Proposta de Trabalho

Estruturas de dados não lineares e Programação Orientada aos Objetos (POO) aplicadas ao domínio dos transportes

Linguagens de Programação II

Rui Silva Moreira
rmoreira@ufp.edu.pt
Beatriz Gomes
argomes@ufp.edu.pt

Março 2016

Universidade Fernando Pessoa
Faculdade de Ciência e Tecnologia

1. Definição do problema

Pretende-se que os alunos modelizem, implementem e testem uma aplicação Java para apoiar a movimentação de passageiros numa rede de transportes, com recurso a informação referente a cada paragem, linhas, zonas, coordenadas, etc.

A aplicação deverá aceitar, como dados de entrada, a informação relativa a cada paragem, tal como a designação da paragem, a lista de linhas na qual está inserida e respectiva informação. Também deverá aceitar a introdução de informação relativa a cada passageiro, tal como o número de identificação, nome, data de nascimento, etc.

Pretende-se que utilizem uma estrutura do tipo Graph para estruturar e armazenar a informação das paragens. Cada paragem terá um conjunto de ligações ou arestas com outras paragem. As ligações têm dois nodos ou vértices, vários custos (e.g., valores monetários e temporais) e a lista de linhas a que pertence. Uma linha refere-se a um determinado percurso efetuado pelo veículo de transporte e pode ter paragens e ligações comuns a outras linhas. A aplicação deverá permitir combinar a informação dos passageiros com as ligações, de modo a determinar qual o custo da deslocação para um determinado passageiro. Os passageiros, dependendo da sua idade, poderão usufruir de descontos.

Pretende-se ainda que utilizem uma estrutura do tipo Tree para armazenar a informação dos passageiros. Cada passageiro deverá ter associado um saldo que deverá ser diminuído consoante deslocações do passageiro e aumentado no caso do passageiro efetuar carregamentos.

Embora o sistema pudesse ser distribuído, para facilitar a implementação, irá utilizar-se uma arquitetura *standalone*, ou seja, uma implementação que deverá funcionar num único PC. Os alunos deverão utilizar pacotes de software pré-existentes que oferecem estruturas de dados genéricas (cf. grafos e árvores), que possam ser reutilizadas na implementação do problema proposto. Desta forma não terão que implementar as estruturas de dados básicas, podendo concentrar-se na lógica e requisitos funcionais da aplicação proposta.

1.1. Requisitos funcionais

Pretende-se que os alunos sigam uma abordagem orientada aos objetos na modelização e implementação do problema proposto. Em concreto deverão desenhar os diagramas de classes necessários que permitam modelizar o problema, reutilizando pacotes/classes pré-existentes (cf. grafo, árvores, *hashmap*, etc.) através de herança, composição ou agregação. Em concreto a aplicação deverá cumprir os seguintes requisitos:

R1. O modelo de dados deverá permitir representar uma rede de paragens e respectivas ligações, bem como toda a meta-informação associada a estas entidades; deverá ser ainda integrada a informação sobre os passageiros; a ligação entre paragens deverá refletir, através de vários pesos, os custos monetários, espaciais e temporais da deslocação entre as paragens;

R2. Cada nó ou vértice do grafo representa uma paragem que deverá ter um conjunto de atributos/propriedades principais (e.g., nome, zona, etc.) e outros atributos que possam vir a ser necessários;

R3. O modelo de dados deve prever a utilização de algoritmos genéricos de gestão e verificação de grafos, nomeadamente:
   a) Algoritmos de cálculo: do caminho mais curto (com base na distância entre nós) entre duas paragens selecionadas; do caminho mais barato (com base no custo monetário entre nós); do caminho mais rápido (com base no custo temporal entre os nós); do caminho mais rápido com ou sem transbordo (com base na informação das linhas)
   b) Verificar se o mapa de paragens (cf. grafo) é conexo;
   c) Selecionar um sub-grafo e aplicar-lhe os mesmos algoritmos ou funcionalidades descritas anteriormente;
   d) Utilizar uma árvore para armazenamento dos dados dos passageiros e posteriormente permitir a manipulação conjunta de forma a facilitar as pesquisas e as ações de gestão de informação; Poderá usar-se, por exemplo, uma árvore `Red-Black` (RB) para guardar a lista de passageiros assim como toda a informação associada de forma a facilitar a pesquisa de passageiros bem como a manipulação da informação acerca do perfil desses passageiros;

R4. Deverá ser criada uma interface gráfica para visualizar o grafo de paragens bem como da árvore de pessoas, distinguindo de alguma forma os tipos de nós;

R5. Deverá ser possível gerir o grafo através da adição e remoção de nós/vértices e arcos/ ramos bem como edição dos seus atributos;

R6. Deverá ser possível efetuar e combinar vários tipos de pesquisas sobre as paragens, ligações e pessoas como, por exemplo:
   a) Listar as linhas que passam numa determinada paragem;
   b) Procurar e listar paragens de uma determinada zona;
   c) Efetuar pesquisas com vários critérios combinados por operadores booleanos (cf. *and*, *or*); Por exemplo, pesquisar passageiros com menos de 18 anos e mais de 65 anos;
   d) Determinar, para as coordenadas de um passageiro, qual a paragem mais próxima da sua localização, e para as coordenadas de destino, determinar qual a paragem mais próxima do destino final; depois de saber as paragens de inicio e fim poderão aplicar-se algoritmos de cálculo de caminhos mais curtos;

R7. A manipulação e gestão de toda a informação e das respectivas pesquisas deverão ser suportadas pela interface gráfica;

R8. Todos os dados referentes à aplicação e respectivas pesquisas deverão poder ser gravados em ficheiros de texto para consulta posterior dos utilizadores;

R9. Deverá ser ainda possível importar e exportar os dados do modelos de dados (cf. Grafo e dados relacionados) para ficheiros binários e de texto.

2. Objectivos

Pretende-se que os alunos modelizem e implementem a aplicação descrita, cumprindo todos os objectivos propostos. Deverão nomeadamente:

- Modelizar o problema através de diagramas de classes (UML), reutilizando estruturas de dados base (grafo, árvore, etc.) e respectivos métodos/operações (propriedades, travessias, pesquisas, etc.) que permitam representar e manipular os dados necessários;
- Implementar os algoritmos principais para a pesquisa e processamento da informação de acordo com as funcionalidades solicitadas;
- Implementar o modelo de dados OO utilizando a linguagem Java; em particular devem implementar todos os requisitos funcionais enumerados e outros que se revelem úteis ou necessários;
- Implementar uma interface gráfica que suporte a visualização e gestão de toda a informação (e.g., visualização da rede, gestão e edição dos nós, execução de pesquisas, etc.);
- Implementar a leitura e escrita de informação baseada em ficheiros de texto e binários.

3. Ficheiros e documentos a entregar

O projeto proposto deve ser implementado numa linguagem orientada aos objetos. Recomenda-se que o código seja complementado com os comentários apropriados, que facilitem a compreensão do mesmo e a respectiva geração automática de documentação.

O relatório final deve ser composto por 3 componentes complementares: i) o modelo de classes definido para o projeto (ficheiro **zargo**); o código fonte do projeto (diretório **src** das classes implementadas e testadas); ii) a documentação do código (páginas de HTML geradas automaticamente pela ferramenta Javadoc). Estes 3 componentes do projeto devem ser entregues num único ficheiro zip através do sistema de *elearning* até ao dia registado no *assignment*.

O projeto deverá ser também apresentado e defendido de “viva voz” no final do semestre em data a combinar com os docentes.