



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

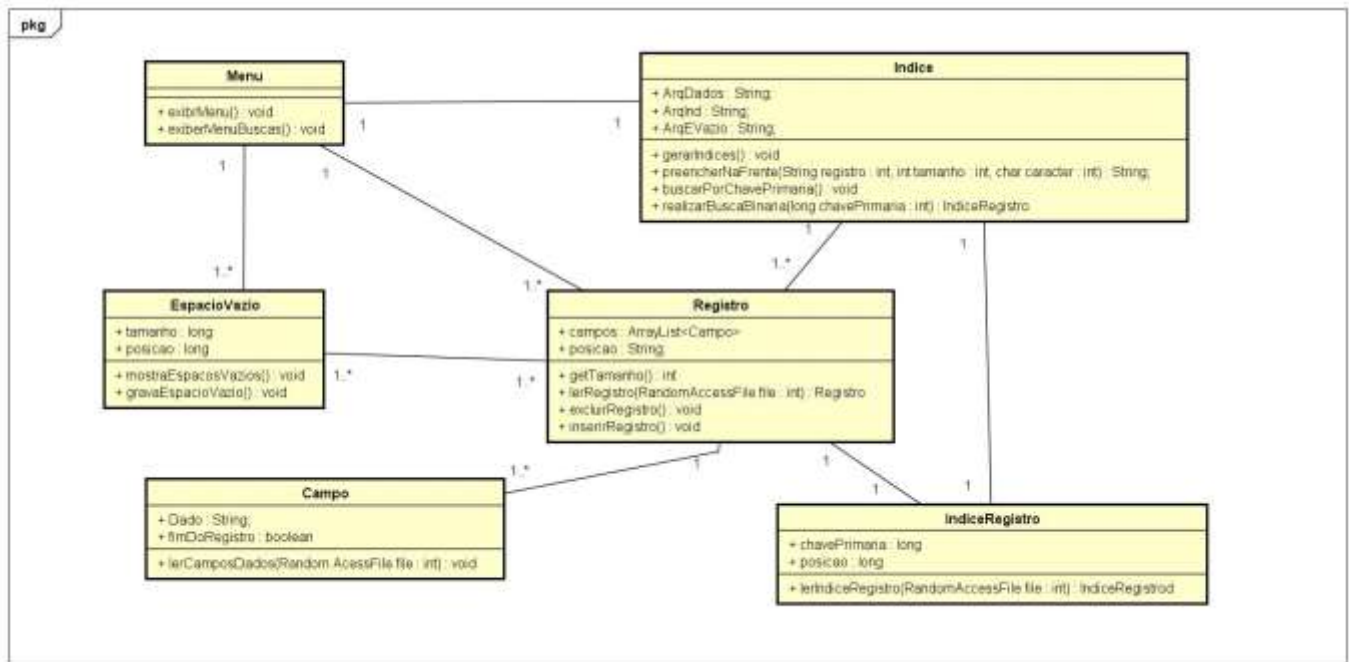
Este documento contém o trabalho da disciplina Estrutura de Arquivos, orientado pelo Prof. Celmar Guimarães da Silva com entrega no dia 21 de novembro de 2016.

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

Diagrama UML.....	4
Instruções de compilação e execução	5
Relatório detalhado	6
Referências	14

