Lab V.

Objetivos

Os objetivos deste trabalho são:

- Identificar e utilizar padrões relacionados com a construção de objetos
- Aplicar boas práticas de programação por padrões em casos práticos

V.1 Empresa de aluguer de automóveis

Pretende-se implementar um programa para simular uma empresa de aluguer de automóveis, partindo do problema 8.1 da Unidade Curricular POO que frequentou no passado.
O programa permite manipular entidades como Motociclo, Automóvel ligeiro, Táxi, Pesado de mercadorias e Pesado de passageiros.
Reveja o código desenvolvido, nomeadamente as interfaces e classes pedidas nesse problema, e construa o código necessário para que o cliente possa executar pedidos como os apresentados no método *main* seguinte:

```
public class EmpresaAluguer {
    public static void main(String[] args) {

        Rental r = new Rental("Rental", "1234-567", "a@ua.pt");
        r.addVeiculo(VehicleFactory.createMotociclo("00-AB-12", "Honda", "CBR 600", 100, "desportivo"));
        r.addVeiculo(VehicleFactory.createAutomovelLigeiro("22-CD-34", "Volkswagen", "Golf", 110, "ABC123456789", 350));
        r.addVeiculo(VehicleFactory.createTaxi("44-EF-56", "Mercedes-Benz", "E-Class", 150, "GHI123456789", 400, "TAXI123"));
        r.addVeiculo(VehicleFactory.createPPEletrico("77-HI-89", "Tesla", "Model X", 500, 2000, "PPE123456789", 50, 100, 200));
        r.addVeiculo(VehicleFactory.createALEletrico("88-IJ-90", "Tesla", "Model 3", 500, "ALE123456789", 500, 1000, 2000));
        r.addVeiculo(VehicleFactory.createPesadoMercadorias("66-GH-78", "Volvo", "FH", 500, "MNO123456789", 20000, 40000));
        r.addVeiculo(VehicleFactory.createPesadoPassageiros("99-JK-00", "Volvo", "FH", 500, 2000, "PPE123456789", 50));

        for(Veiculo v : r.getStock()){
            System.out.println(v);
        }
    }
}
```

V.2 Serviço de comidas PagaLeva

Pretende-se criar um pequeno programa que dado um conjunto conhecido de comidas escolha o recipiente mais adequado ao seu transporte. Considere que os diferentes tipos de comidas são definidos segundo a seguinte interface.

```
public interface Portion {
        public Temperature getTemperature();
        public State getState();
}
public enum State {
        Solid, Liquid;
}
public enum Temperature {
        WARM, COLD;
}
```

Deve criar um conjunto de entidades que modelem as seguintes comidas:

Comida	Estado	Temperatura	Outros
Milk	Liquid	Warm	
FruitJuice	Liquid	Cold	FruitName
Tuna	Solid	Cold	
Pork	Solid	Warm	

Analogamente, devem ser criadas entidades para modelar os seguintes recipientes:

Recipiente	Adequado a:	
	Estado	Temperatura
PlasticBottle	Liquid	Cold
TermicBottle	Liquid	Warm, Cold
Tupperware	Solid	Warm, Cold
PlasticBag	Solid	Cold

Modele a solução e construa o código necessário para que o cliente possa executar pedidos como os apresentados no método *main* seguinte. Deverá criar as entidades/métodos tendo em conta os requisitos de temperatura e de estado de cada alimento.

```
public static void main(String[] args) {
    final int MENUS = 4;
    Portion[] menu = new Portion[MENUS];
    menu[0] = PortionFactory.create("Beverage",Temperature.COLD);
    menu[1] = PortionFactory.create("Meat",Temperature.WARM);
    menu[2] = PortionFactory.create("Beverage",Temperature.WARM);
    menu[3] = PortionFactory.create("Meat",Temperature.COLD);

    System.out.println("---- Thank you for choosing your meal! ----");
    for (Portion p : menu)
        System.out.println(p);

    Container[] containers = new Container[MENUS];
    for (int m = 0; m < MENUS; m++) {
        containers[m] = Container.create(menu[m]);
    }

    System.out.println("---- Take the packages: ----");
    for (Container c : containers) {
        System.out.println(c);
    }
}
```

Output:

```
---- Thank you for choosing your meal! ----
FruitJuice: Orange, Temperature COLD, State Liquid
Pork: Temperature WARM, StateSolid
Milk: Temperature WARM, State Liquid
Tuna: Temperature COLD, State Solid
---- Take the packages: ----
PlasticBottle with portion = FruitJuice: Orange, Temperature COLD, State Liquid
Tupperware with portion = Pork: Temperature WARM, StateSolid
TermicBottle with portion = Milk: Temperature WARM, State Liquid
PlasticBag with portion = Tuna: Temperature COLD, State Solid
```

V.3 Classe Calendar

Analise a implementação da classe *java.util.Calendar* e identifique padrões de construção usados nesta classe. *Nota*: pode consultar este código em https://github.com/openjdk/jdk/blob/master/src/java.base/share/classes/java/util/Calendar.java

Reporte as suas observações no ficheiro *lab05/calendar.txt*.