

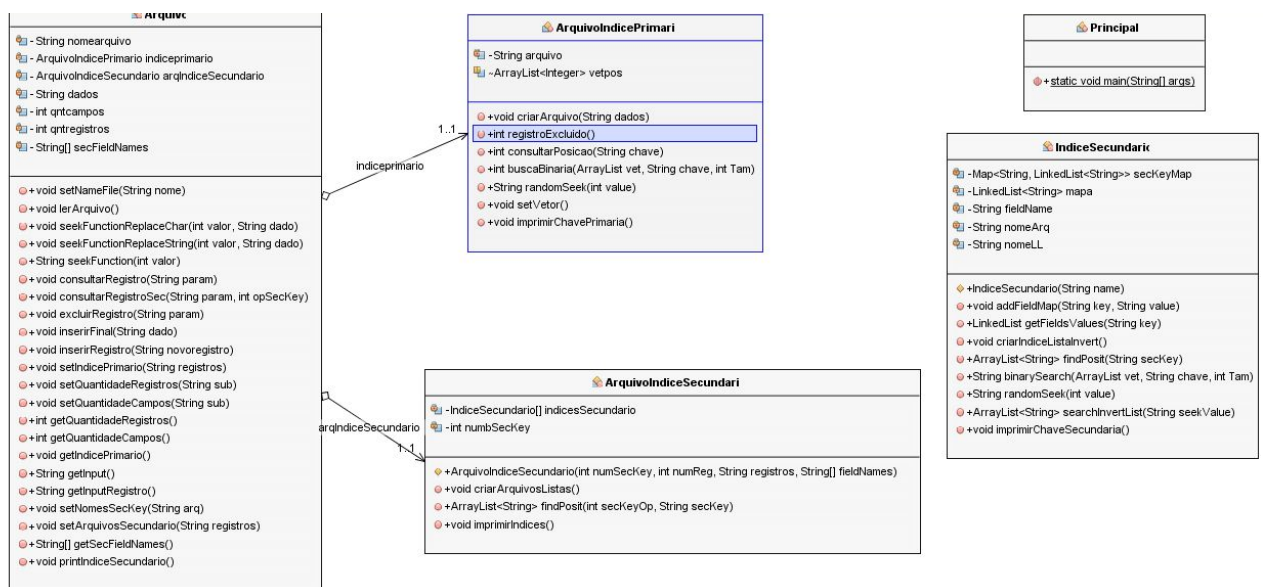
Trabalho Estruturas de Arquivos - ST562-A

Integrantes:

Alexandre Martins Montebelo	RA 157645
Antonio Alberto Pereira Junior	RA 157693
Brandon Caires Nelsen	RA 154810
Giovana Lodde Girardi	RA 155553
Pedro Henrique Bianchi	RA 185663

1. Diagrama UML sobre as classes implementadas e explicações sobre cada uma;

Diagrama:



- Classe Arquivo:

Atributos

```
private String nomearquivo = "";  
private final ArquivoIndicePrimario indiceprimario = new ArquivoIndicePrimario();  
private String dados = "";  
private int qntcampos = 0;  
private int qntregistros = 0;
```

Métodos

Modifica o nome do arquivo.

```
public void setNameFile(String nome) {...}
```

Método responsável por ler o arquivo.

```
public void lerArquivo(){...}
```

Método que troca o primeiro caractere do registro quando é excluído.

```
public void seekFunctionReplaceChar(int valor, String dado) {...}
```

Método que troca o registro quando é inserido em um registro excluído.

```
public void seekFunctionReplaceString(int valor, String dado){...}
```

Método que implementa a função seek.

```
public String seekFunction(int valor) {...}
```

Método de consultar o registro passando chave primária como parâmetro.

```
public void consultarRegistro(String param) {...}
```

Método de consultar o registro passando chave secundária como parâmetro e a opção de qual chave secundária utilizar.

```
public void consultarRegistroSec(String param, int opSecKey) {...}
```

Método que exclui o registro dado sua chave primária.

```
public void excluirRegistro(String param) {...}
```

Método que insere no final do registro.

```
public void inserirFinal(String dado){...}
```

Método que implementa a função de inserção passando como parâmetro os valores do registro.

```
public void inserirRegistro(String novoregistro) {...}
```

Método que modifica o arquivo de índice

```
public void setIndicePrimario(String registros){...}
```

