Lista 6 - Encapsulamento e interfaces.

Prazo de entrega: 31/05/2024 até 23:59 (Sexta)

Instruções de entrega: Você deve ter um repositório em seu github para a disciplina de POO chamado **Programacao-Orientado-Objetos**.
Crie uma pasta dentro dele chamada Listas e dentro dela crie uma pasta chamada Lista6.
Para cada exercício da lista deve ser criado um projeto java.
Ao terminar a lista, suba seus exercícios no github e envie o link no formulário abaixo.

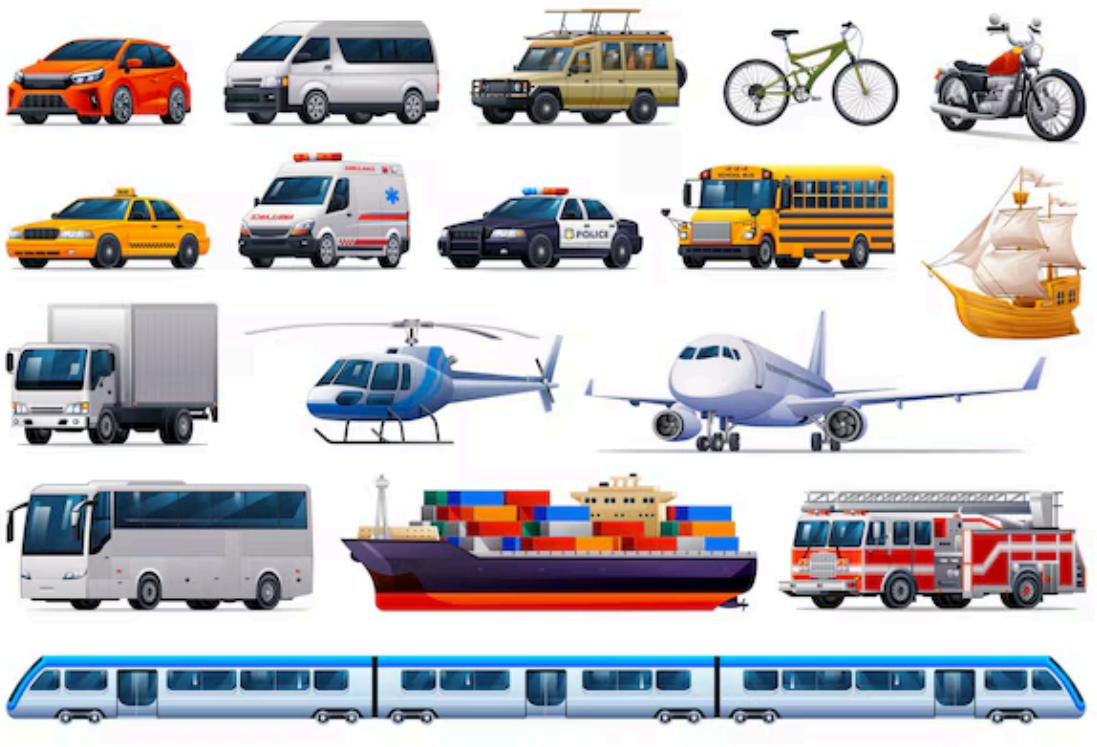
Link para o formulário de envio: https://forms.gle/34ExvF3XCC5ZxDYg8

MONITORIAS: Serão realizados quatro dias de monitorias para auxiliar com a resolução dos exercícios. As datas e horários das monitorias são:

Dia	Horários		
Segunda-Feira	19h até 21h		
Terça-Feira	19h até 21h	-	-
Quarta-Feira	09h até 11h	19h até 21h	-
Quinta-Feira	19h até 21h	-	-
Sexta-Feira	09h até 11h	19h até 21h	22h até 23:59

OBS: Se você não conseguir participar das monitorias nos dias e horários propostos, você pode solicitar uma monitoria em uma data e horário diferente, desde que seja comunicado com pelo menos um dia de antecedência 🙂

EXERCÍCIO 1



Existem muitos tipos de veículos, e desde o Skate ao Trem, eles possuem coisas em comum que nos fazem identificar eles como veículos.

Fomos contratados para trabalhar em um novo jogo de mundo aberto, e neste jogo, o personagem principal poderá pilotar tudo quanto é tipo de veículo, e por isso, nos foi dada a tarefa de desenvolver uma interface que os represente.

Além da interface **VEICULO** também nos foi pedido para desenvolver a classe **CARRO** e **ALGUM OUTRO** veículo que for de nosso interesse.

Instruções:

1 - Crie um pacote chamado **VEICULOS**.

2 - Dentro do pacote **VEICULOS** crie a interface **VEICULO**.
A interface deve ter os seguintes métodos: **acelerar** e **frear.**

3 - Crie a classe **CARRO** e outra classe para representar **ALGUM OUTRO** veículo **DE SUA PREFERÊNCIA**. (Não esqueça que ela deve implementar a interface **VEICULO)**.

4 - Desenvolva ao menos 2 atributos para cada classe.

5 - Os atributos de ambas as classes devem ser privados.

