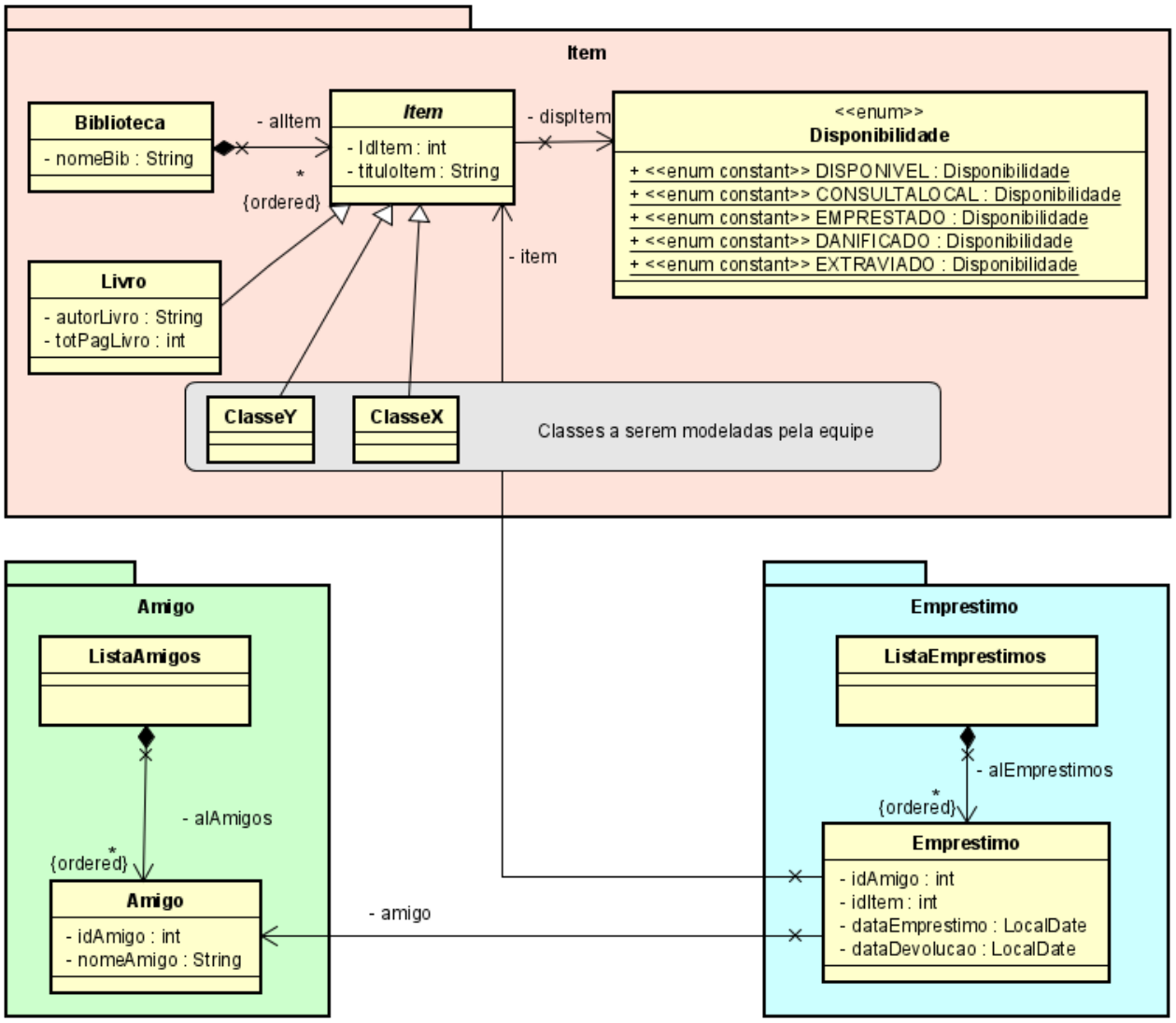
UNIVERSIDADE FEDERAL DO PARANÁ
CURSO DE ENGENHARIA ELÉTRICA
TE 353 - PROGRAMAÇÃO ORIENTADA A OBJETOS

Atividade PjBL 2: Controle de biblioteca pessoal



Contexto

Um estudante da disciplina de POO, percebendo que sua coleção de livros e de outros ítens que costuma emprestar aos seus colegas do departamento, como quadrinhos, CDs, DVDs etc., estava perdendo o controle dos seus objetos e resolveu implementar em Java um pequeno sistema de gestão para suas necessidades de acompanhamento da sua biblioteca/mediateca.

