

Sumário <ul style="list-style-type: none">Herança Documentação complementar Java: <ul style="list-style-type: none">Inheritance

Nota: Gere o JavaDoc para o(s) projeto(s) utilizado(s) na resolução desta ficha de trabalho.

Parte 1

Desenvolva uma API (*Application Programming Interface*) que permita armazenar informação relativa a um conjunto de bicicletas disponíveis para venda numa loja especializada. A empresa produz e comercializa essencialmente bicicletas de dois tipos: montanha e de estrada.

Independentemente do seu tipo, uma bicicleta possui as seguintes características

- Identificador (`id`)
- Número de velocidade (`numberOfGears`)
- Cor principal (`mainColor`)
- Diâmetro da roda (`wheelSize`)
- Tipo de travões (`brakes`)
- Material de construção (`material`)
- Preço (`price`)
- Anos de garantia (`guarantee`)

Os travões podem ser de pinças ou hidráulicos e o material de fabrico pode variar entre carbono e alumínio. Apesar das características gerais previamente identificadas, um determinado tipo de bicicleta possui outras características próprias:

- Bicicleta de montanha
 - Número de luzes (`numberOfLights`)
 - Tipo de suspensão (`suspension`) – pode ser simples, dupla ou sem suspensão
 - Conjunto de utensílios (`bikeTools`) cujo limite será de 5 utensílios (garrafa de água, kit reparação pneu, conta-quilómetros, alforje e suporte para telemóvel). Não deverão ser permitidos utensílios em duplicado.
- Bicicleta de estrada
 - Identificação das fitas no guiador (`handlebelt`)
 - Tamanho do quadro (`frameSize`)
 - Conjunto de observações (`observations`) – Texto livre (limitado a 50 caracteres) que o cliente poderá indicar para a construção da bicicleta. Caso o cliente insira uma frase com um tamanho superior, esta deverá ser truncada a 50 caracteres.
 - Por omissão este tipo de bicicleta é construída em carbono e com travões hidráulicos.

Na resolução dos exercícios propostos considere que deve:

- Garantir o encapsulamento de todas as classes criadas
- Criar os métodos de acesso necessários para as classes criadas
- Criar, num *package* específico, as enumerações necessárias para suportar o problema apresentado
- Criar métodos específicos para manipulação de coleções, ou seja, não deverá ser permitido o acesso direto às variáveis que representam coleções, devendo existir métodos para adicionar, remover, editar e listar elementos.

Antes de resolver os exercícios, deve estruturar as classes de acordo com as relações entre elas e os *packages* a que pertencem. Atente à Figura 1 que representa um diagrama de classes. Este diagrama está propositalmente incompleto.

