UNIVERSIDADE DE COIMBRA
FACULDADE DE CIÊNCIAS E TECNOLOGIA
Departamento de Engenharia Informática

Programação Orientada aos Objetos

Projeto 2018/19
Viagem de alunos do DEI

Data de Entrega: 17 de Dezembro

Descrição do Problema

Os alunos do DEI pretendem organizar uma viagem de estudo e querem desenvolver uma aplicação de suporte ao planeamento da viagem. A viagem deve contemplar 3 locais, cada um com vários pontos de interesse.

Os locais a visitar são caraterizados pelo nome da cidade onde estão situados. Cada local pode ter um ou mais pontos de interesse, incluindo universidades, museus, parques e bares. As universidades são caracterizadas pelo nome e lista de cursos relacionados com Engenharia Informática. Os museus, além do nome, possuem a descrição da sua temática (por exemplo, história natural, arte moderna, ou aviação). Os parques podem ser culturais ou de diversões (por exemplo, parques aquáticos). Os parques aquáticos possuem várias piscinas e equipamentos (por exemplo escorregas) e podem ou não possuir espetáculos de animais. Os bares são caraterizados pela classificação média dada pelos clientes. Todos os pontos de interesse possuem um horário de funcionamento. Alguns pontos de interesse têm custos de entrada associados. Também podem existir outras despesas, como por exemplo a entrada em espetáculos ou o consumo de bebidas.

A aplicação deve solicitar as preferências do utilizador em função dos locais e pontos de interesse a visitar. Todos os utilizadores devem indicar o montante máximo que pretendem gastar. Além das preferências referidas, os alunos de licenciatura podem indicar um ponto de interesse "hot" - que não querem perder, e os alunos de mestrado podem indicar um local que querem evitar (por exemplo uma cidade com elevados níveis de poluição).

Os locais a visitar em cada viagem devem poder ser visualizados por ordem crescente e decrescente de custo. O custo de cada percurso é calculado pela soma dos seguintes fatores (que devem ser aplicados de acordo com a natureza dos pontos de interesse visitados):

- Preço de entrada (museus e parques, considerando as várias atividades que podem ser realizadas)
- Despesas no ponto de interesse (parques e bares)
- Deslocação entre locais – considera-se que as deslocações entre pontos

de interesse no mesmo local têm custo zero.

Os resultados de todas as pesquisas realizadas pelos utilizadores devem ser armazenados de forma a possibilitar o tratamento de dados.

A aplicação deve permitir realizar as seguintes operações:

1. Registo dos utilizadores
2. Introdução das preferências do utilizador contemplando locais e pontos de interesse
3. Introdução do montante máximo que o utilizador pretende gastar
4. Apresentação das viagens que satisfazem a restrição de custo máximo, obedecem às preferências do utilizador e que visitam pelo menos um museu
5. Visualização da viagem que satisfaz as preferências do utilizador, incluindo a distância entre os vários locais, os custos em cada local e o custo total da viagem
6. Listagem dos locais e pontos de interesse mais populares – esta operação deve ter em consideração as interações dos utilizadores anteriores.

A interação com o utilizador deverá ser realizada através de uma **interface gráfica**. A aplicação deve ser disponibilizada com **ficheiros** contendo dados relativos aos locais e pontos de interesse (pelo menos 20 locais, cada um com um mínimo de 3 pontos de interesse diferentes) e também com a distância entre os locais. Após o registo do primeiro utilizador, todos os dados devem ser guardados em **ficheiros de objetos** e carregados sempre que a aplicação for iniciada.

Implementação

A aplicação deve ser implementada na linguagem Java e deverá ter em conta os seguintes aspetos:

1. Cada classe deve gerir internamente os seus dados, pelo que deverá cuidar da proteção das suas variáveis e métodos

2. Cada objeto deverá ser responsável por uma tarefa ou objetivo específico, não lhe devendo ser atribuídas funções indevidas

3. Utilize a *keyword* **static** apenas quando tal se justifique e não para contornar erros do compilador

Elabore um diagrama com as suas classes (em UML) antes de iniciar a implementação, para prever a estrutura do projeto.

Tenha ainda em conta os seguintes pontos que serão importantes na avaliação:

1. Comentar as classes, métodos e variáveis públicas segundo o formato Javadoc. Isto permitir-lhe-á gerar automaticamente uma estrutura de

ficheiros HTML, descritivos do seu código, que deve incluir no seu relatório

2. Comentar o restante código sempre que a leitura dos algoritmos não seja óbvia

3. Tal como sugerido acima, evitar o uso abusivo de **static** e de variáveis e métodos **public**

4. Na escolha de nomes para variáveis, classes e métodos, seguir as convenções adotadas na linguagem **Java**

5. Na organização das classes deverá ser evitada a redundância dos dados

Prazos de entrega

A entrega do trabalho compreende duas metas distintas:

Meta 1 – Análise do projeto e diagrama de classes (entrega até **9 de Novembro**);

Meta 2 – Versão final (entrega até **17 de Dezembro**).

MUITO IMPORTANTE:

Os trabalhos serão comparados (tanto entre os trabalhos da disciplina como com código disponível na Internet), no sentido de detetar eventuais fraudes por cópia. Nos casos em que se verifique que houve cópia de trabalho total ou parcial, os grupos envolvidos terão os projetos anulados, reprovando à disciplina. Serão aplicadas as regras da Universidade de Coimbra relativamente a plágio.

Material a entregar

Cada grupo deve entregar obrigatoriamente:

Meta 1: Diagrama de classes em UML

1. Upload em pdf do diagrama de classes no InforEstudante

2. Entrega de uma cópia impressa do diagrama no momento da defesa na aula teórico-prática

Meta 2: Aplicação

1. Upload de zipFile no InforEstudante com:
    - Todas as classes .java
    - Executável
    - Ficheiros de dados para teste
    - JavaDoc
    - Relatório (em formato pdf)

2. Relatório em papel, entregue no cacifo do docente da turma teórico-prática a que os alunos pertencem, descrevendo o programa do ponto de vista técnico e que deve incluir:
    - Estrutura geral do programa
    - Diagramas de classes inicial e final
    - Descrição das principais estruturas de dados e de ficheiros usados
    - Breve explicação de como o programa se executa

Avaliação do trabalho

Para a avaliação do trabalho contam fatores de dois tipos:

- Caixa preta (tal como é percecionado pelo utilizador):
    - Conjunto de funcionalidades implementadas;
    - Robustez do programa;
    - Qualidade da interface.
- Caixa branca (a forma como está construído):
    - Qualidade das soluções técnicas encontradas para os problemas em causa;
    - Estruturação do código;
    - Qualidade dos comentários.

A avaliação em cada uma das metas é feita individualmente, independentemente dos grupos serem constituídos por 2 alunos.

Nota: Não se aceitam trabalhos que apresentem erros de compilação no momento da defesa e que não estejam corretamente estruturados do ponto de vista da Programação Orientada aos Objetos.

Composição dos grupos

O trabalho deve ser realizado em grupos de 2 elementos da mesma turma

teórico-prática. A defesa dos trabalhos é feita individualmente, bem como a respetiva avaliação.

Defesa final do trabalho

O trabalho deve ser defendido através de uma discussão presencial e individual. Para isso, cada grupo deve inscrever-se num horário que esteja disponível para essa defesa no InforEstudante.