 <small>ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO</small>	Tipo de Prova Avaliação Contínua – Momento de Avaliação 2	Ano letivo 2019/2020	Data
	Curso LEI/LSIRC	Hora	
	Unidade Curricular Paradigmas de Programação	Duração	

## Objetivos

Com a realização do trabalho prático, pretende-se que os alunos ponham em prática todos os conhecimentos adquiridos na utilização do paradigma de programação orientado a objetos (POO) e a sua implementação na linguagem de programação Java, demonstrando as suas apetências em:

- Conhecer e compreender os conceitos fundamentais associados ao paradigma da programação orientada a objetos;
- Conceber e implementar, para problemas concretos, soluções que tenham por base o paradigma da programação orientada a objetos.
- Reconhecer e compreender a semântica e a sintaxe da linguagem Java.
- Reutilizar, alterar e desenvolver código recorrendo à linguagem Java tendo em vista um determinado problema com regras semânticas específicas.

Considere ainda que:

- Não é permitida a utilização de API's/conceitos Java que não tenham sido alvo de lecionação no ano letivo corrente da unidade curricular Paradigmas de Programação. Os alunos que pretendam utilizar API's adicionais devem atempadamente pedir autorização a um dos docentes da unidade curricular.
- Não é permitida a utilização de coleções Java predefinidas ([Java Collections Framework](#)).
- Os recursos de suporte ao trabalho referenciados no enunciado, são de utilização obrigatória.

## Enunciado

A empresa *PackSolutions* tem como principal atividade a disponibilização de soluções dedicadas ao empacotamento de encomendas de grande dimensão das mais diversas empresas da região.

Como resultado de um processo de modernização tecnológica, a empresa pretende desenvolver uma API em linguagem Java capaz de suportar os requisitos de uma ferramenta de suporte à acomodação de caixas (**Box**) dentro de contentores (**Container**). Desta forma, os colaboradores da *PackSolutions* podem mais facilmente gerir e visualizar os itens expedidos para cada encomenda.

## Requisitos gerais

A solução a desenvolver deverá ter capaz de criar encomendas para envio (**ShippingOrders**). Cada encomenda tem pelo menos um contentor (**Container**). Um **Container** representa uma “caixa” de grandes dimensões especialmente orientada para acomodar um conjunto de **Items** (caixas/**Box**) de menor dimensão que armazenam os itens a transportar, como ilustrado na Figura 1.

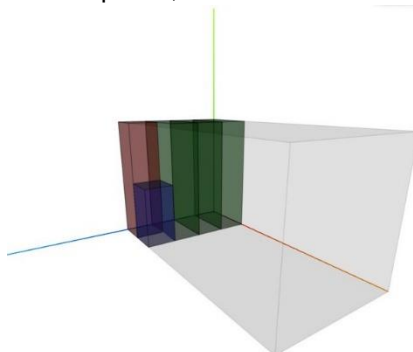



Figura 1. Exemplo de um container de uma encomenda (a cinzento o **Container** e os **Items** internamente representados em diversas cores)

A utilização da API deverá contemplar a especificação das dimensões do **Container** e tendo por base essas dimensões, deverá definir os **Items** que pretende acomodar e em que posições (considerando as coordenadas cartesianas).

 <small>ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO</small>	Tipo de Prova Avaliação Contínua – Momento de Avaliação 2	Ano letivo 2019/2020	Data
	Curso LEI/LSIRC	Hora	
	Unidade Curricular Paradigmas de Programação	Duração	

O objetivo passa por facilitar o processo de acomodação de itens em contentores, otimizando os custos de expedição das encomendas. Note que a API deverá apenas suportar o posicionamento de caixas de forma manual (com coordenadas cartesianas especificadas pelo utilizador).

