# ELE078 - PROGRAMAÇÃO ORIENTADA A OBJETOS

## Trabalho Prático

## valor: 25 Pontos

### I – As seguintes regras devem ser observadas na confecção do trabalho:

1) **O trabalho é individual**. É permitido discutir os problemas e estratégias de solução com outros alunos, mas quando se tratar de escrever ou implementar computacionalmente as soluções, isto deve ser feito individualmente. Utilizar o trabalho dos outros, como se fosse seu, é plágio. Desonestidade acadêmica será punida com severidade.

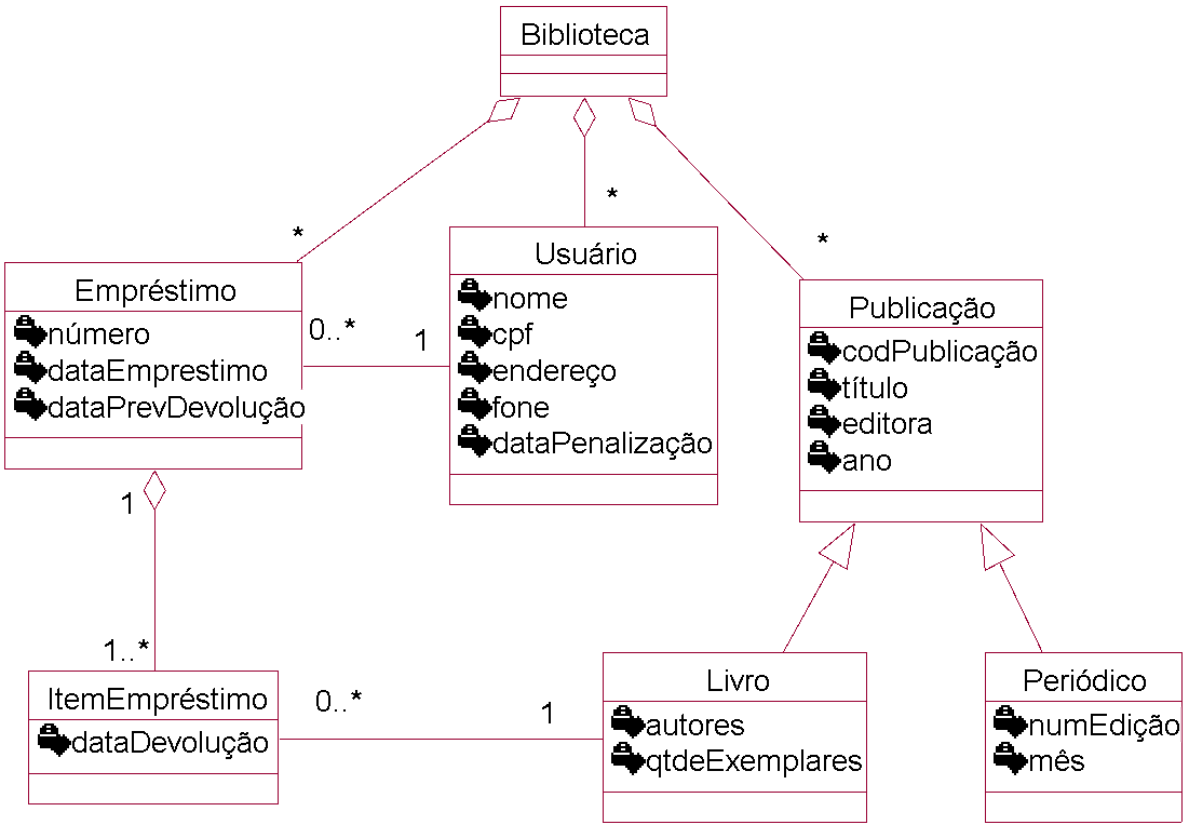
2) **Forma de entrega:** O trabalho deve ser entregue em formato digital por meio do *Moodle*. Utilizar a opção *"Link para Envio do TP".* Anexe um único arquivo .zip (ou .rar) contendo todos os arquivos do trabalho (códigos-fonte, executáveis, documentação, etc.). O nome do arquivo .zip (ou .rar) a ser enviado deve ser o nome completo do aluno. Exemplo:

***Cristiano_Leite_Castro.zip*.**

3) O link para envio do TP no Moodle terá uma tolerância de 24 horas para recebimentos dos trabalhos. Após esse período, os trabalhos não serão recebidos.

