UM SISTEMA PARA BIBLIOTECAS - PDS II

Componentes do Grupo:

Artur Lazzarini
Arthut Lisboa
Gabriel Cerqueira
Guilherme Tolentino

BELO HORIZONTE (MG) Novembro de 2018

SUMÁRIO

A) Introdução – página 3
- a.1) Apresentação – página 3

B) Objetivos – página 4
- b.1) Objetivo Geral – página 4
- b.2) Objetivos Específicos – página 4

C) Metodologia – página 4
- c.1) Implementação e Estudo Técnico – página 4
- c.2) Fluxo do Programa– página 4
- c.3)Fluxo do Programa: O Usuário – página 5
- c.4)Fluxo do Programa: O Funcionário – página 5
- c.5) Resolução de Problemas – página 5

D) Conclusão - página 6

E) Referências Bibliográficas – página 6

F) Apêndice - Cartões CRC – página 6

A)Introdução

Ao utilizar os conhecimentos adquiridos durante o curso de Programação e Desenvolvimento de Software II, nos propusemos a criar um sistema de biblioteca, tendo como pilar motivacional a programação orientada a objetos, modelos mentais de abstração e a digitalização de seus processos físicos. As funcionalidades aplicadas ao sistema pretendem aumentar a eficácia do atendimento e proporcionar mais comodidade aos usuários. Uma biblioteca trabalha constantemente com informação. A segurança, encapsulamento e robustez do programa são de valor primordial e erros como o mal funcionamento e perda de arquivo são inaceitáveis. A partir de agora, qualquer usuário poderá realizar pesquisas de itens no acervo, reservas, aluguéis e devoluções de produtos adquiridos na biblioteca.

a.1)Apresentação

A seguinte documentação é um breve roteiro de estudo acerca da implementação em C++ de um simulador de um sistema de bibliotecas. De forma sucinta, você pode ser um Usuário (cliente) da biblioteca ou um Funcionário. Os menus apresentados e as devidas apresentações de interface se modificam de acordo com sua escolha. Virtualmente, um usuário pode alugar, reservar ou devolver um item. Um funcionário pode cadastrar um novo cliente ou um novo funcionário, além de ser responsável pelas modificações no acervo virtual da biblioteca. A interface principal comunica com o usuário de maneira eficiente, uma vez que conceitos da engenharia de usabilidade foram aplicados. Um sistema de cadastro com login e senha também foi criado.

No ato do empréstimo ou reserva, ocorre a verificação da disponibilidade do item no acervo. Quando solicitado pelo usuário, o sistema deverá gerar uma lista dos itens que já tenham sido alugados ou reservados por ele. Já por parte do funcionário, o sistema permite adicionar e remover multimídias do acervo, cadastrar novos funcionários e usuários e imprimir as coleções de itens.

A definição de casos de uso e regras de negócio foram de extrema importância para a construção do sistema. Depreender tais quesitos fez com que o projeto se desenvolvesse de maneira fluida.

B)Objetivos

b.1) Objetivos Gerais

- Colocar em prática os conceitos de POO
- Maximizar os conhecimentos adquiridos em sala de aula
- Criar um sistema funcional
- Trabalhar com os princípios da Metodologia Ágil

b.2) Objetivos Específicos

- Comunicar o sistema ao usuário de maneira eficiente
- Promover o trabalho em equipe
- Desenvolver melhores práticas de programação

C) Metodologia

c.1)Implementação e Estudo Técnico

Para a modelagem do sistema e construção do diagrama de classes, foi-se criada uma classe principal chamada *Biblioteca*. *Pessoa* e *Item* são classes base - não instanciadas no programa - que possuem características gerais das classes *Usuários* e *Funcionário*. Estas herdaram de *Pessoa*, por possuírem características específicas. Da mesma maneira, as classes *Livro* e *Multimídia* herdam de *Item*. A decisão de implementação de uma classe para *Multimídia* mostrou-se necessária após uma conversa com o monitor Armando, o qual nos ajudou a identificar as futuras regras de negócio que viriam a ser depreendidas pelo sistema. Uma outra classe foi criada para representar o *Acervo*, a qual gere tudo relacionado ao arquivo de itens. A classe *Gerência* é responsável por funções administrativas, como pesquisa de usuário/funcionário e suas devidas criações. A classe *LoginInvalido* representa a exceção de uma tentativa fracassada de entrar no sistema ocasionada por um cadastro/login mal feito.

c.2)Fluxo do Programa

O programa se inicia em uma função da classe biblioteca que inicializa a interface - interfacePrincipal(). Nela são criados os principais objetos em conjunto com um funcionário base. Ocorre também a inicialização do acervo, que por sua vez coleta as informações dos dados do arquivo .txt para a memória principal em tempo de execução. Em seguida é feita a determinação das interfaces, onde há a seleção usuário/funcionário.

c.3)Fluxo do Programa: O Usuário

Como usuário há a verificação da existência de seu cadastro. Caso possua e seja validado, é realizado seu login no sistema. Em caso contrário é aberta a opção de realizar um novo cadastro. Não obstante, a validação de conflito de cadastros já existentes são tratados ao lançar a exceção do tipo LoginInvalido.

