Trabalho Final (2016/2)

Disciplina de Técnicas de Programação

A) Data de entrega: 30/11/2016
Apresentações: 30/11/2016

B) Objetivo:

O objetivo deste trabalho é consolidar o conhecimento sobre conceitos e construção de sistemas orientados a objetos em Java em arquiteturas multicamadas através da exploração dos tópicos discutidos na disciplina de Técnicas de Programação.

C) Enunciado do problema:

Estamos interessados em um sistema de informação para armazenamento e cálculo de dados relativos a consumo de combustível de automóveis. De um modo geral, o sistema deve permitir o seguinte conjunto de operações por parte do usuário:

- cadastrar automóvel;
- alterar informações do automóvel;
- visualizar resumo das informações do automóvel;
- cadastrar abastecimento do automóvel;
- visualizar relatórios de consumo de combustível.

Cadastro de automóvel:

Um automóvel deve possuir o seguinte conjunto de dados: placa do veículo, modelo, ano, fabricante, capacidade do tanque de combustível e valor atual do odômetro em km. A um automóvel poderão ser associados vários abastecimentos.

Cadastro de abastecimentos:

Um abastecimento deve possuir o seguinte conjunto de dados: tipo de combustível, data, valor do odômetro do automóvel no momento do abastecimento, quantidade em litros abastecida, custo total e preço por litro.

Além disso o motorista deve ser capaz de informar se este abastecimento é o início de uma nova série. Exemplo: imagine que o motorista esqueceu de informar uma série de abastecimentos. Como o cálculo da quilometragem percorrida é feito em relação ao último abastecimento, não seria mais possível calcular o consumo do veículo. Então deve ser possível indicar que este abastecimento é o início de uma nova série sem perder os dados que já estão armazenados.

Resumo das informações dos automóveis:

O sistema deve apresentar o resumo das informações para todos os automóveis cadastrados em sua tela inicial:

- Sobre o automóvel: placa do veículo, modelo.
- Sobre o consumo de combustível (média dos últimos 6 meses, considerando os reinícios de série): consumo em litros por quilômetro, custo por quilômetro.

Relatório de consumo:

O sistema deve permitir a visualização dos seguintes relatórios (com filtragem por data de início, data de fim e placa do veículo) tabulados mês a mês (considerando os reinícios de série):

- Eficiência média do veículo em quilômetros por litro;

- Total de distância percorrida;
- Total de gastos em combustível.

Relatório de abastecimentos:

O sistema deve permitir a visualização de todos os abastecimentos efetuados (com filtragem por data de início, data de fim e placa do veículo).

Extras:

Os seguintes itens possuem implementação facultativa:

- Incluir a foto do veículo cadastrado no sistema;
- Apresentar os relatórios através de gráficos.

D) Requisitos:

Os seguintes itens são obrigatórios na implementação do sistema:

- Arquitetura multicamada;
- Uso dos padrões de projeto explorados em sala de aula, sendo obrigatoriamente:
    - Uso do padrão "MVC" na camada de interface gráfica do usuário (interface textual de console não será aceita);
    - Uso do padrão "Facade" para isolar a camada de domínio da camada de interface gráfica;
    - Uso do padrão arquitetural "Domain Model" na camada de domínio;
    - Uso do padrão "DAO" na camada de persistência (a qual deve ser implementada sobre arquivos).
- Implementação em Java;
- Tratamento correto do encapsulamento de exceções entre as camadas.
- Comprovação da funcionalidade via teste unitário com casos de teste suficientes e completos.
- Os arquivos de dados a serem importados deverão ter sido previamente populados com, no mínimo, os valores necessários para uma boa cobertura de casos de teste.

E) Desenvolvimento, apresentação e avaliação do trabalho:

- O trabalho pode ser realizado individualmente ou em grupos de, no máximo, 3 alunos.
- Os trabalhos serão apresentados no laboratório. Durante a apresentação, TODOS os alunos devem estar presentes e aptos a responder às perguntas. Respostas insatisfatórias por um aluno ou a sua ausência acarretarão descontos na nota final.
- A apresentação do trabalho é de inteira responsabilidade dos alunos (configuração da máquina, do ambiente de software, etc.) e o código-fonte utilizado deverá ser o mesmo entregue ao professor. É tarefa do grupo garantir que o sistema esteja apto a ser executado no dia da apresentação.
- Sistemas que não consigam ser executados ou apresentados no dia da apresentação receberão nota zero.
- Mensagens de erro apresentadas durante a execução do programa, mesmo que a aplicação não pare de executar, serão consideradas como erros de execução, e acarretarão descontos na nota do trabalho.
- Em caso de erro de sintaxe (compilação), o peso final do trabalho será valorado em zero.
- Em caso de erro de semântica (conteúdo), o peso final do trabalho sofrerá uma redução.
- Os trabalhos serão avaliados de acordo com critérios a serem estabelecidos pelo professor da disciplina, considerando o que é pedido no enunciado e o que foi realizado com sucesso pelo sistema. Também serão avaliadas a modelagem do sistema (correta criação das classes necessárias, com seus atributos e métodos, encapsulamento, coesão, acoplamento, e correto estabelecimento de relações entre as classes) e sua implementação de acordo com os conceitos de orientação a objetos e arquitetura multicamada.
- A comprovação do uso de teste unitário, padrões de projeto e de programação por contratos será levada em conta na avaliação do trabalho.

- ***Trabalhos copiados resultarão em nota zero para todos os alunos envolvidos.***

F) Entrega do trabalho:

- Todos os arquivos necessários a execução do sistema, bem como os arquivos-fonte e os arquivos de documentação, deverão ser empacotados em um único arquivo (.zip) e submetidos através do sistema Moodle até a data de entrega.
- Devem fazer parte da documentação pelo menos:
    - Diagrama de classes do sistema. O diagrama de classes deverá ser entregue junto com documentação em texto salientando os pontos onde foram utilizados padrões de projeto na implementação da solução. É importante que as classes estejam agrupadas em pacotes de acordo com a arquitetura de camadas do sistema. Diagramas gerados diretamente via engenharia reversa de código, sem qualquer organização lógica, serão desconsiderados na avaliação.
    - Os diagramas devem estar disponíveis em imagens com resolução suficiente e de fácil visualização (inclusive em documentos PDF). Não serão aceitos diagramas que estejam em formato original da ferramenta de desenho (como Visio, Astah, e outros).
    - Qualquer classe cujos métodos dependam de um contrato bem-definido e cuja interpretação da operação realizada não seja trivialmente definida deve apresentar a definição de pré e pós condições de forma adequada (não é necessário utilizar uma documentação JML, mas documentação JavaDoc será observada).
    - Casos de teste utilizados no teste unitário (de preferência em uma tabela que relacione entradas e saídas esperadas).
- Não serão aceitos trabalhos enviados por correio eletrônico.
- Não serão aceitos trabalhos enviados fora do prazo estabelecido.