Diagrama UML



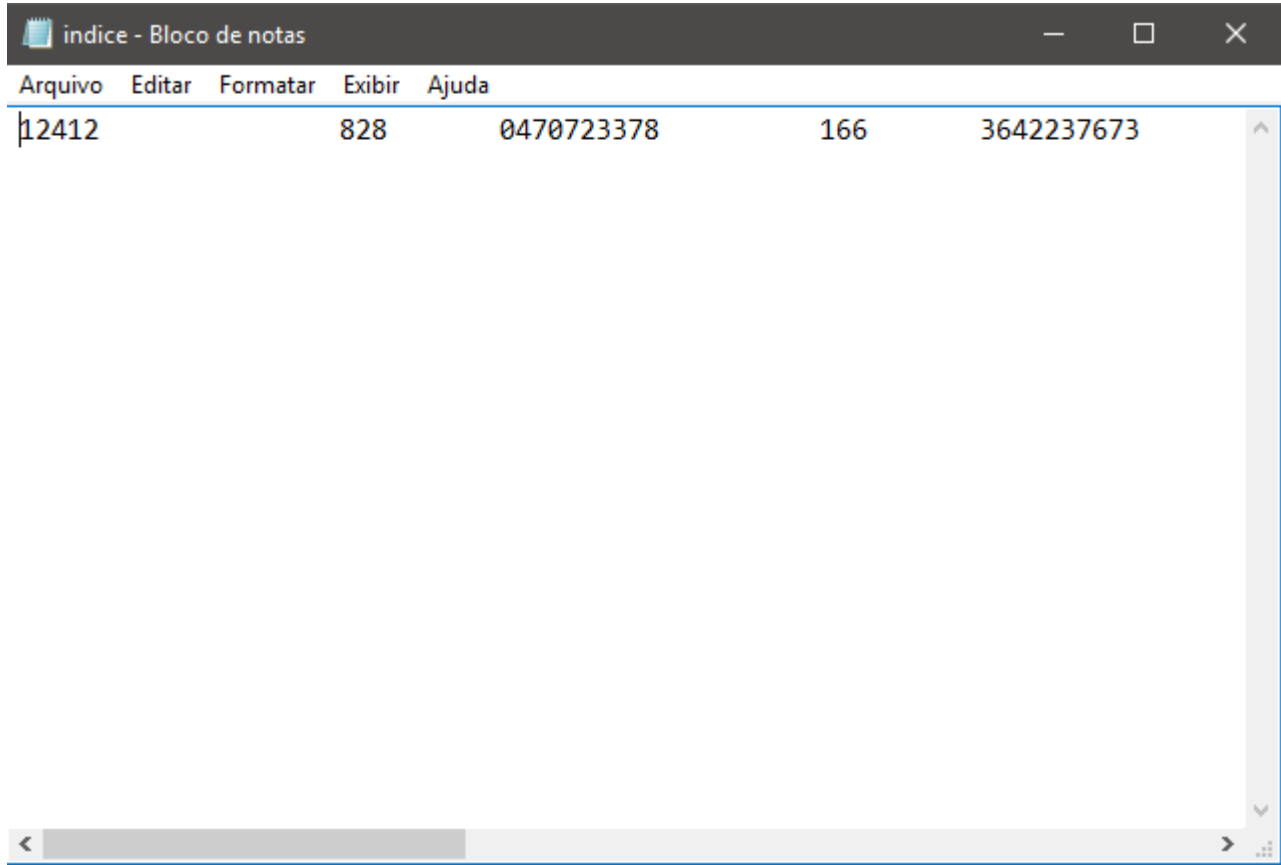
No projeto há a classe Menu que será responsável na escolha das opções que o usuário fizer. Já a classe indice tem a função de gerar um arquivo de indices, buscar chaves primárias e realizar busca binária . A classe registro tem a função de ler os registros de um arquivo de dados, inserir registros e excluir, sendo possível além disso obter os campos e a posição de um determinado registro. A classe espaço vazio tem a função de mostrar os espaços vazios e grava-los , para isso é necessário saber o tamanho e a posição do registro. A classe Campo tem a função de ler os campos de dados de um registro e encontrar o final dos registros. A classe Indice Registro, a partir da chave primária e da posição de um registro é capaz de ler o indice do registro.

