PRÁTICA LABORATORIAL 09

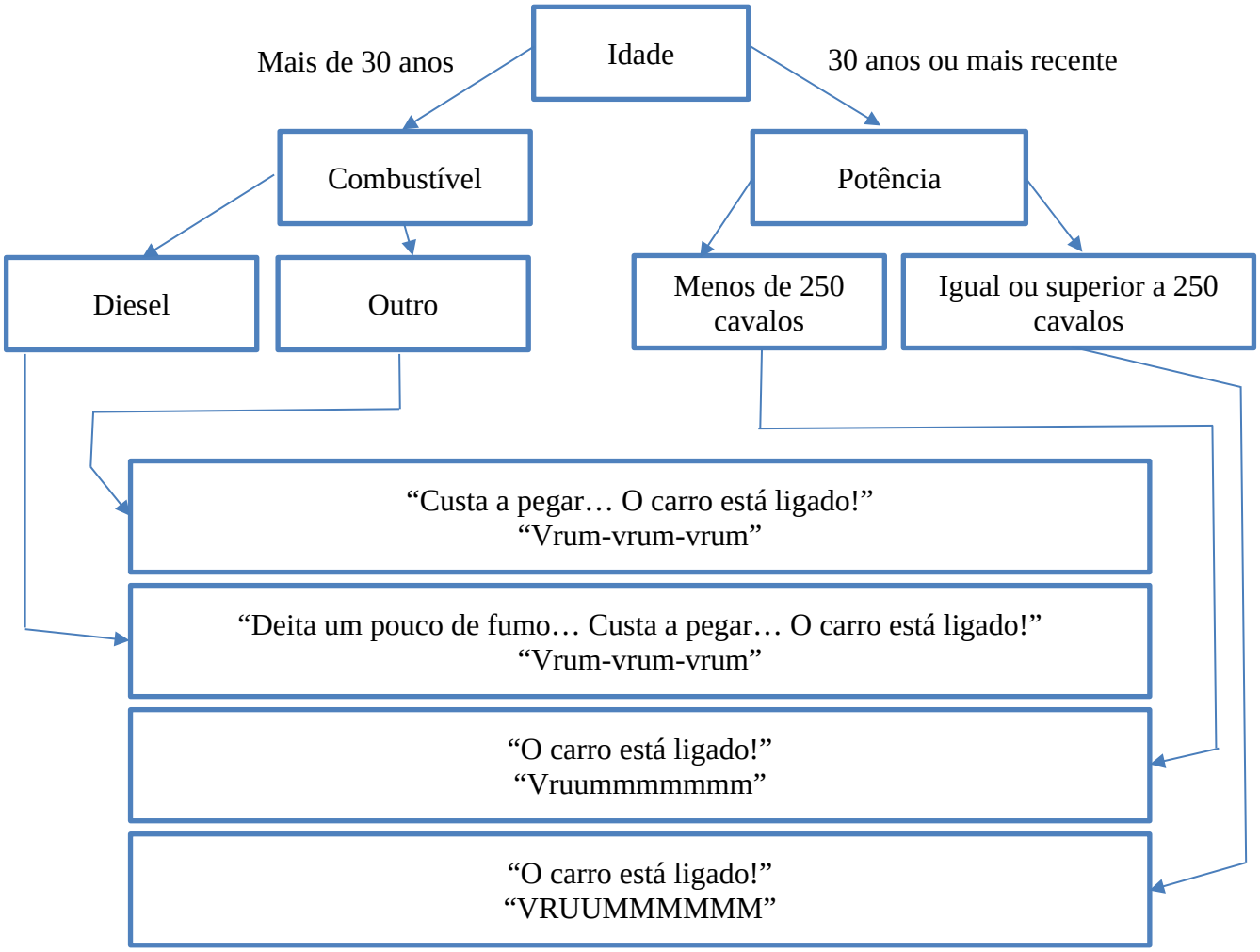
Objetivos:

