Universidade Federal de Lavras
Departamento de Ciência da Computação
GCC – 110 – Programação Orientada a Objetos

Trabalho Prático 1:

I – Observações:

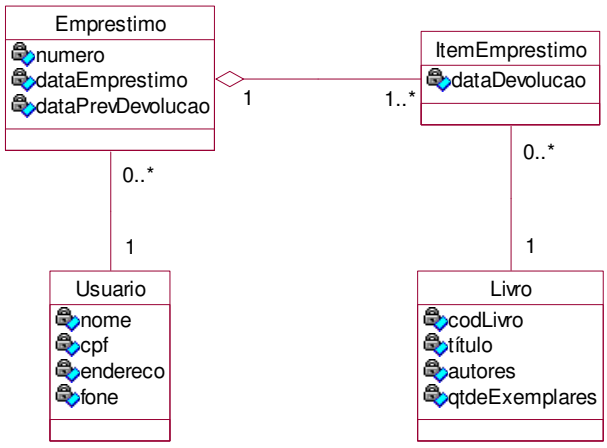
1) O trabalho pode ser feito em grupo de (no máximo) 4 pessoas. É permitido discutir os problemas e estratégias de solução com outros grupos, mas quando se tratar de escrever ou implementar computacionalmente as soluções, isto deve ser feito separadamente. O trabalho deve ser feito em uma linguagem de programação orientada a objetos, tal como Java ou C++.

2) Forma de entrega: O trabalho deve ser entregue em formato digital por meio do ***Moodle***. Utilizar a opção ***"Entrega do Trabalho Prático 1".*** Anexe **um único arquivo .zip** contendo todos os arquivos do trabalho**.** O nome do arquivo .zip deve conter o primeiro e o último nome de cada aluno integrante do grupo. Por exemplo: **CristianoCastro_WilianLacerda.zip. Obs:** O arquivo deve ser .zip mesmo! Não vale .tar, .tgz, .bz2, e outros formatos de compactação.

3) Trabalhos copiados receberão nota zero para todas as cópias. Trabalhos com erros de compilação não serão avaliados e receberão nota zero**.** Em qualquer erro deve-se emitir uma mensagem adequada ao erro. O programa deve ser desenvolvido seguindo as boas normas de programação.

Sistema de Controle de Biblioteca

Implemente um conjunto de classes para um sistema de controle de biblioteca. A biblioteca necessita manter informações sobre seus livros, usuários e empréstimos. Para isso, foi feito o seguinte projeto e você deve implementá-lo:



- **Classe Usuario**:
  - Atributos: nome (String), cpf (String), endereco (String) e fone (String). Todos com acesso privado (*private*);
  - Método construtor que inicializa todos os atributos através de parâmetros;
  - Médodos *get* e *set* para obter e modificar cada um dos valores dos atributos.

- **Classe Livro**:
  - Atributos: codLivro (int), titulo (String), autores (String, nome de todos os autores) e qtdeExemplares (int). Todos com acesso privado (*private*);
  - Método construtor que inicializa todos os atributos através de parâmetros;
  - Método construtor que inicializa os atributos codLivro, titulo e autores através de parâmetros e o atributo qtdeExemplares com zero;
  - Médodos *get* e *set* para obter e modificar os valores dos atributos titulo e autores;
  - Métodos *get* para obter os valores dos atributos codLivro e qtdeExemplares;
  - Método para incrementar a quantidade de exemplares de uma unidade;
  - Método para decrementar a quantidade de exemplares de uma unidade. Não permitir o decremento se a quantidade for igual a zero.

- **Classe Emprestimo**:
  - Atributos: numero (int), dataEmprestimo (GregorianCalendar), dataPrevDevolucao (data prevista para devolução, GregorianCalendar), usuario (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 de *array* ou da classe Vector. Todos com acesso privado (*private*);
  - Método construtor que inicializa os atributos com os seguintes valores: numero (próximo número sequencial), dataEmprestimo (data corrente do sistema), dataPrevDevolucao e usuario (passados como parâmetros);
  - Métodos *get* para obter os valores dos atributos numero, dataEmprestimo, dataPrevDevolucao e usuario;
  - Método para adicionar um item (livro) ao empréstimo. Parâmetro: o livro. A quantidade de exemplares do livro deve ser decrementada de uma unidade (verificar se a quantidade é suficiente);
  - Método para excluir um item (livro) do empréstimo. Parâmetro: o livro a ser excluído. A quantidade de exemplares do livro deve ser incrementada de uma unidade;
  - Método para devolver de um item (livro) do empréstimo. Parâmetro: 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;
  - Método para devolver de 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 (GregorianCalendar) e livro (Livro). Todos com acesso privado (*private*);
  - Método construtor que inicializa os atributos da seguinte forma: livro (passado como parâmetro) e dataDevolucao com valor nulo;
  - Métodos *get* para obter os valores dos atributos dataDevolucao e livro;
  - Método para atualizar o atributo dataDevolucao com a data do sistema.