6 - Crie getters e setters para todos os atributos de ambas as classes.

7 - Crie construtores com parâmetros para iniciar os atributos mais importantes dos objetos de ambas as classes.

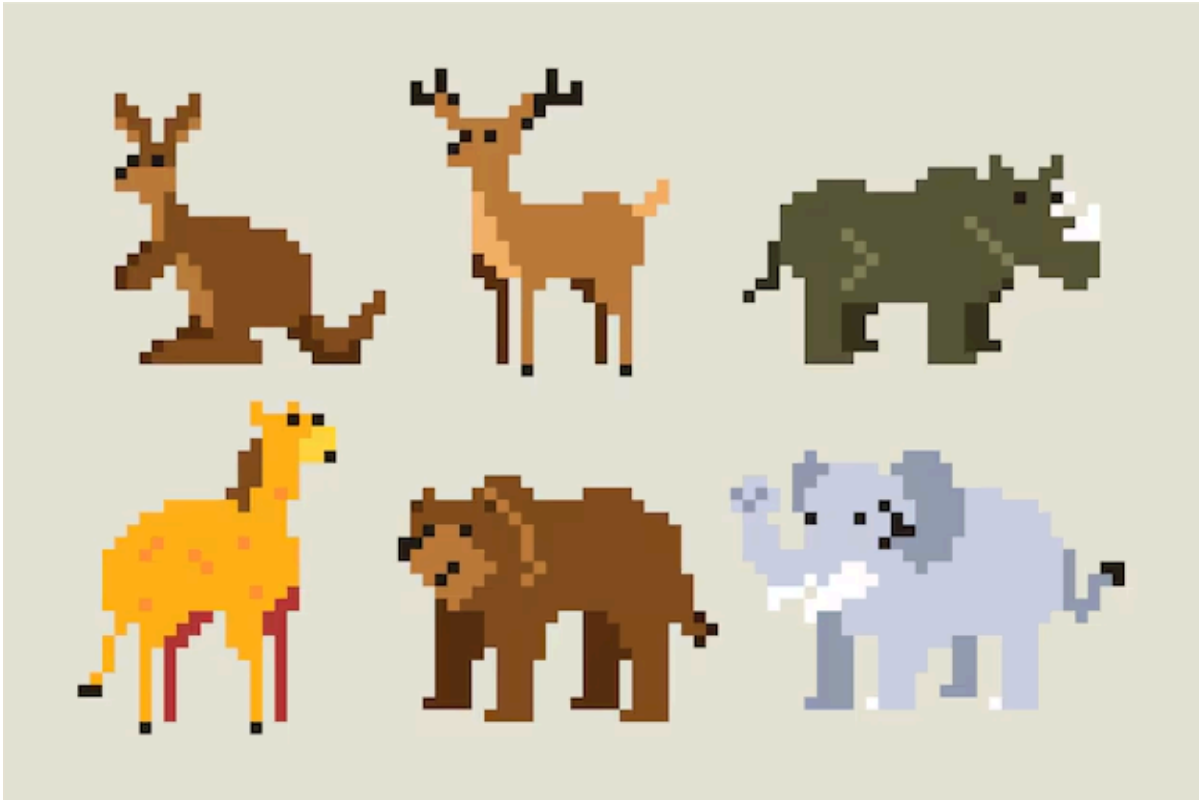
8 - Implemente os métodos **acelerar** e **frear**. (Os métodos devem imprimir na tela uma onomatopeia para representar o som que o veículo faz quando acelera e quando freia em seus respectivos métodos).

9 - Na classe APP:

.Crie uma instância de cada classe utilizando o construtor com parâmetro.

.Chame os métodos **acelerar e frear** de cada objeto.

EXERCÍCIO 2



O mesmo jogo também terá uma fauna bastante diversa com muitos animais diferentes, e também nos foi dada a tarefa de desenvolver a interface **ANIMAL** e algumas implementações da interface.

Além da interface **ANIMAL**, também nos foi solicitado para desenvolver a classe **GATO** e **OUTRA CLASSE** para representar outro animal **DE SUA ESCOLHA**.

Instruções:

1 - Crie um pacote chamado **ANIMAIS**.

2 - Dentro do pacote **ANIMAIS** crie a interface **ANIMAL**.
A interface deve ter os seguintes métodos: **comer** e **emitirSom**.

3 - Crie a classe **GATO** e **OUTRA CLASSE** para representar outro animal **DE SUA PREFERÊNCIA**. (Não esqueça que ela deve implementar a interface **ANIMAL**).

4 - Desenvolva ao menos 2 atributos para cada classe.

5 - Os atributos de ambas as classes devem ser privados.

6 - Crie getters e setters para todos os atributos de ambas as classes.

7 - Crie construtores com parâmetros para iniciar os atributos mais importantes dos objetos de ambas as classes.

8 - Implemente os métodos **comer** e **emitirSom**. (Os métodos devem imprimir na tela uma descrição do comportamento do animal ao comer e o som que o animal emite).

9 - Na classe APP:
.Crie uma instância de cada classe utilizando o construtor com parâmetro.
.Chame os métodos comer e emitirSom de cada objeto.