## Descrição Técnica

Como suporte ao desenvolvimento da API, são disponibilizados um conjunto de recursos (*MA02\_Resources*), de utilização obrigatória e que definem os contratos que permitem o desenvolvimento da API. Os conteúdos fornecidos são um complemento ao presente enunciado, contendo informação específica sobre as particularidades de implementação de cada funcionalidade. A utilização dos contratos constitui um ponto de partida, cujos ficheiros não podem ser alterados. **Caso não utilize os recursos disponibilizados, todo o trabalho é invalidado.**

Deverá realizar a implementação do código necessário para suportar cada uma das operações definidas nos contratos. Teste o mais exaustivamente possível o código que desenvolveu como resposta aos requisitos apresentados. Recorra a comentários JavaDoc e não só de modo a documentar, o mais exaustivamente possível, o código que desenvolveu.

A existência dos contratos não deve ser impeditiva para a implementação de novas funcionalidades e/ou novos métodos ou classes.

Para complementar o processo de validação da API a desenvolver, é disponibilizado um componente que com base no código desenvolvido, apresenta uma *interface* gráfica. Poderá utilizar esta componente para **validação do código** produzido. A componente de interface gráfica **já se encontra implementada**, devendo ser utilizada após a implementação e validação da API.

A interface gráfica é exposta através da classe: `PackingGUI` que disponibiliza um método: `validate` (responsável por validar o ficheiro JSON<sup>1</sup> com os dados de entrada) e `render` (responsável por apresentar uma interface gráfica, utilizando o *browser* definido por defeito). O seguinte excerto de código apresenta um exemplo de invocação do método `render`:


```
(...)  
PackingGUI.render(pathToFile);  
(...)
```

Como argumento, o método `render` recebe uma *string* com o caminho para um ficheiro JSON. O JSON é um formato compacto para troca de dados entre sistemas, que suporta o armazenamento de dados no formato atributo-valor. No contexto da API, o ficheiro JSON suporta o armazenamento dos dados de uma encomenda, incluindo os seus contentores e os itens armazenados em cada contentor. Na Figura 2, é disponibilizado um documento JSON de exemplo que apresenta a estrutura necessária de forma a que a interface gráfica disponibilizada realize a correta interpretação dos dados produzidos pela API.

A estrutura do ficheiro, considera:

- Os dados gerais da encomenda: identificador (`orderId`), estado(`status`), dados do cliente com respetivas moradas (`customer`) e da pessoa de destino com respetiva morada (`destination`) para entrega da encomenda.
- Os dados de cada container (armazenados num array), contendo a informação do contentor: referência (`reference`), o volume máximo e ocupado (`volume` e `occupiedVolume`), as dimensões (`length`, `height`, `depth`) e a cor do contentor e das bordas (`color` e `colorEdge`).
- Os dados de cada item (armazenados num array) dentro de um contentor, contendo a informação de cada item: a referência (`reference`), descrição (`description`), as dimensões (`length`, `height`, `depth`), cor do item e das bordas (`color` e `colorEdge`) e as coordenadas cartesianas do item no contentor (`x`, `y` e `z`).

<sup>1</sup> JavaScript Object Notation

 <small>ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO</small>	Tipo de Prova Avaliação Contínua – Momento de Avaliação 2	Ano letivo 2019/2020	Data
	Curso LEI/LSIRC	Hora	
	Unidade Curricular Paradigmas de Programação	Duração	

```
{
  "orderId": 1,
  "status": "RECEIVED",
  "customer": {
    "id": 1,
    "name": "John Doe",
    "address": {
      (...)
    },
  },
  "destination": {
    "name": "Jane Doe",
    "address": {
      "country": "country A2",
      "number": 2,
      "street": "street A2",
      "city": "city A2",
      "state": "state A2"
    },
  },
  "billingAddress": {
    (...)
  },
  "containers": [
    {
      "reference": "c1",
      "volume": 125000,
      "occupiedVolume": 36,
      "depth": 50,
      "length": 50,
      "height": 50,
      "color": "white",
      "colorEdge": "black",
      "closed": true,
      "items": [
        {
          "reference": "ITEM1",
          "description": "ITEM1",
          "depth": 1,
          "length": 1,
          "height": 1,
          "color": "aqua",
          "colorEdge": "aqua",
          "x": 0,
          "y": 0,
          "z": 0,
        },
        (...)
      ],
    },
    (...)
  ],
}
```


Figura 2. Excerto de um documento JSON com dados de uma encomenda

Pode utilizar uma biblioteca Java para facilitar a interpretação do documento. A escolha da biblioteca de manipulação de documentos JSON a utilizar fica ao critério de cada grupo. A título de exemplo, é disponibilizado na plataforma moodle um excerto de código demonstrativo de utilização da biblioteca: json-simple<sup>2</sup>).

