UNIVERSIDADE FEDERAL DE MINAS GERAIS
Departamento de Ciência da Computação
Bacharelado em Sistemas de Informação

Filipe Martins
Gabriel Resende
Jean George
João Gontijo

Trabalho Prático - Grupo 7
Sistema para Biblioteca

Trabalho elaborado com vista à avaliação na disciplina de *Programação e Desenvolvimento de Software II,* ministrada pelo Profº Douglas Guimarães Macharet.

Belo Horizonte
Outubro/2018

SUMÁRIO

1. Introdução

Este documento apresenta o relatório técnico referente ao sistema de uma biblioteca, feito em C++, desenvolvido pelo grupo composto pelos alunos Filipe Martins, Gabriel Resende, Jean George e João Gontijo, do curso de Sistemas de Informação.

C++ é uma linguagem de programação compilada multi-paradigma (seu suporte inclui linguagem imperativa, orientada a objetos e genérica) e de uso geral. Desde os anos 1990 é uma das linguagens comerciais mais populares, sendo bastante usada também na academia por seu grande desempenho e base de utilizadores.

Bjarne Stroustrup desenvolveu o C++ (originalmente com o nome C with Classes, que significa C com classes em português) em 1983 no Bell Labs como um adicional à linguagem C. Novas características foram adicionadas com o tempo, como funções virtuais, sobrecarga de operadores, herança múltipla, gabaritos e tratamento de exceções. Após a padronização ISO realizada em 1998 e a posterior revisão realizada em 2003, uma nova versão da especificação da linguagem foi lançada em dezembro de 2014, conhecida informalmente como C++17.

O sistema feito pelo grupo ajuda o usuário, no caso um bibliotecário, a gerir toda uma biblioteca, com seus livros, usuários e transações.

No trabalho foi utilizado o GitHub, que nos auxiliou muito no decorrer do desenvolvimento e controle de versão do sistema a ser desenvolvido.

Nosso objetivo, é de que o usuário possa gerir uma biblioteca com um sistema que é simples e eficiente, tendo mais informações em suas mãos e assim facilitar o leitor da biblioteca a ter acesso aos livros disponíveis.

2 - Implementação

Seguindo a especificação, foi criado um sistema de biblioteca, onde há o login do bibliotecário, e as funcionalidades de Busca, Reserva e Empréstimo implementadas, além de algumas funcionalidades extras, como a aplicação de dias de bloqueio do Usuário em caso de devolução atrasada. Para a implementação do trabalho, foram implementadas as classes: Reserva, Empréstimo, Livro, Pessoa, Usuário, Bibliotecário, Opcao, Tela, TelaInicial, TelaLivros, TelaPrincipal, TelaTransacoes, TelaUsuarios, Arquivo, AbrirArquivoException, Data, DataInvalidaException e a Principal: a classe Biblioteca.

- Classe Reserva:
    - Faz a reserva de um livro para um usuário caso o número de exemplares seja 0.
        - void setIdLivro(int idLivro): atualiza o valor da variável idLivro na classe Reserva.
        - void setIdUsuario(int idUsuario): atualiza o valor da variavel idUsuario na classe Reserva.
        - void setIdReserva(int idReserva): atualiza o valor da variavel idReserva na classe Reserva.
        - int getIdReserva(): Retorna id da reserva.
        - int getIdUsuario(): Retorna a variável idUsuario.
        - getIdLivro(): Retorna a variável idLivro.
        - string toString(): Formata a saída do objeto, e é usada para retornar a string com os dados, usada para imprimir.
        - string toStringToSave(): Organiza os dados para colocar no txt com os devidos separadores.

- Classe Empréstimo:
    - Classe usada para relacionar Usuários e Livros criando vínculos entre eles(Alugar, devolver, renovar)
        - int getIdEmprestimo(): Retora o valor da variavel idEmprestimo.
        - void setIdEmprestimo(int idEmprestimo): Atualiza a variável idEmprestimo da classe Emprestimo
        - int getIdLivro(): Retorna o valor da variável idLivro.
        - void setIdLivro(int idLivro): Atualiza a variável idLivro da classe Emprestimo
        - int getIdUsuario(): Retorna o valor da variavel idUsuario.
        - void setIdUsuario(int idUsuario): Atualiza a variável idUsuario da classe Emprestimo
        - int getValorMulta(): Retorna o valor da variável valorMulta.
        - void setValorMulta(int valorMulta): Atualiza a variável valorMulta da classe Emprestimo
        - int getDiaDevolucao(): Retorna o valor da variável diaDevolucao.
        - int getMesDevolucao(): Retorna o valor da variável mesDevolucao.
        - int getAnoDevolucao(): Retorna o valor da variável anoDevolucao.
        - void setDiaRealizacao(int): Atualiza a variável diaRealizacao da classe Emprestimo.
        - void setMesRealizacao(int): Atualiza a variável mesRealizacao da classe Emprestimo
        - void setAnoRealizacao(int): Atualiza a variável anoRealizacao da classe Emprestimo
        - void setDiaDevolucao(int): Atualiza a variável diaDevolucao da classe Emprestimo.

