PROF. ALEXANDRE DE SOUZA JR.
@alexandrejunior.dev

TREINA RECIFE
DESENVOLVIMENTO GERENCIAL E PROFISSIONAL

LISTA DE EXERCÍCIOS - ORIENTAÇÃO A OBJETOS

Questão 1: Transporte

Implemente a classe abaixo:

Transporte
- id : Integer - ano : Integer - modelo : String - cargaMaxima : Double - potencia : Double
+ getters *+ setters* + calcularConsumo() : Double

(a) O método **calcularConsumo()** deve retornar **a potência do motor x carga máxima (kg) x 100**.

(b) Adicione na classe o construtor padrão e um construtor que receba todos os parâmetros para inicializar os dados de um transporte.

Questão 2: Produto

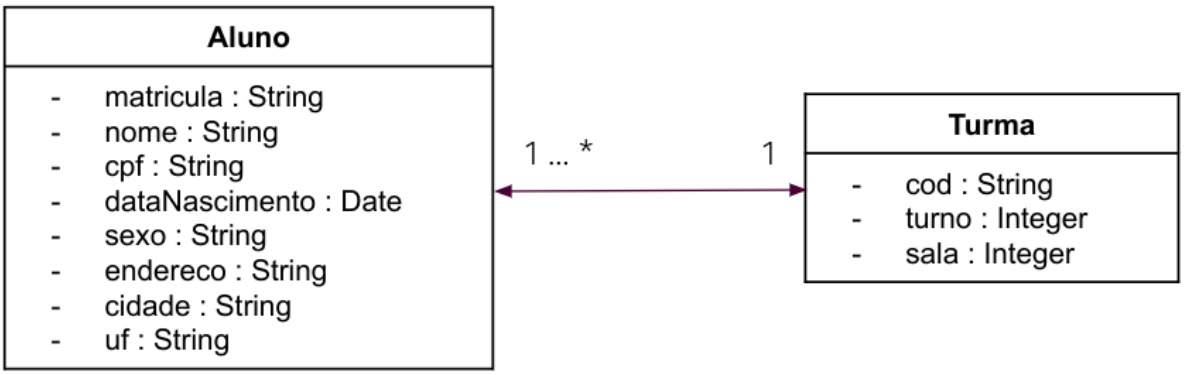
Implemente a classe abaixo:

Produto
- id : Integer - nome : String - descricao : String - validade : LocalDate
+ getters *+ setters* + calcularDiasRestantes() : Integer

O método **calcularDiasRestantes()** deve retornar a quantidade de dias que restam para o prazo de validade do produto, conforme definido no atributo **validade**.

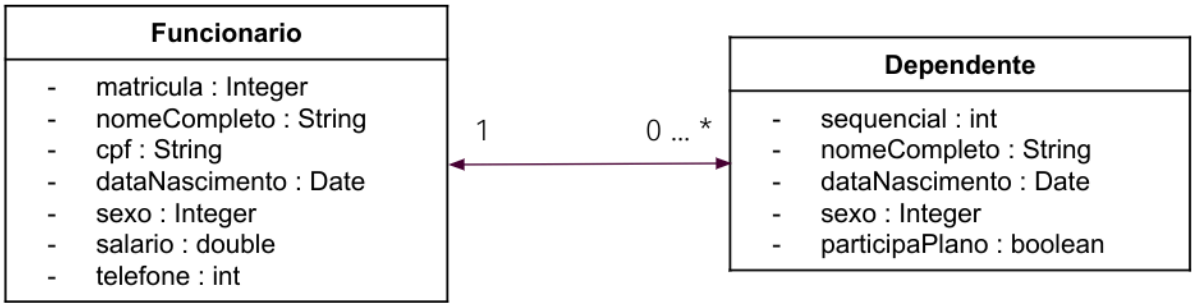
Questão 3: Aluno e Turma

Implemente o modelo abaixo:



