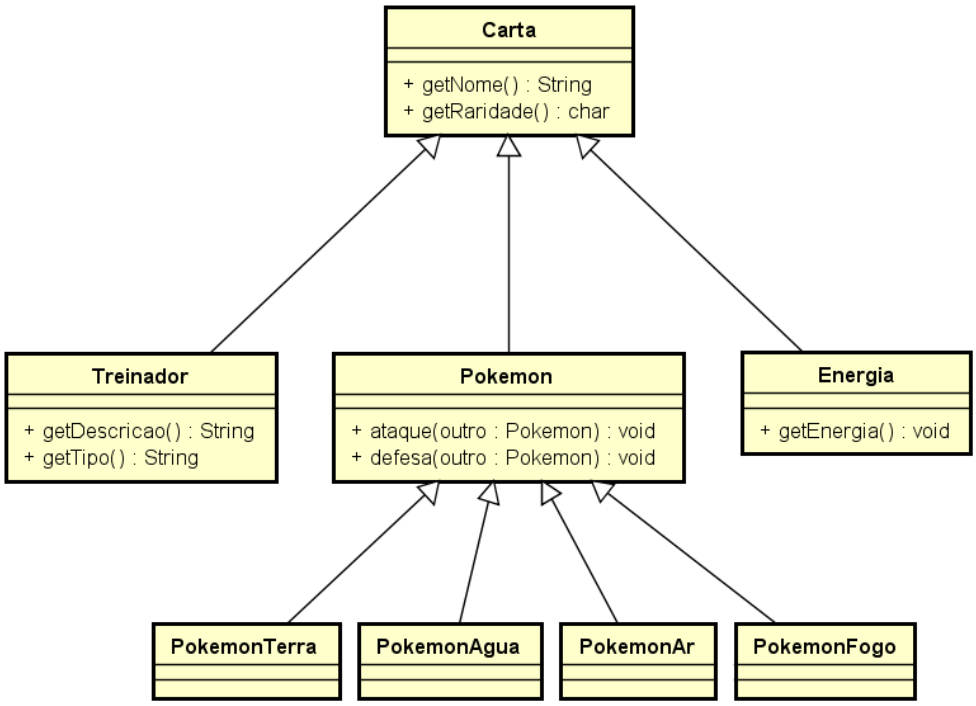
Programação Orientada a Objetos
Trabalho Final – 2022/2

Magic: the Gathering, **M:TG**, **MTG** ou simplesmente **Magic**, é um jogo de cartas colecionáveis (*TCG*, *Trading Card Game*) criado por Richard Garfield, no qual os jogadores utilizam um baralho de cartas construído de acordo com o seu modo individual de jogo para tentar vencer o baralho adversário. Em 2003, na comemoração do aniversário de 10 anos de lançamento do Magic, a revista "*Games*" selecionou-o para o seu Hall da Fama[1]. Criado em 1993, *Magic* foi o primeiro TCG produzido e continuado até hoje, quando conta com aproximadamente 12 milhões de jogadores ao redor do mundo. (https://pt.wikipedia.org/wiki/Magic:_The_Gathering)

Objetivos: o objetivo deste trabalho é demonstrar domínio dos conceitos básicos de programação orientada a objetos em especial herança e polimorfismo, coleções, conhecimentos básicos de organização de código e criação de interface com o usuário rica usando JavaFX.

Tema do trabalho: o trabalho deverá implementar um jogo ao estilo de "Jogo de Cartas Colecionável" como por exemplo "Pokemon" ou "Magic". O tema e as regras do jogo podem ser definidos livremente desde que sejam respeitadas as seguintes restrições:

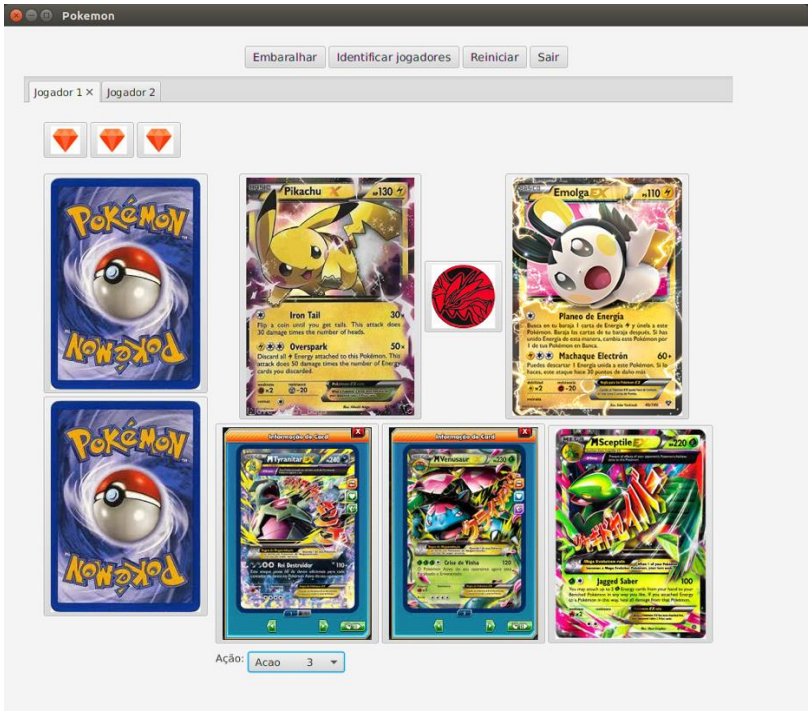
I. Restrições de implementação:
   - Devem existir diferentes tipos de cartas que justificadamente possam ser representadas por uma hierarquia de herança. O diagrama abaixo é uma sugestão inicial de como poderia ser estruturada a hierarquia de classes.



- A interface com o usuário deve ser construída usando JavaFX.
- As classes devem ser coesas e ter suas responsabilidades claramente definidas. O conjunto de classes que modelam a interface com o usuário deve-se limitar a entrada e saída de dados. É obrigatório que toda a lógica de funcionamento do jogo esteja implementada em um conjunto de classes independente da interface com o usuário e que estas classes possam ser testadas de forma independente.

II. Restrições no jogo (adequar ao jogo escolhido pelo seu grupo):
- Devem existir pelo menos 5 tipos de cartas diferentes
- O objetivo do jogo deve ser eliminar todas as cartas do adversário
- A cada ataque de um oponente a carta do adversário perde “vidas”, sendo eliminada da mesa quando não restarem mais vidas.
- A cada jogada um jogador pode colocar ou retirar uma carta da mesa ou acionar algum efeito da carta.
- Na vez de cada jogador o “computador” deve anunciar de quem é a vez e só depois de confirmada a presença do jogador mostrar a “sua tela” (ou mão).
- O “computador” deve ser capaz de detectar o final do jogo.

III. Requisitos da interface com o usuário:
- Devem ser usadas imagens na representação das cartas que são apresentadas em cada jogada;
- Os componentes da interface devem permitir reiniciar o jogo, colocar os nomes dos dois jogadores, verificar o número de “vidas”, embaralhar e sacar as cartas (por exemplo), além de outras funcionalidades necessárias para o jogo;
- A imagem abaixo apresenta um exemplo de como a interface deve parecer:



Entrega e Apresentação:

- Os trabalhos podem ser realizados individualmente ou em grupos de até três alunos.
- A implementação deve seguir as orientações dadas em aula quanto a convenções Java para nomes de identificadores e estrutura das classes.
- Não serão aceitos trabalhos com erros de compilação. Programas que não compilarem corretamente terão nota ZERO.
- Uma versão em formato "zip" (fazer download neste formato do projeto no github) deve ser entregue via sistema Moodle contendo a implementação feita (todas as pastas e os arquivos .java). Este arquivo deve ter o nome e sobrenome do(s) aluno(s), da seguinte forma: nome_ultimosobrenome-nome_ultimosobrenome.zip. Deve ser feito o upload deste arquivo na tarefa indicada para isto no Moodle até a data e horário especificados. Alternativamente o professor da sua turma pode solicitar apenas o link para o repositório (privado) do github, neste caso ele deve ter sido convidado como colaborador do projeto para poder realizar o download.
- Trabalhos entregues, mas não apresentados, terão sua nota anulada pelo professor. Durante a apresentação será avaliado o domínio da resolução do problema, podendo inclusive ser possível invalidar o trabalho quando constatada a falta de conhecimento sobre o código implementado.
- A cópia parcial ou completa do trabalho terá como consequência a atribuição de nota ZERO ao trabalho dos alunos envolvidos. O sistema Moss (https://theory.stanford.edu/~aiken/moss/) será utilizado para detecção de cópias, logo é importante que até a data de entrega do TF seu trabalho seja mantido em um repositório privativo
- O Fórum disponibilizado pode ser utilizado para discutir qualquer tema relacionado ao trabalho: sugestões, críticas, estratégias. Mas não é permitido postar código.