        - void setMesDevolucao(int): Atualiza a variável mesDevolucao da classe Emprestimo.
        - void setAnoDevolucao(int): Atualiza a variável anoDevolucao da classe Emprestimo.
        - string toString(): Formata a saída do objeto, e é usada para retornar a string com os dados, usada para imprimir.
        - string toStringToSave(): Organiza os dados para colocar no txt com os devidos separadores.

- Classe Livro:
    - Classe livro contém todos dados do livro como título, autor, quantidade de cópias, genero e um id.
        - void aumentarQuantidade(): Função para aumentar a quantidade de exemplares do livro(1 unidade).
        - string toString(): Formata a saída do objeto, e é usada para retornar a string com os dados, usada para imprimir.
        - string toStringToSave(): Organiza os dados para colocar no txt com os devidos separadores.
        - string getAutor(): Retorna a string da variavel autor.
        - void setAutor(string autor): Atualiza a variável autor da classe Livro.
        - string getGenero(): Retorna a string da variavel genero.
        - void setGenero(string genero): Atualiza a variável genero da classe Livro.
        - int getId(): Retorna o valor da variável id.
        - void setId(int id): Atualiza o valor da variável id da classe Livro
        - int getQuantidadeDeCopias(): Retorna o valor da variável quantidadeDeCopias.

        - void setQuantidadeDeCopias(int quantidadeDeCopias): Atualiza o valor da váriavel quantidadeDeCopias da classe Livros.
        - string getTitulo(): Retorna a string da variável titulo.
        - void setTitulo(string titulo): Atualiza a variável titulo da classe Livro.

- Classe Pessoa:
    - Pessoa é a classe base para as classes Usuário e Bibliotecario.
        - int getId(): Retorna o valor da variável id da classe Pessoa.
        - string getCpf(): Retorna a string da variável cpf.
        - void setCpf(string cpf): Atualiza a variável cpf da classe Pessoa.
        - string getDataNascimento(): Retorna a string da variavel dataNascimento.
        - void setDataNascimento(string): Atualiza a variável dataNascimento da classe Pessoa.
        - string getNome(): Retorna a string da variável nome.
        - void setNome(string): Atualiza a variavel nome da classe Pessoa.
        - string getSobrenome(): Retorna a string da variável sobrenome.
        - void setSobrenome(string): Atualiza a variável sobrenome da classe Pessoa.
        - char getSexo(): Retorna o char da variável sexo.
        - void setSexo(char): Atualiza a variável sexo da classe Pessoa.
        - string toString(): Formata a saída do objeto, e é usada para retornar a string com os dados, usada para imprimir.
        - string toStringToSave(): Organiza os dados para colocar no txt com os devidos separadores.

- Classe Usuário:

- Classe que armazena e manipula as informações do usuário cadastrado.
- Funções:
  - string toString() override: Formata a saída do objeto, e é usada para retornar a string com os dados, usada para imprimir.
  - string toStringToSave() override: Organiza os dados para colocar no txt com os devidos separadores.
  - string getEmail(): Retorna a string da variável email.
  - void setEmail(string email): Atualiza a variavel email da classe Usuario.
  - bool getStatus() : retorna o status do usuário.
  - void setStatus(bool status): muda o status do usuário.
  - string getTelefone(): retorna o telefone do usuário.
  - void setTelefone(string telefone): muda o telefone do usuário.
  - void setDiasBloqueados(int): muda o valor dos dias de bloqueio do usuário.
  - int getDiasBloqueados(): retorna os dias de bloqueio do usuário.
  - static bool getCpfValido(string& cpf, bool bloquearEntradaVazia): verifica se o CPF é válido.
  - static bool getSexoValido(string& sexo): verifica o sexo válido.
  - static bool getStatusValido(string& status): verifica o status válido.
  - int gerarID() override: gera um ID para o usuário.

- Classe Bibliotecário:
  - Classe que armazena e manipula as informações do bibliotecário cadastrado.
  - Funções:
    - bool logarNoSistema(const vector<Bibliotecario>& bibliotecariosCadastrados): loga o bibliotecário no sistema.