Métodos get e set referente ao arquivo de dados.

```
public void setQuantidadeRegistros(String sub){...}
```

```
public void setQuantidadeCampos(String sub){...}
```

```
public int getQuantidadeRegistros(){...}
```

```
public int getQuantidadeCampos(){...}
```

Método que invoca a impressão do arquivo de índice primário.

```
public void getIndicePrimario() throws IOException {...}
```

Método que auxilia na criação dos índices secundários, pegando o nome dos campos dos índices secundário no cabeçalho do arquivo.

```
public void setNomesSecKey(String arq){...}
```

Método que prepara e dispara a ação de criar os arquivos de índices secundários e suas lista invertidas.

```
public void setArquivosSecundario(String registros){...}
```

Método que imprime os arquivos de índice secundários.

```
public void printIndiceSecundario() throws IOException{...}
```

- Classe ArquivoIndicePrimario:

Atributos

```
private final String arquivo = "dados/indicePrimario.txt";
```

```
File file = new File(arquivo);
```

```
ArrayList<Integer> vetpos = new ArrayList<>();
```

Métodos

Método responsável por criar o arquivo de índice primário;

```
public void criarArquivo(String dados) {...}
```

Método responsável por retornar a posição do registro excluído;

```
public int registroExcluido() {...}
```

Método responsável por retornar a posição do registro usando o método buscaBinaria.

```
public int consultarPosicao(String chave) {...}
```

Método que implementa a busca binária.

```
public int buscaBinaria(ArrayList vet, String chave, int Tam) {...}
```

Método de seek que auxilia a busca binária.

```
public String randomSeek(int value) {...}
```

Método que imprime na tela as chaves primária dos registros contidas no arquivo.

```
public void imprimirChavePrimaria(){...}
```

- Classe ArquivoIndiceSecundario:

Atributos

```
private IndiceSecundario[] indicesSecundario;
```

```
private int numbSecKey, numbReg;
```

Construtor da classe, que prepara a classe para a criação dos arquivos de Índices Secundários e suas listas invertidas.

```
public ArquivoIndiceSecundario(int numSecKey,int numReg , String registros, String[] fieldNames){...}
```

Métodos

Método que invoca a criação de cada índice secundário e as listas invertidas através da classe IndiceSecundario.

```
public void criarArquivosListas(){...}
```

Método que retorna a lista das chaves primárias encontradas na busca através da chave secundária.

```
public ArrayList<String> findPosit(int secKeyOp, String secKey){...}
```

Método que invoca a impressão das chaves secundárias através da classe IndicesSecundario

```
public void imprimirIndices() throws IOException{...}
```

- Classe IndiceSecundario:

Atributos

```
private Map<String, LinkedList<String>> secKeyMap = new LinkedHashMap<String,
LinkedList<String>>();
private LinkedList<String> mapa = new LinkedList<String>();
private String fieldName;
private String nomeArq = "dados/indiceSecundario_";
private String nomeLL = "dados/listaLigada_";
```

Métodos

Método que define o nome do campo que o IndiceSecundario irá gerenciar, com índice e lista invertida.

```
public IndiceSecundario(String name){...}
```

Método que auxilia na criação arquivo de Índice Secundário e na criação da lista invertida (Só é utilizado para criar os arquivo, não para consulta).

```
public void addFieldMap (String key, String value){...}
```

Método que cria os arquivos de Índice Secundário e suas listas invertidas.

```
public void criarIndiceListaInvert(){...}
```

Método que realiza uma busca binária no Índice Secundário através da chave secundária, retornando a chave primária encontrada.

```
public String binarySearch(ArrayList vet, String chave, int Tam) throws IOException {
```

Método que retorna a lista de chaves primária encontrada no arquivo de lista invertida referente a busca realizada com chave secundária, utilizando o binarySearch para encontrar a chave primária a ser buscada na lista invertida.

```
public ArrayList<String> findPosit(String secKey){...}
```

Método auxiliar da busca binária, ele faz uma leitura no arquivo na posição passada e retorna o registro na posição.

```
public String randomSeek(int value) throws IOException {...}
```

Método que faz a busca na lista invertida, utilizando uma chave primária passada, e retorna uma lista com as chaves primárias encontradas.

```
public ArrayList<String> searchInvertList (String seekValue){...}
```

Método que imprime na tela a chave secundária.

```
public void imprimirChaveSecundaria() throws IOException {...}
```

2. Instruções de como o sistema deve ser compilado e executado;

Baixe o SistemaConsulta.zip no seu computador, na pasta desejada. Extraia o arquivo. Na pasta nomeada “dados” insira os arquivos desejados para realizar a consulta. Inicialize o NetBeans, vá em File -> Open Project (também pode ser acessado pelo atalho Ctrl+Shift+O) e selecione, na pasta em que o projeto está, a pasta SistemaConsulta. Selecione o projeto (com o ícone do Java), clique em “Open Project”.