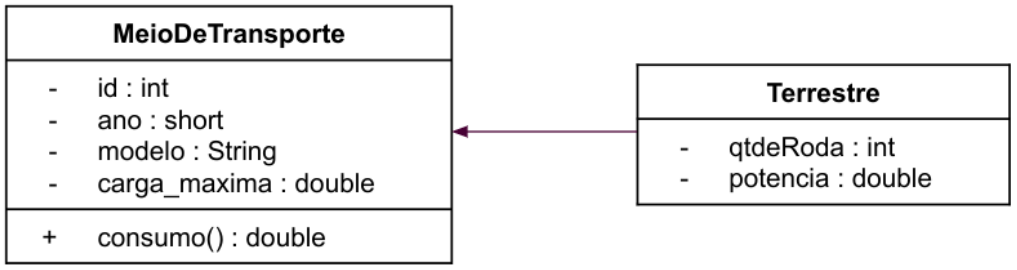
Questão 4: Funcionário e Dependente

Implemente o modelo abaixo:



Questão 5: Meio de Transporte

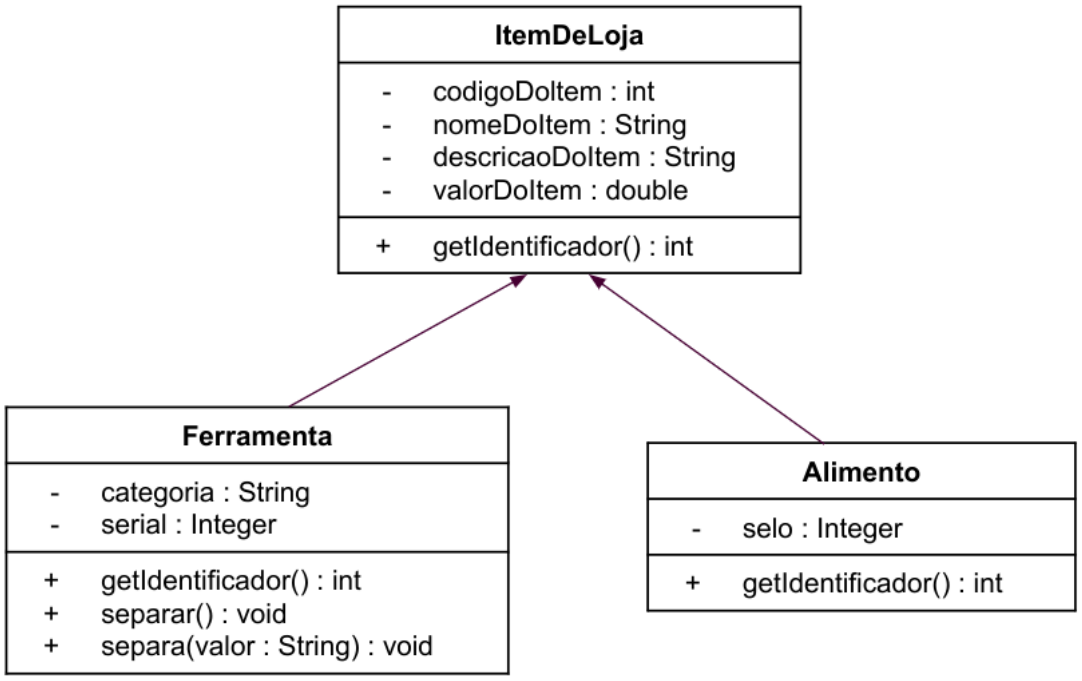
Implemente o modelo abaixo:



O método **consumo()** deve retornar o consumo médio do transporte. Para Terrestre, o consumo médio representa a **potência do motor x carga máxima (kg) x 100**.