### II - Sistema de Controle de Biblioteca

Implemente um conjunto de classes para um sistema de controle de uma Biblioteca. A biblioteca necessita manter informações sobre as publicações (livros e periódicos) disponíveis, usuários e empréstimos. Para isso, foi feito o seguinte projeto em UML (Diagrama de Classes) e você deve implementá-lo usando C++:



**Detalhamento do Sistema:**

- Classe Usuário:
  - Atributos: nome (string), cpf (string), endereco (string), fone (string) e dataPenalização (tipo Date, a ser definido).
  - Deve existir um método construtor que inicializa todos os atributos através de argumentos com exceção do atributo dataPenalização. Este deve ser inicializado com a data corrente, sinalizando que o Usuário já poderá tomar um novo empréstimo a partir de seu cadastro. O significado deste atributo será explicado mais à frente.

A Biblioteca controla livros e periódicos (revistas e outras publicações editadas periodicamente). Porém, somente os livros podem ser emprestados. Os periódicos são de acesso exclusivo dentro da biblioteca, não podem ser levados pra casa. Os livros e periódicos são chamados genericamente de Publicação.

- Classe Publicação:
  - Atributos: codPublicacao (int), titulo (string), editora (string) e ano (int);
- Classe Livro:
  - Atributos: autores (string, nome de todos os autores) e qtdeExemplares (int);
  - Deve existir um método construtor que inicializa todos os atributos através de parâmetros;
  - Deve existir um método construtor que inicializa os atributos codPublicacao, titulo, editora, ano e autores através de argumentos e o atributo qtdeExemplares com zero;
  - Deve existir métodos para incrementar e decrementar a quantidade de exemplares de uma unidade; Não permitir o decremento se a quantidade for igual a zero.
- Classe Periodico:
  - Atributos: numEdição (int) e mês (string);
  - Deve existir um método construtor que inicializa todos os atributos através de parâmetros;

A biblioteca também possui um controle de penalização por atraso na devolução de livros. Se a devolução for efetuada após a data prevista de entrega, o usuário deve ser penalizado em 3 dias sem tomar um novo empréstimo para cada dia de atraso. Deve ser efetuado o cálculo do número de dias de penalização e a data a partir da qual o usuário poderá tomar um novo empréstimo deve ser armazenada no atributo "dataPenalização" do objeto "Usuário". Ao tentar efetuar um novo empréstimo esta data deve ser verificada e não deve ser permitido que um usuário com penalização efetue empréstimo;

- Classe Emprestimo:
  - Atributos: numero (int), dataEmprestimo (tipo Date, a ser definido), dataPrevDevolucao (data prevista para devolução), usuário (Usuario), itens (lista de ItemEmprestimo) e proximoNumero (int). O atributo proximoNumero deve ser **estático** e será usado para armazenar o próximo número de empréstimo. A lista de itens de empréstimos pode ser implementada através da classe **Vector**;
  - deve existir um método construtor que inicializa os atributos com os seguintes valores: numero (próximo número sequencial), dataEmprestimo (data corrente do sistema), dataPrevDevolucao e usuário (passados como argumentos);
  - deve existir um método para adicionar um item (livro) ao empréstimo. Argumento: o livro. A quantidade de exemplares do livro deve ser decrementada de uma unidade (verificar se a quantidade é suficiente);