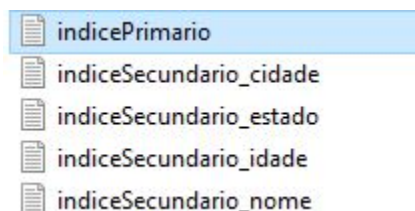
Clique no ícone de play verde (Run Project) ou, pelo atalho, F6. O projeto irá inicializar no terminal do NetBeans.

Outra opção é executar via terminal o arquivo “SistemaConsulta.jar”, um Executable Jar File, que se encontra na pasta dist do nosso projeto. Porém, atente-se que deve ser executado via terminal pois o não há uma GUI implementada no sistema, logo funciona somente no terminal.

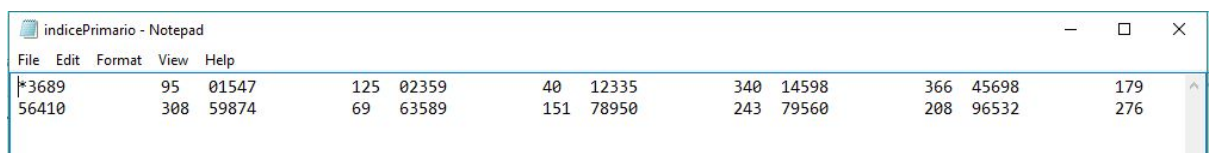
3. Provas de que o sistema atende às especificações.

Todos as capturas de telas foram feitas executando o sistema com o arquivo “arquivo1.txt” da pasta “Dados”

• **R1: O sistema deve criar índices sobre todos os campos do arquivo de dados.**

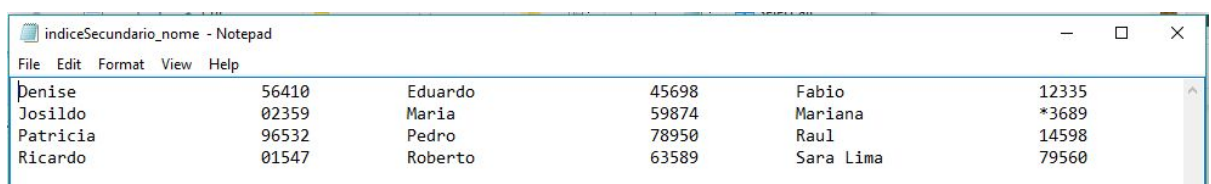


arquivos de índices criados



3689	95	01547	125	02359	40	12335	340	14598	366	45698	179
56410	308	59874	69	63589	151	78950	243	79560	208	96532	276

arquivo de índices primários



Denise	56410	Eduardo	45698	Fabio	12335
Josildo	02359	Maria	59874	Mariana	*3689
Patricia	96532	Pedro	78950	Raul	14598
Ricardo	01547	Roberto	63589	Sara Lima	79560

arquivo de índices para o campo Nome

indiceSecundario_cidade - Notepad					
File	Edit	Format	View	Help	
Bahia	01547	Belo Horizonte	78950	Campinas	02359
Campo Alegre	56410	Curitiba	45698	Limeira	59874
Maringa	63589	Porto Alegre	79560	Sao Paulo	*3689
Ubatuba	12335	Uberlandia	96532		

arquivo de índices para o campo Cidade

indiceSecundario_estado - Notepad					
File	Edit	Format	View	Help	
BH	01547	MG	78950	MS	56410
PR	63589	RS	79560	SP	02359

arquivo de índices para o campo Estado

indiceSecundario_idade - Notepad					
File	Edit	Format	View	Help	
12	45698	15	79560	21	*3689
23	01547	24	12335	26	78950
30	96532	32	63589	45	59874
49	56410	53	02359	60	14598

arquivo de índices para o campo Idade

- **R2: O sistema deve permitir ao usuário consultar registros por chave primária ou por chave secundária (sempre com índice).**

