 <small>ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO</small>	<b>Tipo de Prova</b> Trabalho prático 1	<b>Ano letivo</b> 2020/2021	<b>Data</b>
	<b>Curso</b> LEI	<b>Hora</b>	
	<b>Unidade Curricular</b> Engenharia de Software 2	<b>Duração</b>	

## Enunciado

### Contexto e Objetivos

Considere a existência de uma API de suporte a uma ferramenta que apoia no processo de acomodação de caixas (Box) dentro de contentores (Container), de modo a simplificar a gestão e visualização de itens expedidos para uma determinada encomenda.

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

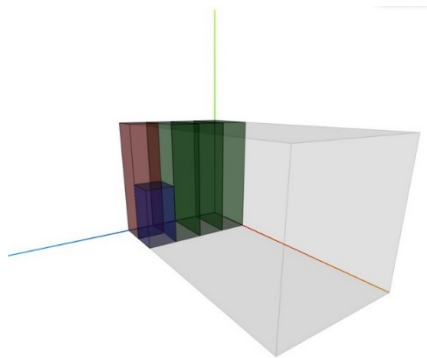



Figura 1. Exemplo de um container de uma encomenda (a cinzento o Container e os itens internamente representados em diversas cores)

A API permite a especificação das dimensões do Container e, com base nas dimensões fornecidas, define os itens que pretende acomodar de acordo com determinada ordem e posição (considerando as coordenadas cartesianas). O objetivo passa por facilitar o processo de acomodação de itens em contentores, procurando otimizar o tempo para acomodação das caixas no espaço disponível e consequentemente diminuir os custos de expedição das encomendas. Note que a API apenas suporta o posicionamento de caixas de forma manual (com coordenadas cartesianas especificadas pelo utilizador).

A API, antes de entrar em modo de produção, é necessário assegurar a qualidade do software desenvolvido.

Face ao exposto pretende-se:

- Verificar o uso de boas práticas de programação usadas durante o processo de desenvolvimento;
- Verificar o comportamento da API.

 <small>ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO</small>	<b>Tipo de Prova</b> Trabalho prático 1	<b>Ano letivo</b> 2020/2021	<b>Data</b>
	<b>Curso</b> LEI	<b>Hora</b>	
	<b>Unidade Curricular</b> Engenharia de Software 2	<b>Duração</b>	

As verificações supra-identificadas, devem ser implementadas considerando os métodos, técnicas e práticas abordados na unidade curricular de Engenharia de Software II.

## Descrição Técnica

A API disponibiliza um conjunto de contratos e respetiva implementação que permitem gerir a acomodação de itens de encomendas nos containers.

A API disponibiliza uma interface gráfica (classe PackingGUI). Esta interface admite um documento JSON com os dados de entrada e permite validar (método validate) e apresentar, visualmente, através de um browser (método render) a acomodação de itens num Container. No contexto da API, o ficheiro JSON armazena os dados de uma encomenda, incluindo os seus contentores e os itens armazenados em cada contentor. Nos recursos disponibilizados junto com a API, 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.

De modo a complementar a descrição técnica, é fornecido a documentação javadoc. Adicionalmente, os recursos para o trabalho encontram-se disponíveis em: [https://github.com/ESTG-ES2/TP1\\_2020\\_2021](https://github.com/ESTG-ES2/TP1_2020_2021). O recurso não contém o código fonte da API, apenas a biblioteca compilada no formato jar que deve ser usada para efeitos de análise de testes. O comportamento dos métodos deve ser avaliado apenas pelo javadoc disponível e pelo conteúdo deste enunciado.

Adicionalmente é também fornecido uma pequena demonstração da utilização da API num programa JAVA, que deve ser apenas utilizada para se familiarizarem com a API.

## Atividades a desenvolver


### A.1. Análise do problema

- Proceder à análise funcional do sistema e identificar as funcionalidades associadas à gestão de Containers e ShippingOrder. Pode recorrer a Use Cases.
- Seguir uma estratégia de testes de caixa preta para desenho de casos de teste, aplicando técnicas de Equivalence Class Partitioning (ECP) e Boundary Value Analysis (BVA);

### A.2. Identificação dos casos de teste

Com base na análise ECP e BVA, identificar um conjunto de casos de teste com uma breve descrição. A identificação dos casos de teste deverá conter a seguinte informação.

