Universidade Estadual de Maringá

Departamento de Informática

Disciplina: Programação orientada a objetos

Curso: Informática

Professora: Juliana Keiko Yamaguchi

Trabalho prático – Parte 01

Instruções

1. O trabalho deve ser implementado utilizando a linguagem Java.
2. Poderá ser desenvolvido individualmente ou em equipe de, **no máximo**, duas pessoas.
3. Devem ser entregues os seguintes artefatos compactados em um arquivo:
   3.1. Código fonte do programa;
   3.2. Bibliotecas de terceiros (Jar) para o funcionamento correto do programa, se forem utilizadas.
   3.3. Documento que descreve as classes definidas no trabalho e a forma de funcionamento do sistema – ver Apêndice 1.
4. O trabalho deve ser entregue via Moodle. Haverá um link de entrega no sistema para fazer o upload do arquivo. O trabalho deverá ser entregue até as 23:55hs do dia **17/05/19**. Coloque seu nome e R.A. como nome do arquivo compactado. Exemplo: Nome123456.zip (sem espaço). Se o trabalho for feito em dupla, separe os nomes e respectivos R.A.'s por um "_" (underline). Exemplo: FulanoTal12345_SicranoOutro67890.zip (sem espaço);
5. O trabalho deve ser apresentado a partir da data de entrega, em ordem alfabética constante na chamada.
   5.1. Embora o trabalho possa ser feito em dupla, isto não garante mesma nota para os integrantes da equipe.
   5.2. A apresentação deve abranger:
      a) as decisões de projeto do sistema;
      b) implementação das classes de projeto.

Objetivo do trabalho

Aplicar os conceitos de orientação a objetos na programação de um sistema de software.

Descrição do Trabalho

O trabalho consiste na implementação do jogo Detetive, com regras simplificadas. Detetive é um jogo de adivinhação cujo objetivo principal é descobrir a solução de um crime, isto é, identificar o autor do crime, qual a arma utilizada e o local onde aconteceu.

O jogo apresenta um conjunto de suspeitos, um conjunto de armas e um conjunto de locais. No início do jogo, o crime é "cometido", ou seja, são sorteados um suspeito, uma arma e um local a partir dos objetos existentes no jogo.

Durante a implementação do trabalho, pode ser considerado somente um jogador, que tentará dar os palpites para resolver o crime. O jogo faz as verificações sobre os palpites conforme as entradas recebidas e retorna quais palpites ele acertou e errou. Quando o jogador decide que pode resolver o crime, ele deve fazer uma acusação de um suspeito, de uma arma e de um local. O jogo verifica a

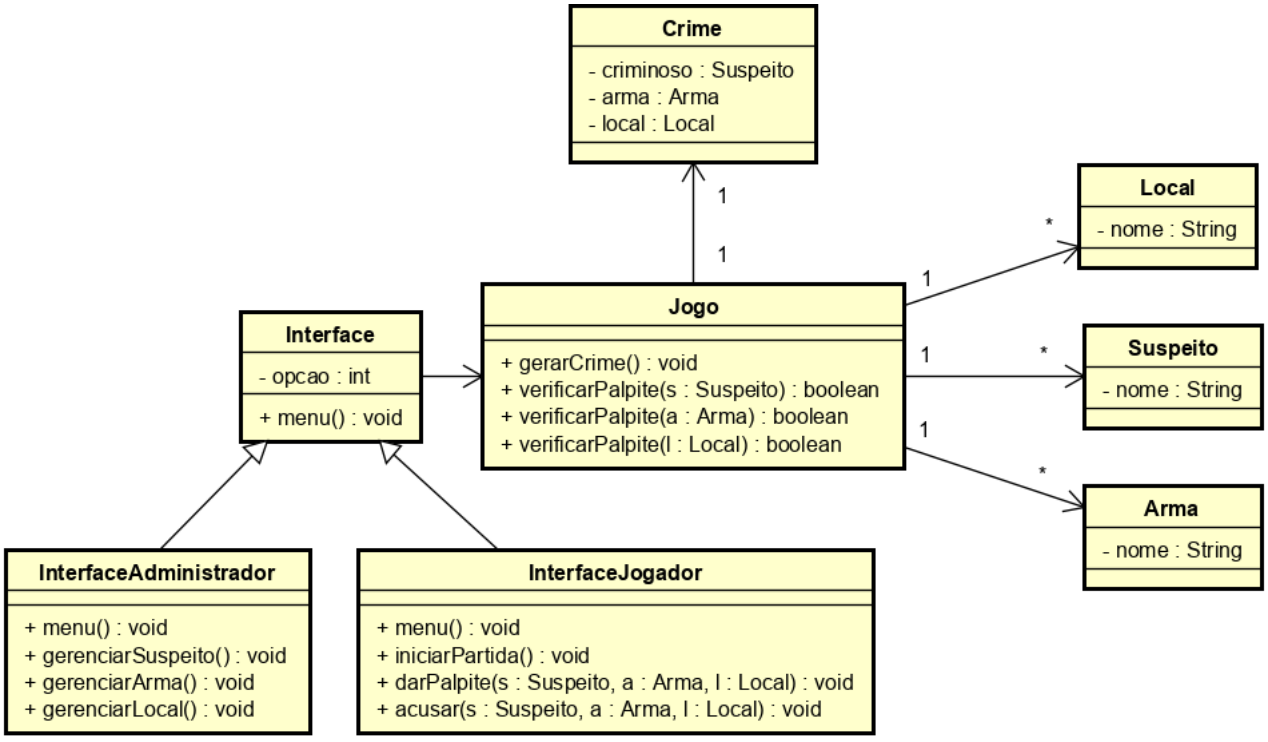
acusação e, se todos as entradas estiverem corretas, o jogador vence, caso contrário, ele perde.