Chave primária:

```

Sistema de Consulta
1. Consultar
2. Incluir
3. Excluir
4. Visualizar Indices
5. Consultar (Chaves Secundaria)
0. Fim
Opcao: (Escolha um número relacionado ao que deseja executar)
1
Digite chave primaria - Consultar
01547
125

Busca binária fez: 3 seeks
Seek para posição: 125
Registro: 01547|Ricardo|Bahia|BH|23
Tempo de Consulta: 2.0ms

```

Chave secundária

```
Sistema de Consulta
1. Consultar
2. Incluir
3. Excluir
4. Visualizar Indices
5. Consultar (Chaves Secundaria)
0. Fim
Opcao: (Escolha um número relacionado ao que deseja executar)
5
0. nome
1. cidade
2. estado
3. idade
Gostaria de utilizar qual campo para pesquisa da chave secundaria:
Opção: (Escolha o número relacionado a chave que deseja pesquisar)
0
Digite chave secundaria - Consultar
Denise

Busca binária da chave secundaria fez: 3 seeks
308

Busca binária fez: 1 seeks
Seek para posição: 308
Registro: 56410|Denise|Campo Alegre|MS|49
Tempo de Consulta: 3.0ms
```

Sistema de Consulta

```
1. Consultar
2. Incluir
3. Excluir
4. Visualizar Indices
5. Consultar (Chaves Secundaria)
0. Fim
Opcao: (Escolha um número relacionado ao que deseja executar)
5
0. nome
1. cidade
2. estado
3. idade
Gostaria de utilizar qual campo para pesquisa da chave secundaria:
Opção: (Escolha o número relacionado a chave que deseja pesquisar)
2
Digite chave secundaria - Consultar
SP

Busca binária da chave secundaria fez: 2 seeks
40

Busca binária fez: 2 seeks
Seek para posição: 40
Registro: 02359|Josildo|Campinas|SP|53
Tempo de Consulta: 5.0ms
-----
69

Busca binária fez: 3 seeks
Seek para posição: 69
Registro: 59874|Maria|Limeira|SP|45
Tempo de Consulta: 9.0ms
```

Consulta por chave secundária com repetição no arquivo

- R3: O sistema deve permitir ao usuário inserir registro, atualizando todos os arquivos adequadamente.

```
Sistema de Consulta
1. Consultar
2. Incluir
3. Excluir
4. Visualizar Indices
5. Consultar (Chaves Secundaria)
0. Fim
Opcao: (Escolha um número relacioando ao que deseja executar)
2
Inserir campo 0
14598
Inserir campo 1
Raul
Inserir campo 2
Campinas
Inserir campo 3
SP
Inserir campo 4
60
Posição do registro Excluido: 0
368
Tempo de Inserção 10.0ms
```

- R4: O sistema deve permitir ao usuário remover registro (dada sua chave primária), atualizando todos os arquivos adequadamente.

```
Sistema de Consulta
1. Consultar
2. Incluir
3. Excluir
4. Visualizar Indices
5. Consultar (Chaves Secundaria)
0. Fim
Opcao: (Escolha um número relacioando ao que deseja executar)
3
Digite chave primaria - Excluir
13689
95

Busca binária fez: 4 seeks
Seek para posição: 95
*3689|Mariana|Sao Paulo|SP|21
Tempo de Exclusão: 12.0ms
```

- R5: 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.

```

Sistema de Consulta
1. Consultar
2. Incluir
3. Excluir
4. Visualizar Indices
5. Consultar (Chaves Secundaria)
0. Fim
Opcao: (Escolha um número relacionado ao que deseja executar)
4

```

Imprimir Indices

Chave Primária	Posição
01547	125
02359	40
12335	340
13689	95
14598	366
45698	179
56410	308
59874	69
63589	151
78950	243
79560	208
96532	276

Chave Secundaria	Chave InvList
Denise	56410
Eduardo	45698
Fabio	12335
Josildo	02359
Maria	59874
Mariana	13689
Patricia	96532
Pedro	78950
Raul	14598
Ricardo	01547
Roberto	63589
Sara Lima	79560

Chave Secundaria	Chave InvList
Bahia	01547
Belo Horizonte	78950
Campinas	02359
Campo Alegre	56410
Curitiba	45698
Limeira	59874
Maringa	63589
Porto Alegre	79560
Sao Paulo	13689
Ubatuba	12335
Uberlandia	96532

	Chave Secundaria	Chave InvList	
BH		01547	
MG		78950	
MS		56410	
PR		63589	
RS		79560	
SP		02359	

	Chave Secundaria	Chave InvList	
12		45698	
15		79560	
21		13689	
23		01547	
24		12335	
26		78950	
30		96532	
32		63589	
45		59874	
49		56410	
53		02359	
60		14598	

• R6: O sistema deve ser interativo (não necessariamente em tela gráfica), permitindo ao usuário informar os nomes ou valores de campos conforme as opções disponibilizadas.