        - string toString() override: printa os dados do bibliotecário na tela.
        - string toStringToSave() override: salva os dados do bibliotecário nos arquivos
        - bool receberEntradas(string& inputLogin, string& inputSenha): recebe as entradas passadas pelo usuário durante o login
        - bool validarEntradasLogin(const vector<Bibliotecario>& bibliotecariosCadastrados, string inputLogin, string inputSenha): verifica a validade das informações passadas no login.
        - int gerarID() override : gera um ID para o bibliotecário.

- Classe Opcao:
    - Cria uma opção que será utilizada na Classe Tela, contendo um valor e uma descrição.

- Classe Tela:
    - Cria uma tela onde é apresentada diversas opções de acordo com a escolha do bibliotecário, que será sobrescrita pelas classes TelaInicial, TelaLivros, TelaPrincipal, TelaTransacoes e TelaUsuarios.

- Classe TelaInicial:
    - Fornece opções ao bibliotecário em relação a:
        - Realizar login
        - Cadastrar um novo Bibliotecário

- Classe TelaLivros:
    - Fornece opções ao bibliotecário em relação a:
        - Voltar para o Menu Principal
        - Visualizar todos os livros ou um livro específico
        - Atualizar dados de um livro
        - Adicionar ou Remover um livro na base de dados

- Classe TelaPrincipal:
    - Fornece opções ao bibliotecário em relação a:
        - Gerenciar Livros
        - Gerenciar Usuarios
        - Gerenciar Transacoes

- Classe TelaTransacoes:
    - Fornece opções ao bibliotecário em relação a:
        - Voltar para o Menu Principal
        - Visualizar todos os Empréstimos ou um Empréstimo específico
        - Visualizar todas as Reservas ou uma Reserva específica
        - Realizar ou Renovar um Empréstimo
        - Realizar uma Devolução
        - Realizar ou Cancelar uma Reserva

- Classe TelaUsuarios:
    - Fornece opções ao bibliotecário em relação a:
        - Voltar para o Menu Principal
        - Visualizar todos os usuários ou um usuário específico
        - Atualizar dados de um usuário
        - Adicionar ou Remover um usuário na base de dados

- Classe Arquivo:
    - Faz as manipulações necessárias com os arquivos .txt.
    - Funções:
        - void atualizarDados (const vector<Bibliotecario>& vectorBibliotecarios): atualiza os dados do vector alterado no arquivo bibliotecario.txt.
        - void atualizarDados(const vector<Usuario>& vectorUsuarios): atualiza os dados do vector alterado no arquivo usuarios.txt.
        - void atualizarDados(const vector<Emprestimo>& vectorEmprestimos): atualiza os dados do vector alterado no arquivo emprestimos.txt.
        - void atualizarDados(const vector<Livro>& vectorLivros): atualiza os dados do vector alterado no arquivo livros.txt.
        - void atualizarDados(const vector<Reserva>& vectorReservas): atualiza os dados do vector alterado no arquivo reservas.txt.
        - template<class T, class A> void obterDados(vector<T, A> &vectorParam): busca os dados do arquivo .txt e aloca no vector passado por parâmetro.
        - void alocarDadosLinha(string linha, vector<Bibliotecario>& vectorBibliotecarios): aloca os dados buscados na linha do arquivo .txt no vectorBibliotecarios.

        - void alocarDadosLinha(string linha, vector<Usuario>& vectorUsuarios): aloca os dados buscados na linha do arquivo .txt no vectorUsuarios.
        - void alocarDadosLinha(string linha, vector<Emprestimo>& vectorEmprestimos): aloca os dados buscados na linha do arquivo .txt no vectorEmprestimos.
        - void alocarDadosLinha(string linha, vector<Livro>& vectorLivros): aloca os dados buscados na linha do arquivo .txt no vectorLivros.
        - void alocarDadosLinha(string linha, vector<Reserva>& vectorReservas): aloca os dados buscados na linha do arquivo .txt no vectorReservas.
        - void abrirModoLeitura(): abre para a leitura o arquivo .txt
        - void abrirModoEscrita(): abre para a escrita o arquivo .txt
        - void fecharModoLeitura(): fecha para a leitura o arquivo .txt
        - void fecharModoEscrita(): fecha para a escrita o arquivo .txt
    - Variáveis:
        - string diretorio: armazena o diretorio do arquivo a ser trabalhado.
        - ifstream arquivoLeitura: armazena o arquivo para leitura.
        - ofstream arquivoEscrita: armazena o arquivo para escrita.

- Classe AbrirArquivoException:
    - Realiza verificações de exceção na abertura do arquivo, na criação de um arquivo, e se o arquivo estiver vazio.