- deve existir um método para excluir um item (livro) do empréstimo. Argumento: o livro a ser excluído. A quantidade de exemplares do livro deve ser incrementada de uma unidade;
- deve existir um método para devolver um item (livro) do empréstimo. Argumento: o livro a ser devolvido. A quantidade de exemplares do livro deve ser incrementada de uma unidade. A data de devolução do item deve ser atualizada pela data do sistema;
- deve existir um método para devolver todos os livros do empréstimo. A quantidade de exemplares de todos os itens (livros) do empréstimo deve ser incrementada de uma unidade. A data de devolução de todos os itens deve ser atualizada pela data do sistema;

- Classe ItemEmprestimo:
  - Atributos: dataDevolucao (tipo Date) e livro (Livro);
  - Deve existir um método construtor que inicializa os atributos da seguinte forma: livro (passado como argumento) e dataDevolucao com valor nulo;
- Classe Biblioteca:
  - Atributos: usuários (lista de Usuário), livros (lista de Publicacao) e emprestimos (lista de Emprestimo). As listas podem ser implementadas através da classe Vector.
  - Deve existir um método construtor que inicializa os atributos listas como vazias;
  - Deve existir um método para inserir um novo usuário na lista de usuários. Argumento: o usuário;
  - Deve existir um método para inserir uma nova Publicacao na lista de Publicacoes. Argumento: a publicacao;
  - Deve existir um método para inserir um novo empréstimo na lista de empréstimos. Argumento: o empréstimo;
  - Deve existir um método para inserir um novo item de empréstimo para um empréstimo. Argumento: o empréstimo e o item de empréstimo;
  - Deve existir um método para excluir um usuário da lista de usuários. Argumento: o usuário. O usuário não pode ser excluído se existir algum empréstimo para ele;
  - Deve existir um método para excluir uma publicacao da lista de publicacoes. Argumento: a publicacao. Se a publicação for um livro, este não pode ser excluído se existir algum empréstimo para ele;
  - Deve existir um método para excluir um empréstimo da lista de empréstimos. Argumento: o empréstimo;
  - Deve existir um método para excluir um item de empréstimo de um empréstimo. Argumentos: o empréstimo e o item de empréstimo;
  - Deve existir um método para devolver um item (livro) de um empréstimo. Argumento: o empréstimo e o livro a ser devolvido.
  - Deve existir um método para devolver todos os livros de um empréstimo. Argumento: o empréstimo.
  - Deve existir um método para pesquisar publicações (livro ou periódico) por título. Argumento: uma string especificando parte do título da publicação. O método deve retornar uma lista com todas as publicações que contêm a string no título;
  - Deve existir um método para pesquisar livros por autor. Argumento: uma string especificando parte do nome do autor. O método deve retornar uma lista com todos os livros que contêm a string no nome dos autores;
  - Deve existir um método para obter a lista de usuários;
  - Deve existir um método para obter a lista de publicacoes;

- Deve existir um método para obter a lista de empréstimos;
- Deve existir um método para gravar os dados em arquivo;
- Deve existir um método para ler os dados armazenados em arquivo.

