

# Exame de Programação Orientada aos Objectos

MiEI e LCC  
DI/UMinho

28/06/2017

Duração: 2h

*Leia o teste com muita atenção antes de começar*

*Assuma que gets e sets estão disponíveis, salvo se forem explicitamente solicitados.*

RESPONDA A CADA PARTE EM FOLHAS SEPARADAS.

---

## PARTE I - 7 VALORES

Considere a seguinte estrutura fornecida por um dos grupos que entregou o projecto prático da UMeR.

```
public class UMeR {  
    private Map<String, Cliente> clientes;  
    private Map<String, Veiculo> veiculos;  
  
    ...  
}
```

Considere também que cada instância de Cliente está definida da seguinte forma:

```
public class Cliente {  
    private String email; // código de identificação de cliente  
    private String nome;  
    private String nif;  
    private List<Viagem> viagens;  
  
    ...  
}
```

A classe Viagem está definida como:

```
public class Viagem {  
    private String codViagem; // código de identificação de uma viagem  
    private String codVeiculo;  
    private String codCliente;  
    private int duracao;  
    private LocalDateTime dataViagem;  
    private double preco;  
  
    ...  
}
```

A classe Veiculo tem a definição que se apresenta:



```
public abstract class Veiculo implements Serializable {
    private String matricula; //código de identificação de um veículo
    private String marca;
    private String modelo;
    private double velocidadeMedia;
    private double precoBase;
    private int factorDeFiabilidade;
    private int autonomia;
    private double totalKms;
}
```

1. Implemente o método toString da classe Cliente.
2. Codifique o método public void insereVeiculo(Veiculo v) throws VeiculoExisteException, da classe UMeR. Não necessita fazer a classe de excepção.
3. Na classe UMeR desenvolva o método public List<Viagem> getViagensByDate(String cliente), que devolve as viagens de um cliente ordenadas por ordem cronológica de ocorrência (note que a classe LocalDateTime tem um método isBefore que devolve um valor booleano que indica que uma data é anterior a outra com que está a ser comparada). Poderá, caso necessário, dividir este método em métodos auxiliares.
4. Crie um método da classe UMeR, public Map<String, Set<String>> veiculosPorCliente(), que apresenta a relação entre um cliente e os veículos (identificador dos veículos) em que teve viagens realizadas.

————— Mude para uma nova folha —————

## PARTE II - 8 VALORES

Considere ainda no contexto da UMeR que, após os protesto dos taxistas, o governo exigiu à UMeR o desenvolvimento de um sistema automático de facturas. Os requisitos exigidos são:

- (a) Uma factura deve conter a seguinte informação: código de factura, NIF (Número de Identificação Fiscal) da UMeR, NIF do cliente, data da viagem e preço.
- (b) A UMeR deve conter um histórico de todas as facturas passadas, e uma factura deve ser eficientemente consultada dada o seu código.
- (c) No fim do mês a UMeR deve enviar às finanças as facturas passadas nesse mês, ordenadas crescentemente por data.

Responda agora às seguintes questões:

5. Defina a classe Factura de forma que respeite o requisito a), codificando a declaração de variáveis e o construtor parametrizado.
6. Actualize a classe UMeR de modo a implementar os requisitos a) e b).
7. Escreva um método public void facturasDoMes(int mes, String ficheiro) throws IOException, que recebe um inteiro com o mês em causa e produz um ficheiro de texto com todas as facturas desse mês, tal como requerido em c). Note que o método public Month getMonth(), da classe LocalDateTime devolve o mês de uma data (Janeiro é o mês 1).



8. Considere que por regulamentação legal foi decidido introduzir um regime especial de facturação para clientes internacionais. As facturas passadas aos clientes internacionais (FacturasInternacionais) deverão prever a existência de dois métodos distintos para dar informação financeira:

```
public double precoSemImpostos();  
public double valorImposto();
```

, sendo que o valor do imposto a pagar em Portugal é de 13% do valor da viagem. Responda às seguintes questões:

- (a) Apresente o código necessário para acrescentar estas definições e codifique os métodos acima apresentados.
- (b) Codifique o método da classe UMeR, public double montanteImposto(), que calcula o montante de imposto acumulado nas facturas internacionais.

————— Mude para uma nova folha —————

### PARTE III - 5 VALORES

Considere também que existem veículos do tipo VeiculoElectrico, CarrinhaCarga e BicicletaTandem. Os VeiculoElectrico acrescentam informação sobre o número de baterias e a capacidade eléctrica das mesmas (em KWh) e as BicicletaTandem sobre o material do quadro (ex: alumínio, carbono, etc.) e a marca da suspensão.

Os VeiculoElectrico e BicicletaTandem possuem um método que apresenta o valor da autonomia corrigido da seguinte forma: é de mais 40% nos VeiculoElectrico e de mais 25% nos BicicletaTandem. O factor de fiabilidade de um VeiculoElectrico é de mais 2/3 do apresentado por um Veículo.

- 9. Escreva as declarações necessárias necessárias para representar as classes VeiculoElectrico e BicicletaTandem, nomeadamente as declarações de variáveis e os métodos que possibilitam o acesso à autonomia e à fiabilidade.
- 10. Considere que se pretende gravar em ficheiro de objectos todos os veículos da classe UMeR que sejam eléctricos ou bicicletas e cujo valor de autonomia seja superior a 50. Para tal codifique o método public void gravaEcologicos(String ficheiro) throws IOException.
- 11. Codifique na classe ~~o método~~ VeiculoElectrico o método public boolean equals().