Questão 6: Item de Loja

Implemente o modelo abaixo:

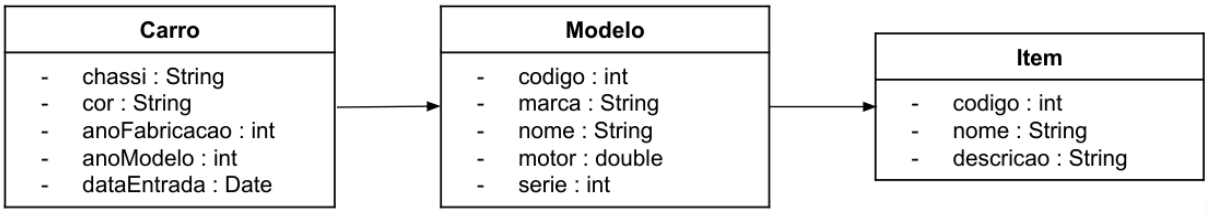


(a) O método **getIdentificador()** deve retornar o identificador da classe instanciada, ou seja, para um **ItemDeLoja** deverá retornar o "codigoDoItem", para um **Alimento** o "selo" e para uma **Ferramenta** o "serial".

(b) O método **separar()** deve imprimir a categoria do produto. Se essa informação não for passada, a categoria a ser impressa deverá ser a categoria "Outros".

Questão 7: Carro, Modelo e Item

Implemente o modelo abaixo:



Questão 8: Conceitos POO I

Sobre as características da Programação Orientada a Objetos (POO) em Java, assinale verdadeiro (V) ou falso (F):

	Java possui menos de 9 tipos primitivos.
	Não é possível comparar tipos primitivos utilizando o operador “==”.
	Atributos de interface são sempre *final*.
	Palavras reservadas do Java são aquelas que não podem ser usadas para nomear membros de classes.
	Em Java toda variável, sem exceção, deve ser declarada.
	As subclasses podem adicionar membros próprios.
	Variáveis polimórficas podem referenciar objetos de uma classe (subclasse) da superclasse declarada.
	Em métodos, o modificador *abstract* obriga que suas subclasses não abstratas implementem o método.
	Um método definido como *final* pode ser sobreposto apenas por uma classe descendente.
	Classe com modificador *final* só pode ser especializada por herança.
	O modificador *final* estabelece que um atributo não pode ter seu valor modificado.
	A visibilidade definida por *public* permite que um determinado atributo seja acessível a partir de quaisquer métodos, objetos e classes.
	O modificador *protected* não restringe acesso oriundo de outro pacote.
	Os atributos e métodos privados de uma classe são acessíveis apenas nos métodos da própria classe.
	A estrutura *switch* aceita qualquer tipo de dado primitivo do Java.
	Encapsulamento consiste em proteger os atributos de acessos e modificações não controladas, centralizando o gerenciamento e a validação dos dados antes de serem armazenados pelos objetos.
	O encapsulamento permite definir o grau de visibilidade dos atributos de uma classe, estabelecendo restrições e permissões por métodos ao sistema.
	Não se implementa o encapsulamento em interface.
	A implementação de uma interface obriga a classe a implementar todos os métodos definidos, a não ser que a classe seja definida como

	abstrata, podendo assim deixar a implementação para as suas subclasses não abstratas.
	O polimorfismo de sobrecarga pode ser utilizado para distinguir, em uma determinada classe, dois métodos com o mesmo nome, mas com parâmetros diferentes.
	Polimorfismo é o conceito que define que mais vários métodos, com o mesmo nome, podem Implementar diferentes formas de executar, dependendo de como ele é acionado.
	Override é um tipo de Polimorfismo que só ocorre em caso de herança.
	Na herança, todos os atributos são herdados, inclusive os privados.
	Em Java as subclasses herdam atributos e métodos da classe *Object*.
	Uma classe Java pode herdar de uma única classe na herança simples, e de várias na herança múltipla.
	Além de herdar entidades de sua classe-pal, uma classe derivada pode modificar métodos herdados, inclusive podendo até acrescentar novas entidades, sem afetar a estrutura da classe que a originou.

