Algoritmos e Estruturas de Dados I

Prof. Jurair Rosa

Trabalho prático - Biblioteca

Centro Federal de Educação Tecnológica Celso Suckow da Fonseca

O CEFET começou a reconstruir o acervo de sua biblioteca, e para o gerenciamento virtual da biblioteca foi decidido que seria selecionado um aluno da Engenharia de Computação para criar um novo *software* de gerenciamento. Na hora da seleção, você tirou a sorte grande e foi escolhido para desenvolver esse *software*. Agora a biblioteca do CEFET está em suas mãos e cabe a você trazê-la à sua antiga glória!

Os seus supervisores lhe mandaram um conjunto de especificações a serem seguidas. Essas especificações detalham as funcionalidades que o *software* de gerenciamento precisa ter para acomodar as necessidades da biblioteca:

1) Gerenciamento de Alunos: os alunos que desejam realizar atividades na biblioteca podem ter seu cadastro realizado. O cadastro é composto pelo nome e matrícula do aluno (composta de números e letras), assim como um ID gerado pela biblioteca. Deve ser possível adicionar e remover cadastros, assim como listar todos os cadastros e também buscar o cadastro de um aluno a partir de seu ID. Não deve ser possível remover o cadastro de um aluno que possui alguma pendência com a biblioteca (pegou algo emprestado e ainda não devolveu);
2) Gerenciamento de Livros: a biblioteca também possui seu acervo de livros físicos. Para gerenciar esse acervo é necessário um registro digital desses livros. Os livros possuem nome, ano de publicação, categoria e um ID gerado pela biblioteca. Deve ser possível adicionar e remover livros nesse registro, assim como listar todos os livros e também buscar os livros por ID e por categoria. Outra funcionalidade necessária é o empréstimo de livros. No empréstimo de livros, deve ser possível que alunos peguem ou devolvam um livro emprestado, mudando seu estado. Para isso, o registro do livro também deve conter um estado (*disponível* ou *emprestado*), assim como o ID de um aluno atribuído ao livro caso o estado seja emprestado. Vale lembrar que essas informações também devem aparecer na listagem ou busca dos livros, sendo que ao invés do ID do aluno deve ser mostrada a sua matrícula. Também não deve ser possível remover o registro de um livro que ainda não foi devolvido;
3) Gerenciamento de Salas, Computadores e Armários: algumas salas, computadores e armários da biblioteca podem ser emprestados para os alunos. Como a infraestrutura da biblioteca pode mudar (novas salas podem ser construídas, computadores podem quebrar, etc) logo deve haver um registro para esses recursos. Um recurso possui um ID inserido manualmente, um tipo (sala, computador ou armário), um estado (*livre* ou *ocupado*) e um ID de aluno atribuído ao recurso caso o mesmo esteja ocupado. Deve ser possível adicionar, remover e listar esses recursos, assim como buscá-los por ID. Também deve ser possível que alunos ocupem ou desocupem esses recursos, mudando seu estado. Não deve ser possível remover um recurso que está ocupado. Vale lembrar que na listagem ou busca, a matrícula do aluno que está ocupando o recurso deve ser mostrada ao invés do ID desse aluno;
4) Exportação/importação dos dados (*bônus valendo 0,5 pto*): seria interessante que as informações criadas durante o uso fossem armazenadas ao fechar o sistema (exportar) e recuperadas na próxima inicialização (importar). Além disso, para evitar que os dados sejam visualizados por terceiros caso estes sejam extraviados, ao armazenar as informações estas devem ser criptografadas e, por sua vez, descriptografadas na importação dos dados;
5) UI (*bônus valendo 0,5 pto*): é desejável, e foi requisitado pelos gestores, que uma interface gráfica seja desenvolvida para tornar o uso do sistema mais amigável.

Vale lembrar que o *software* de gerenciamento é utilizado pelos atendentes da biblioteca, ou seja, os alunos não interagem diretamente com o mesmo. A listagem e a busca dos elementos deve mostrar todas as informações referentes ao mesmo. O *software* deve ser desenvolvido em linguagem C e possuir interface de linha de comando. Em caso de implementação do item 5, todo o código fonte (incluindo a UI) poderá ser implementado em C++.

Instruções:

- O estudo e compreensão do problema faz parte do trabalho, incluindo a escolha das estruturas de dados que serão utilizadas. No entanto, dúvidas podem ser sanadas junto ao monitor e/ou professor;
- O trabalho é individual;
- Pontuação (0,0 - 7,0 pts):
  - Para todo `arquivo.c` (com exceção do `main`), deve haver seu respectivo cabeçalho (`arquivo.h`), o qual deve ser documentado[1]. Se nunca fez algo dessa forma, sugiro que assista essa videoaula. Lembrando que tanto a documentação quanto a formatação do código fonte serão avaliados;
  - Boas práticas de programação: modularização, indentação e otimização dos recursos utilizados (memória);
  - Apresentação do seu trabalho em funcionamento, onde mostrará os tópicos mais importantes da implementação, bem como o uso do *software*, e se ele atende aos requisitos especificados;
  - **Os itens 4 e 5 são bônus.** Acrescentar-se-á 0,5 ponto por item concluído, podendo atingir 7,0 pontos no total. Caso contrário, a pontuação máxima passa a ser 6,0 pontos.
- O trabalho (código fonte e documentação em HTML, gerada pelo Doxygen) deverá ser entregue até às 23h59m do dia 31 de julho de 2022;
- As apresentações ocorrerão presencialmente entre os dias 01 e 03 de agosto de 2022, com dia e horário distintos pra cada discente;
- É imprescindível que os arquivos a serem entregues tenham seu nome e sobrenome. Utilize a seguinte formatação:
  - **nome-sobrenome-src.zip**: código fonte (arquivos .c e .h);
  - **nome-sobrenome-doc.zip**: documentação associada ao código fonte, gerada pelo Doxygen.
- O trabalho deverá ser executado em um computador do laboratório (à definir), ou no computador pessoal do aluno.

---

[1] O código fonte deverá ser comentado no formato Doxygen. Com isso será possível gerar uma documentação automática do código fonte (em HTML). O documento final, gerado automaticamente pelo Doxygen, também deverá ser entregue.