- Classe Data:
    - Classe que faz os cálculos de datas para facilitar o usuário a adicionando e manipulando a automaticamente no sistema.

    - Funções:
        - void setData(int, int, int): troca o valor da data armazenada.
        - Data &operator++(): incrementa a data.
        - Data operator++(int): incrementa a data de acordo com o valor passado por parâmetro.
        - const Data &operator+=(int): adiciona dias de acordo com o valor passado por parâmetro.
        - bool anoBissexto(int) const: verifica se o ano é bissexto ou não.
        - bool fimDoMes(int) const: verifica se o fim do mês ou não.
        - int getMes() const: retorna o mês.
        - int getDia(): retorna o dia.
        - int getMesValor(): retorna o mês.
        - int getAno(): retorna o ano.
        - void auxiliaIncremento(): função auxiliar no incremento da data
    - Variáveis:
        - int mes: armazena o mês.
        - int dia: armazena o dia.
        - int ano: armazena o ano.
        - static const int dias[]: armazena os dias de cada mes.

- Classe DataInvalidaExpection:
    - Verifica as possíveis exceções relacionadas à datas inválidas.

- Classe Biblioteca:
    - Faz todo o controle do sistema.
    - Funções:
        - string getNome(): retorna o nome da biblioteca.

- void setNome(string): muda o nome da biblioteca.
- string getEndereco(): retorna o endereco da biblioteca.
- void setEndereco(string): muda o endereco da biblioteca.
- string getTelefone():retorna o telefone da biblioteca.
- void setTelefone(string):retorna o telefone da biblioteca.
- void iniciarSistema(): realiza a inicialização do sistema
- bool carregarArquivos(): carrega os arquivos na classe os armazenando no vector definido.
- void exibirConfirmacaoLogin(): mostra na tela o sucesso no login
- void exibirMenu(): mostra o menu com as opções iniciais
- void mostrarData(): mostra a data na tela
- void atualizarStatus(): atualiza o status do usuário
- bool getEntradaValida(string& entrada, bool bloquearEntradaVazia): Verifica se a entrada digitada é valida
- bool getEntradaInt(int& entrada): Verifica se a entrada digitada é um inteiro
- bool confirmarAcao(string): Exibe tela mensagem de confirmação e faz a validação se o digitado foi Sim ou Não.
- void removerEspacos(string& linha): Remove todos os espaços de uma variável.
- void verificarOpcaoPorTela(string opcao): Verifica qual foi a opção digitada pelo usuário na tela inicial e chama a função que trata os dados da tela digitada.
- void verificarOpcaoTelaPrincipal(string opcao): Trata os dados digitados na tela principal.
- void verificarOpcaoTelaLivros(string opcao): Trata os dados digitados na tela de livros.
- void verificarOpcaoTelaUsuarios(string opcao): Trata os dados digitados na tela de usuários.

- void verificarOpcaoTelaTransacoes(string opcao): Trata os dados digitados na tela de transações.
- void listarLivros(): Exibe uma lista com todos os livros e seus dados.
- void pesquisarLivro(): Pesquisa e exibe, se encontrado, um livro com título informado pelo bibliotecário.
- void atualizarLivro(): Recebe entrada, pesquisa um livro e, se encontrado, salva seus dados de acordo com as entradas informadas pelo bibliotecário.
- bool obterDadosAtualizarLivro(Livro& l): Recebe as entradas do bibliotecário para definir os dados do livro a ser atualizado. Se a entrada for vazia, mantém o valor antigo.
- void adicionarLivro(): Adiciona um novo livro e salva.
- bool existeLivro(string nome, string autor): Verifica se existe um livro com determinado título e autor.
- bool construirNovoLivro(Livro*& l): Recebe entradas e tenta criar um novo livro.
- void removerLivro(): Remove um livro com ID (inteiro) informado pelo bibliotecário e salva.
- void listarUsuarios(): Exibe uma lista com todos os usuarios e seus dados.
- void pesquisarUsuario(): Pesquisa e exibe, se encontrado, um usuário com CPF (apenas números) informado pelo bibliotecário
- void atualizarUsuario(): Recebe entrada, pesquisa um usuário e, se encontrado, salva seus dados de acordo com as entradas informadas pelo bibliotecário.
- bool obterDadosAtualizarUsuario(Usuario& u): Recebe as entradas do bibliotecário para definir os dados do usuário a ser atualizado. Se a entrada for vazia, mantém o valor antigo.