Questão 9: Conceitos POO II

Marque a afirmativa **incorreta**:

(a) Encapsulamento representa a proteção das propriedades referentes a um determinado objeto.
(b) O encapsulamento permite concentrar as regras de negócio da classe dentro dela mesma.
(c) Encapsulamento representa a proteção das propriedades referentes a um determinado objeto.
(d) O encapsulamento é obrigatório para os atributos de uma classe.
(e) Um atributo encapsulado não pode ser acessado diretamente por outro objeto ou classe.
(f) Os métodos de acesso do encapsulamento (getters/setters) não devem ser privados.

Questão 10: Conceitos POO III

A habilidade de duas ou mais classes, derivadas da mesma superclasse, responderem à mesma solicitação cada qual à sua maneira, é o conceito de:

(a) Simetria
(b) Abstração

(c) Polimorfismo
(d) Encapsulamento
(e) Reutilização
(f) Herança

Questão 11: Palavras Reservadas

A respeito das palavras reservadas da linguagem Java, associe a coluna da esquerda à coluna da direita.

(A) super	(1) Método sem retorno
(B) default	(2) Define constantes
(C) new	(3) Tipo de dado
(D) this	(4) Implementa herança
(E) throw	(5) Declara importação
(F) void	(6) Implementa uma interface
(G) int	(7) Classe
(H) implements	(8) Incremento
(I) ++	(9) Laço de repetição
(J) class	(10) Levanta uma exceção
(K) get	(11) Sem significador
(L) static	(12) Instancia a class
(M) public	(13) Encerra a execução do método
(N) return	(14) Define membro de classe
(O) String	(15) Definição de uma classe
(P) abstract	(16) Auto referência
(Q) do	(17) Referencia classe pai
(R) extends	(18) Define classe sem objeto
(S) final	(19) *Else* do switch
(T) import	(20) Modificador de visibilidade

Questão 12: Classe vs. Objeto

Diferencie com suas palavras: Classe e Objeto.

Questão 13: Métodos

O que compõe a assinatura de um método? Exemplifique.

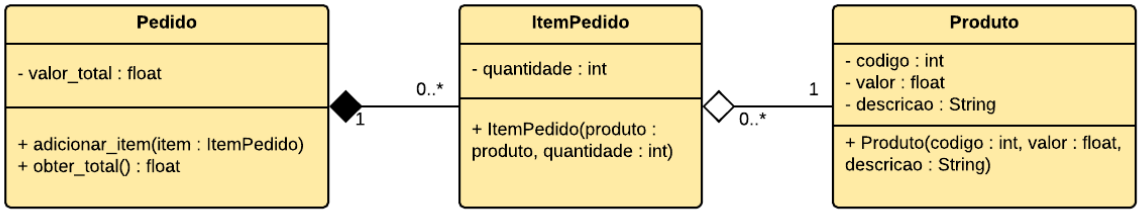
Questão 14: Polimorfismo

A partir da classe abaixo, extraia o seu padrão/modelo (superclasse/interface). Atente aos métodos herdados, explicitados pela anotação **@Override**.

```
import java.util.*;

public class DadosImpl implements Dados {
    private ArrayList<Transporte> lista = new ArrayList<>();

    @Override
    public void adicionar(Transporte t) throws Exception {
        lista.add(t);
    }

    @Override
    public void excluir(Transporte t) {
        lista.remove(t);
    }
}
```

Questão 15: Diagrama de Classes I

Implemente as classes descritas no diagrama abaixo:



