigências da organização para classes que implementam memórias de cálculo.

Questões de concursos

[CESGRANRIO 2023 Banco do Brasil – Agente de Tecnologia] Nesse cenário, com que fragmento de código o programador deve começar, de forma correta, a implementação da classe?

- [A] class MemoriaCalculoVenda extends MemoriaCalculo
- [B] class MemoriaCalculoVenda implements MemoriaCalculo
- [C] class MemoriaCalculoVenda imports MemoriaCalculo
- [D] class MemoriaCalculoVenda inherits MemoriaCalculo
- [E] class MemoriaCalculoVenda uses MemoriaCalculo

Questões de concursos

[CESGRANRIO 2023 Banco do Brasil – Agente de Tecnologia] Nesse cenário, com que fragmento de código o programador deve começar, de forma correta, a implementação da classe?

- [A] class MemoriaCalculoVenda extends MemoriaCalculo
- [B] class MemoriaCalculoVenda implements MemoriaCalculo
- [C] class MemoriaCalculoVenda imports MemoriaCalculo
- [D] class MemoriaCalculoVenda inherits MemoriaCalculo
- [E] class MemoriaCalculoVenda uses MemoriaCalculo

Enums

TIPOS ENUMERADOS EM JAVA

Um enum define um tipo com um **conjunto fixo de constantes** nomeadas, verificado em tempo de compilação.

Java

Anatomia do enum TipoConta

```
public enum TipoConta {                          — Tipo enum
    CORRENTE(0.0), POUPANCA(0.005);              — Constantes
    private final double rendimento;             — Atributo
    TipoConta(double r) {                        — Construtor
        this.rendimento = r;
    }
    public double getRendimento() { ... }        — Método
}
```

Regras de prova

- Constantes são **public static final** implícitas
- Construtor é **private** implícito
- **Não** se instancia com **new**
- Constantes vêm **primeiro**; com membros, a lista termina em **;**

Uso

`TipoConta t = TipoConta.POUPANCA;` A variável só aceita as constantes do enum (ou **null**).

Comparação

Cada constante é uma **instância única**, então **==** é seguro para comparar.

Métodos

RECURSOS DOS ENUMS

Todo enum estende **java.lang.Enum** implicitamente, herda seus métodos e pode ser usado em **switch**.

Java

Métodos usados em prova

```
TipoConta.values()                          — Todas, em ordem
TipoConta.valueOf("CORRENTE")               — Busca pelo nome
POUPANCA.name()                             — "POUPANCA"
POUPANCA.ordinal()                          — 1 (começa em 0)
CORRENTE.compareTo(POUPANCA)                — Negativo
switch (t) {
    case POUPANCA: render(); break;         — Nome simples
}
```

Pegadinhas

- **valueOf** com nome inexistente lança **IllegalArgumentException**
- **valueOf** diferencia maiúsculas
- **ordinal()** começa em **0**
- No **case**, o nome qualificado só é aceito a partir do **Java 21**

Pode

Ter **atributos**, **construtores** e **métodos**, e **implementar interfaces**.

Não pode

Estender outra classe, pois já estende java.lang.Enum, nem ser instanciado com **new**.

Questões de concursos

[CESGRANRIO 2024 BNDES – Analista – Análise de Sistemas – Desenvolvimento] Um desenvolvedor Java 21 precisa implementar uma enumeração chamada Dia Da Semana, que representa os dias da semana, de forma a garantir que esteja acessível em todos os pacotes usados no sistema.

Questões de concursos

[CESGRANRIO 2024 BNDES – Analista – Análise de Sistemas – Desenvolvimento] Considerando-se esse contexto, qual extrato de código se deve usar para fazer essa implementação?

- [A] enum DiaDaSemana {SEGUNDA, TERCA, QUARTA, QUINTA, SEXTA, SABADO, DOMINGO}
- [B] enumeration DiaDaSemana {SEGUNDA, TERCA, QUARTA, QUINTA, SEXTA, SABADO, DOMINGO}
- [C] public enum DiaDaSemana {SEGUNDA, TERCA, QUARTA, QUINTA, SEXTA, SABADO, DOMINGO}

Questões de concursos

[CESGRANRIO 2024 BNDES – Analista – Análise de Sistemas – Desenvolvimento] Considerando-se esse contexto, qual extrato de código se deve usar para fazer essa implementação?

- [D] enum DiaDaSemana: {SEGUNDA, TERCA, QUARTA, QUINTA, SEXTA, SABADO, DOMINGO}
- [E] public enumeration DiaDaSemana {SEGUNDA, TERCA, QUARTA, QUINTA, SEXTA, SABADO, DOMINGO}

Questões de concursos

[CESGRANRIO 2024 BNDES – Analista – Análise de Sistemas – Desenvolvimento] Considerando-se esse contexto, qual extrato de código se deve usar para fazer essa implementação?

- [C] public enum DiaDaSemana {SEGUNDA, TERCA, QUARTA, QUINTA, SEXTA, SABADO, DOMINGO}

CESGRANRIO | REVISÃO DE RETA FINAL

XML E JAVA

Obrigado!

Bons estudos e
bora para a aprovação.

ANTES DA PROVA, REVISE

1. **Bem-formado × válido**: sem esquema, só dá para verificar se é bem-formado.
2. A tabela de **modificadores de acesso** e a nota do **protected**.
3. **final**, **abstract** e **static** em classes, atributos e métodos.
4. Java: herança **simples** de classes, **múltipla** de interfaces.
5. Enums: constantes **public static final**, construtor **private**, **ordinal()** começa em 0.

@destrinchandoti | @profrogeraoaraujo | rogeraoaraujo.com.br

destrinchandoTI

Prof. Rogério Araújo
@destrinchandoti