


| | | | |
|--|---|-------------------------|------|
|  <small>ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO</small> | Tipo de Prova Trabalho prático – Época de Normal | Ano letivo 2022/2023 | Data |
| | Curso Licenciatura em Engenharia Informática Licenciatura em Segurança Informática em Redes de Computadores | Data entrega | |
| | Unidade Curricular Estruturas de dados | | |

Observações

Este trabalho destina-se a todos os estudantes inscritos na unidade curricular de **Estruturas de Dados** (ED) e irá servir para avaliar a respetiva componente prática. Os estudantes deverão juntar-se em grupos de **2 elementos** de modo a dividir, da melhor forma, as tarefas definidas neste trabalho. Excecionalmente, e quando se justifique, poderão ser considerados grupos com outro número de elementos.

Objetivos

- Utilizar os conhecimentos sobre estruturas de dados para escolher as estruturas de dados que melhor se aplicam à resolução do problema proposto;
- Desenhar e implementar, eficaz e eficientemente, o algoritmo de resolução do problema proposto.

Implementação

- Deverá ser usada a linguagem Java;
- O código deverá estar comentado através do JavaDoc;
- Não pode ser usada nenhuma coleção da plataforma de coleções do Java, sempre que for necessário terá de selecionar a estrutura de dados com o comportamento desejado desenvolvida durante as aulas (cada grupo deverá usar as **suas** versões).

Resumo

Pretende-se que seja desenvolvida uma *Application Programming Interface* (API) para a gestão de um jogo online tendo por base os pontos de interesse de uma cidade.

Os pontos de interesse (por exemplo, estátuas, memoriais, igrejas, etc.) referem-se a pontos turísticos relevantes que representam parte essencial do jogo. No contexto do jogo, os pontos de interesse são conhecidos como portais (**Portals**). Os portais estão conectados diretamente a outros portais ou a determinados locais conhecidos como **Connectors**. Os **Portals** e **Connectors** representam as possíveis rotas para alcançar os locais existentes. A Figura 1 apresenta uma representação de como as rotas estão definidas no jogo, traduzindo o caminho que os jogadores terão de realizar para chegar aos **Portals**.

No contexto do jogo, os jogadores utilizam o *smartphone* para interagir com estes locais de interesse. Cada jogador poderá dirigir-se a um local e a partir desse ponto calcular a rota que terá de realizar para chegar aos locais seguintes. Quando chega aos locais, o jogador pode também realizar ações específicas de acordo com determinadas condições.

Existem duas equipas às quais os jogadores podem pertencer: **Sparks** e os **Giants**. Cada uma destas equipas tem como objetivo “conquistar” os vários **Portals** existentes e assim controlar o maior número de **Portals** da região.

Cada jogador tem energia (valor entre $0-\infty$) que necessita para realizar as várias interações. O jogador pode:

- Para conquistar um **Portal** “sem equipa”, o jogador terá de o carregar com pelo menos 25% da energia máxima.

| | | | |
|---|---|-------------------------|------|
| P.PORTO ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO | Tipo de Prova Trabalho prático – Época de Normal | Ano letivo 2022/2023 | Data |
| | Curso Licenciatura em Engenharia Informática Licenciatura em Segurança Informática em Redes de Computadores | Data entrega | |
| | Unidade Curricular Estruturas de dados | | |

- Conquistar o **Portal** da equipa adversária, o que envolve descarregar a energia do **Portal**, consumindo a energia do jogador. Para conquistar o portal para a sua equipa, o jogador terá de o carregar com pelo menos 25% da energia máxima. Caso contrário fica como “sem equipa”.
- Caso o **Portal** seja da sua equipa, o jogador poderá descarregar energia para o **Portal** para o tornar mais “resistente” (somando a energia descarregada pelo jogador à energia do portal) até ao máximo pré-estabelecido para o portal.
- Quando o jogador interage com um **Connector** pode recarregar a sua energia em função da capacidade do **Connector**. O **Connector** não contém limitações de energia, não descarregando com o seu fornecimento aos jogadores. No entanto, cada **Connector**, tem um limite de tempo mínimo para voltar a servir o jogador, devendo para isso guardar sempre pelo menos a última interação com cada jogador (o que engloba o armazenamento da data da interação) para determinar se pode ou não voltar a fornecer um jogador específico.

