CM Kotlin Challenge

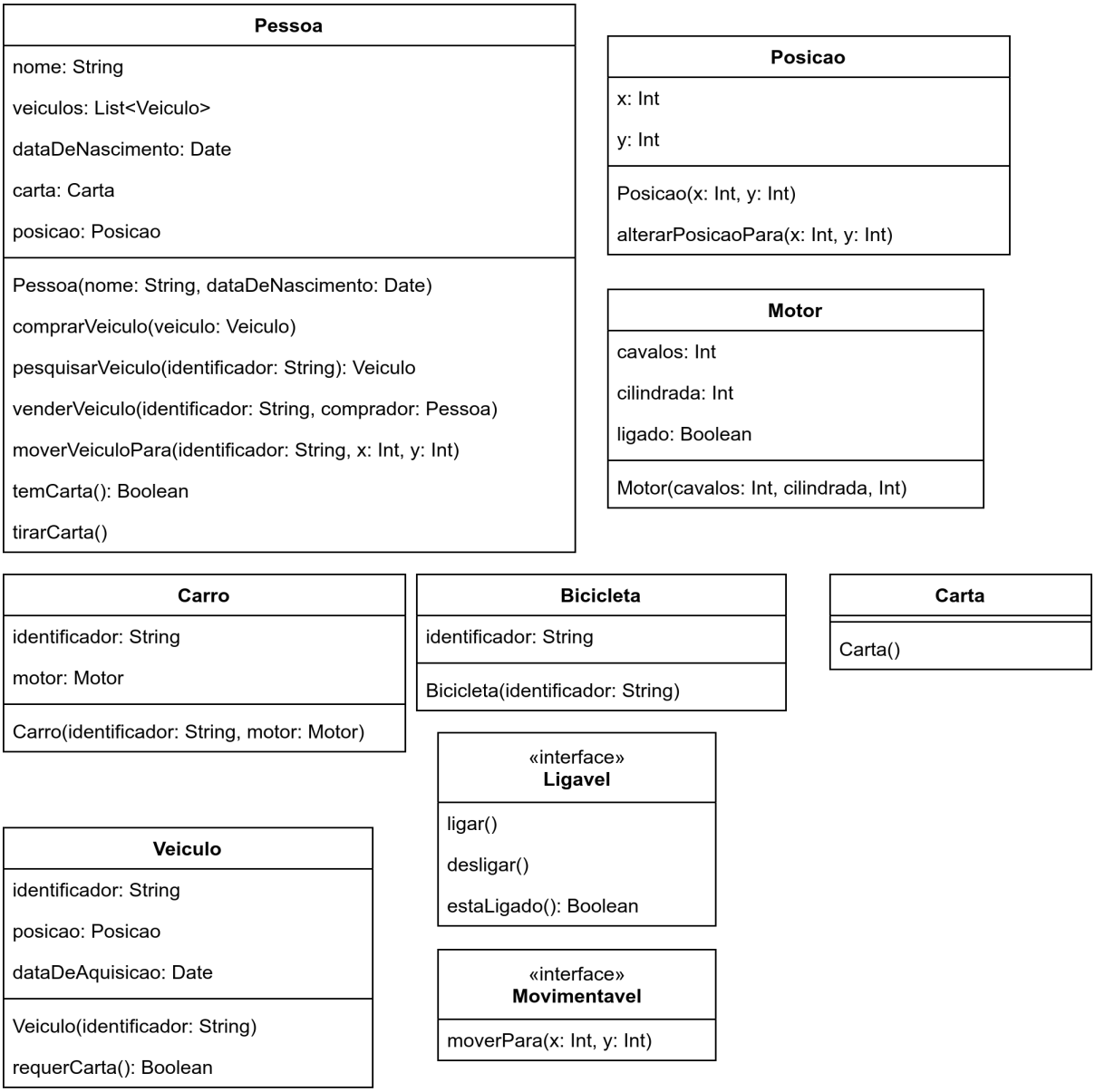
Seleccione ou arraste para aqui o ficheiro .zip com o projecto

Desafio de Kotlin para Computação Móvel

Este exercício não conta para avaliação, é apenas uma forma de treinares a linguagem Kotlin.