O sistema proposto terá algumas classes devidamente organizadas e relacionadas permitindo operações mínimas de controle dos itens pertencentes a sua biblioteca pessoal. O projeto (modelo de classes inicial acima) e cujo esqueleto foi apresentado em sala e está disponível no arquivo Biblioteca.zip, já contém a base das classes. **Item** é uma classe abstrata de um item da biblioteca que pode ser empretado, que possui como subclasse concreta **Livro**. As **ClasseX** e **ClasseY** são de implementação livre da equipe, podem ser outros itens (ex: CD e DVD), pode ser uma classe de item e um enum (ex: Filme e enum Gênero), pode ser uma classe abstrata e uma concreta (Ex: torna Livro abstrata e cria LivroDidático e LivroLiteratura) etc. : o importante é criar duas classes novas alinhadas como o conceito de herança.

Sessão Típica

Uma sessão típica consiste em:

1) cadastrar item: efetua o cadastro de um novo item cujo **idItem** é gerado automaticamente (baseado na quantidade de itens de **alItem**), incluindo os atributos de **Item** (título) e da subclasse específica que está sendo cadastrada (no caso de um **Livro**, o autor e o total de páginas, para as Classes C e Y, depende da modelagem da equipe);

2) cadastrar amigo: efetua o cadastro de um novo amigo cujo **idAmigo** é gerado automaticamente (baseado na quantidade de amigos de **alAmigos**), incluindo o atributo **nomeAmigo**;

3) emprestar: o sistema pergunta o código do item a ser emprestado (atributo **idItem**), depois qual o amigo (pode mostrar todos para escolher o código do amigo, ou fazer busca pelo nome para identificar e pegar código). Efetua o empréstimo na data corrente com objeto da classe **Emprestimo** incluído no **alEmprestimos**. Ou seja, **idAmigo** empresta item **idItem** em **dataEmprestimo** colocando **dispItem** em **Item** para **Disponibilidade.Emprestado**;

4) devolver: o sistema pergunta o código do item a ser emprestado (atributo **idItem**), e efetua a devolução deste Item que estava fora, ou seja aquele objeto cujo **idItem** indica que está emprestado. Ao devolver um item, não remover o objeto de **alEmprestimos** (que portanto conterá empréstimos já concluídos – um histórico, e os empréstimos em andamento);

5) listar empréstimos atuais: o sistema mostra todos os items emprestados: seu título, quem pegou e em que data pegou emprestado (para isso percorre o alEmprestados identificando quem está fora, ou seja, **dispItem** igual **Disponibilidade.Emprestado**);

6) listar histórico de empréstimos: o sistema pergunta o código de um item (**idItem**) e lista todas as movimentações daquele item, ou seja, todos os empréstimos já feitos (quem, data empréstimo e data devolução) e o empréstimo atual (se existir) para aquele item em particular;

7) listar biblioteca: o sistema mostra todos os items ordenados pelo seu título em ordem alfabética ascendente, mostrando o título do item e sua situação (**dispItem**);

8) alterar estado: o sistema pergunta o código de um item (**idItem**) e permite mudar sua disponibilidade para Consulta Local (se não estive Emprestado), ou Danificado ou Extraviado (independente de estar anteriormente Disponível ou Emprestado).

Requisitos

A organização do menu, linear como sugerido acima, ou em níveis, bem como a sequência/nomenclatura ficam livres para a equipe decidir, o importante é permitir que sejam possíveis as operações listadas, ou seja, as funcionalidades do sistema descritas na sessão típic acima.

Para facilitar a apresentação prática do código funcionando, deixar alguns objetos cadastrados prontos nas listas, com criação dos objetos *harcoded*, logo no início da *main*.

A tabela a seguir apresenta as características que serão avaliadas:

Uso adequado das funções *static* na main com reuso quando módulos compatíveis	Funcionalidade de empréstimo de items baseado no código do item
Construção das classes X e Y adequadas ao problema em questão	Funcionalidades de devolução de items baseado no código do item
Encapsulamento e métodos das classes empregados adequadamente	Alteração de Estado conforme regras especificada e uso do enum
Funcionalidades de cadastro de itens e cadastro de amigos	Empregar tags JavaDoc adquedamente para documentação das classes de domínio da aplicação
Funcionalidades de listagem de empréstimos, do histórico de um item e do total de itens	Aplicação de persistência de objetos (salvar e ler as classes do sistema em disco)