

MC322 – Programação Orientada a Objetos

Laboratório 07 – 2s2021

TESTE PRÁTICO 2

Leonardo Montecchi (Professor)

Thales Eduardo Nazatto
Leonardo de Sousas Rodrigues

Ângelo Renato Pazin Malaguti

Para perguntas ou dúvidas usem o Discord (<https://discord.gg/Xzpz9epNTG>)

1 Submissão

Data de entrega

- 10/10/2021 até às 23h59.

Submissão

- Ao criar o projeto Java no Eclipse (ou ferramenta equivalente), selecionar **JavaSE-11** como JRE
- IMPORTANTE: Nomear o projeto na forma **RA_Lab07** e o pacote base na forma **com.unicamp.mc322.lab07**. Substitua RA com o seu *Registro Acadêmico* (matrícula).
- Submeta o trabalho no link de entrega na página do Classroom da disciplina, em formato de arquivo compactado (zip) com o nome **RA_Lab07.zip**. Substitua RA com o seu *Registro Acadêmico* (matrícula). Entregas feitas de outras formas não serão consideradas.
- O arquivo compactado deve conter o projeto inteiro ("File / Export" no Eclipse, ou crie o arquivo manualmente).

Critérios de avaliação

- **Este laboratório VALE nota.**

Faz parte da avaliação saber quais classes devem ser criadas e em quais classes cada método deve ser implementado. Em particular, o código será avaliado de acordo com os seguintes aspectos:

1. Definição de Classes (30%).
 - Ex: *Classes utilizadas, hierarquia das classes.*
2. Aplicação de princípios de Programação Orientada a Objetos (POO) (30%).
 - Ex: *Reutilização de códigos da herança, utilização do polimorfismo, encapsulamento de atributos, responsabilidades de classes.*
3. Funcionalidades implementadas (40%).

2 Especificação do Sistema

Você foi contratado para desenvolver o AirPOO, um software para reserva de apartamentos para curta temporada. O aplicativo deve implementar as seguintes funcionalidades.

- Existem vários tipos de *residências* que podem ser alugadas. Cada residência possui um nome e uma posição, que pode ser representada como um par de coordenadas (x, y) . Cada residência possui um valor base de aluguel por dia.

- Um *apartamento* possui informações como: número de quartos, número de banheiros e o andar em que ele se encontra. Ainda, pode ou não ter sacada.
- Uma *casa* possui informações como: número de quartos, número de banheiros e se possui piscina ou não.
- Uma *mansão* é definida com base nos metros quadrados.
- O sistema permite também alugar *redes*. Uma rede tem apenas um preço fixo por dia.
- O sistema permite também comprar *experiências*. Cada experiência possui uma posição (onde a experiência será desenvolvida) e um número máximo de participantes. Uma experiência tem um preço base e um preço descontado para menores de idade.
- O sistema mantém um cadastro dos elementos que podem ser alugados e deve permitir a adição e remoção deles. *Pode-se considerar que a classe que implementa o sistema receba diretamente os objetos para ser adicionados ao cadastro.*
- Cada residência e cada experiência possui um identificador alfanumérico que deve ser único no sistema. Isto é, na fase de cadastro não devem ser admitidos identificadores duplicados.
- O sistema permite adicionar “pontos de interesse” da região, isto é, lugares turísticos ou de interesse para visitantes. Lugares de interesse são definidos por um nome e por um par de coordenadas (x, y) .
- Uma *reserva* contém o identificador do elemento que foi alugado (ou comprado, no caso das experiências), o número de pessoas totais, quantas dessas são menores de idade, e o número de dias da reserva. Se o número de dias não for especificado, deve-se assumir que o aluguel é de 1 dia.
- O valor de uma reserva é calculado de acordo com as seguintes regras:
 - Para apartamentos, o preço base é incrementado de 1% para cada andar. Isto é, +3% caso o apartamento esteja no terceiro andar. Se o apartamento tiver sacada, é adicionada mais uma diária ao valor total da reserva.
 - Para casas, a diária é ajustada com um fator baseado na proporção entre banheiros e quartos. A diária efetivamente aplicada é calculada como: $d_{efetiva} = (n_{banheiros}/n_{quartos}) \cdot d_{base}$. Se tiver piscina, é cobrada uma diária extra ($d_{efetiva}$) por dia, mas apenas pelos primeiros 7 dias de reserva.
 - Para mansões, o preço da diária aumenta exponencialmente a cada dia de estadia, com um fator baseado na proporção entre o número de pessoas da reserva e os metros quadrados da imóvel. A diária efetiva do k -ésimo dia é calculada como:

