Professor: José Cleyton
Turma: ADS 2AN
Disciplina: POO

Definição do Projeto: Sistema de Adoção de Animais ■

Contexto

ONGs e protetores independentes precisam organizar cadastros de animais e o fluxo de adoção. O sistema vai registrar animais, adotantes e processos de adoção, aplicando regras simples — porém exigindo domínio de POO em Java.

Objetivo

Criar um sistema em Java (CLI obrigatório; GUI opcional/extra), com persistência de dados obrigatória, que:
- Gerencie animais disponíveis para adoção.
- Gerencie adotantes.
- Registre adoções, validando regras de negócio.
- Utilize apenas os conceitos: classes, objetos, métodos, atributos, encapsulamento, sobrecarga, herança, sobrescrita, interface, classe abstrata e exceptions personalizadas.

Requisitos Funcionais

1. CRUD de Animais (cadastrar, listar, atualizar, remover).
2. CRUD de Adotantes (cadastrar, listar, atualizar, remover).
3. Registro de Adoção (associar animal disponível a adotante, atualizar status, contar adoção).
4. Listagem de Adoções com filtros simples (por adotante e/ou por período).

Regras de Negócio

- Um adotante pode ter no máximo 3 animais adotados simultaneamente.
- Um animal só pode ser adotado se o status for DISPONIVEL.
- Violações devem lançar exceptions personalizadas (e serem tratadas no fluxo da aplicação).

Persistência (Obrigatória)

Os dados não podem ficar apenas em memória. Escolha uma das abordagens e documente:
- Arquivos CSV (ou TXT delimitado) por entidade (Animais, Adotantes, Adoções).
- JSON por entidade.
- Serialização Java.
- Banco de Dados SQLite via JDBC.

Requisitos mínimos:
- Operações CRUD devem ler/escrever no armazenamento escolhido.
- Ao iniciar, carregar registros existentes.
- Ao encerrar (ou a cada operação), salvar registros.

- Isolar acesso em módulo de repositório (ex.: Repositorio).

Interface de Uso

Obrigatório: CLI (menu em console).
Extra (Opcional, valendo bônus): GUI em JavaFX ou Swing.
- Telas básicas para os mesmos fluxos (listar, criar/editar, excluir, adotar).
- A GUI pode compensar até 20% dos pontos perdidos.
- A GUI deve usar o mesmo serviço/repositório (sem duplicar lógica).

Estrutura de POO (Conceitos exigidos)

- Classe abstrata Animal (com emitirSom).
- Subclasses: Cachorro, Gato (sobrescrevem emitirSom).
- Interface CuidadosEspeciais (vacinar, vermifugar).
- Classe Adotante (encapsulamento e sobrecarga).
- Classe Adocao (liga Animal + Adotante, sobrescreve toString).
- Exceptions personalizadas: LimiteAdocoesException, AnimalIndisponivelException.
- Sobrecarga: construtores e métodos de busca.
- Sobrescrita: emitirSom e toString.

Fluxo de Adoção

1. Usuário escolhe “Realizar Adoção”.
2. Sistema busca Adotante e Animal (persistência).
3. Serviço valida:
- quantidade de adoções < 3? senão lança LimiteAdocoesException.
- animal disponível? senão lança AnimalIndisponivelException.
4. Se ok: cria Adocao, marca animal como ADOTADO, incrementa contador do adotante e persiste alterações.

Critérios de Avaliação

- Modelo OO e Encapsulamento (20 pts).
- Herança, Classe Abstrata, Interface (25 pts).
- Sobrecarga e Sobrescrita (15 pts).
- Regras de Negócio + Exceptions (20 pts).
- Persistência funcionando (CRUD real) (20 pts).
- Bônus GUI: até +20% da nota obtida (sem ultrapassar 100).

Entregáveis

- Código-fonte organizado em pacotes.
- README com persistência escolhida e instruções de execução.
- Script/semente de dados (ex.: CSV inicial ou método seed()).