UmCarroJá:
Análise e Teste de Software

DI, Universidade do Minho

Ano lectivo 2019/2020

João Saraiva & José Nuno Macedo

1 UmCarroJá: Sistema de Software

No ano lectivo 2018/2019, no contexto da disciplina de *Programação Orientada a Objectos* (POO) leccionada no Departamento de Informática da Universidade do Minho, os alunos tiveram de desenvolver em grupo uma aplicação Java para gerir um serviço de aluguer de veiculos particulares pela internet. O enúnciado desse trabalho, onde se descrevem todos os requisitos do sistema de software a desenvolver, encontra-se anexado a este documento.

No contexto da disciplina de Análise e Teste de Software (ATS) pretende-se que os alunos apliquem as técnicas de análise e teste de software, estudadas nas aula, de modo a analisar a qualidade de (algumas) soluções desenvolvidas pelos alunos de POO[1]. Assim, os alunos de ATS deverão:

- Analisar a qualidade do código fonte do(s) sistema(s) de software. Nesta análise os alunos deverão identificar *bad smells* no código fonte e o seu *technical debt*.
- Aplicar refactorings de modo a eliminar os *bad smells* encontrados, e deste modo reduzir (eliminar?) o *technical debt*.
- Testar o software de modo a ter mais garantias que ele cumpre os requisitos do enúnciado da aplicação UmCarroJá.
- Gerar inputs aleatórios para a aplicação UmCarroJá que simulem execuções reais (tal como foi fornecido em POO)
- Analisar a performance (tempo de execução e consumo de energia) da versão inicial do software (i.e., com *smells*) e a obtida depois de eliminados smells.

2 Tarefa 1- Qualidade do Código Fonte da Aplicação UmCarroJá

Nesta tarefa os alunos irão utilizar sistemas como o sonarQube, IDEs do Java, etc para analisar a qualidade do código da aplicação desenvolvida pelos alunos de POO. Métricas de software serão também utilizadas para encontrar *bad smells*.

[1] Algumas soluções desenvolvidas por alunos de POO estão disponíveis na página de ATS.

Tarefa Extra: Definir regras no sonarqube para identificar *red smells* (ou qualquer outro *smell* não suportado pelo sonarQube) numa aplicação Java.

3 Tarefa 2- Refactoring da Aplicação UmCarroJá

Nesta tarefa serão utilizadas ferramentas como o autorefactor, IDEs do Java que suportam refactoring, ou o jStanley para identificar e eliminar os *bad smells* e *red smells* existentes no software fornecido.

Um estudo detalhado sobre os *smells* encontrados na(s) aplicação(ões) fornecidas, os refactorings aplicados e o *technical debt* obtidos deverão ser incluídos no relatório.

4 Tarefa 3- Teste da Aplicação UmCarroJá

Nesta tarefa serão utilizadas técnicas de teste de software, para efectuar o teste unitário e o teste de regressão da aplicação UmCarroJá. Serão ainda utilizadas sistemas para a geração automática de casos de teste para gerar testes unitários e ainda inputs para simular a execução real da aplicação.

(mais informação em breve)

5 Tarefa 4- Análise de Desempenho da Aplicação UmCarroJá

Nesta tarefa será feita uma análide detalhada do desempenho da aplicação UmCarroJá, nomeadamente em termos de tempo de execução e consumo de energia. A aplicação será executada utilizando a framework de geração de inputs reais definidos na Tarefa 3. Duranta a execução da aplicação o tempo e consumo de energia serão monitorizados.

De modo a analisar a influência dos *bad smells* e *red smells* no desempenho do software as diferentes versões (com e sem *smells*) do software UmCarroJá serão comparados. Inputs de tamanhos diferentes deverão ser também considerados na análise.

(mais informação em breve)

Tarefa Extra: Efectuar uma análise detalhada por *smell*. Isto é, fazer um estudo sobre como cada smell individualmente influência (melhora ou piora) o desempenho do software.

6 Entrega e Avaliação

A entrega do projeto será feita no dia 14/01/2020. Na entrega pretende-se que os projetos apresentem o projeto oralmente (por exemplo, usando slides), e depois junto dos docentes mostrem a ferramenta construída a funcionar.