#### Nota:

A interface gráfica deve apenas ser utilizada após a implementação e validação de todo o código desenvolvido. Isto significa que tem de realizar a instanciação das diversas classes de forma a proporcionar a exportação (através da interface *IExporter*) do documento JSON que é utilizado pela interface gráfica.

<sup>2</sup> <https://code.google.com/archive/p/json-simple/>

 <small>ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO</small>	Tipo de Prova Avaliação Contínua – Momento de Avaliação 2	Ano letivo 2019/2020	Data
	Curso LEI/LSIRC	Hora	
	Unidade Curricular Paradigmas de Programação	Duração	

## Elaboração do trabalho

Este trabalho é realizado em grupo que deverá ser composto no **por 2 alunos** da unidade curricular. Os alunos devem comunicar atempadamente o seu grupo de trabalho na plataforma moodle, até ao dia **30 de Maio 2020**.

## Datas e considerações

O trabalho deve ser entregue até às **23:55** horas do dia **5 de junho de 2020**, devendo a entrega ser feita através da página da unidade curricular de Paradigmas de Programação em <http://moodle.estg.ipp.pt>.

A defesa do trabalho será realizada de acordo com a seguinte configuração:

- LSIRCT1 – 12/06/2020
- LSIRCT2 – 12/06/2020
- LEIT1 – 09/06/2020
- LEIT2 – 09/06/2020
- LEIT3 – 09/06/2020
- LEIT4 – 12/06/2020

A defesa será realizada por turnos e a data exata para cada aluno (tendo em consideração o horário da respetiva turma) será comunicada na plataforma moodle após a entrega do trabalho. No caso de um grupo ser constituído por alunos de turmas diferentes, a marcação do horário será aleatória.

Considera-se por defesa satisfatória, quando o aluno demonstra que realizou o trabalho submetido e que **domina todos os conceitos de programação orientada a objetos aplicados na resolução do trabalho**. Tentativas de fraude, resultarão na avaliação do trabalho como: **Fraude Académica**.

## Formato da entrega

Os trabalhos entregues deverão evitar (se possível) utilizar caminhos absolutos ou endereços específicos, de modo a que possam ser facilmente utilizados em qualquer máquina. Para além disso, e no sentido de facilitar a receção dos vários trabalhos recebidos, estes deverão observar as seguintes regras:

- Todos os elementos do grupo deverão submeter o trabalho no link respetivo (Entrega do Trabalho);
- O trabalho desenvolvido deverá ser entregue através do moodle, através da submissão de um ficheiro com o nome PP\_AC\_<nr\_do\_aluno>\_<nr\_do\_aluno>.zip, contendo:
  - Os ficheiros criados incluindo o(s) projeto(s) do IDE Netbeans e uma pasta com a distribuição (jar) da solução proposta.
  - Recorra a comentários JavaDoc, e não só, de modo a documentar, o mais exaustivamente possível, o código desenvolvido.
  - Cada ficheiro de código entregue por cada grupo terá de possuir no início do mesmo um comentário com pelo menos a seguinte informação (com as adaptações óbvias para cada aluno/grupo):

```

/*
 * Nome: <Nome completo do aluno>
 * Número: <Número mecanográfico do aluno>
 * Turma: <Turma do aluno>
 *
 * Nome: <Nome completo do colega de grupo>
 * Número: <Número mecanográfico do colega de grupo>
 * Turma: <Turma do colega de grupo>
 */

```

Os alunos que não realizem a entrega do trabalho até à data/hora definida serão sujeitos a **penalização** ou a **invalidação do trabalho**.