Instruções de compilação e execução

- Foi utilizado a linguagem Java.
- O Desenvolvimento do projeto ocorreu no NetBeans (Oracle).
- 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



Índice sobre a chave primária (ISBN) gerado pelo código



Código para gerar índices.

```

3  import java.io.IOException;
4  import java.util.Scanner;
5
6  public class Menu {
7
8      public static void exibirMenu() throws IOException {
9
10         Indice.gerarIndices();
11         EspacoVazio.gravaEspacoVazio();
12
13         int escolha = 0;
14
15         while (escolha != 5) {
16             System.out.println("-----");
17             System.out.println("1-Inserir um registro");
18             System.out.println("2-Excluir um registro");
19             System.out.println("3-Buscar um registro");
20             System.out.println("4-Listar índices");
21             System.out.println("5-Sair");
22             System.out.println("-----");

```

Código para gerar índices via “Indice.gerarIndices()”

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

```

1-Inserir um registro
2-Excluir um registro
3-Buscar um registro
4-Listar índices
5-Sair

```

Menu de opções -> Buscar um registro

```

Saida - Gerenciador de Arquivos (run)
$
1-Buscar por chave primária
2-Voltar
1
Digite a Chave Primária:
978562832043

```

Opção “Buscar por chave primária”

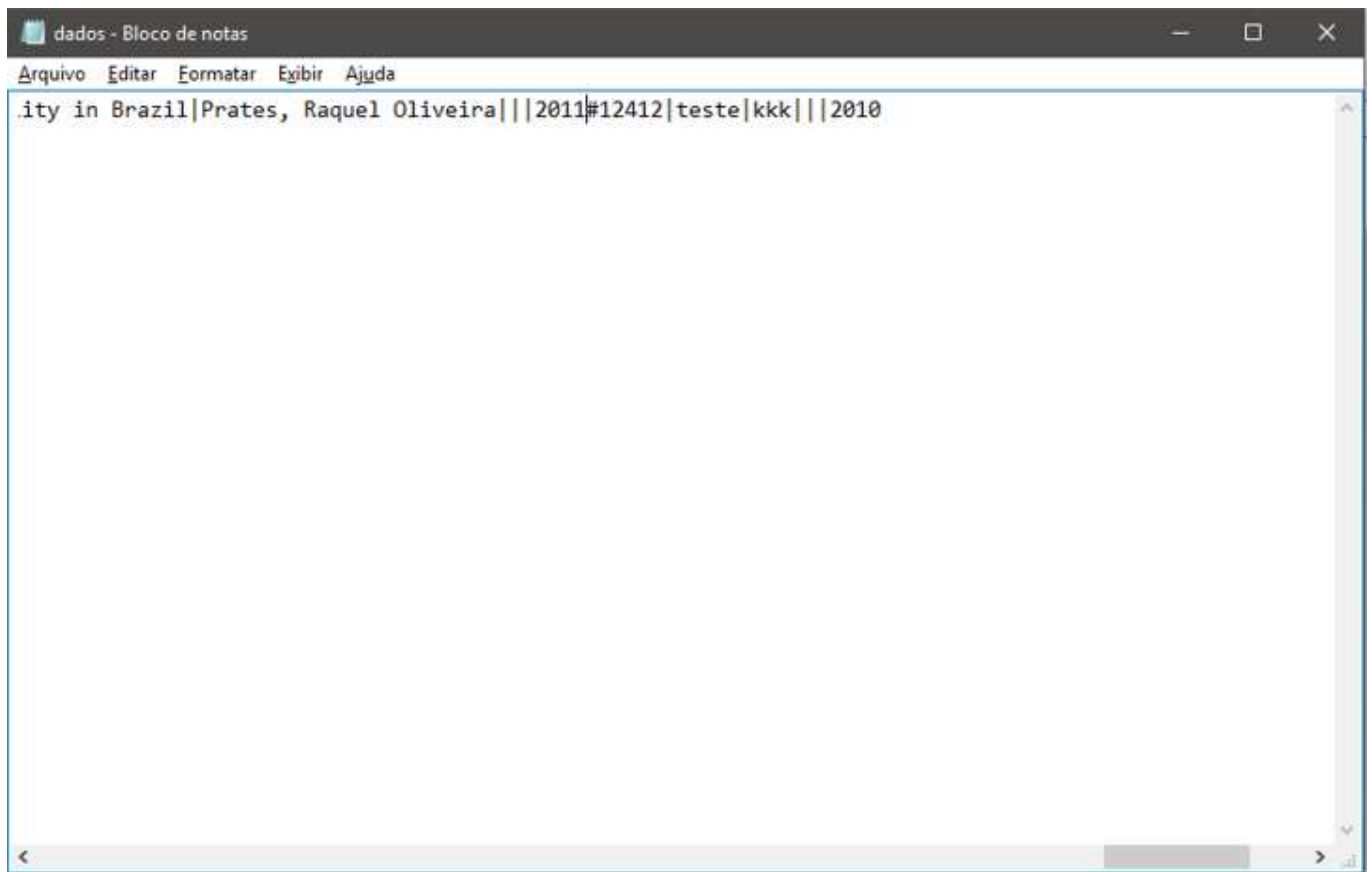
```

978562832043
ISBN: 978562832043
Titulo: Design e avaliação de interfaces humano-computador
Autor1: Rocha, Heloisa Vieira da
Autor2: Serebrennikov, Maria Cecilia Calani
Autor3:
Ano: 2008
Tempo gasto para buscar por chave primária: 72,131 ms

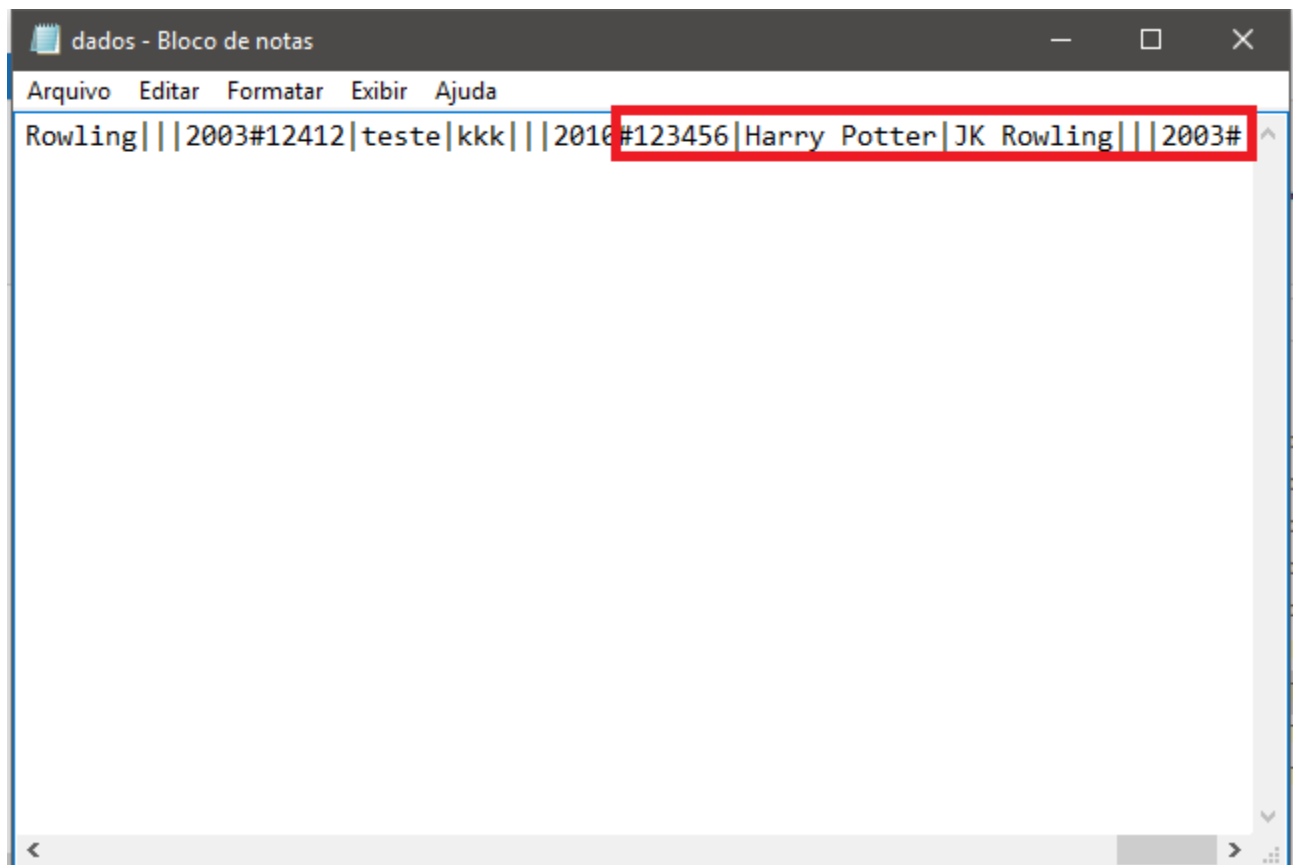
```