A avaliação do projeto terá em conta as seguintes componentes:

Tarefa 1:	10%
Tarefa 2:	10%
Tarefa 3:	15%
Tarefa 4:	20%
Extras:	15%
Relatório final:	15%
Apresentação/Defesa:	15%

POO (MiEI/LCC)

2018/2019

Enunciado do Trabalho Prático

Conteúdo

1 UM carro já!

Uma empresa de alunos de POO pretende criar um serviço de aluguer de veiculo particulares pela internet, muito parecido com o serviço de aluguer de casas *Airbnb*. Na aplicação a desenvolver, um proprietário de um automóvel poderá registar o seu veiculo na aplicação *UMCarroJá!* e este ser alugado por um cliente registado nessa mesma aplicação[1]. Tal como nos serviços *Uber* e *Airbnb* após o cliente escolher o veiculo a alugar, o proprietário terá de aceitar (ou não) o aluguer pedido. Para tal, um histórico de *feedback* de aluguer de clientes tem ser ser mantido, bem como do automóvel (para permitir aos clientes melhor escolherem o veiculo que desejam). Uma vez que os veículos são particulares, eles encontram-se estacionados na rua, e não concentrados numa agência de aluguer. Assim, a localização (coordenadas GPS) de um veiculo serão importantes para a escolha por parte de um potencial cliente. Por exemplo, um cliente pode desejar escolher o veiculo que está mais perto da sua localização. Um veiculo tem ainda a indicação do preço que é cobrado por quilometro percorrido, a quantidade de combustível existente no carro (percentagem de gasolina e/ou bateria), o consumo médio por quilómetro, bem como informação geral do automóvel e do seu proprietário.

Pretende-se que a aplicação a ser desenvolvida dê suporte a toda a funcionalidade que permita um utilizador alugar e realizar uma viagem num dos veículos da *UMCarroJá!*. O processo deve abranger todos os mecanismos de criação de clientes, proprietários, automóveis e posteriormente a escolha e aluguer de um automóvel, e a imputação do preço quando o mesmo é devolvido. Pretende-se ainda que o sistema guarde registo de todas as operações efectuadas e que depois tenha mecanismos para as disponibilizar (exemplo: alugueres de um cliente, extracto de aluguer de um carro num determinado período, valor facturado por um proprietário num determinado período, etc.).

Cada perfil de utilizador deve apenas conseguir aceder às informações e funcionalidades respectivas.

- Os clientes da *UMCarroJá!* poderão:
  - solicitar o aluguer de um carro mais próximo das suas coordenadas;
  - solicitar o aluguer do carro mais barato;
  - solicitar o aluguer do carro mais barato dentro de uma distância que estão dispostos a percorrer a pé;
  - solicitar o aluguer de um carro específico;
  - solicitar o aluguer de um carro com uma autonomia desejada.

  O proprietário do automóvel é a única pessoa que pode abastecer o seu depósito/bateria. Assim, no acto de aluguer o cliente deve indicar o destino da viagem de modo a ser possível verificar se o automóvel tem a autonomia para o alcançar. Não será permitido alugar um automóvel que não tenha a autonomia para alcançar o destino.
- Os proprietários poderão:
  - sinalizar que um dos seus carros está disponível para aluguer;
  - abastecer o veiculo;

[1] Uma aplicação com objetivos semelhante existe para Android e IOS com o nome *drivenow*

- alterar o preço por km;
- aceitar/rejeitar o aluguer de um determinado cliente;
- registar quanto custou a viagem.

2 Actores do sistema

Propõe-se a existência de dois tipos distintos de actores no sistema, que partilham a seguinte informação:

- email (que identifica o utilizador);
- nome;
- password;
- morada;
- data de nascimento.

Para além disso, cada actor tem um conjunto de dados específicos, como explicado de seguida.

2.1 Cliente

O Cliente representa a pessoa que solicita e efectua o aluguer de um carro. O cliente está sempre numa determinada localização (expressa em coordenadas x e y, isto é, num espaço 2D). O cliente tem também uma relação de todo o historial de alugueres que fez, com toda a informação relativa a cada aluguer que realizou.

2.2 Proprietário

O proprietário pode ter vários automóveis registados na *UMCarroJá!* e além da informação geral sobre o proprietário possui também dados relativos a:

- classificação do proprietário, dado numa escala de 0 a 100, calculada com base na classificação dada pelos clientes que alugaram os seus carros;
- histórico dos alugueres realizados;

3 Os Veículos da *UMCarroJá!*

O ecossistema da *UMCarroJá!* contempla diferentes tipos de veículos de aluguer. Neste momento estão em funcionamento os seguintes tipos de viaturas:

- carros a gasolina;
- carros eléctricos;
- carros híbridos.

Mas a gestão da *UMCarroJá!* considera que em breve poderá aumentar o tipo de veículos que fazem parte da sua oferta (neste momento está em estudo a utilização de bicicletas, trotinetes e pranchas de surf eléctricas!).

