 <small>ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO</small>	Tipo de Prova Avaliação Contínua – Momento de Avaliação 2	Ano letivo 2020/2021	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 competências 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 monitorização de parâmetros ambientais (por exemplo, de qualidade do ar ou nível de ruído) tem como objetivo controlar a qualidade ambiental, suportando a adoção de medidas que contribuam para o bem-estar da população de uma cidade.

Para suportar este processo de monitorização, existe uma rede de estações, cada uma delas equipada com diferentes sensores, que suportam a medição de parâmetros de qualidade relacionados com o ar, ruído, estado do tempo e até tráfego de uma cidade.

Pretende-se que desenvolva uma API em linguagem Java que seja capaz de suportar os requisitos de uma ferramenta de monitorização de parâmetros de qualidade ambientais, recolhidos através de vários sensores instalados nas diversas estações de controlo existentes.

Requisitos gerais


A solução a desenvolver deverá ter capaz de registar cidades (**ICity**) com diversas estações meteorológicas (**IStation**). Neste contexto, uma estação meteorológica é caracterizada por uma zona com um conjunto de sensores (**ISensor**) instalados. Cada sensor possui um identificador único assim como as coordenadas (geográficas e cartesianas) da sua localização. O código do sensor (com 10 caracteres) segue um padrão que permite identificar, o tipo de sensor (**SensorType**), o tipo de medição (associado ao tipo do sensor) e o parâmetro avaliado. Por exemplo: **QA0NO20001**:

- QA – Qualidade do ar
- NO2 – Sigla do Dióxido de Azoto

Cada sensor é de apenas um único tipo, nomeadamente, Qualidade do ar (QA), Ruído (RU) e de Meteorologia/Estado do Tempo (ME). Para cada tipo, são monitorizados diferentes parâmetros:

Qualidade do ar:

- Dióxido de Azoto (NO2) - $\mu\text{g}/\text{m}^3$
- Ozono(O3) - $\mu\text{g}/\text{m}^3$
- Partículas (PM2.5)– $\mu\text{g}/\text{m}^3$
- Partículas (PM10) – $\mu\text{g}/\text{m}^3$

 <div> ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO </div>	Tipo de Prova Avaliação Contínua – Momento de Avaliação 2	Ano letivo 2020/2021	Data
	Curso LEI/LSIRC	Hora	
	Unidade Curricular Paradigmas de Programação	Duração	

- Dióxido de enxofre (SO₂) – µg/m³
- Benzeno (C₆H₆) – µg/m³
- Monóxido de Carbono(CO) –mg/m³

Ruído

- Nível sonoro contínuo equivalente – (DB)

Estado do tempo

- Pressão atmosférica – mbar
- Humidade relativa - %
- Temperatura - °C
- Vento intensidade – km/h
- Vento direção - ° (grau)
- Precipitação – mm
- Radiação global – W/m²
- Radiação Ultravioleta – índice UV

De notar que os dados recolhidos, por vezes possuem erros que impedem a sua correta interpretação (e como tal não devem ser considerados). De notar, igualmente, que nem sempre existem dados para todos os parâmetros e podem existir novos parâmetros não documentados (na API). A API deverá suportar os parâmetros documentados e considerar como exceções os parâmetros não documentados.

Para cada sensor, é necessário registar as medições (**IMeasurement**), que são caracterizadas pelo valor da medição, a unidade de medida (que deverá ser validada de acordo com o tipo do sensor) e ainda a etiqueta temporal em que a medição foi capturada.

Descrição Técnica

Como suporte ao desenvolvimento da API, são disponibilizados um conjunto de recursos (*MA02_Resources*), de utilização obrigatória e que definem os contratos e definem o ponto de partida para o desenvolvimento da API. Os conteúdos fornecidos são um complemento ao presente enunciado, contendo informação específica sobre as particularidades de implementação de cada funcionalidade.


Deverá realizar a implementação do código necessário para suportar cada uma das operações definidas nos contratos. A existência dos contratos não deve ser impeditiva para a implementação de novas funcionalidades e/ou novos métodos ou classes. 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.**

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.

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

No contexto da API, são utilizados documentos JSON¹. O JSON é um formato compacto para troca de dados entre sistemas, que suporta o armazenamento de dados no formato atributo-valor. Este tipo de documentos suportam o armazenamento dos dados relacionados com a geração de visualizações disponíveis em: <https://quickchart.io/>.

¹ <https://www.json.org/json-en.html>

 <div> <div>ESCOLA</div> <div>SUPERIOR</div> <div>DE TECNOLOGIA</div> <div>E GESTÃO</div> </div>	Tipo de Prova Avaliação Contínua – Momento de Avaliação 2	Ano letivo 2020/2021	Data
	Curso LEI/LSIRC	Hora	
	Unidade Curricular Paradigmas de Programação	Duração	

O *quickchart* é uma API web que com base num documento JSON que contém não só a configuração da visualização, mas também dos dados a visualizar, permite gerar uma imagem correspondente ao tipo de visualização que se pretende obter. Os mecanismos para invocação da API já se encontram implementados nos recursos disponibilizados (classe `Dashboard`). Cada grupo de trabalho terá apenas que gerar o documento JSON adequado à visualização (deverá consultar o website da API para compreender a estrutura de documentos JSON que terá de gerar) e fornecer esse documento como argumento de método `render`.