- Identificação do caso de testes
- Requisito/Use Case/Funcionalidade
- Pré-condições / estado(s) inicial

 <small>ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO</small>	<b>Tipo de Prova</b> Trabalho prático 1	<b>Ano letivo</b> 2020/2021	<b>Data</b>
	<b>Curso</b> LEI	<b>Hora</b>	
	<b>Unidade Curricular</b> Engenharia de Software 2	<b>Duração</b>	

- Inputs
- Valores de Inputs
- Resultados Esperados

### A.3. Codificação e execução dos testes

Codificar os testes usando o framework: JUnit 5<sup>1</sup> e executar os testes usando o gradle.

### A.4. Reporting

O reporting deve incluir não apenas a apresentação dos resultados da execução dos testes, mas documentar todo o processo e respetivas decisões de abordagem aos testes, itens a testar e critérios de passagem ou de falha.

## Documentação

Para auxílio à documentação deve considerar os documentos *Test Case Specification* e *Test Case Outline*. Para cada um destes documentos será disponibilizado um *template* no *moodle*. A par com cada um destes documentos, será disponibilizada a norma IEEE 829 que define a documentação relacionada com software testing.

Para a análise da aplicação e especificação dos casos de teste mais adequado, tenha em consideração as restrições específicas ao comportamento de cada componente descritas na documentação java disponibilizada como suporte à API.

## Submissão dos resultados do Trabalho Prático


O código desenvolvido deverá ser disponibilizado através de um repositório GIT da ESTG em <https://gitlab.estg.ipp.pt> e que deverão identificar devidamente usando o seguinte padrão para definição do nome do projeto: LEI\_ESII2021\_TP1\_GRUPO<X>. Ao projeto deverão ser dadas permissões para os professores.

No repositório GIT deverão estar refletidas as contribuições de cada membro do grupo, assim como a evolução do projeto. O relatório deverá estar integrado no repositório GIT, onde poderá ser estruturado recorrendo à linguagem de marcação *Markdown*<sup>2</sup>, ou via documento *pdf* incluído no repositório.

---

<sup>1</sup> <https://junit.org/junit5/>

<sup>2</sup> <https://github.com/adam-p/markdown-here/wiki/Markdown-Cheatsheet>

 <small>ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO</small>	<b>Tipo de Prova</b> Trabalho prático 1	<b>Ano letivo</b> 2020/2021	<b>Data</b>
	<b>Curso</b> LEI	<b>Hora</b>	
	<b>Unidade Curricular</b> Engenharia de Software 2	<b>Duração</b>	

O trabalho deverá ser realizado por grupos de 3 elementos (máximo). A identificação dos grupos deverá ser indicada na plataforma moodle até dia 20 de Novembro.

O período de entrega estará disponível até dia **7 de Dezembro pelas 23:59**. A entrega apenas é considerada após identificação no moodle do link para o repositório GIT, através de atividade pra o efeito.

## Considerações sobre critérios de avaliação

Serão objeto de avaliação os seguintes artefactos:

- Documentação produzida no âmbito das atividades realizadas. Contribui para a avaliação a descrição clara, objetiva, correta e completa das estratégias e técnicas utilizadas, bem como a adequada representação, usando, sempre que possível, as recomendações da IEEE;
- Os testes unitários codificados.

É ainda considerado para avaliação:

- A padrão de utilização da plataforma GitLab;
- A apresentação/Discussão do projeto.

Nota #1: Sempre que se justificar, alguns grupos podem ser convidados a realizar uma discussão de projeto mais alargado com o docente, em data e hora a combinar.

Nota #2: Aquando da apresentação, se algum estudante demonstrar, de forma inequívoca, um alheamento ao tema e conteúdo do projeto, ser-lhe-á atribuída nota "zero".

**Os Docentes:**

Cristóvão Sousa (regente)

Bruno Oliveira

Fábio Silva