Resultado da busca por chave primária via índice

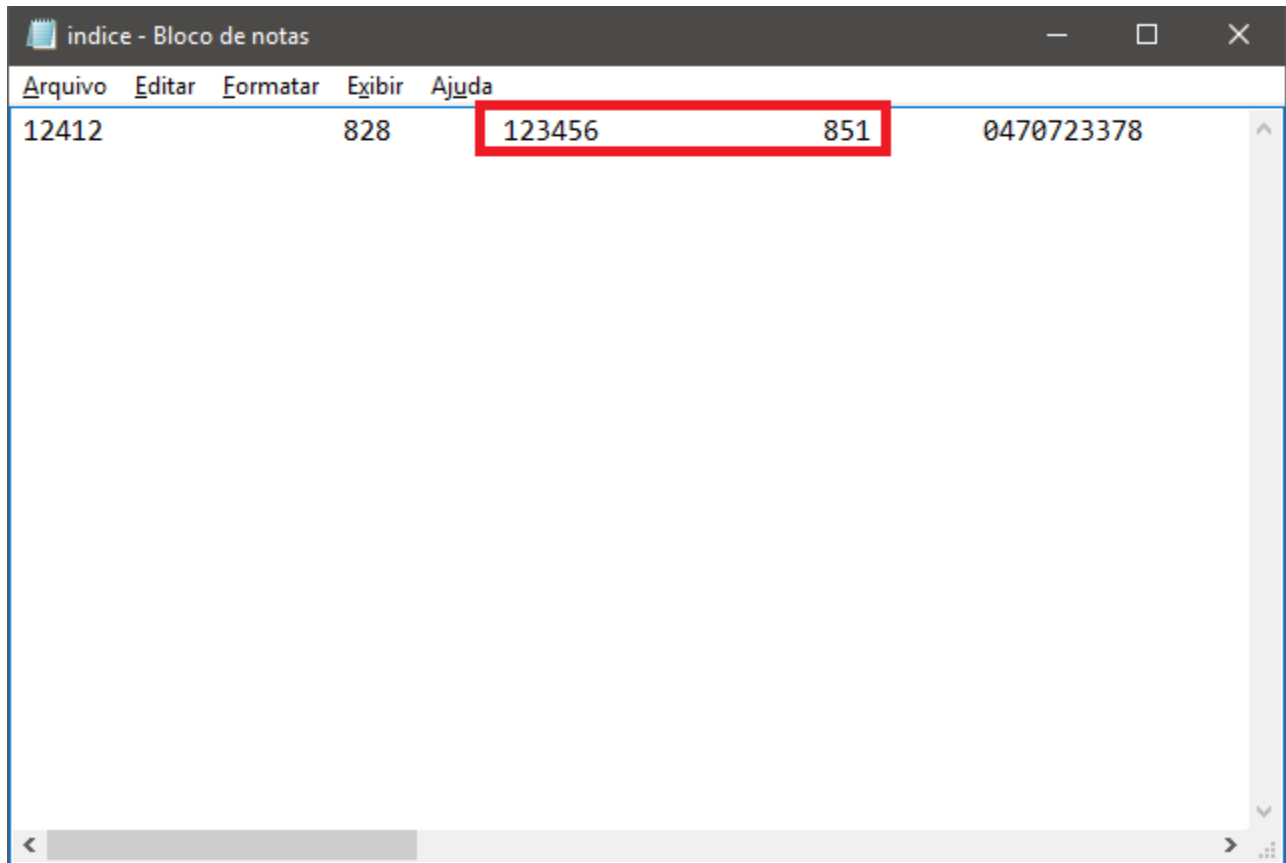
- O sistema deve permitir ao usuário inserir registro, atualizando todos os arquivos adequadamente



Arquivo “dados.txt” antes da nova inserção.