- Classe Interface:
  - Método para apresentar um menu (interface de caracteres) com opções para: cadastrar um novo usuário, cadastrar um novo livro, cadastrar um novo periodico, cadastrar um novo empréstimo, inserir um novo item de empréstimo, excluir um usuário, excluir um livro, excluir um periodico, excluir um empréstimo, excluir um item de empréstimo, devolver todos os livros do empréstimo, devolver um livro do empréstimo, pesquisar publicações por título, pesquisar livros por autor, listar todos os usuários, listar todos as prublicações (livros e periódicos), listar todos os empréstimos e seus itens e sair do programa;
  - Método para cadastrar um novo usuário. Deve permitir que o usuário entre com os dados do usuário a ser inserido;
  - Método para cadastrar um novo livro. Deve permitir que o usuário entre com os dados do livro a ser inserido;
  - Método para cadastrar um novo periodico. Deve permitir que o usuário entre com os dados do periodico a ser inserido;
  - Método para cadastrar um novo empréstimo. Deve permitir que o usuário entre com os dados do empréstimo a ser inserido e de cada um de seus itens.
  - Método para inserir um novo item de empréstimo a um empréstimo já existente. Deve permitir que o usuário escolha o empréstimo e insira os dados do novo item;
  - Método para excluir um usuário. Deve permitir que o usuário escolha o usuário a ser excluído;
  - Método para excluir um livro. Deve permitir que o usuário escolha o livro a ser excluído;
  - Método para excluir um periodico. Deve permitir que o usuário escolha o periodico a ser excluído;
  - Método para excluir um empréstimo. Deve permitir que o usuário escolha o empréstimo a ser excluído;
  - Método para excluir um item de empréstimo de um empréstimo já existente. Deve permitir que o usuário escolha o empréstimo e o item a ser excluído;
  - Método para devolver todos os livros de um empréstimo. Deve permitir que o usuário escolha o empréstimo a ser devolvido;
  - Método para devolver um livro de um empréstimo. Deve permitir que o usuário escolha o empréstimo e o livro a ser devolvido;
  - Método para pesquisar publicações (livros e periódicos) por título. Deve permitir que o usuário entre com parte do título da publicação a ser pesquisada. Deve listar os dados de todas as publicações encontradas.
  - Método para pesquisar livros por autor. Deve permitir que o usuário entre com parte do nome do autor do livro a ser pesquisado. Deve listar os dados de todos os livros encontrados.
  - Método para listar todas as publicações. Deve listar os dados de todas os livros e periódicos;
  - Método para listar todos os empréstimos. Deve listar os dados de todos os empréstimos com seus respectivos itens;
  - Método *main* para apresentar o menu principal.

- **Observações a serem seguidas:**

- Observe a separação entre as classes de negócio (Emprestimo, ItemEmprestimo, Usuário, Publicações, Livro e Periodico) e a classe de interface. As classes de negócio não devem possuir nada que crie uma dependência da interface. Desta forma, se posteriormente você decidir mudar a interface, por exemplo, usar uma interface gráfica, as classes de negócio não precisam ser alteradas. A classe Biblioteca atua como uma classe gerente, fazendo a interligação das classes de negócio com a classe de interface. Em um projeto mais profissional, provavelmente existiriam várias interfaces e classes gerente. A mesma ideia deveria ser usada para separar as classes de negócio e as classes de persistência. Por simplicidade, nesse trabalho o armazenamento dos dados será feito em listas (containers do tipo Vector) e as próprias classes de negócio e de gerência sabem como fazer isso, o que significa que se decidirmos passar a usar um banco de dados teremos que alterar essas classes. Em um projeto mais profissional, deveríamos criar uma camada de persistência, separando as classes de negócio da estrutura de armazenamento.

- Todas as listas (lista de empréstimos, lista de usuários e lista de publicações da classe Biblioteca e lista de itens de empréstimos da classe Empréstimo) podem ser implementadas usando a classe Vector;

- Todas as situações de exceção devem ser tratadas. Por exemplo, não é permitido emprestar um periódico, não é permitido emprestar livro com quantidade de exemplares igual a zero, não é permitido emprestar livros para um usuário com penalização, não é permitido excluir um livro que já tenha sido emprestado, não é permitido excluir um usuário que já tenha feito algum empréstimo etc. Crie classes de exceção e faça todo o tratamento necessário. Impressão de mensagens de erros devem ser apresentadas somente na classe de interface;

- **Use polimorfismo sempre que possível**. Um bom teste para verificar o uso correto do polimorfismo em seu projeto é imaginar a inclusão de um novo tipo de publicação ao sistema. Por exemplo, se for solicitada uma alteração no projeto para incluir o controle de jornais, as alterações no código deveriam ser a criação da nova classe e a adaptação na classe de interface. As outras classes do sistema não deveriam sofrer nenhuma alteração;

- Leitura de dados do usuário e impressão na tela devem ser feitos somente nas classes de interface;

- Gere documentação para as classes que compõem o trabalho.