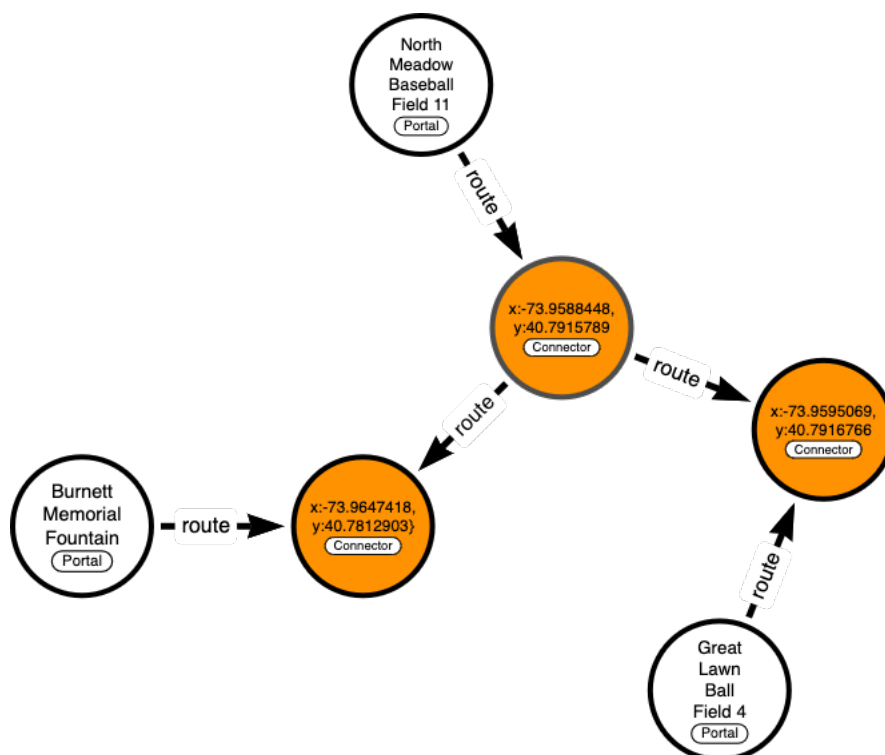



Figura 1. Exemplo de rotas envolvendo connectors e portals

Todos os locais possuem: Identificador único, longitude e latitude e a quantidade de energia que possuem).

Para além dos dados dos locais, os **portals** possuem o estado (neutro ou a equipa a que se encontra associado) e ainda o jogador que o conquistou.

Para além dos dados dos locais, os **connectors** contêm o intervalo específico de fornecimento de energia (a.k.a., *cooldown*) a um jogador após uma interação. Por exemplo, um local que só forneça energia de 5 em 5 minutos apenas permite que um jogador possa voltar a recarregar após esse período.

Para além da energia atual, o jogador possui um nome e a equipa a que pertence. O jogador possui ainda um nível que é definido à custa dos pontos de experiência obtidos nas interações, que deverão ser diferentes em função de cada tipo de interação (conquista de *portal*, reforço de energia de **connector**,

| | | | |
|--|---|-------------------------|------|
|  <small>ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO</small> | Tipo de Prova Trabalho prático – Época de Normal | Ano letivo 2022/2023 | Data |
| | Curso Licenciatura em Engenharia Informática Licenciatura em Segurança Informática em Redes de Computadores | Data entrega | |
| | Unidade Curricular Estruturas de dados | | |

etc), definidos como configurações globais do jogo. A atribuição dos pontos de experiência resulta no cálculo do nível de jogador através da fórmula:

$$(level/x)^y$$

com o valor do x a afetar a quantidade de pontos de experiência (valores mais baixos = mais pontos de experiência necessário por nível) e o y a determinar a rapidez com que os pontos de experiência necessários por nível devem aumentar (valores mais altos = maiores diferenças entre os níveis)¹. Em cada nível, a energia máxima do jogador deverá aumentar uma percentagem do valor da sua energia atual.

Para a componente de administração de jogo, a API deverá suportar:


- Gestão de **Portals e Connectors**:
 - Adicionar, editar e remover **Portals e Connectors**.
 - Para os **Connectors**, deverá ser possível adicionar e remover registos de interações que os jogadores realizam. É necessário armazenar a última interação de cada jogador.
 - Listar os **Portals/Connectors** vários critérios (por exemplo, ordenados por um determinado parâmetro).
 - Deve ser possível importar/exportar um ficheiro JSON com toda informação relativa a **Portals/Connectors**.
- Gestão de rotas:
 - Criar e remover rotas entre **Portals e Connectors**.
 - Deve ser possível importar/exportar um ficheiro JSON com toda informação relativa às rotas existentes e os **Portals/Connectors** de interesse associados.
- Gestão de jogadores:
 - Adicionar, atualizar ou remover jogadores.
 - Associar e desassociar jogadores a equipas.
 - Listar os jogadores e por equipa, por nível e por número de **Portals** atualmente conquistados.
 - Deve ser possível importar/exportar um ficheiro JSON com toda informação relativa aos jogadores existentes assim como de estatísticas relacionadas (ver ponto anterior).
- Gestão de jogo:
 - Calcular o caminho mais curto entre dois pontos (**Portals** e/ou **Connectors**).
 - Calcular o caminho mais curto considerando a necessidade de passar obrigatoriamente por locais para recarregar ou passar obrigatoriamente apenas por **Portals e Connectors** (se possível).
 - Deve ser possível exportar um ficheiro JSON com toda informação relativa aos caminhos mais curtos gerados indicando a informação de cada **Portals/Connectors** envolvidos.
 - Deve ser possível importar/exportar um ficheiro JSON com as configurações do jogo.