A interface gráfica é exposta através da classe: `Dashboard` que disponibiliza o método `render` (responsável por apresentar uma *dashboard*, utilizando o *browser* definido por defeito). O seguinte excerto de código apresenta um exemplo de invocação do método `render`:

```
(...)  
Dashboard.render(arrayWithJsonFiles);  
(...)
```

Como argumento, o método `render` recebe um array com a lista de `strings` que representam as visualizações a apresentar.

```
{  
  "type": "bar",  
  "data": {  
    "labels": [  
      "Q1",  
      "Q2",  
      "Q3",  
      "Q4"  
    ],  
    "datasets": [  
      {  
        "label": "Users",  
        "data": [  
          50,  
          60,  
          70,  
          180  
        ]  
      },  
      {  
        "label": "Revenue",  
        "data": [  
          100,  
          200,  
          300,  
          400  
        ]  
      }  
    ]  
  }  
}
```

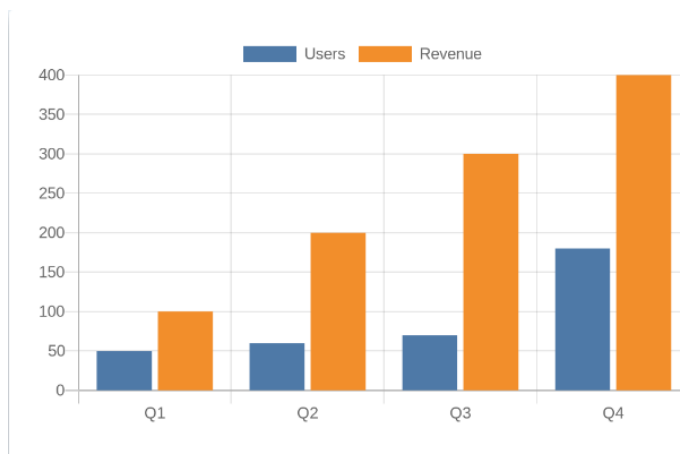



Figura 1. Excerto de um documento JSON e do respetivo output

Na Figura 1, é apresentado um documento JSON de exemplo (em anexo ao presente enunciado) que apresenta a estrutura necessária de forma que a *dashboard* disponibilizada realize a correta interpretação dos dados produzidos pela API do *quickchart*.

Como suporte ao trabalho prático, é disponibilizado na Figura 2 um documento JSON com um conjunto de leituras realizadas para a cidade de Lisboa, incluindo informação relacionada com estações, sensores e medições.

 <small>ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO</small>	Tipo de Prova Avaliação Contínua – Momento de Avaliação 2	Ano letivo 2020/2021	Data
	Curso LEI/LSIRC	Hora	
	Unidade Curricular Paradigmas de Programação	Duração	

```
[
  {
    "id": "QA00NO0001",
    "date": "202105121800",
    "dateStandard": "UTC",
    "value": 44,
    "unit": "µg/m3",
    "address": "Calçada da Ajuda",
    "coordinates": {
      "x": -92781,
      "y": -106663,
      "z": 0,
      "lat": 38.70263097,
      "lng": -9.199692206
    }
  },
  {
    "id": "QA0NO20001",
    "date": "202105121800",
    "dateStandard": "UTC",
    "value": 151,
    "unit": "µg/m3",
    "address": "Calçada da Ajuda",
    "coordinates": {
      "x": -92781,
      "y": -106663,
      "z": 0,
      "lat": 38.70263097,
      "lng": -9.199692206
    }
  },
  (...)
]
```

A estrutura do ficheiro, considera:

- O identificador do sensor (**id**) que contém informação sobre o seu tipo.
- A data (**date**), o valor (**value**) e a unidade (**unit**) da medição (a chave: **dateStandard** apenas é utilizada como metadados e como tal não deverá ser considerada)
- A morada (**address**) representa o nome da estação
- O objeto **Coordinates** representa as coordenadas do sensor.

Aconselha-se a utilização da biblioteca Java: JSON simple: json-simple², para facilitar a interpretação e escrita de documentos JSON.

Deve ainda incorporar a API numa aplicação funcional que permita ao utilizador através da consola importar o ficheiro com as medições, visualizar o relatório de importação (IOStatistics) e permitir a seleção das visualizações (deve implementar no mínimo 2 visualizações à escolha) que o utilizador pretende obter (considerando as consultas descritas no contrato: ICityStatistics).


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 do documento JSON que é utilizado pela *dashboard*. Pode (e deve) testar a API sem o documento JSON com as medições e pode também criar outros documentos JSON mais simples para testar a API.

Elaboração do trabalho

Este trabalho é realizado em grupo que deverá ser composto **por 2 alunos** da unidade curricular.

² <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 2020/2021	Data
	Curso LEI/LSIRC	Hora	
	Unidade Curricular Paradigmas de Programação	Duração	

Datas e considerações

Os alunos devem comunicar atempadamente o seu grupo de trabalho na plataforma moodle, até ao dia **8 de junho de 2021**.

O trabalho deve ser entregue até às **23:55** horas do dia **14 de junho de 2021**, 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 na semana de **15 a 18 de junho**.

A defesa será realizada por turnos e a data exata para cada aluno será comunicada na plataforma moodle após a entrega do trabalho.

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**.