- void verificarDefinirDadosUsuario(Usuario& u, string dados[8]): Verifica se cada dado é vazio. Se for vazio não, atualiza o atributo, e se não for vazio, atualiza o atributo.
- void adicionarUsuario(): Adiciona um novo usuário e salva.
- bool existeUsuario(string cpf): Verifica se existe um usuário com determinado CPF.
- bool construirNovoUsuario(Usuario*& u): Recebe entradas e tenta criar um novo usuário.
- void removerUsuario(): Remove um usuário com ID (inteiro) informado pelo bibliotecário e salva.
- int getIdNovoLivro(): Busca o ultimo ID de livro cadastrado, acrescenta 1 e retorna esse valor.
- int getIdNovoUsuario(): Busca o ultimo ID de usuário cadastrado, acrescenta 1 e retorna esse valor.
- int getIdNovoEmprestimo(): Busca o ultimo ID de empréstimo cadastrado, acrescenta 1 e retorna esse valor.
- int getIdNovaReserva(): Busca o ultimo ID de reserva cadastrado, acrescenta 1 e retorna esse valor.
- void listarEmprestimos(): Exibe uma lista com todos os empréstimos e seus dados.
- void pesquisarEmprestimo(): Lê o ID do empréstimo e remove empréstimo do arquivo, caso o ID seja válido.
- void realizarEmprestimo(): Lê o ID do Usuário e do livro, cria o emprestimo e salva no arquivo.
- void renovarEmprestimo(): Lê o ID do empréstimo e realiza, faz as validações necessárias e adiciona o empréstimo no arquivo.
- void realizarDevolucao(): Remove um usuário com ID (inteiro) informado pelo bibliotecário e salva.

        - void listarReservas(): Exibe uma lista com todas as reservas e seus dados.
        - void pesquisarReserva(): Lê o ID da reserva e realiza uma pesquisa no arquivo, mostrando os dados da reserva caso encontre.
        - void realizarReserva(): Lê os dados da reserva, faz as validações necessárias e cria uma reserva adicionando-a no arquivo de reservas.
        - void cancelarReserva(): Lê o ID da reserva e remove a reserva do arquivo, caso o ID seja válido.
    - Variáveis:
        - Bibliotecario *bibliotecarioAtual: Armazena qual é o bibliotecário logado no sistema.
        - Arquivo *arquivoBibliotecarios, *arquivoUsuarios, *arquivoLivros, *arquivoEmprestimos, *arquivoReservas: Realizam na abertura dos arquivos .txt e obtenção dos dados dos arquivos .txt para os vectors
        - Tela* telaAtual: Armazena em qual tela o programa está no momento.
        - vector<Bibliotecario> vectorBibliotecarios: Armazena os dados recebidos do arquivo bibliotecarios.txt.
        - vector<Usuario> vectorUsuarios: Armazena os dados recebidos do arquivo usuarios.txt.
        - vector<Livro> vectorLivros: Armazena os dados recebidos do arquivo livros.txt.
        - vector<Emprestimo> vectorEmprestimos: Armazena os dados recebidos do arquivo emprestimos.txt.
        - vector<Reserva> vectorReservas: Armazena os dados recebidos do arquivo reservas.txt.

- Data data: Armazena a data atual.
- string nome: Armazena o nome da biblioteca.
- string endereco: Armazena o endereço da biblioteca.
- string telefone: Armazena o telefone da biblioteca.

3 - Conclusão

O desenvolvimento do presente trabalho nos possibilitou aprender os conceitos de uma nova linguagem de programação. Além disso garantiu a evolução das técnicas de lógica e programação lecionadas neste curso.

Uma dificuldade encontrada de início foi pensar e planejar o sistema, de modo com que facilitasse a codificação do mesmo, mas isso foi rapidamente contornado graças aos User Stories e CRC´s bem feitos que nos deram uma boa ideia de como desenvolver nosso sistema de biblioteca.

Outra dificuldade foi pesquisar e adaptar funcionalidades do C++ para o uso no trabalho, que demandou de pesquisas em sites e fóruns da internet relacionados a linguagem.

De forma geral, o trabalho foi um sucesso, implementando todos os requisitos necessários, um sistema completo, com informações e menu intuitivos, e tudo isso de uma forma simples e eficiente.

BIBLIOGRAFIA E REFERÊNCIAS

C++ - Wikipédia. 2018. Disponível em: <https://pt.wikipedia.org/wiki/C%2B%2B>. Acesso em: 15 out. 2018.

GITHUB. 2018. Disponível em:<https://github.com/>. Acesso em: 19 out. 2017.

Stringstream - C++ reference. 2018. Disponível em: <http://www.cplusplus.com/reference/sstream/stringstream/>. Acesso em: 19 out. 2018.