Cada um dos três tipos de viaturas tem associada:

1. uma velocidade média por km;
2. um preço base por km;
3. um consumo de gasolina/bateria por km percorrido;
4. acesso a um histórico de alugueres realizados;
5. classificação do carro, dado numa escala de 0 a 100, calculada com base na classificação dada pelos clientes no final do aluguer;

Um carro sabe sempre a localização (em x e y) onde está. Quando realiza um aluguer desloca-se para as coordenadas indicadas pelo cliente e fica aí parado até que seja solicitado um novo serviço. O carro tem também conhecimento da sua autonomia. Quando esta descer abaixo dos 10%, o seu proprietário deve ser imediatamente notificado e não é possível efectuar viagens nele.

4 Alugar na *UMCarroJá!*

O processo de aluguer na *UMCarroJá!* segue as seguintes passos:

1. o cliente indica as coordenadas x e y em que se encontra e para onde se pretende deslocar;
2. o cliente decide qual carro pretende alugar (de acordo com as possibilidades oferecida pela aplicação e descritas anteriormente);
3. De seguida é indicado ao cliente se o carro pretendido tem autonomia para efectuar a viagem, e caso o tenha, é indicado o custo estimado da viagem, tendo em conta o deslocamento que é necessário efectuar;
4. Caso o veículo tenha autonomia, o proprietário é notificado do pedido de aluguer e do tempo que o cliente demora a chegar a pé ao veiculo[2]. O proprietário poderá aceitar ou não o pedido de aluguer.
5. de acordo com a fiabilidade do carro (e de outros factores que pode considerar: a destreza do condutor, as condições meteorológicas, etc.) é calculado o tempo real da viagem.
6. o carro fica no ponto definido como fim da viagem à espera de - novo aluguer;
7. após a viagem o cliente e o proprietário podem dar uma avaliação ao carro ao cliente, e ficam com o registo relativo à viagem guardado nas suas áreas pessoais.

[2]Por uma questão de simplificação, os clientes/carros deslocam-se sempre em linha recta pelo que o cálculo da distância entre o cliente e o carro é feito pela cálculo da distância euclidiana (exemplo: se o cliente estiver em (0,0) e o veiculo em (2,2) a distância é de 2.8284 kms). Assume-se ainda que a caminhar uma pessoa percorre 4km por hora.

Como factor de diferenciação na solução que cada grupo de POO vai encontrar para resolver este problema, podem pensar que o tempo de viagem (logo o custo) além de ser função de um factor associado ao veículo, pode também ser função das condições atmosféricas ou condicionantes de trânsito. Nesse caso será necessário acrescentar informação a passar nos métodos que suportem estes requisitos.

5 Abastecimento dos veiculos

Actualmente temos carros que são apenas a combustão (gasolina ou gasóleo), híbridos plugin e totalmente eléctricos. É necessário saber aquando do abastecimento qual é o tipo de combustível necessário. Existem carros que necessitam de abastecimento de combustível (os gasolina/gasóleo e híbridos plugin) e carros que necessitam de abastecimento eléctrico (híbridos plugin e eléctricos). O sistema deve conseguir responder a questões como:

1. conseguir fornecer listas com os carros que cada um dos tipos (para permitir que os clientes escolham o tipo de carro em função do trajecto pretendido)
2. determinar quanto combustível (gasolina/gasóleo ou eléctrico) que o carro tem em determinado momento
3. determinar a autonomia do carro dependente do tipo de combustível que possui. Para tal é necessário determinar o consumo em termos do motor de combustão e do motor eléctrico (e no cado dos plugins da combinação dos dois tipos de "combustível").

6 Requisitos básicos

Identificam-se, de seguida, os requisitos básicos que o programa deverá cumprir: O programa deverá ainda fornecer uma interface de utilizador (em modo texto) que permita o acesso às funcionalidades.

- O estado da aplicação deverá estar pré-carregado com um conjunto de dados significativos, que permita testar toda a aplicação no dia da entrega.
- Registar um utilizador, quer cliente quer proprietário.
- Validar o acesso à aplicação utilizando as credenciais (email e password), por parte dos clientes e dos proprietários;
- criar e inserir viaturas;
- solicitar, por parte de um cliente, uma viagem de ponto P(x,y) para o ponto R(x,y), escolhendo uma viatura ou então solicitando a viatura mais próxima. Caso a viatura seja das que possuem lista de espera, inserir na lista de espera;
- classificar a viatura, após a viagem;
- ter acesso, no perfil de cliente, à listagem dos alugueres efectuados (entre datas);
- ter acesso, no perfil de proprietário, à listagem das alugueres efectuados (entre datas);
- indicar o total facturado por uma viatura num determinado período;

- determinar a listagens dos 10 clientes que mais utilizam o sistema (em número de vezes e em kms percorridos);
- gravar o estado da aplicação em ficheiro, para que seja possível retomar mais tarde a execução.