Arquivo “dados.txt” com novo registro inserido.



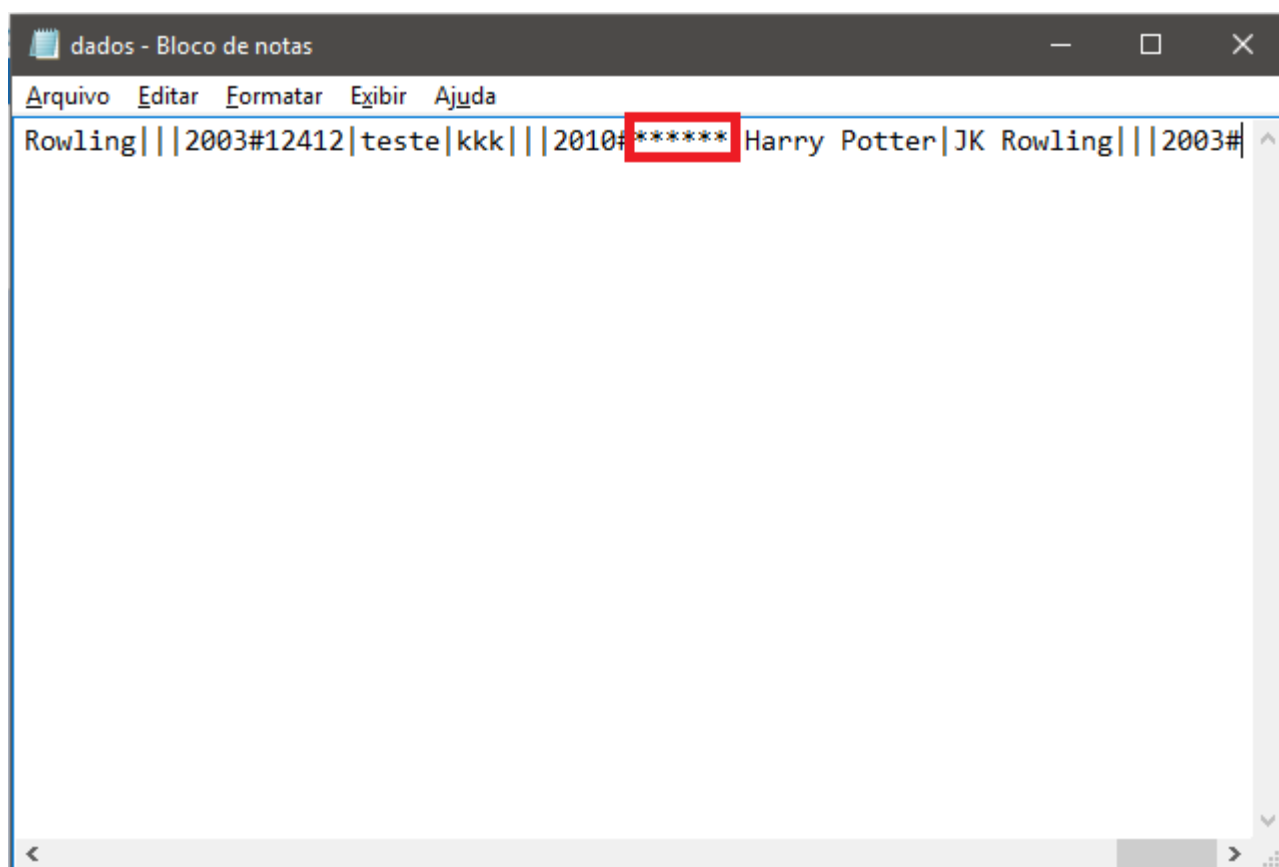
12412	828	123456	851	0470723378
-------	-----	--------	-----	------------

Arquivo de índice utilizado para inserção do registro.