Com o cadastro pessoal de usuário feito, a função de biblioteca menuUsuario gera um menu interativo mostrando todas as operações que podem ser feitas pelo sistema. Desde pesquisas por itens no acervo, aluguéis, reservas e devoluções, até listagens de seus próprios aluguéis e reservas. Nas operações com referência direta ao item, é sempre verificada sua disponibilidade, concretizando-a se tiver o item requerido no acervo. Já nas operações de pesquisa por livros ou multimídia, o programa oferece filtros, podendo-se realizar por título, autor ou editora, de modo a facilitar os comandos do usuário.

c.4)Fluxo do Programa: O Funcionário

Entretanto se o login é feito por um funcionário, o mesmo possui operações mais gerenciais dentro do sistema. Adicionar ou remover itens do acervo, cadastrar novos usuários ou funcionários e fazer a listagem de toda coleção de itens da biblioteca é de sua responsabilidade. Sempre quando o acervo é modificado é feita a atualização do arquivo texto que contém os dados dos itens. E então ao final da execução é chamada a função desalocaDados(), no qual realiza todo o processo de desalocar os blocos de memória usados. O vazamento de memória é inadmissível, pois trata-se de um sistema utilizado não eventualmente, mas todos os dias e por milhares de usuários.

c.5) Resolução de Problemas

A principal dificuldade deu-se na primeira semana de implementação. Abstrair em modelos mentais e levar para o código as decisões de implementação em grupo tornou-se um grande desafio. A criação e separação de classes, manutenção do contrato e comunicação eficiente do programa com a utilização dos conceitos de POO estiveram presentes durante todo o trabalho. Entretanto, a busca por soluções viáveis e a integração forte do grupo permitiu que o trabalho fluísse sem muitos empecilhos.

O armazenamento do acervo da biblioteca foi um dilema enfrentado. A decisão em conjunto entre carregar o arquivo *.txt* em memória ou mexer constantemente com a abertura dos mesmos demandou tempo.

Por fim, recorreu-se a ajuda do monitor Armando, o qual nos esclareceu que o carregamento em memória, por mais que pareça inviável para arquivos enormes de um acervo, apresentaria melhor desempenho em tempo e segurança.

D)Conclusão

O objetivo principal do trabalho é a consolidação dos principais conceitos de programação orientada a objetos vistos durante o semestre. A concepção de herança, encapsulamento, polimorfismo e principalmente abstração foram melhor visualizadas durante a implementação prática. A percepção do papel de cada um foi importante no tratamento e organização da informação, e só assim, ser feita as decisões de implementação.

E)Referências Bibliográficas

Acess Specifiers. cppREFERENCE. Acesso em: 17 de Novembro de 2018 às 02:19. Disponível em: <https://en.cppreference.com/w/cpp/language/access>

Constructors and member initializer lists. Belo Horizonte, 15 de Junho de 2018. Acesso em: 04

de Novembro de 2018 às 10:30. Disponível em: <https://en.cppreference.com/w/cpp/language/initializer_list>

LearnCPP. Disponível em: <http://www.learncpp.com/cpp-tutorial/8-5a-constructor-member-initializer-lists/>

Slides do Prof. Douglas Macharet. Disponível em: <*https://virtual.ufmg.br/20182/course/view.php?id=1008*>

Catch 2 and C++ the Community. Belo Horizonte, 14 de Dezembro de 2016. Acesso em: 11 de Novembro de 2018 às 04:30. Disponível em:

<http://cppcast.com/2016/12/phil-nash/>

F)Apêndice - Cartões CRC

Classe: Pessoa	
Responsabilidades: ● Nome ● ID ● Senha	**Colaborações:** ● Item ● Usuário ● Funcionário

Classe: Usuário	**Superclasse:** Pessoa
Responsabilidades: ● Reservar itens ● Alugar itens ● Devolver itens ● Procurar itens	**Colaborações:** ● Pessoa ● Funcionário ● Item ● Livro ● Multimídia

Classe: Funcionário	**Superclasse:** Pessoa
Responsabilidades: ● Adicionar itens ● Remover itens ● Cadastrar usuários	**Colaborações:** ● Pessoa ● Usuário ● Item ● Livro ● Multimídia

Classe: Item	
Responsabilidades: ● Título ● Autor ● Editora	**Colaborações:** ● Pessoa ● Usuário ● Funcionário ● Livro ● Multimídia

Classe: Livro	**Superclasse:** Item
Responsabilidades: ● Disponibilidade	**Colaborações:** ● Item ● Usuário ● Funcionário

Classe: Multimídia	**Superclasse:** Item
Responsabilidades: ● Tipo de multimídia ● Disponibilidade	**Colaborações:** ● Item ● Usuário ● Funcionário

Classe: Gerência	
Responsabilidades: ● Armazena usuários cadastrados ● Armazena funcionários cadastrados ● Pesquisa pessoas cadastradas	**Colaborações:** ● Item ● Pessoa ● Acervo

Classe: Acervo	
Responsabilidades: ● Armazena itens ● Conta estoque ● Pesquisa por itens	**Colaborações:** ● Item ● Livro ● Multimídia ● Gerência

Classe: Biblioteca	
Responsabilidades: ● Interface do sistema ● Menus e opções ● Cria objetos	**Colaborações:** ● Item ● Pessoa ● Acervo ● Gerência