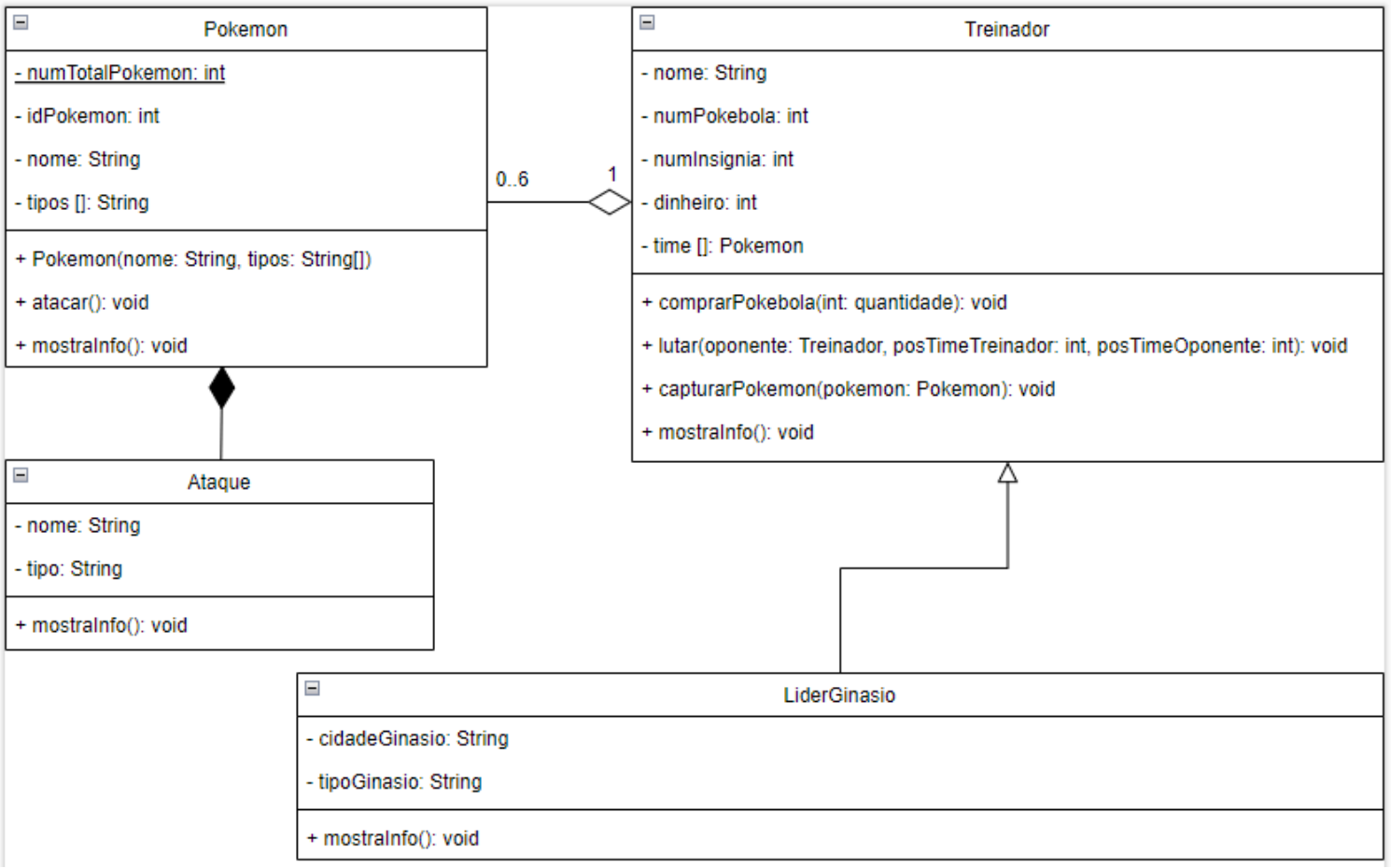
Pokémon é um dos animes e jogos mais icônicos do mundo, com uma vasta base de fãs e um universo complexo. Foi pedido a você que desenvolvesse a seguinte UML simplificada, para explicar o funcionamento dos jogos:



NOTAS IMPORTANTES:

- Se necessário, crie apenas os getters e setters que for utilizar. A criação de um getter ou setter desnecessário irá acarretar na perda de pontos;
- Siga a UML, não segui-la irá acarretar na perda de pontos.

Classes:

- **Pokemon:**
  - Pokemons podem ter um ou dois tipos, portanto o atributo “tipos” deve ser um array com limite de 2 tipos, exemplo: Fogo, Água, Planta, Elétrico, etc;
  - Dica: O atributo do Ataque para a composição deve ser um atributo público
  - O construtor da classe deve incrementar o atributo estático “numTotalPokemon” para cada objeto criado e usá-lo para preencher o atributo “idPokemon”;
  - atacar() deve apenas fazer um print indicando o ataque (exemplo: “pokemon X está atacando”) e em seguida mostrar todas as informações do ataque do pokemon;
  - mostraInfo() deve fazer um print de todos os atributos de um pokemon e de seu ataque.
- **Ataque:**
  - mostraInfo() deve fazer um print de todas as informações do ataque
- **Treinador:**

    - Treinadores podem ter um time de até 6 pokemon, portanto o atributo “time” deve ser um array com limite de 6 pokemon;
    - comprarPokebola() deve receber como parâmetro uma quantidade de pokebolas que o treinador deseja comprar e, considerando que cada pokebola custa 100 dinheiros, o método deve calcular o preço total e verificar se o treinador possui essa quantidade de dinheiro, se sim, o dinheiro do treinador é subtraído, a quantidade de pokebolas do treinador é incrementada e é feito um print (exemplo: “treinador X agora tem X dinheiro e X pokebolas”), se não, deve ser feito o print de uma mensagem (exemplo: “treinador X não tem dinheiro suficiente”)
    - lutar() deve receber como parâmetro o treinador oponente e os valores “posTimeTreinador” e “posTimeOponente”, que são as posições no array “time” do treinador que está chamando o método e do treinador oponente, respectivamente. O método deve apenas chamar o método “atacar()” do pokemon que está naquela posição do array. Após a luta, caso o treinador oponente seja uma instância de **LiderGinasio**, deve-se incrementar 1 no atributo “numInsignia” do treinador que chamou o método.
    - capturarPokemon(): deve inserir o pokemon que recebe como parâmetro no array “time”;
    - mostraInfo() deve fazer um print de todas as informações do treinador e dos pokemon de seu time.
- **LiderGinasio**:
    - mostraInfo() deve fazer um print de todas as informações do líder de ginásio
    - **Obs.:** Lembre-se de utilizar “super” no construtor e em mostraInfo!

No arquivo Main, crie as instâncias de todas as classes possíveis e chame todos os métodos que não foram chamados dentro de algum método