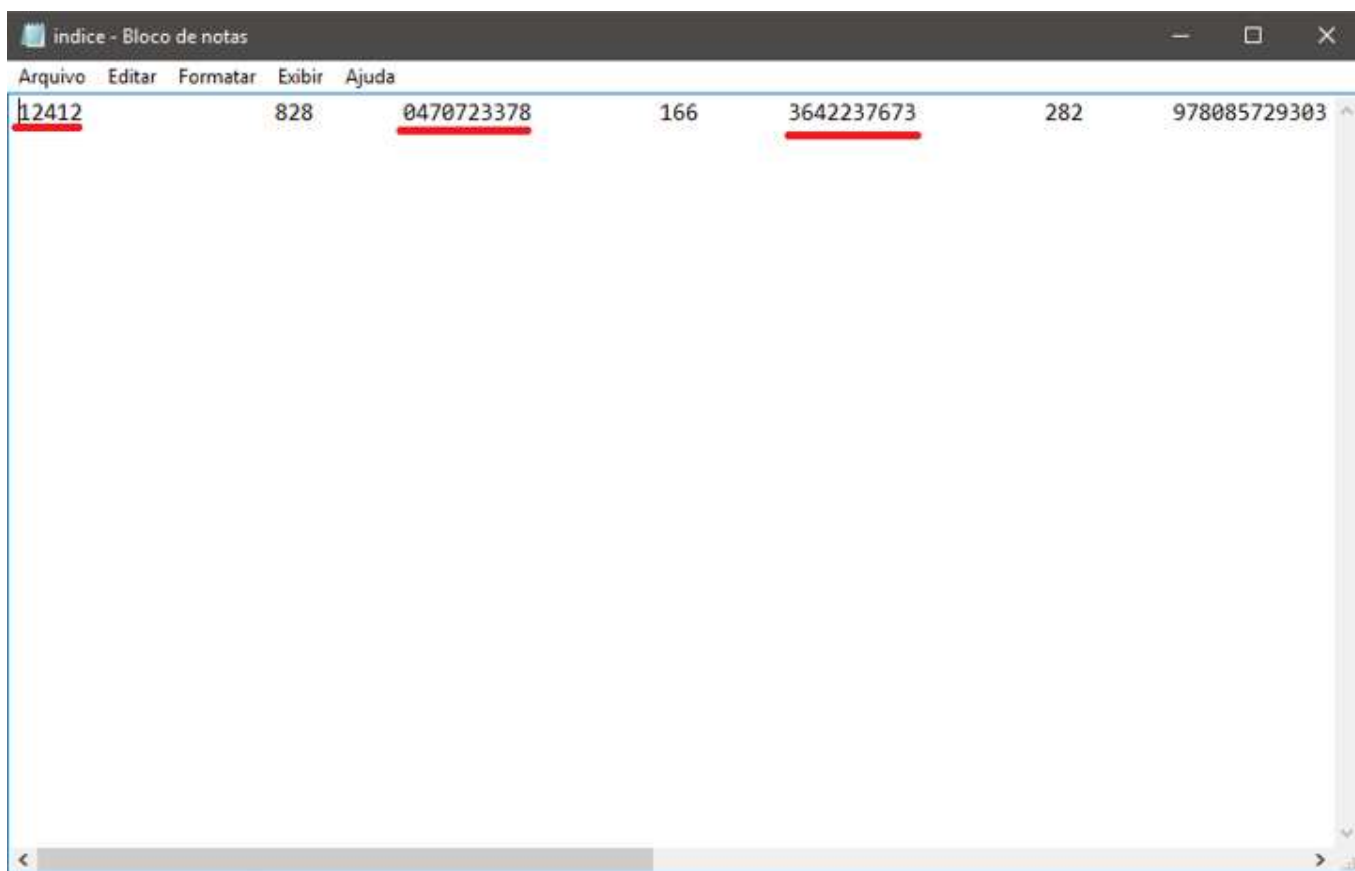
4. O sistema deve permitir ao usuário remover registro (dada sua chave primária), atualizando todos os arquivos adequadamente

```
1-Inserir um registro
2-Excluir um registro
3-Buscar um registro
4-Listar Índices
5-Sair
-----
2
Digite a Chave Primária do registro que deseja Excluir:
|
```

Menu de opções -> Excluir um registro



Registro foi excluido



Arquivo de índices foi atualizado.