- **Classe Biblioteca**:
  - Atributos: usuarios (lista de Usuario), livros (lista de Livro) e emprestimos (lista de Emprestimo). As listas podem ser implementadas através de *array* ou da classe Vector. Todos com acesso privado (*private*);
  - Método construtor que inicializa os atributos listas como vazias;
  - Método para inserir um novo usuario na lista de usuarios. Parâmetro: o usuario;
  - Método para inserir um novo livro na lista de livros. Parâmetro: o livro;

- Método para inserir um novo empréstimo na lista de empréstimos. Parâmetro: o empréstimo;
- Método para inserir um novo item de empréstimo para um empréstimo. Parâmetros: o empréstimo e o item de empréstimo;
- Método para excluir um usuario da lista de usuarios. Parâmetro: o usuario. O usuario não pode ser excluído se existir algum empréstimo para ele;
- Método para excluir um livro da lista de livros. Parâmetro: o livro. O livro não pode ser excluído se existir algum empréstimo para ele;
- Método para excluir um empréstimo da lista de empréstimos. Parâmetros: o empréstimo;
- Método para excluir um item de empréstimo de um empréstimo. Parâmetros: o empréstimo e o item de empréstimo;
- Método para devolver de um item (livro) de um empréstimo. Parâmetro: o empréstimo e o livro a ser devolvido.
- Método para devolver de todos os livros de um empréstimo. Parâmetro: o empréstimo.
- Método para pesquisar livros por título. Parâmetro: uma string especificando parte do título do livro. O método deve retornar uma lista com todos os livros que contêm a string no título;
- Método para pesquisar livros por autor. Parâmetro: 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;
- Método para obter a lista de usuarios;
- Método para obter a lista de livros;
- Método para obter a lista de empréstimos;
- Método para gravar os dados em arquivo;
- Método para ler os dados armazenados em arquivo.

- **Classe Interface**:
  - Método para apresentar um menu (interface de caracteres) com opções para: cadastrar um novo usuario, cadastrar um novo livro, cadastrar um novo empréstimo, inserir um novo item de empréstimo, excluir um usuario, excluir um livro, excluir um empréstimo, excluir um item de empréstimo, devolver todos os livros do empréstimo, devolver um livro do empréstimo, pesquisar livros por título, pesquisar livros por autor, listar todos os usuarios, listar todos os livros, listar todos os empréstimos e seus itens e sair do programa;
  - Método para cadastrar um novo usuario. Deve permitir que o usuário entre com os dados do usuario 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 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 usuario. Deve permitir que o usuário escolha o usuario a ser excluído;
  - Método para excluir um livro. Deve permitir que o usuário escolha o livro 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 livros por título. Deve permitir que o usuário entre com parte do título do livro a ser pesquisado. Deve listar os dados de todos os livros encontrados.
- 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 todos os livros. Deve listar os dados de todos os livros;
- Método para listar todos os empréstimos. Deve listar os dados de todos os empréstimos com seus respectivos itens;
- Método *main* para ler os dados armazenados em arquivo, apresentar o menu principal e, antes de sair do programa, gravar os dados em arquivo.

Observe a separação entre as classes de negócio (Emprestimo, ItemEmprestimo, Usuario e Livro) 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 idéia 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 arquivo 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.

Observações a serem seguidas:

- As classes, atributos e métodos descritos constituem o projeto básico do sistema. Pode-se adicionar outros métodos e atributos necessários para completar o sistema. Porém qualquer mudança “radical” no projeto deve ser discutida com o professor;
- Todas as classes, exceto Interface, devem estar em um pacote chamado **biblioteca**. Defina a variável CLASSPATH corretamente;
- Faça comentários no formato JavaDoc para todas as classes e todos os métodos públicos. Gere uma documentação usando este utilitário.