

Exame de Programação Orientada aos Objectos

MiEI e LCC
DI/UMinho

25/06/2018
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 - 6 VALORES

1. Considere que se quer criar uma aplicação que faça a gestão de tickets (pedidos) de suporte técnico (por exemplo, num operador de telecomunicações). Para tal definiu-se a classe TicketSuporte como contendo as seguintes variáveis de instância:

```
public class TicketSuporte {  
    private String nome0; // quem originou o pedido  
    private LocalDateTime horaOcorrencia; // instante do report do ticket  
    private String assunto;  
    private String descrição;  
    private String identTecnico; //técnico que resolveu o ticket  
    private LocalDateTime horaFecho; //instante do fecho do ticket  
    ...  
    //método que efectua a resolução de um ticket.  
    //O algoritmo de resolução está codificado neste método.  
    public void executaTarefa() {...}  
    ...  
}
```

Criou-se também a classe SistemaTickets que é a plataforma onde são guardados e geridos os tickets de suporte. Note-se que os tickets de suporte são atendidos pela ordem pela qual chegam ao sistema. Considere que a declaração das variáveis de instância desta classe foi feita da seguinte forma:

```
public class Tecnico {  
    private String ident; // identificador do técnico  
    private String nome; //nome do técnico  
    private int numTickets; // número de tickets resolvidos  
    ...  
}  
  
public class Supervisor extends Tecnico {  
    private List<String> equipa;
```

```

    }
    ...

    public class SistemaTickets {
        private Map<String,Tecnico> funcionarios; // funcionarios da empresa
        private List<Ticket> ticketsPorResolver; //tickets ainda não tratados
        private List<Ticket> ticketsResolvidos; //tickets já satisfeitos
        ...
    }

```

Codifique os seguintes métodos:

- (a) public void adicionaTicket(Ticket t), da classe SistemaTickets, que adiciona um ticket ao sistema.
- (b) public void resolveTicket(String ident) throws TecnicoNaoExisteException, da classe SistemaTickets, que resolve um ticket do sistema.
- (c) public Map<String,List<Ticket> ticketsPorTecnico(), da classe SistemaTickets, que devolve um map em que associa a cada identificador de técnico os tickets por ele resolvidos.
- (d) public String supervisorTop(), da classe SistemaTickets, que devolve o identificador do supervisor que mais tickets resolveu.

PARTE II - 5 VALORES

2. Considere a definição de Eletrodomestico, conforme descrito abaixo. Um Eletrodomestico pode estar ligado ou desligado e que tem capacidade de registrar os seus consumos. Quando ligado tem um consumo por milissegundo que é uma característica do electrodoméstico (quando desligado o consumo é zero!).

```

public class Eletrodomestico {
    private String ident;
    private LocalDateTime inicio;
    private LocalDateTime parcial;
    private double consumoLigada;
    private double consumoTotal;
    private double consumoParcial;
    private boolean estado; // false - desligada, true - ligada

    public Eletrodomestico(String ident, double consumo) {...}
    // liga o eletrodoméstico
    public void EletrodomesticoON() {...}
    // desliga o electrodoméstico
    public void EletrodomesticoOFF() {...}
    // devolve o consumo desde o início
    public double totalConsumo() {...}
    //devolve o consumo desde o último reset
    public double periodoConsumo() {...}
    public void efectuarResetConsumo() {...}
}

```

Crie agora a classe SmartHome que permite representar a associação de cada identificador de eletrodoméstico ao respectivo objecto, e faz a gestão das operações que a casa disponibiliza. Considere que todos os métodos equals, clone, toString estão definidos.

Responda às seguintes questões:

- (a) declaração de variáveis de instância e o construtor que aceita um Map com eletrodomésticos a adicionar à casa,
`public SmartHome(Map<String,Eletrodomestico> novosEletrodomesticos).`
- (b) `public void addEletrodomestico(Eletrodomestico e)`, que adiciona mais um eletrodoméstico à casa.
- (c) `public void desligaEletrodomestico(String id) throws...`, que desliga o eletrodoméstico identificado (apresente todas as definições necessárias).

PARTE III - 6 VALORES

3. Considere que os eletrodomésticos podem ser inteligentes, ecológicos ou ambos.

Os eletrodomésticos inteligentes têm a capacidade de programar a hora de ligar e desligar e implementam os métodos

```
public void turnOn(LocalDateTime d)
public void turnOff(LocalDateTime d)
```

Os eletrodomésticos ecológicos só podem funcionar num determinado período de horas, definido de acordo com: `public void setPeriodo(LocalDateTime don, LocalDateTime doff)`

- (a) Defina `EletrodomesticoInteligente` e `EletrodomesticoEcologico`
- (b) Que alterações devem ser realizadas na classe `SmartHome` para suportar estes 2 novos tipos de eletrodomésticos?
- (c) Implemente na casa inteligente os seguintes métodos:
 - i. `gravaEletrodomesticosEcologicos(String filename)` que grava no ficheiro de text o filename os eletrodomésticos Ecologicos
 - ii. `consumoParcial()` calcula o valor total dos consumos parciais dos electrodomésticos Inteligentes e Ecológicos

PARTE IV- 3 VALORES

Considere que a conhecida classe `Ponto` está já definida - tal como feito nas aulas - bem como a classe `Poligono`. Um fragmento desta última classe apresenta-se de seguida

```
public abstract class Poligono {
    private List<Ponto> pontos;
    ...
    public Poligono() {...}
    public Poligono(Lista<Ponto> pts) { ... }
    public List<Ponto> getPontos() { ... }
    public void addPonto(Ponto p) { ... }
    public abstract double area();
}
```

Resolva os seguintes exercícios:

- 4. Sabendo que um polígono convexo é um polígono, apresente a definição da classe `PoligonoConvexo` e codifique os seus métodos:

- (a) `public PoligonoConvexo(List<Ponto> pts) throws NaoConvexoException,`
assumindo que já está definida a classe `NaoConvexoException` e o método (da classe `PoligonoConvexo`) `public boolean eConvexo()` (não precisa de o codificar).
- (b) `public List<Triangulo> triangula(),` que calcula uma lista de triângulos cuja soma das áreas é igual à área do polígono convexo. Na resolução deste método considere o construtor da classe `Triangulo` da alínea seguinte.
- (c) Um triângulo é um polígono convexo. Defina a classe `Triangulo` e codifique o construtor parametrizado `public Triangulo (Ponto x, Ponto y, Ponto z)` (sabendo que um polígono tem de ser definido por uma linha poligonal fechada, isto é, o último ponto tem de ser igual ao primeiro) e ainda o método `area()` de um triângulo, que é calculado pela fórmula de Heron da seguinte forma:

```

a = distancia(x, y)
b = distancia(y, z)
c = distancia(z, x)
s = (a+b+c) / 2      // semi-perimetro
area = sqrt (s*(s-a)*(s-b)*(s-c))

```

Na resolução deste exercício assuma que a classe `Ponto` disponibiliza o método `public double distancia(Ponto p).`