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

```
-----
1-Inserir um registro
2-Excluir um registro
3-Buscar um registro
4-Listar Índices
5-Sair
-----
4
ChavePrimaria-----Posicao
12412                      4828
0470723378                4166
3642237673                4282
9780857293039             4728
9781598299441             4598
9783642217074             4442
9788588833043             437
Tempo gasto para exibir indices: 0,003 ms
```

Exibição dos índices utilizados

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

```
-----
1-Inserir um registro
2-Excluir um registro
3-Buscar um registro
4-Listar Índices
5-Sair
-----
```

Menu com as opções disponibilizadas

7. Índices devem ser implementados como listas, e em arquivo
8. Operações de consulta devem ser feitas diretamente em arquivo (não vale trazer o arquivo inteiro para a RAM para então fazer consulta)

```

public static void buscarPorChavePrimaria() throws IOException {
    System.out.println("Digite a Chave Primária: ");

    Scanner scan = new Scanner(System.in);
    long chavePri = scan.nextLong();
    IndiceRegistro reg = realizarBuscaBinaria(chavePri);

    if (reg == null) {
        System.out.println("Registro não encontrado!");
        return;
    }

    File f = new File(ArqDados);

    RandomAccessFile arq = new RandomAccessFile(f, "rw");
    arq.seek(0);

    Registro registro = Registro.lerRegistro(arq);
    arq.seek(reg.posicao);

    Registro novo = Registro.lerRegistro(arq);

    for (int i = 0; i < novo.campos.size(); i++) {
        System.out.println(registro.campos.get(i).dado + ":" + novo.campos.get(i).dado);
    }
    arq.close();
}

```

Operações realizadas diretamente em Disco.

9. Buscas feitas nos índices devem ser buscas binárias

```

public static IndiceRegistro realizarBuscaBinaria(long chavePrimaria) throws IOException {
    File arq = new File(ArqInd);
    RandomAccessFile file = new RandomAccessFile(arq, "rw");
    file.seek(0);

    long totalRegistros = arq.length() / 30;
    long inicio = 0;
    long fim = totalRegistros - 1;

    IndiceRegistro RegAchou = null;

    while (true) {
        if (fim - inicio == 1) {
            file.seek(inicio * 30);
            IndiceRegistro reg = IndiceRegistro.lerIndiceRegistro(file);

            if (reg.chavePrimaria == chavePrimaria) {
                RegAchou = reg;
                break;
            }

            file.seek(fim * 30);
            reg = IndiceRegistro.lerIndiceRegistro(file);

            if (reg.chavePrimaria == chavePrimaria) {
                RegAchou = reg;
            }
        }
    }
}

```

```

        break;
    }
    break;
}

long meio = ((fim - inicio) / 2) + inicio;
file.seek(meio * 30);
IndiceRegistro reg = IndiceRegistro.lerIndiceRegistro(file);

if (reg.chavePrimaria > chavePrimaria) {
    fim = meio;
} else if (reg.chavePrimaria < chavePrimaria) {
    inicio = meio;
} else {
    RegAchou = reg;
    break;
}
}
file.close();
return RegAchou;
}
}

```

Busca Binária

10. O sistema deve usar listas invertidas de chaves primárias como apoio à implementação de índices secundários

11. Operações de consulta devem exibir tempo gasto

```

1
Digite a Chave Primária:
9788588833043
ISBN:9788588833043
Titulo:Design e avaliação de interfaces humano-computador
Autor1:Rocha, Heloisa Vieira da
Autor2:Baranauskas, Maria Cecilia Calani
Autor3:
ano:2003
Tempo gasto para buscar por chave primária: 33,266 ms

```

Operação de Consulta e seu respectivo tempo gasto.

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.

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)