O desenvolvimento do trabalho contempla duas etapas:

1. Preparação do jogo;
2. Implementação das regras do jogo.

A preparação do jogo consiste em inserir dados no jogo para que posteriormente uma partida possa ser jogada, o que compreende a realização do cadastro dos conjuntos de suspeitos, de armas e de locais. Todos os objetos serão armazenados em memória. A implementação das regras do jogo consiste em gerar um crime a partir dos objetos cadastrados e realizar a interação com o usuário (jogador).

O diagrama de classes ilustrado a seguir apresenta uma sugestão de classes que podem ser implementados no projeto.



A seguir uma explicação sobre as duas etapas de desenvolvimento.

Etapa 1: Preparação do jogo

O usuário, no papel de administrador ou planejador do jogo, realiza o gerenciamento dos suspeitos, das armas e dos locais. O gerenciamento consiste das operações de criar, recuperar, alterar e excluir objetos do jogo. O usuário pode decidir quantos objetos de cada tipo (Suspeito, Arma e Local) devem existir no jogo (a quantidade pode ser baseada nas regras do jogo real original).

Etapa 2: Implementação das regras

Uma vez que o jogo possui todos os objetos cadastrados, o usuário, no papel de jogador, poderá iniciar a partida. Quando uma partida do jogo começa, o método `iniciarPartida()` é chamado,

significando que deve ser invocado o método **`gerarCrime()`** de Jogo, que sorteará um objeto de cada lista (Local, Suspeito e Arma) para compor o crime que deve ser solucionado.

Dado que existe um objeto Crime, o usuário deverá fornecer os palpites para o suspeito, para a arma e para o local, por meio do método **`darPalpites()`**. Esses dados serão verificados pelo método **`verificarPalpite()`** da classe Jogo, que retorna verdadeiro, se o palpite estiver correto e falso, caso contrário. Enquanto a partida estiver em andamento, o jogador pode dar quantos palpites quiser. Quando o jogador achar que pode resolver o crime, ele deverá formalizar uma acusação, chamando o método **`acusar()`**, fornecendo o nome do suspeito, a arma e o local. O jogo deve realizar as verificações sobre essas entradas e, se todos os palpites forem verdadeiros, o jogador vence, caso algum palpite seja falso, ele perde a partida.

Pontos importantes a serem considerados durante a realização do trabalho:

- Originalidade da implementação das classes concretas e classes de projeto.
- Implementação das regras de forma correta.
- Implementação dos conceitos de Programação Orientada a Objetos
  - Encapsulamento
  - Herança
  - Polimorfismo
  - Implemente o trabalho o mais orientado a objetos possível.
- Organização das classes em pacotes (seguindo a convenção de Java)
- Documentação do sistema
- Usabilidade do sistema:
  - Clareza na execução das jogadas: qual o estado do jogo antes e após cada jogada, o estados dos objetos, opções de jogadas intuitivas;
- Organização dos menus

Apêndice 1

Documentação do sistema

Além do código da implementação e apresentação do mesmo, deve ser entregue um documento especificando:

1. **Passo-a-passo** para a compilação e execução do programa.
2. **Funcionamento geral** do programa
   a) Descreva quais funções o programa executa.
   b) Como as entradas devem ser fornecidas para que as funções sejam executadas.
   c) Qual o processamento realizado a partir das entradas
   d) Qual(is) a(s) saída(s) esperada(s).
3. **Conceitos de Orientação a Objetos**

   Escolha classes utilizadas no projeto para exemplificar como foram implementados os seguintes conceitos de orientação a objetos:
   - Encapsulamento
   - Herança (Se houver)
   - Polimorfismo
4. **Decisões de projeto**
   a) Explicação do funcionamento geral do sistema:
   - Descreva como funcionam os principais métodos (e indique a qual classe pertence) que participam da execução do programa, considerando o fluxo principal de execução.
   - Apresente trechos de código para melhorar a compreensão dos métodos descritos.

   b) Definição das classes

   Explique o funcionamento das classes desenvolvidas quanto:
   - Classes de negócio: explique porque a importância de cada classe.
   - Classes controladoras implementadas
   - Classes de interação com o usuário (menus de opções)
   - Tratamento de exceções (se houver)
   - Organização das classes em pacotes (coesão e acoplamento)

   c) Diagrama de classes de projeto

   Apresente o diagrama de classes de projeto do sistema.