Teste de Programação Orientada aos Objectos

2015.06.09
Duração: **2h**

Leia o teste com muita atenção antes de começar.
RESPONDA A CADA PARTE EM FOLHAS SEPARADAS.

Assuma que os métodos `get` e `set` de que necessitem estão disponíveis, a menos que sejam explicitamente solicitados.

PARTE I - 3 valores

1. Considere a seguinte implementação do método equals, da classe Aluno (cf. aulas teóricas),

```
1 public boolean equals(Object o) {
2           if (this == o)
3               return true;
4            if ((o == null) || (this.getClass() != o.getClass()))
5               return false;
6             else {
7               Aluno umAluno = (Aluno) o;
8               return(this.nome.equals(umAluno.getNome()) &&
                              this.nota == umAluno.getNota() &&
                             this.numero == umAluno.getNumero());
9             }
10         }
```

   De acordo com o que foi discutido nas aulas teóricas responda, de forma muito directa e objectiva, às seguintes questões:

   (a) o que fazem as linhas 4, 7 e 8 ?

   (b) no caso de termos agora uma subclasse de Aluno, seja AlunoTE, em que o método equals seja:

```
1 public boolean equals(Object o) {
2           if (this == o)
3               return true;
4            if ((o == null) || (this.getClass() != o.getClass()))
5               return false;
6             else {
7               AlunoTE umAluno = (AlunoTE) o;
8               return(this.nomeEmpresa.equals(umAluno.getNomeEmpresa()) &&
                              super.equals(umAluno);
9              }
10         }
```

como é que a linha 4, do equals da superclasse dá o resultado esperado?

Parte II - 6 valores

2. Como sabe, no fim de semana passado terminou a etapa portuguesa da Volvo Ocean Race (VOR) Além de acolher durante duas semanas a comitiva da prova, a organização solicitou também a um grupo de alunos que fizesse uma aplicação para gestão do evento. A VOR é constituída por equipas, que podem ter vários barcos em prova, e cada barco tem um skipper e tripulantes.

   Considere que a classe Barco contém um identificador único, informação sobre as milhas percorridas, a categoria do barco (catamaran, veleiro, etc.) e informação sobre a sua autonomia. A informação da Equipa deve prever que regista o nome da equipa e a informação dos vários barcos que por ela correm.

```
public class Barco {
  private String id;
  private double milhas;
  private String categoria;
  private double autonomia;
  private Pessoa skipper;
  private Set<Pessoa> tripulantes;
}

public class Equipa {
  private String nome;
  private Map<String,Barco> barcos;

}
```

   Tendo em conta esta descrição, responda às seguintes questões

   (a) Declare a classe VOR, com as variáveis de instância necessárias.

   (b) Faça o método `Barco getBarco(String idEquipa, String idBarco) throws InvalidBoatException`, que enviado à VOR devolve a instância de barco correspondente (caso esta exista).

   (c) Obter lista de barcos de uma equipa com mais de X milhas percorridas, `List<Barco> getBarcos(String idEquipa, double milhas)`

   (d) Remover barco dado o seu identificador, `void removeBarco(String idEquipa, String idBarco) throws InvalidBoatException`

Parte III - 8 valores

Considere agora que um Barco possui uma lista de registos para as diversas etapas que foram efectuadas. A classe RegistoEtapa contém 2 variáveis, que são 2 GregorianCalendar (um para a data de início, outro para a data de fim), além do nome da etapa e do número de milhas percorridas.

3. Codifique na classe Equipa, os seguintes métodos:

   (a) Obter o tempo total gasto em horas em prova (i.e., o somatório de todos os tempos Finais - Iniciais) para um dado barco,
   `double totalEmProva(String idBarco)`

   (b) Registo mais longo, obtido pelos diversos barcos da equipa,
   `double registoMaisLongo()`

4. Defina uma nova classe BarcoHibrido. Este tipo de barco tem um motor elétrico adicional ao motor de combustão, adicionando a capacidade da bateria (kWh) e potência elétrica (kW), bem como autonomia elétrica. Defina:

   (a) a nova classe (variáveis de instância, construtor parametrizado e toString)

   (b) implemente o método que determina a autonomia total (autonomia elétrica mais autonomia convencional), `double getAutonomia()`

Parte IV - 3 valores

5. Considere agora que existe uma hierarquia dos barcos, com a classe Barco como superclasse abstracta e tendo já como subclasses as classes BarcoCatamaran (que acrescenta às variáveis de instância de Barco a área de vela existente), BarcoRemos (que tem como característica a largura) e BarcoHibrido (que tem associada a potência do motor electrico).

   Considere que se quer identificar os barcos que podem ter um seguro associado, BarcoCatamaran e BarcoHibrido, e para esses barcos o valor do seguro é equivalente a 2% da distância percorrida até ao momento.

   (a) Explique como implementar e codifique as alterações necessárias nas classes BarcoCatamaran e BarcoHibrido.

   (b) Codifique um método que devolva o conjunto dos barcos que tenham seguro, ordenado crescentemente pelo montante de seguro a pagar.

   (c) Crie o método, `void gravaFicheiroTextoSeguraveis(String fich)`, da classe VOR que grava em ficheiro de texto os barcos que pagam seguro.