Instruções

1. Considera e cria uma implementação que faça sentido com as classes abaixo indicadas:



2. Cria um ficheiro Kotlin chamado Main.kt com o seguinte conteúdo

```
package pt.ulusofona.cm.kotlin.challenge

fun main() {
   // aqui escreves o código do programa

}
```

3. Para concluires o exercício deverás utilizar a seguinte estrutura de pacotes: pt.ulusofona.cm.kotlin.challenge.

- exceptions { Onde irás colocar as tuas exceções. }
- interfaces { Aqui irás colocar as tuas interfaces }
- models { Onde irás colocar os teus modelos, como por exemplo: Pessoa, Carro, entre outros }

4. Para resolveres o exercício terás também de implementar algumas exceções:

- AlterarPosicaoException { Quanto tentas um movimento para a posição onde já te encontras. }
- MenorDeIdadeException { Quando uma pessoa tenta tirar a carta e é menor de idade. }
- PessoaSemCartaException { Quanto uma pessoa sem carta tenta conduzir um veículo que necessita de carta. }
- VeiculoNaoEncontradoException { Quando pesquisas por um veículo que uma pessoa não possui. }

- VeiculoDesligadoException { Quanto uma pessoa tenta desligar um veículo e o mesmo já se encontra desligado. }
- VeiculoLigadoException { Quando uma pessoa tenta ligar um veículo e o mesmo já se encontra ligado. }

5. Todas as classes deste modelo terão de implementar o método toString com a seguinte formatação:

```
"Nome da classe | atributo1 | atrubuto2 | ..."
```

```
"Bicicleta | identificador | data de aquisicao | Posicao | x:0 | y:0"
```

```
"Carro | identificador | data de aquisicao | Posicao | x:0 | y:0"
```

```
"Pessoa | nome | data de nascimento | Posicao | x:0 | y:0"
```

```
"Posicao | x:0 | y:0"
```

```
"Motor | cavalos | cilindrada"
```

Nota que, o nome da classe no método toString deve estar escrito da mesma forma que o nome do ficheiro, isto é, com a primeira letra maiúscula.

6. As datas deverão deverão ser apresentadas no formato exemplo: 01-02-2020. Uma vez que diversas classes usam esta formatação da data, será recomendável que utilizes uma classe dedicada para o efeito.

7. Por defeito qualquer posição deverá assumir x:0 e y:0. Outros valores poderão ser fornecidos através do construtor da classe ou através do método altera

8. A PessoaSemCartaException deve conter a seguinte mensagem:

```
        "Nome da Pessoa não tem carta para conduzir o veículo indicado"
```

Pistas

1. Repara que, tanto a pessoa como o carro e bicicleta têm a capacidade de se deslocarem, embora de formas diferentes.

2. Por omissão, uma pessoa não tem carta e só após ser maior de idade é que poderá tirar a carta de condução.

3. O método moverVeiculoPara deve ser capaz de mover qualquer tipo de veículo, seja ele ligável ou não. De referir que, caso o veículo seja ligável, o mesmo deverá permanecer no estado desligado assim que a sua movimentação estiver concluída. Deverão recorrer a polimorfismo para implementar este método.

4. O método que lida com a venda de veículos deve lidar com todo o processo de venda, isto é, a remoção da lista do vendedor, adicionar ao comprador e atualizar a data de aquisição do respetivo veículo.

O ficheiro tem que ser um zip com a seguinte estrutura:

```
AUTHORS.txt  (contém linhas NUMERO_ALUNO;NOME_ALUNO, uma por aluno do grupo)
+ src
|--- pt
|------ ulusofona
|--------- cm
|------------ kotlin
|--------------- challenge
|------------------ Main.kt
|------------------ ...   (outros ficheiros com código do projecto)
```

Nota: Todos os outros ficheiros devem ser omitidos do ficheiro zip, nomeadamente os ficheiros que resultam da compilação