Lab XIII.

Objetivos

Os objetivos deste trabalho são:

- Aplicar conceitos de programação adquiridos em aulas anteriores;
- Rever e consolidar competências de desenvolvimento de software.

Como preparação para o exame prático, procure resolver estes problemas apenas com a ajuda da documentação partilhada na pasta e-learning e a documentação de JAVA.

XIII.1 Software para a empresa "ToShare"

Pretende-se construir um programa para gerir a empresa *ToShare*, de partilha de produtos. Como base, são fornecidos os seguintes ficheiros:

- *Product.java* – Interface que deve ser respeitada por qualquer Produto. Considere como possíveis entidades: *Car*, *Motorcycle* e *Van*.
- *OldJeep.java* – Classe antiga que se pretende usar como um *Product* em *ToShare*.
- *Client.java* – Cliente da *ToShare*, que pode partilhar e alugar produtos.
- *PDS2122.java* – Programa principal, que simula um cenário de uso da *ToShare*.

1. Crie entidades e associações adequadas à gestão da *ToShare*, de modo a executar no final o método *question1()*. Considere que a entidade que representa a *ToShare* deve implementar os seguintes métodos (entre outros que venham a ser necessários):
    - *public boolean add(Product p)*
    - *public Product remove(String code)*
    - *public Product remove(Product p)*
    - *public boolean share(String code, Client user)*
        - *Product with 'code' will be shared with 'user' and will be unavailable until giveBack*
    - *public boolean share (Product p, Client user)*
    - *public boolean giveBack(String code)*
    - *public boolean giveBack(Product p)*
    - *public String allAlugados()*

2. Acrescente código que permita à classe *ToShare* ler um conjunto de produtos (List<Product>) a partir de qualquer fonte de dados (por exemplo diferentes formatos de ficheiros de texto ou binários). A implementação da leitura não deverá ser da responsabilidade de *ToShare*.

   Por simplificação, considere apenas duas fontes de dados: 1) uma entidade que fornece os mesmos produtos do exemplo da alínea 1 (vetor *cars*); 1I) uma entidade que fornece os produtos listados no ficheiro TXT fornecido; Complete o código do método *alinea2* para testar a solução desenvolvida.

a)

3. Alguns clientes usam frequentemente o mesmo produto. Contudo nem sempre está disponível. Para resolver este problema, pretende-se:
   a. Criar um serviço que notifique os clientes sempre que um produto é devolvido (*giveBack*). Só deverão ser notificados os Clientes que manifestarem interesse nesse produto (invocação do método *share*, quando o produto já foi emprestado).
   b. Se um produto estiver emprestado, os novos pedidos de empréstimo que forem solicitados para esse produto (invocação dos métodos *share*), deverão ficar registados em *ToShare*. Logo que o produto seja devolvida, para além das notificações realizadas na alínea a) deverá ser atribuída ao cliente seguinte. Por exemplo:

```
   shareIt.share("UA0001", u1);        // true
b) shareIt.share("UA0001", u2);        // false (já está emprestada)
c) shareIt.giveBack("UA0001");         // true (devolvida por u1, emprestado
   a u2)
d) shareIt.giveBack("UA0001");         // true (devolvida por u2)
e) shareIt.giveBack("UA0001");         // false
```

   Desenvolva uma solução adequada e escreva no método *alinea3()* o código para validar essa solução.

   Nota: crie um novo package a3 para não alterar as soluções anteriores.

```
Question 1) ---------------------------------

--- All Products :
Van [code=AA22BB, descr=Chevrolet Chevy, 2020, points=180.0]
Car [code=BB44ZB, descr=Ford Mustang, Red, 2021, points=150.0]
Car [code=ZA11ZB, descr=Tesla, Grey, 2021, points=100.0]
Motorcycle [code=ZA33ZB, descr=Touring, 750, 2022, points=85.0]
--- Shared Products :
Total : 3
        Car ZA11ZB shared with Client [number=187, name=Peter Pereira]
        Car BB44ZB shared with Client [number=826, name=Mary Monteiro]
        Motorcycle ZA33ZB shared with Client [number=957, name=Anne Marques]
--- Shared Products :
Total : 1
        Motorcycle ZA33ZB shared with Client [number=957, name=Anne Marques]
--- All Products :
Van [code=AA22BB, descr=Chevrolet Chevy, 2020, points=180.0]
Car [code=BB44ZB, descr=Ford Mustang, Red, 2021, points=150.0]
Jeep [code=JJ0011, descr=Some old SUV, points=88.5]
Motorcycle [code=ZA33ZB, descr=Touring, 750, 2022, points=85.0]

Question 2 (output example) ---------------------------------

--- All Products :
Total : 13
        Motorcycle [code=900009, descr=BMW R1150R Boxer Naked Roadster Cinza ABS,
points=195.0]
        Motorcycle [code=900010, descr=Harley-Davidson Pan America 1250, points=205.0]
        Van [code=970005, descr=Jeep Grand Cherokee, points=150.0]
        Van [code=970006, descr=Alfa Romeo AR6, points=130.0]
        Van [code=980001, descr=GMC Safari, points=130.0]
        Van [code=980002, descr=Lancia Voyager, points=120.0]
        Car [code=990000, descr=Ford Maverick, points=120.0]
        Motorcycle [code=990001, descr=Ducati DesertX, points=185.0]
        Car [code=990002, descr=MAZDA MX-5 Miata, points=100.0]
        Van [code=AA22BB, descr=Chevrolet Chevy, 2020, points=180.0]
        Car [code=BB44ZB, descr=Ford Mustang, Red, 2021, points=150.0]
        Car [code=ZA11ZB, descr=Tesla, Grey, 2021, points=100.0]
        Motorcycle [code=ZA33ZB, descr=Touring, 750, 2022, points=85.0]

Question 3) ---------------------------------
```

