MC302
Primeiro semestre de 2018

Laboratório 3

Professora: Esther Colombini (esther@ic.unicamp.br)
PEDs: Nathana Facion (nathanafacion@gmail.com), Rafael Tomazela (sohakes@gmail.com), Luis Fernando Antonioli (luisfernandoantonioli@gmail.com))
PAD: Anderson Cotrim (ander.cotrim@gmail.com)

---

1 Descrição Geral

Neste laboratório, vamos continuar com a construção do sistema de carona on-line, reaproveitando o que foi feito nos dois laboratórios anteriores.

2 Objetivo

O objetivo desta atividade consiste na familiarização do aluno com a construção de classes que possuam atributos *final*, além do uso de *arrays* e *ArrayList*.

3 Atividade

3.1 Definições

Adicionaremos a classe referente à carona nessa atividade. A classe deve conter o atributo *final* caronante, além de um array com os caroneiros. Na figura 1 apresentamos o diagrama UML desta classe.

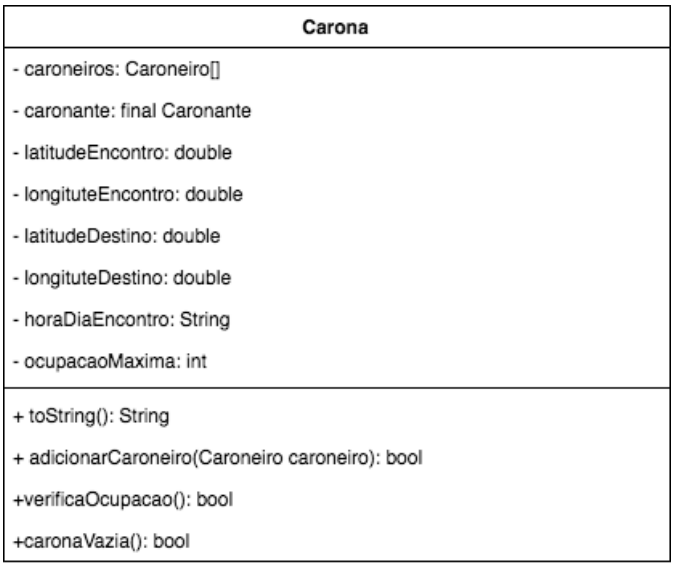

Figura 1: Diagrama UML da classe Carona

Como tarefa:

1. Implemente a nova classes conforme apresentado no diagrama UML 1. Os *getters*, *setters* e construtores foram omitidos mas eles precisam ser criados para cada atributo, a não ser quando estabelecido explicitamente. Não crie um *setter* para o *array de caroneiros*, ao invés disso instancie um *array* vazio em todos os construtores que achar necessário criar. Lembre-se de que o array precisa ser criado e a sua capacidade máxima deve ser definida. Além disso, atribua o valor do atributo *assentosDisponiveis* do caronante, ao valor de *ocupacaoMaxima* como um possível valor padrão;

2. Implemente o método *adicionarCaroneiro(Caroneiro caroneiro)*, que deve adicionar o objeto *caroneiro* passado por parâmetro na lista *caroneiros*. Caso haja mais caroneiros do que a ocupação máxima, o método não deve adicionar o caroneiro, e deve retornar falso. Senão, o método deve retornar verdadeiro;

3. Implementes os métodos que indicam: o número de caroneiros atual (verificaOcupacao()), se a carona não tem caroneiros (caronaVazia() - retorna true se o número de caroneiros for 0 e false caso contrário);

4. Modifique o atributo que corresponde à carteira de motorista na classe *Caronante* para que seja *final*;

5. Sobrescreva o método *toString()* da classe *Carona* de forma que imprima todas as informações, incluindo a informação de cada caroneiro presente na lista *caroneiros*;

6. No método *main*, crie um objeto da classe *Caronante* e quatro objetos da classe *Caroneiro*. Construa então um objeto *Carona* que tem o caronante criado anteriormente como o atributo caronante. Atribua com um *setter* o atributo *ocupacaoMaxima* para 3 (ou crie o caronante com o atributo *assentosDisponiveis* como 3, que deverá ter o mesmo efeito dado o item 1). Agora utilize o método *adicionarCaroneiro* para adicionar os 4 caroneiros criados anteriormente na carona. Imprima a saída do método, e verifique se o retorno foi verdadeiro nas três primeiras adições, e falsa na última. Também imprima o objeto carona (que utilizará o método *toString()*), e verifique se só 3 dos caroneiros foram adicionados.