- Classes
- Objetos
- Métodos Construtores
- Métodos Set e Get

EXERCÍCIOS

Parte 1

1. Atualize a classe Carro, para além de marca, modelo e ano, deve ter os seguintes parâmetros: potência, cilindrada, tipoCombustivel, litros100Km.
    a. Tipo Combustível deve ser uma enumeração: GASOLINA, DIESEL, GPL.
    b. Altere o método ligar para que imprima uma mensagem de acordo com as seguintes especificações:



c. Crie um método chamado corrida que receba um adversário como parâmetro e retorne o carro vendedor. O vencedor é o carro com mais potência (caso seja a mesma o fator de desempate é a cilindrada (o maior ganha), caso contrário o fator de desempate é a idade (o mais recente ganha), caso contrário temos empate).
d. Crie um método que receba como parâmetro uma distância (em km), e que calcule o valor consumido em litros do combustível.
e. Teste a classe instanciando dois carros ao seu gosto e faça uma corrida entre os dois, imprima o vencedor. Seguidamente, calcule quanto é que os dois carros iam gastar numa viagem de 97km. E imprima na consola quanto gastaria cada um, e qual dos dois é que ia gastar mais.

2. Atualize a classe Conta: para além dos atributos número da conta, saldo e titular da conta, deve também ter ano de abertura, margem de empréstimo e valor em divida.
    a. O método de construtor não muda a **nível de parâmetros**.
    b. O ano de criação da conta deve ser 2023.
    c. A margem de empréstimo deve ser 90% do saldo (ex: se a conta tem 1000€, a conta pode fazer um empréstimo até 900€). Sempre que o saldo alterar, também esta margem deve mudar.
    d. Crie o método pedirEmprestimo, deve receber como parâmetro o valor a pedir. Seguidamente, avalie se a conta pode pedir um empréstimo com esse valor. Se houver qualquer valor em dívida, o pedido deve ser automaticamente desconsiderado (só pode ter um empréstimo de cada vez). Caso seja possível, atualize o saldo e valor em dívida com o dinheiro pedido e retorne true. Caso não seja possível, não atualize saldo nem valor em dívida e retorne false.

3. Crie uma classe animal, que tenha como atributos: nome, espécie, país de origem, peso (em kg) e alimentação (array de Strings).
    a. Crie um método para o animal comer, receba como parâmetro o alimento e o peso (em g). Se o alimento constar na lista de alimentos que o animal come, incremente o seu peso com o peso da refeição e imprima na consola “O animal comeu”. Caso contrário, não deve atualizar e imprimir na consola “O animal recusou essa comida”.
    b. Teste a classe instanciando um animal e apresentado uma refeição que ele goste, seguidamente imprima o seu novo peso. A seguir, apresente uma refeição que ele não goste e imprima o seu peso.

4. Crie uma classe Imóvel, que tenha como atributos: rua, número da porta, cidade, tipo (apartamento, casa, mansão), acabamento (para restauro, usada, nova, nova com alto acabamento), área, número de quartos, número de casas de banho e área da piscina.
    a. Crie um método para calcular o valor do imóvel com base nas seguintes regras:
        i. Apartamento: cada m2 vale 1000€.
        ii. Casa: cada m2 vale 3000€
        iii. Mansão: cada m2 vale 5000€
        iv. Para Restauro: preço total tem 50% de desconto
        v. Usada: preço total tem 10% de desconto
        vi. Nova: preço reflete sem desconto
        vii. Nova com alto acabamento: preço total valoriza 25%
        viii. Cada quarto incrementa 7500€ ao custo
        ix. Cada casa de banho incrementa 10500€ ao custo
        x. A piscina custa 1000/m2
    b. Crie um método para mudar o estado de uma casa.
    c. Crie um método chamado imprimirDescricao que imprima as especificações do imóvel.
    d. Crie um método chamado compararImoveis, que receba outro imóvel como parâmetro e retorne qual o imóvel mais caro.

5. Crie uma classe chamada "Pessoa" que armazena informações básicas sobre uma pessoa, como nome, idade, email e telemóvel. De seguida, crie uma classe chamada "Agenda" que armazena um conjunto de objetos Pessoa num array. A classe Agenda terá um método para adicionar novas pessoas e outro para listar todas as pessoas registadas.

6. Crie uma classe chamada "Atleta" que armazena informações sobre um atleta, incluindo nome, modalidade desportiva, altura, peso e país de origem. Cada atleta pode participar numa ou mais competições.
    a. De seguida, crie a classe Competicao, que armazena informações sobre uma competição, incluindo o nome da competição, país, e a lista de atletas participantes.
    b. Crie métodos para adicionar atletas.
    c. No Main crie, pelo menos, 6 atletas.
    d. No Main crie, pelo menos, 2 competições (onde são adicionados os participantes).
    e. Crie um método para imprimir as informações de uma competição, inclusive a lista de atletas.

Bom trabalho! ☺