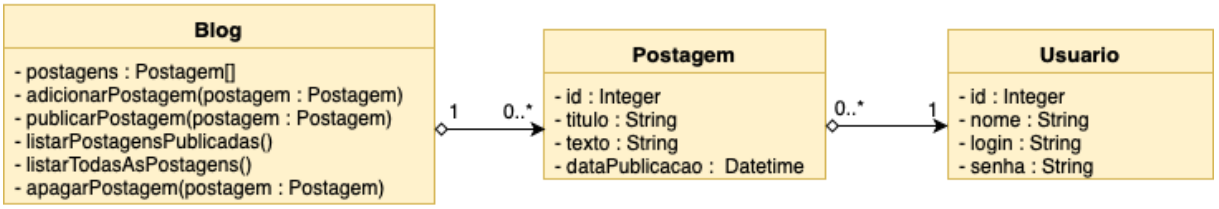
(a) Um pedido é composto por um conjunto de itens pedidos.

(b) Um item pedido associa-se com um produto.

(c) O cálculo do valor total do pedido deverá ser feito mediante a soma do preço de cada produto incluído nos itens pedidos.

Questão 16: Diagrama de Classes II

Crie um programa para representar um blog. O blog contém várias postagens que são cadastradas por usuários autenticados. Siga o diagrama de classes abaixo para a implementação:



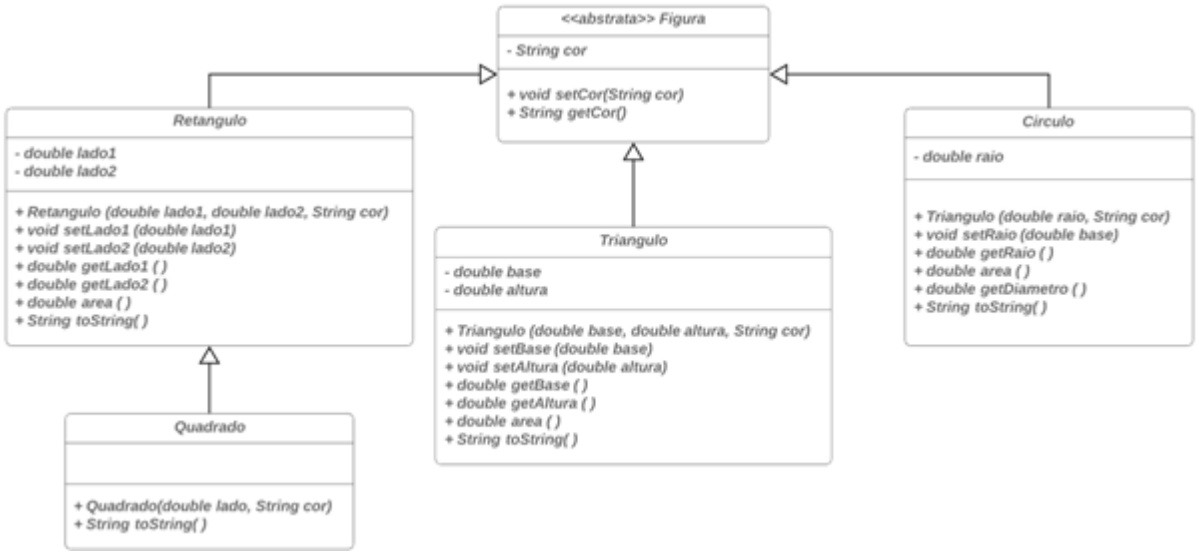
(a) O blog contém uma lista de postagens.

(b) O método **listarPostagensPublicadas()** deverá listar apenas as postagens cuja data de publicação já foi atingida. Do contrário, as postagens serão visíveis apenas pelo método **listarTodasAsPostagens()**.

(c) A postagem deverá identificar o usuário que fez a sua criação.

Questão 17: Diagrama de Classes III

Implemente o modelo abaixo. Para os métodos, apenas a título de exercício, ao ser chamado, faça com que seja impresso o nome da classe e o nome do método.



Questão 18: Diagrama de Classes IV

Implemente o modelo abaixo. Para os métodos, apenas a título de exercício, ao ser chamado, faça com que seja impresso o nome da classe e o nome do método.