Digite o nome do arquivo texto a ser utilizado, sem a extensão:

arquivol

Chave Primária	Posição
01547	125
02359	40
12335	340
13689	95
45698	179
56410	308
59874	69
63589	151
78950	243
79560	208
96532	276

Sistema de Consulta

1. Consultar
 2. Incluir
 3. Excluir
 4. Visualizar Indices
 5. Consultar (Chaves Secundaria)
 0. Fim
- Opcao: (Escolha um número relacionado ao que deseja executar)

tela inicial

Também conta que esse requisito foi atendido com todas as telas das situações passadas.

- R7: Índices devem ser implementados como listas (como visto em aula), e em arquivo.

```

public void setIndicePrimario(String registros) throws FileNotFoundException {
    HashMap<String, Integer> map = new HashMap<>();
    List list = new ArrayList<>();
    int quantfim = getQuantidadeRegistros();
    int cont = 0;
    int vlors = 0;
    String refSize;
    String chave;
    for (int i = 0; i < quantfim; i++) {
        chave = registros.substring(0, (registros.indexOf("|")));
        if (chave.length() < 15) {
            for (int p = chave.length(); p < 15; p++) {
                chave = chave.concat(" ");
            }
        } else if (chave.length() > 15) {
            chave = chave.substring(0, 15);
        }
        //String nome = registros.substring((registros.indexOf("|") + 1), (registros.indexOf("|") + 1));
        //nome = nome.substring(0, 5);
        //chave = chave.concat(nome);

        String reg = registros.substring(0, registros.indexOf('#'));
        if (i == 0) {
            cont = 0;
            vlors = 0;
        }

        registros = registros.substring(reg.length() + 1, registros.length());
        map.put(chave, vlors);
        list.add(chave);
        vlors += reg.getBytes().length + 1;
        cont = cont + (reg.length() + 1);
    }
    Collections.sort(list);
    int ultimo = list.size();
    list.remove(ultimo - 1);
    Iterator itr = list.iterator();
    while (itr.hasNext()) {
        Object element = itr.next();
        refSize = map.get(element).toString().trim();
        if (refSize.length() < 5) {
            for (int p = refSize.length(); p < 5; p++) {
                refSize = refSize.concat(" ");
            }
        } else if (refSize.length() > 5) {
            refSize = refSize.substring(0, 5);
        }
        //this.dados += element.toString() + "|" + map.get(element).toString() + "#" + "\n";
        this.dados += element.toString() + refSize + "\n";
    }

    this.indiceprimario.criarArquivo(dados);
    this.dados = "";
}

```

O método setIndicePrimario

indicePrimario - Notepad											
File Edit Format View Help											
*3689	95	01547	125	02359	40	12335	340	14598	366	45698	179
56410	308	59874	69	63589	151	78950	243	79560	208	96532	276

Arquivo de índice primário, organizado em ordem crescente

• **R8: Operações de consulta devem ser feitas diretamente em arquivo (não vale trazer o arquivo inteiro para a RAM para então fazer consulta).**

O sistema foi feito todo com consultas diretamente em arquivo, por meio dos arquivos de índices e posições.

