|  |  |
| --- | --- |
| Centro Universitário de Belo Horizonte | **DEPARTAMENTO DE CIÊNCIAS EXATAS E TECNOLOGIA - DCET**Análise e Projeto de Algoritmos |
| Valor: 30 pontos  Datas de entrega:08/05/2012 e 25/06/2012  Datas de apresentação: 08/05/2012 e 25/06/2012 | |
| O trabalho deverá ser realizado por grupos de no máximo 2 (dois) alunos, e ser entregue até as datas definidas para cada etapa. É fundamental a presença de todos os integrantes do grupo durante a apresentação do trabalho, quando serão avaliados e o aluno que não demonstrar conhecimentos sobre trabalho perderá pontos. O programa deve ser desenvolvido seguindo todas as técnicas de orientação por objetos. Não basta funcionar, tem que seguir os princípios OO.  Implemente um conjunto de classes para um sistema de controle de biblioteca. A biblioteca necessita manter informações sobre as publicações (livros e periódicos) disponíveis, usuários e empréstimos. Para isso, você deve implementar o projeto a seguir usando a linguagem de programação Java:    A biblioteca controla além de livros também a sua relação de 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.  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) devem ser implementadas usando a classe ArrayList.  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 excluir um livro que já tenha sido emprestado, não é permitido excluir um usuário que já tenha feito algum empréstimo etc. Faça todo o tratamento necessário, gerando mensagens usando a classe Exception. 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.  Faça comentários no formato JavaDoc para todas as classes e todos os métodos públicos. Gere uma documentação usando este utilitário.  Para cada classe, deve-se obdecer as seguintes orientações:  Classe Usuário:   * Atributos: nome (String), cpf (String), endereco (String) e fone (String). Todos com acesso privativo (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 Publicação:   * Atributos: codPublicacao (int), titulo (String), editora (String), ano (int). Todos com acesso privativo (private); * Método construtor que inicializa todos os atributos através de parâmetros; * Médodos get e set para obter e modificar os valores dos atributos titulo e editora; * Métodos get para obter os valores dos atributos codPublicacao e titulo;   Classe Livro (subclasse de Publicação):   * Atributos: autores (String, nome de todos os autores) e qtdeExemplares (int). Todos com acesso privativo (private); * Método construtor que inicializa todos os atributos através de parâmetros; * Método construtor que inicializa os atributos codPublicacao, titulo, editora, ano 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 codPublicacao 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 Periódico (subclasse de Publicação):   * Atributos: numEdicao (int), mes (String). Todos com acesso privativo (private); * Método construtor que inicializa todos os atributos através de parâmetros; * Métodos get para obter os valores dos atributos numEdicao e mes;   Classe Emprestimo:   * Atributos: numero (int), dataEmprestimo (GregorianCalendar), dataPrevDevolucao (data prevista para devolução, GregorianCalendar), usuário (Usuário), 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 ArrayList. Todos com acesso privativo (private); * Método construtor que inicializa os atributos com os seguintes valores: numero (próximo número sequencial), dataEmprestimo (data corrente do sistema), dataPrevDevolucao e usuário (passados como parâmetros); * Métodos get para obter os valores dos atributos numero, dataEmprestimo, dataPrevDevolucao e usuário; * 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 privativo (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: usuários (lista de Usuário), livros (lista de Livro) e emprestimos (lista de Emprestimo). As listas podem ser implementadas através da classe ArrayList. Todos com acesso privativo (private); * Método construtor que inicializa os atributos listas como vazias; * Método para inserir um novo usuário na lista de usuários. Parâmetro: o usuário; * 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 usuário da lista de usuários. Parâmetro: o usuário. O usuário 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 usuários; * 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 InterfaceComUsuario: * Método para cadastrar um novo usuário. Deve permitir que o usuário entre com os dados do usuário a ser inserido; * Método para cadastrar uma nova publicação. Deve permitir que o usuário entre com os dados da publicação a ser inserida; * 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 uma publicação. Deve permitir que o usuário escolha o livro ou o períodico 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 por título. Deve permitir que o usuário entre com parte do título a ser pesquisado. Deve listar os dados de todos 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 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 apresentar o menu principal.   Observe a separação entre as classes de negócio (Emprestimo, ItemEmprestimo, Usuário e Livro) e a classe de InterfaceComUsuario. 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.  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.  Cronograma de entregas:  1ª. Etapa:   * **O que entregar**: Implementação das classes Publicação, Livro, Periódico, Biblioteca e InterfaceComUsuario (com as funcionalidades inerentes as publicações). * **Prazo**: 08 de maio * **Valor**: 15 pontos   2ª. Etapa:   * **O que entregar**: Implementação do sistema completo com todos os requisitos especificados devidamente implementado e funcionando + JavaDoc do projeto. * **Prazo**: 25 de junho * **Valor**: 15+3 pontos   Pontos Extras:  Trabalho com interface gráfica ou Web para a classe Interface (3 pontos extras);  Gravar todos os dados do sistema em arquivos de objetos. Leitura de dados do usuário e impressão na tela devem ser feitos somente nas classes de interface. Deve-se separar as classes de negócio e as classes de persistência. Por simplicidade, nesse trabalho o armazenamento dos dados pode ser 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 (3 pontos extras/válidos somente em caso da não aplicação dos pontos de interface gráfica).  Os pontos extras serão avaliados somente para os alunos que implementarem todas os requisitos solicitados e souberem explicar detalhadamente o funcionamento do seu trabalho.  Vale ressaltar que não será admitida cópia de trabalhos, de espécie alguma. Entendam não admitir cópia como sendo premiação com nota zero para ambos os grupos. Trabalhos com erros de compilação ou de execução não serão avaliados.  Bom trabalho a todos!. | |
|  | |
|  | |