Implementação

Deverá implementar um programa que permita testar a API. O programa deve disponibilizar um menu para interagir com as diversas funcionalidades da API.

O programa deverá:


¹ Ver: <https://blog.jakelee.co.uk/converting-levels-into-xp-vice-versa/>

| | | | |
|---|---|-------------------------|------|
|  <div> ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO </div> | Tipo de Prova Trabalho prático – Época de Normal | Ano letivo 2022/2023 | Data |
| | Curso Licenciatura em Engenharia Informática Licenciatura em Segurança Informática em Redes de Computadores | Data entrega | |
| | Unidade Curricular Estruturas de dados | | |

- Iterativamente identificar o jogador e pedir a movimentação que pretende efetuar no “mapa” de jogo, validando os caminhos possíveis para que o jogador possa realizar a movimentação.
- Quando um jogador está posicionado num **Connector**, poderá ou não realizar o carregamento instantâneo da energia do jogador. Se o jogador voltar a acionar a mesma opção (antes do *cooldown* do **Connector**) o Conector deverá rejeitar a interação. Após o *cooldown*, o **Connector** fica recarregado para aquele jogador. O *cooldown* é aplicado a cada jogador individualmente.
- Quando um jogador está posicionado num **Portal**:
 - Pode ou não conquistar o **Portal** se estiver em modo neutro.
 - Pode ou não atacar o **Portal**, que resulta no desconto da sua energia na energia do **Portal**. Se a energia não for suficiente para derrubar o **Portal**, o **Portal** ficará com a energia restante até que seja atacado/recarregado;
 - Pode ou não recarregar o **Portal** com uma quantidade de energia selecionada.

Deverá ainda ser possível importar os dados do problema utilizando o seguinte formato JSON:

```
{
  "locals": [
    {
      "id": 1414,
      "type": "Portal",
      "name": "Jacqueline Kennedy Onassis Reservoir",
      "coordinates": {
        "latitude": 40.7656918,
        "longitude": -73.9737489
      },
      "gameSettings": {
        "energy": 180,
        "maxEnergy": 200,
        "ownership": {
          "player": "LoboMau"
        }
      }
    },
    {
      "id": 5212,
      "type": "Connector",
      "coordinates": {
        "latitude": 40.7697989,
        "longitude": -73.9723702
      },
      "gameSettings": {
        "energy": 110,
        "cooldown": 3
      }
    }
  ],
  "routes": [
    {
      "from": 1414,
      "to": 5212
    }
  ],
  "players": [
    {
      "name": "LoboMau",
      "team": "Sparks",
      "level": 5,
      "experiencePoints": 24998,
      "currentEnergy": 70
    }
  ]
}
```

| | | | |
|--|---|-------------------------|--------------|
|  <small>ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO</small> | Tipo de Prova Trabalho prático – Época de Normal | Ano letivo 2022/2023 | Data |
| | Curso Licenciatura em Engenharia Informática Licenciatura em Segurança Informática em Redes de Computadores | | Data entrega |
| | Unidade Curricular Estruturas de dados | | |

Pode adicionar, alterar campos do documento, devendo preservar a estrutura base.

Avaliação

- Apenas serão considerados para avaliação os trabalhos entregues antes da data-limite definida pelos docentes da UC e disponibilizada no Moodle. A não submissão do trabalho até esta data invalida a sua avaliação;
- A defesa é obrigatória e será realizada no dia do exame da época correspondente (ver calendário de exames). **A não comparência de um membro do grupo não invalida a defesa dos restantes;**
- Critérios de avaliação:
 - A escolha apropriada das estruturas de dados e o uso destas será o fator de avaliação preponderante em todas a funcionalidades implementadas.
 - Boas práticas:
 - Comentários e JavaDoc.
 - Uso de controlo de versões (desde o início do projeto).
 - Teste unitários.
 - Uso das convenções do Java (ex.: <https://www.geeksforgeeks.org/java-naming-conventions/>).