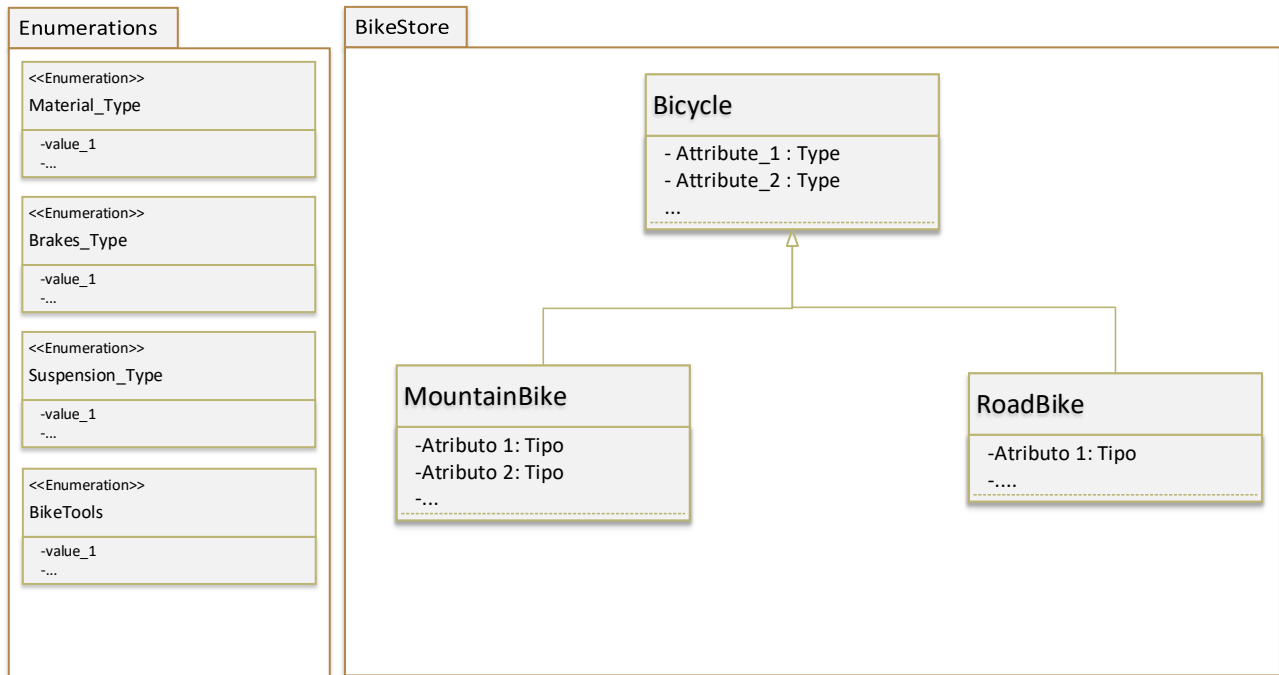


Figura 1 - Diagrama de Classes

Exercício 1

- 1.1) Complete o diagrama presente na Figura 1, substituindo os atributos, tipos e valores exemplo de acordo com o enunciado do problema.
- 1.2) Crie um projeto com o nome `pp_fp07` e, no package `pp_fp07.bikeStore`, implemente o código Java para representar a estrutura descrita anteriormente.
- 1.3) Crie a classe `BikeDemo` de forma a testar as classes implementadas. Inicialize alguns elementos relativos a bicicletas de montanha e de estrada.

Exercício 2

- 2.1) Crie uma classe `BicycleManagement` que permita armazenar um conjunto de bicicletas que estão disponíveis para venda. Implemente os métodos necessários para adicionar, remover e listar as bicicletas. Teste a classe com pelo menos de 20 bicicletas de tipos diferentes.
- 2.2) Crie um método que permita listar apenas bicicletas de montanha.
- 2.3) De que forma podemos implementar uma coleção que permita guardar um número ilimitado de bicicletas? Implemente as modificações necessárias.

Parte 2

Uma instituição é constituída por diversas pessoas que desempenham funções distintas. De um modo geral pode afirmar-se que independentemente do tipo de funções que desempenhe será sempre necessário obter a seguinte informação de uma pessoa

- Nome
- Data de nascimento
- Morada
- Número de cartão de cidadão
- Número de identificação fiscal

Considerando uma instituição de ensino implemente as classes necessários para criar um Professor, Funcionário administrativo e Aluno. Tenha em atenção as seguintes características específicas para cada

- Professor
 - Sigla
 - Tipo de contrato (Parcial ou Integral)
 - Conjunto de unidades curriculares que leciona
- Funcionário
 - Código de funcionário
 - Tipo de contrato (Parcial ou Integral)
- Aluno
 - Código de aluno
 - Conjunto de unidades curriculares que frequenta

Uma Unidade Curricular tem os seguintes atributos:

- Nome
- Sigla
- Nome do curso
- Ano lectivo (ex. 1º)
- Semestre (ex. 2º)

Na resolução dos exercícios propostos considere que deve:

- Garantir o encapsulamento de todas as classes criadas
- Criar os métodos de acesso necessários para as classes criadas
- Criar, num *package* específico, as enumerações necessárias para suportar o problema apresentado
- Criar métodos específicos para manipulação de coleções, ou seja, não deverá ser permitido o acesso direto às variáveis que representam coleções, devendo existir métodos para adicionar, remover, editar e listar elementos.

Exercício 1

- 1.1) Crie um diagrama de classes de forma a estruturar e representar todas as classes e relações entre elas. O diagrama deve ajudá-lo a estruturar a resolução deste problema antes de o codificar no editor de desenvolvimento.
- 1.2) No `package pp_fp07.publicSchool`, implemente o código Java para representar a estrutura descrita anteriormente.
- 1.3) Nas várias classes criadas adicione métodos que permitam imprimir os valores atribuídos nas variáveis das diversas instâncias.
- 1.4) Crie a classe `SchoolDemo` de forma a testar as classes implementadas. Inicialize alguns elementos relativos aos diferentes tipos de pessoas que fazem parte da instituição.

Exercício 2

- 2.1) Crie uma classe `SchoolManagement` que permita realizar a gestão de pessoas da instituição.
- 2.2) Crie um método que permita listar
 - Professores
 - Alunos e Funcionários