UNIVERSIDADE ESTADUAL DE CAMPINAS

FACULDADE DE TECNOLOGIA

**RELATÓRIO**

ESTRUTURAS DE ARQUIVOS – ST562 B

Alunos: Caroline Resende Silveira (165921)

Mirelle Candida Bueno (174909)

Otavio Passarelli Praça (175390)

Vitor Artoni de Marcio (178379)

Professor: Dr. Celmar Guimarães da Silva

Novembro de 2016

Este documento contém o trabalho da disciplina Estrutura de Arquivos, orientado pelo Prof. Celmar Guimarães da Silva com entrega no dia 18 de novembro de 2016.

O objetivo do projeto é implementar um sistema de gerenciamento de arquivo de registros, visando treinar os conceitos aprendidos sobre gerência de arquivos de registros e indexação baseada em listas simples.

**Sumário**

[Resumo 4](#_bookmark0)

[Objetivos 5](#_bookmark1)

[Relatório Final 6](#_bookmark2)

[Plano de Trabalho e Cronograma de Execução 11](#_bookmark3)

[Referências 13](#_bookmark4)

# Diagrama UML

**Struct Registro:** Struct responsável pela implementação dos campos :ISBN, Titulo, Autor1, Autor2, Autor3, Ano;

**Struct noChaveP:** Struct responsável pela implementação dos campos :ISBN, Ativo, Rrn, Proximo;

**Struct noChaveS:** Struct responsável pela implementação dos campos :Chave, Primeiro, Ultimo, Proximo;

**Struct noListaInv:** Struct responsável pela implementação dos campos :Cp, Proximo.

# Instruções de compilação e execução

* Foi utilizado a linguagem C++;
* O Desenvolvimento do projeto ocorreu no Visual Studio 2015(Microsoft);
* A compilação do projeto deve ocorrer diretamente na plataforma ou em algum ambiente de desenvolvimento semelhante;
* O arquivo “dados.txt” deve estar na mesma pasta na qual está o código implementado;

# Relatório detalhado

1. O sistema deve criar índices sobre todos os campos do arquivo de dados **OK**

2. O sistema deve permitir ao usuário consultar registros por chave primária ou por chave secundária (sempre com índice)

3. O sistema deve permitir ao usuário inserir registro, atualizando todos os arquivos adequadamente

4. O sistema deve permitir ao usuário remover registro (dada sua chave primária), atualizando todos os arquivos adequadamente

5. O sistema deve permitir ao usuário ver os índices utilizados. Estes devem ser impressos na tela em formato de tabelas, e devem ser legíveis ao usuário

6. O sistema deve ser interativo (não necessariamente em tela gráfica), permitindo ao usuário informar os nomes ou valores de campos conforme as opções disponibilizadas **OK**

7. Índices devem ser implementados como listas, e em arquivo **OK\*\* → ver 1.**

8. Operações de consulta devem ser feitas diretamente em arquivo (não vale trazer o arquivo inteiro para a RAM para então fazer consulta) **OK\*\* → ver 2.**

9. Buscas feitas nos índices devem ser buscas binárias **OK\*\* → ver 2.**

10. O sistema deve usar listas invertidas de chaves primárias como apoio à implementação de índices secundários **OK\*\* → ver 1.**

11. Operações de consulta devem exibir tempo gasto **OK**

12. O sistema também deve funcionar corretamente com outros arquivos que não o exemplificado neste trabalho, mas que tenham registro de cabeçalho com nomes de registros, em formato semelhante a este. **OK**

# Referências

[1] Folk, M. J. Zoellick, B. File Structures. 2ª ed. Addison-Wesley, 1991.

[2] Neto, J. P. Programação, Algoritmos e Estruturas de Dados. 3ª ed. Escolar, 2014.

[3] Deitel, H. C++ Como Programar. 5ª ed. Pearson, 2006.

[4] Mizrahi, V. V. Treinamento em linguagem C. 2ª ed. Prentice Hall, 2008.

[5] Edicezar, L. N. Estudo de sistemas de arquivos estruturados em IOG e uma proposta de implementação para o ambiente LINUX. UNICAMP, 1998. Disponível em: http://www.bibliotecadigital.unicamp.br/document/?code=000129878 (Acesso em 11 de novembro de 2016)