• **R9: Buscas feitas nos índices devem ser buscas binárias.**

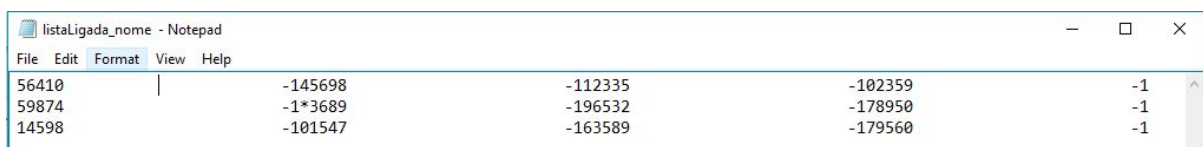
```
public int consultarPosicao(String chave) {  
  
    int tamlinha = 0;  
    int seekvalue = 0;  
    try {  
        try (FileReader arq = new FileReader(arquivo)) {  
            BufferedReader lerArq = new BufferedReader(arq);  
            String linha = lerArq.readLine();  
            vetpos.add(0);  
            while (linha != null) {  
                vetpos.add(tamlinha += linha.length() + 1);  
                linha = lerArq.readLine();  
            }  
            seekvalue = buscaBinaria(vetpos, chave, vetpos.size());  
            arq.close();  
        }  
  
        setVetor();  
    } catch (IOException e) {  
        System.err.printf("Erro na abertura do arquivo.\n",  
            e.getMessage());  
    }  
    return seekvalue;  
}
```

A classe consultarPosicao utiliza a busca binária para retornar a posição do registro no arquivo, nessa classe vetpos seria um vetor temporário para auxiliar na busca binária.

• **R10: O sistema deve usar listas invertidas de chaves primárias como apoio à implementação de índices secundários.**

listaLigada_cidade
listaLigada_estado
listaLigada_idade
listaLigada_nome

Arquivos de listas como apoio dos índices secundários



56410	-145698	-112335	-102359	-1
59874	-1*3689	-196532	-178950	-1
14598	-101547	-163589	-179560	-1

Arquivo de lista do campo Nome

listaLigada_cidade - Notepad						
File	Edit	Format	View	Help		
01547	-178950	-102359	14598	14598	-1	
56410	-145698	-159874		-163589	-1	
79560	-1*3689	-112335		-196532	-1	

Arquivo de lista do campo Cidade

listaLigada_estado - Notepad						
File	Edit	Format	View	Help		
01547	-178950	96532	96532		-1	
56410	-163589	45698	45698		-1	
79560	-102359	59874	59874	*3689		
*3689	12335	12335	14598	14598	-1	

Arquivo de lista do campo Estado

listaLigada_idade - Notepad						
File	Edit	Format	View	Help		
45698	-179560		-1*3689		-1	
01547	-112335		-178950		-1	
96532	-163589		-159874		-1	
56410	-102359		-114598		-1	

Arquivo de lista do campo Idade

```

Sistema de Consulta
1. Consultar
2. Incluir
3. Excluir
4. Visualizar Indices
5. Consultar (Chaves Secundaria)
0. Fim
Opcao: (Escolha um número relacionado ao que deseja executar)
5
0. nome
1. cidade
2. estado
3. idade
Gostaria de utilizar qual campo para pesquisa da chave secundaria:
Opção: (Escolha o número relacionado a chave que deseja pesquisar)
2
Digite chave secundaria - Consultar
SP

Busca binária da chave secundaria fez: 2 seeks
40

Busca binária fez: 2 seeks
Seek para posição: 40
Registro: 02359|Josildo|Campinas|SP|53
Tempo de Consulta: 5.0ms
-----
69

Busca binária fez: 3 seeks
Seek para posição: 69
Registro: 59874|Maria|Limeira|SP|45
Tempo de Consulta: 9.0ms

```

Exemplo de chave secundária com repetição - mostrando a implementação das listas

- **R11: Operações de consulta devem exibir tempo gasto.**

```

Busca binária da chave secundaria fez: 3 seeks
308

Busca binária fez: 1 seeks
Seek para posição: 308
Registro: 56410|Denise|Campo Alegre|MS|49
Tempo de Consulta: 3.0ms

```

- **R12: 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.**

Colocamos 3 arquivos na pasta “Dados” do projeto além do arquivo dado de exemplo, com o mesmo formato de cabeçalho porém diferentes números de registros e tamanho dos campos. Abaixo, o sistema inicializando com todos os arquivos:

Digite o nome do arquivo texto a ser utilizado, sem a extensão:
arquivo1

Chave Primária	Posição
01547	125
02359	40
12335	340
14598	366
45698	179
56410	308
59874	69
63589	151
78950	243
79560	208
96532	276

Digite o nome do arquivo texto a ser utilizado, sem a extensão:
arquivo2

Chave Primária	Posição
11203	335
44560	135
550236	61
77493	274
88503	197

Digite o nome do arquivo texto a ser utilizado, sem a extensão:
arquivo3

Chave Primária	Posição
155183	163
155553	110
157692	36