7 Teste de Sistema

A *UMCarroJá!* armazena num ficheiro de *logs* todas as operações realizadas por clientes e proprietários: registo de carro, registo de proprietário, registo de cliente, pedido de aluguer, etc. O ficheiro de *logs* do ano de 2018 está disponível num ficheiro de texto em que cada linha tem uma operação. Uma operação define o tipo de operação seguida de uma lista de atributos separados por virgula.

```
NovoCarro, <lista de atributos do carro separados por virgula>
NovoProp, <lista de atributos do proprietário separados por virgula>
Novocliente, <lista de atributos do cliente separados por virgula>
Aluguer, <lista de atributos do aluguer separados por virgula>
...
```

Um exemplo de um ficheiro de *logs* será

```
NovoCliente,Jose Silva,20674520,...
NovoCarro,electrico,AX-11-34,Opel,Jose,60,1.6,1
NovoCarro,gasolina,MC-74-94,Opel,Jose,60,1.6,5.5
NovoProp,Antonia

Aluguer,Jose,MaisPerto
ClassCarro,AX-11-34,90
Aluguer,Jose,MaisBarato
```

em que `Jose Silva` é o nome de um cliente com NIF `20674520`. O carro com matricula `AX-11-34` é eléctrico com 60 km/h de velocidade média, 1.6 (euros) é o preço por km, 5.5 é o consumo por km, ..

Uma descrição detalhada da estrutura deste ficheiro será fornecida mais tarde.

Para efectuar o teste de sistema do software desenvolvido o ficheiro (enorme) de logs de 2018 será disponibilizado durante o semestre, e a aplicação deverá ser executada com esse ficheiro como input. Por outras palavras, o software será executado em batch lendo os dados do ficheiro de *log*.

8 Relatório

O relatório deve descrever o trabalho realizado para desenvolver a aplicação solicitada. No mínimo, devem ser abordados os seguintes pontos:

- Capa com identificação da Unidade Curricular e do grupo.
- Breve descrição do enunciado proposto.

- Descrição da arquitectura de classes utilizada (classes, atributos, etc.) e das decisões que foram tomadas na sua definição.
- Descrição da aplicação desenvolvida (ilustração das funcionalidades).
- Discussão sobre como seria possível incluir novos tipos de viaturas na aplicação.

9 Salvaguarda do estado da aplicação

O programa deve permitir que em qualquer momento se possa guardar em ficheiro a informação existente em memória sobre os clientes, viaturas, alugueres, etc. A gravação deve ser feita de forma a permitir que o estado que foi gravado seja recuperado novamente. Na altura da entrega do projecto deve ser também entregue um estado (guardado em ficheiro) que possa ser carregado durante a apresentação.

10 Patamares de classificação do projecto

Este projecto de POO tem previstos dois patamares de dificuldade em função dos requisitos anteriormente identificados:

1. Requisitos básicos de criação de viaturas, clientes, proprietários e registo de alugueres e viagens: nota máxima 16 valores;
2. Itens anteriores mais gestão de factores de aleatoriedade, na duração da viagem e fiabilidade das viaturas, e gestão dos diferentes tipos de combustível dos diferentes carros: nota máxima 20 valores

Para que um projecto seja considerado positivo (com nota maior ou igual a 10) será necessário que se consiga efectuar, durante a apresentação, o registo e posterior consulta de um aluguer de viatura. Obviamente que a estruturação da solução deve respeitar as normas da programação orientada aos objectos, cf aulas teóricas, nomeadamente o encapsulamento, a abstração de implementação e a capacidade de a aplicação evoluir de forma controlada (eg: através da incorporação de novos veículos, etc.)

11 Cronograma

A entrega do projecto far-se-á de forma faseada, nas seguintes *milestones*:

1. Entrega de projecto BlueJ com uma versão inicial das declarações das classes (pelo menos com as variáveis de instância). Entrega da composição do grupo.
   Data Limite: 3 de Maio (esta fase é eliminatória, isto é, os grupos que não entregarem não poderão submeter o projecto final).
2. Entrega final de código e relatório de projecto (feita por via electrónica no elearning)
   Data Limite: 25 de Maio
3. Apresentação presencial do projecto
   Semana de: 3 a 7 de Junho