XIII.2 Software para Gestão Portuária

Considere as seguintes entidades que fazem parte de um sistema de gestão portuária, nomeadamente para o controlo de entrada e de saída de embarcações num porto:

- Ship – Interface que representa qualquer tipo de embarcação.
- Port – Interface que representa um gestor de portos (por exemplo, do Porto de Aveiro, de Leixões, etc.). Os métodos de Port devem fazer o seguinte:
  - add(String code, Ship ship) – Dá entrada no porto de uma embarcação, identificada por um código. Caso o código já exista, a embarcação associada deverá ser substituída.
  - exists(String code) – Verifica se uma embarcação existe no porto.
  - remove(String code) – Devolve e remove uma embarcação do porto.
- PassengerShip e CargoShip – dois exemplos de Ship, com atributos comuns (inferidos pela interface) e distintos (o primeiro tem o número máximo de passageiros, o segundo tem a carga).
- SeaPort – Implementação de um gestor portuário marítimo.
- PDS2122R.java – Programa principal, que simula um cenário de utilização.

a) Complete a implementação com todas as classes e métodos necessário para a solução, de modo a correr o programa principal e a apresentar um output semelhante ao indicado.

b) Pretende-se agora que o SeaPort possa admitir (através do método add) uma nova embarcação (ShipOfSmallShips) que transporta outras embarcações, mas apenas do tipo PassengerShip. Inclua métodos para adicionar e remover PassengerShip, assumindo um número máximo de embarcações. Acrescente ainda o código necessário para apresentar um output semelhante ao indicado.

c) Crie um novo gestor de porto, RiverPort, que implemente Port e que:

i) reuse, por composição, um SeaPort;

ii) qualquer instância/objeto de RiverPort deve ser fornecida por um método fábrica;

iii) não aceite embarcações com carga superior a 10, ou que tenham uma lotação superior a 10 passageiros;

iv) aceite um objeto do tipo RiverLogger, classe que deve criar e que irá registar todas as operações que forem invocadas sobre um RiverPort (add, exists, remove). O formato e o destino das mensagens são à sua escolha, por exemplo, pode guardar numa List<String> ou escrever num ficheiro.

Construa o código necessário e apresente um output à sua escolha.

```
Alínea a) ---------------------------------

Ref: C02 - CargoShip [code=S101, name=Quebra Molas, cargo=155.5]
Ref: C03 - CargoShip [code=S923, name=Madalena, cargo=18.8]
Ref: C11 - CargoShip [code=S732, name=SoPingas, cargo=20.2]
Ref: P06 - PassengerShip [code=S078, name=Costeiro, passengers=25]
Ref: P35 - PassengerShip [code=S185, name=PDS All aboard, passengers=80]
Ref: P54 - PassengerShip [code=S199, name=Bananeiros, passengers=120]

Alínea b) ---------------------------------

Quebra Molas removed
Ref: C01 - Container Ship with 3 ships. Total passagers capacity : 14
               PassengerShip [code=B899, name=Bora, passengers=4]
               PassengerShip [code=B878, name=Riacho, passengers=2]
               PassengerShip [code=B785, name=Turista, passengers=8]
Ref: C03 - CargoShip [code=S923, name=Madalena, cargo=18.8]
Ref: C11 - CargoShip [code=S732, name=SoPingas, cargo=20.2]
Ref: P06 - PassengerShip [code=S078, name=Costeiro, passengers=25]
Ref: P35 - PassengerShip [code=S185, name=PDS All aboard, passengers=80]
Ref: P54 - PassengerShip [code=S199, name=Bananeiros, passengers=120]
Ref: X01 - CargoShip [code=S45, name=Beirão, cargo=81.0]
```