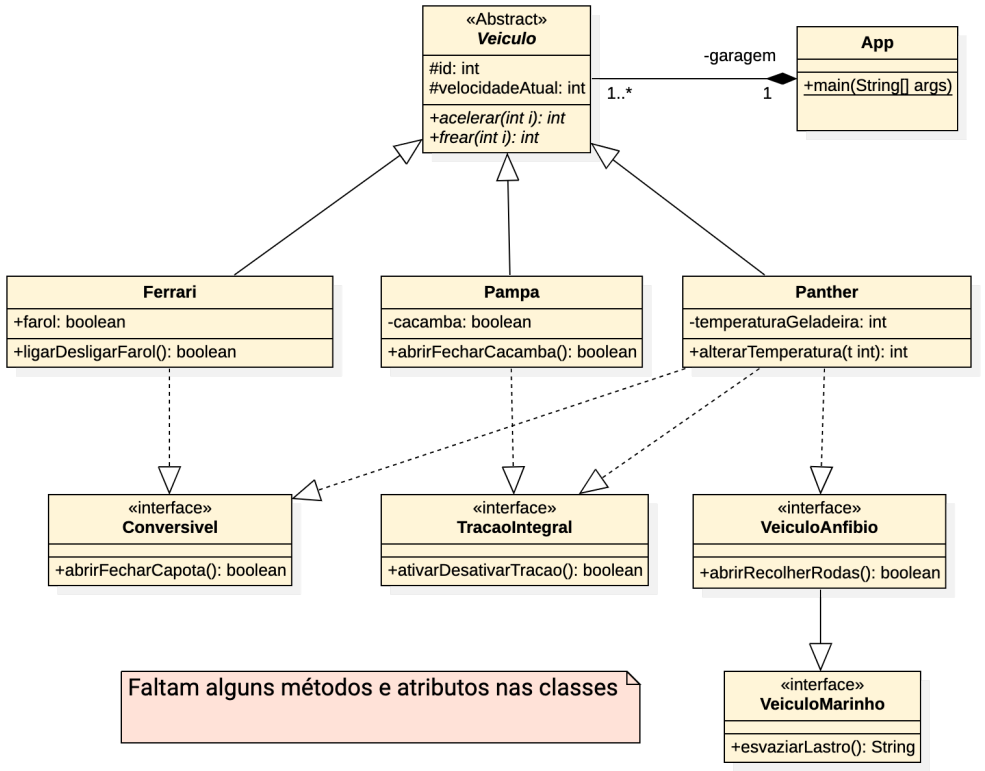
Engenharia de Telecomunicações — Prof. Emerson Ribeiro de Mello
POO129003 – Programação Orientada a Objetos — mello@ifsc.edu.br

Laboratório 03: Herança e polimorfismo

29/07/2024

Figura 1: Diagrama de classes UML de um jogo de corrida com diferentes tipos de carro



Na Figura 1 é apresentado o diagrama de classes UML para um jogo de corrida que permite o jogador escolher diferentes tipos de carro que possui em sua garagem. Cada carro possui um conjunto específico de habilidades, por exemplo, um carro anfíbio pode recolher suas rodas e esvaziar seu lastro para flutuar na água. Já um carro com tração integral pode ativar ou desativar essa funcionalidade para melhorar a tração em terrenos escorregadios. Um carro conversível pode abrir ou fechar a capota, e assim por diante. O diagrama UML citado foi criado com base nas regras de negócio apresentadas a seguir. Talvez o diagrama UML apresentado não esteja completo ou tão detalhado ao ponto de atender completamente a lista abaixo.

1. Todo veículo possui identificador único, velocidade atual e velocidade máxima que pode atingir
2. O limite máximo de velocidade para os carros que não possuem tração integral é 200 km/h. Para veículos que possuem tração integral a velocidade máxima é 140 km/h
3. As ações, abrir ou fechar caçamba; abrir ou fechar capota; ativar ou desativar tração integral, só podem ser executadas se o veículo estiver complementamente parado. Para veículos anfíbios, ativar ou desativar a tração integral também exige que suas rodas não estejam recolhidas
4. Só é possível recolher as rodas de um veículo anfíbio se o mesmo estiver parado e esse procedimento implica obrigatoriamente no esvaziamento do seu lastro
5. A capacidade do lastro (em litros) depende de cada veículo, mas sabe-se que nenhum veículo poderá ter um lastro com capacidade maior que 100 litros

Crie um projeto Java com o gradle e com base no diagrama da Figura 1, e nas regras de negócio apresentados acima, implemente as classes e interfaces. **Mas atenção**, se o diagrama de classes não estiver aderente com as regras, então você precisará fazer as modificações necessárias nas classes antes de implementar. Gere um novo diagrama de classes. A implementação deverá respeitar as seguintes regras:

- Para simular as ações como “abrir capota”, “esvaziar lastro”, “carro em 100km/h” etc. Faça uso de `Strings` que deverão ser impressas no console. Contudo, somente a classe com o método `main` deverá ter instruções de interação com o usuário (e.g. `Scanner` e `System.out`);
- Na classe com o método `main`, crie um atributo do tipo `ArrayList` que permita armazenar objetos de qualquer classe concreta presente no diagrama da Figura 1. Adicione a esse `ArrayList` pelo menos uma instância de cada uma das classes concretas;
- Na classe com o método `main`, e usufruindo do conceito de polimorfismo, invoque métodos dos carros instanciados. Faça um comentário no código fonte (Ex: `// Usando polimorfismo na instrução abaixo`) logo acima da instrução onde você fez uso do polimorfismo.
- Invoque cada um dos métodos dos objetos instanciados. Por exemplo, se um carro anfíbio possui um método `recolherRodas()` e um carro conversível possui um método `abrirCapota()`, então você deverá invocar esses métodos para cada objeto instanciado.