7. Finalmente, repita a construção da classe Carona para que a mesma tenha como atributo caroneiros um *ArrayList de caroneiros* no lugar de um array (Figura 2). Chame essa nova classe de CaronaAL. Realize no main os mesmos procedimentos que foram realizados para o caso da classe Carona.

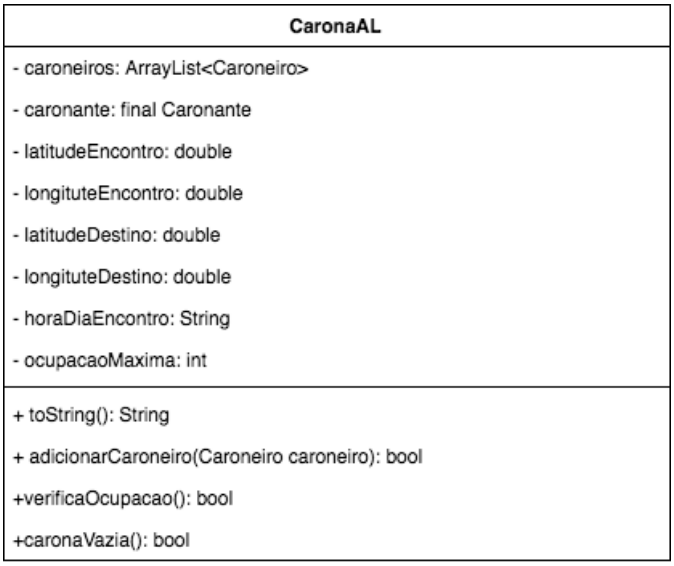

Figura 2: Diagrama UML da classe CaronaAL

3.2 Observações

- Como o atributo *caronante* da classe *Carona* é *final*, é necessário que o construtor da classe receba pelo menos esse atributo. Além disso, deixa de fazer sentido a criação de um *setter* para esse atributo (nem é possível);
- Com a mudança no atributo referente à carteira de motorista da classe *Caronante*, provavelmente será necessário modificar os construtores já existentes dos laboratórios anteriores;
- Não peça, em nenhum momento, dados pela entrada padrão. Construa os objetos com valores inseridos diretamente no código.

4 Questões

Sobre a atividade realizada, responda como comentário no início do código da classe que contém o método *main*.

- Tente modificar o valor da carteira de motorista (que agora é um atributo final) de algum caronante. Crie um *setter* se necessário. Foi possível fazer a modificação? Explique.
- Agora, no método main, crie uma variável *final* do tipo *Caronante*, e instancie ela com os valores que preferir. Tente modificar algum atributo do objeto através de um *setter*, como o atributo referente ao tempo de habilitação. Foi possível modificar esse atributo, mesmo com o objeto sendo *final*? Por quê?
- Agora adicione o modificador *final* no atributo *caroneiros* da classe *Carona*, e execute seu programa novamente. O comportamento do programa mudou? Por que essa alteração fez/não fez efeito?
- No cenário que foi construído, onde a ocupação máxima é definida no objeto carona, qual(is) a(s) vantagem(ns)/desvantagem(ns) (caso existam) de se adotar a solução com array e com ArrayList?

5 Submissão

Para submeter a atividade utilize o Moodle (`https://www.ggte.unicamp.br/ea`). Salve os arquivos dessa atividade em um arquivo comprimido no formato .tar.gz ou .zip e nomeie-o **Lab3-000000.[tar.gz | zip]** trocando '000000' pelo seu número de RA. Submeta o arquivo na seção correspondente para esse laboratório no moodle da disciplina MC302.

Datas de entrega

- Dia **26/03/2018** Turma **ABCD** até às 23:55