$$d_{efetiva}^k = d_{base} \cdot \left(\frac{100 \cdot n_{pessoas}}{mq} \right)^{k-1}$$
 - Para redes, o preço por dia é multiplicado pelo número de pessoas e pelo número de dias. Redes podem ser reservadas por máximo 5 dias. Para reservas mais longas o valor da reserva deve ser zero.
 - Para experiências, o número de dias não influi no preço. O valor é dado simplesmente pelo número de adultos multiplicado pelo preço base, mais o número de menores de idade multiplicado pelo preço descontado. Se o número de pessoas totais da reserva for maior do que o número máximo admitido o valor da reserva deve ser zero.

- Recebendo como input uma lista de reservas, o sistema deve calcular o valor total a ser pago.
- O sistema deve permitir imprimir os detalhes de todas as residências e experiências disponíveis, incluindo, para cada uma delas, a distância de todos os pontos de interesse cadastrados.

3 Exemplo de fluxo de execução

Para se auxiliar no desenvolvimento, considere o seguinte cenário como exemplo de execução. Esse fluxo deve ser considerado com uma das possíveis execuções do programa e não faz parte dos critérios de avaliação. Os critérios de avaliação estão especificados no começo deste documento.

1. Adicione os seguintes pontos de interesse: “Parque Ibirapuera” em posição (20, -30); “Catedral Metropolitana” em posição (5, 40).
2. Crie dois apartamentos: “AP14”, “Taquaral Flat 14” com 3 quartos e 2 banheiros, no primeiro andar, com sacada, posição (10, 10) e diária 150 R\$; e “AP99”, com 2 quartos e 2 banheiros, no nono andar, sem sacada, posição (-15, 7), diária 180 R\$.
3. Crie uma casa “DECOURT01”, “Pousada do Jorge” com 5 quartos, 4 banheiros e com piscina. Posição (37, 81), diária 450 R\$.
4. Crie uma mansão “GASTEI01RIM” com 1.200 mq, diária 2.000 R\$, posição (-10, -10).
5. Crie uma rede “RED1”, “Rede Simples”, posição (1, 1) diária 20 R\$.
6. Crie uma experiência “RODIZIOPIZZA”, preço base 59 R\$, preço reduzido 39 R\$, máximo 10 pessoas.
7. Adicione as residências e experiências ao sistema.
8. Imprima os detalhes de todos os elementos cadastrados no sistema
9. Crie as reservas:
 - “AP14” para 3 dias e 4 pessoas, sendo dois menores de idade.
 - “RODIZIOPIZZA” para 4 pessoas, sendo dois menores de idade.
 - “DECOURT01” para 10 dias e 2 pessoas.
 - “GASTEI01RIM” para 5 dias e 5 pessoas.
 - “RED1” para 10 dias e 5 pessoas.
10. Calcule e imprima o valor das reservas.

4 Observações

- Não é necessário desenvolver interação com o usuário por meio do terminal, além da impressão dos detalhes de residências e experiências, como pedido acima.
- Funções matemáticas podem ser encontradas na classe `java.lang.Math`.
- Tudo o que não está especificado no texto é deixado livre. Caso ache necessário, justifique eventuais escolhas com comentários no código.

5 Boas Práticas

- Nomes de classes devem começar com letra maiúscula;
- Nomes de variáveis e métodos devem começar com letra minúscula;
- Nomes de classes devem ser substantivos;
- Nomes de métodos devem ser verbos ou começar com verbo;
- Nomes compostos por mais de uma palavra devem ser escritos na forma *CamelCase*. Isto é, com a primeira letra de cada palavra maiúscula. Por exemplo: “get something” deve ser escrito como `getSomething` (método), e uma classe que representa *something* poderia ser declarada como `Something`.