

PROPOSTA 1 — Sistema de Gerenciamento de Produtos Naturais

“Catálogo Verde” — Controle de produtos orgânicos e artesanais

Domínio

- **Classe Base:** Produto
- **Subclasses:**
 - ProdutoAlimenticio
 - ProdutoArtesanal

Funcionalidades

- Cadastrar, buscar, remover e listar produtos.
- Preços, categorias, níveis de estoque, produtor local.
- Regra de negócio: preço deve ser positivo e estoque ≥ 0 .
 - Lançar `NegocioException` se violar.

Árvore

- **AVL** indexada por id ou por nome.

Ordenação

- **MergeSort** para ordenar a lista por:
 - Preço,
 - Nome,
 - Categoria (via `Comparator`).

POO Avançada

- Interfaces:
 - `Repositorio<T>`
 - `Arvore<K,V>`
 - `Ordenacao<T>`
- Classe Genérica:
 - `RepositorioHash<T>` com `Function<T,String> idExtractor`.

Anotação

- `@InfoAutor` em `Produto` e em `RepositorioHash`.

PROPOSTA 2 — Sistema Acadêmico de Alunos e Cursos

“Campus Manager” — Mini gerenciador estudantil

Domínio

- **Classe Base:** Pessoa
- **Subclasses:**
 - Aluno
 - Professor

Funcionalidades

- CRUD de alunos e professores.
- Atribuir professores a cursos, média de notas, frequência.
- Regra de negócio:
 - Média deve ser entre 0 e 10, frequência entre 0 e 100.
 - Caso contrário → `NegocioException`.

Árvore

- **ABB** por nome do aluno (ou matrícula).

Ordenação

- **HeapSort** para ordenar alunos por:
 - Média,
 - Nome,
 - Frequência.

POO Avançada

- Interfaces: `Repositorio<T>`, `Ordenacao<T>`, `Arvore<K,V>`.
- Classe Genérica: `RepositorioHash<T>`.

Anotação

- `@InfoAutor` aplicada em Aluno e Pessoa.

PROPOSTA 3 — Biblioteca Digital

“BookTree” — Acervo com autores e livros

Domínio

- **Classe Base:** Publicacao
- **Subclasses:**
 - Livro
 - Revista

Funcionalidades

- Inserir, buscar, listar e remover publicações.
- Pesquisar por autor, título e ano.
- Regra de negócio:
 - Ano deve ser ≥ 1500 .
 - Título não pode ser vazio.
 - Caso contrário \rightarrow NegocioException.

Árvore

- AVL por título da publicação.

Ordenação

- MergeSort por:
 - Ano,
 - Título,
 - Autor.

POO Avançada

- Interfaces: Repositorio<T>, Ordenacao<T>, Arvore<K,V>.
- Classe Genérica: RepositorioHash<T>.
- Uso de Comparator<Publicacao> customizado.

Anotação

- Aplicada em Livro e RepositorioHash.

PROPOSTA 4 — Sistema de Agenda de Compromissos

“TimeMaster” — Gerenciador inteligente de compromissos

Domínio

- **Classe Base:** Compromisso
- **Subclasses:**
 - CompromissoPessoal
 - CompromissoProfissional

Funcionalidades

- Cadastrar, editar, buscar e remover compromissos.
- Checar conflitos de agenda.
- Regra de negócio:
 - Data não pode ser passada.
 - Não pode registrar dois compromissos no mesmo horário → `NegocioException`.

Árvore

- ABB por horário (`LocalDateTime`).

Ordenação

- HeapSort por:
 - Data,
 - Prioridade,
 - Categoria.

POO Avançada

- Interfaces + Repositório genérico.
- Classe genérica de repositório com `Function<Compromisso, String>` para ID.

Anotação

- `@InfoAutor` em `Compromisso` e `RepositorioHash`.

PROPOSTA 5 — Sistema de Cadastro de Veículos

“AutoTrack” — Mini-Gerenciador de Frota e Veículos pessoais

Domínio

- **Classe Base:** Veiculo
- **Subclasses:**
 - Carro
 - Moto

Funcionalidades

- Cadastro com marca, modelo, ano, placa e quilometragem.
- Busca por placa.
- Regra de negócio:
 - Ano \geq 1960.
 - Quilometragem não pode ser negativa.
 - Placa deve seguir padrão.
 - Se violar \rightarrow NegocioException.

Árvore

- AVL por placa (String) ou ano.

Ordenação

- MergeSort para ordenar veículos por:
 - Ano,
 - Modelo,
 - Quilometragem.

POO Avançada

- Interfaces de repositório, árvore e ordenação.
- Classe genérica com HashMap interno.
- Polimorfismo ativo ao imprimir informações de cada tipo.

Anotação

- Aplicada em Veiculo e RepositorioHash.