```
  1: import java.io.BufferedReader;
  2: import java.io.InputStreamReader;
  3: import java.io.FileReader;
  4: import java.io.IOException;
  5: import java.io.FileNotFoundException;
  6: import javax.swing.JOptionPane;
  7:
  8: public class PeopleIndexer {
  9:
 10:    static HumanTree dataCPF;
 11:    static HumanTree dataNome;
 12:    static JOptionPane panel = new JOptionPane();
 13:
 14:    public static void main (String arquivo[]) throws IOException {
 15:
 16:            //Instancia as arvores com um nodo "dummy"
 17:            dataCPF = new HumanTree("0","0","0",'0',0);
 18:            dataNome = new HumanTree("0","0","0",'0',0);
 19:
 20:            System.out.println("");
 21:            System.out.println("PeopleIndexer(Arvores Binarias) - TPV - AEDSII");
 22:            System.out.println("Aluno: Gustavo Campos Ferreira Guimaraes - 2005041291");
 23:            System.out.println("Professor: Roberto Bigonha");
 24:            System.out.println("-----------------------------------------------------");
 25:
 26:            BufferedReader enArquivo = new BufferedReader ( new InputStreamReader ( System.in )
);
 27:            BufferedReader enTeclado = new BufferedReader ( new InputStreamReader ( System.in )
);
 28:            //Recebe as entradas
 29:            try {
 30:                    if (arquivo.length == 0) {
 31:                            System.out.println("Voce deve passar o arquivo de entrada como
parametro");
 32:                            System.exit(0);
 33:                    } else enArquivo = new BufferedReader ( new FileReader(arquivo[0]) );
 34:            }
 35:            catch (FileNotFoundException erro) {
 36:                    System.out.println("Arquivo nao encontrado, confira o nome do arquivo");
 37:                    System.exit(0);
 38:            }
 39:
 40:            boolean endOfFile = false;
 41:            int     regNum = 0;
 42:
 43:            String linha;
 44:            String[] dados = new String[5];
 45:
 46:            do {
 47:                    for (int i = 0; i<5; i++) {
 48:                            try {
 49:                                    linha = enArquivo.readLine();
 50:                                    if ( !linha.equals("endOfFile") ) {
 51:                                            dados[i]=linha;
 52:                                    } else {
 53:                                            endOfFile = true;
 54:                                            break;
 55:                                    }
 56:                            }
 57:                            catch (IOException error) {
 58:                                    System.out.println("Erro"+error);
 59:                            }
 60:                    }
 61:                    if ( !endOfFile ) {
 62:                            dataCPF.insertByCPF ( new HumanTree (
 63:                                            dados[0], dados[1], dados[2], dados[3].charAt(0),
Integer.parseInt(dados[4])
 64:                                    )
 65:                            );
 66:                            dataNome.insertByNome (new HumanTree(
 67:                                            dados[0], dados[1], dados[2], dados[3].charAt(0),
Integer.parseInt(dados[4])
 68:                                    )
 69:                            );
 70:                            regNum++;
 71:
```

```
 72:                 }
 73:             } while (!endOfFile);
 74:             System.out.println("");
 75:             System.out.println("Inseridos "+regNum+" registros");
 76:             System.out.println("");
 77:             while (true) {
 78:                     System.out.println("Escolha uma opcao:");
 79:                     System.out.println("1 - Buscar um registro por CPF");
 80:                     System.out.println("2 - Buscar registros por nome");
 81:                     System.out.println("3 - Apagar um Registro");
 82:                     System.out.println("4 - Visualizar banco ordenado por CPF");
 83:                     System.out.println("5 - Visualizar banco ordenado por Nome");
 84:                     System.out.println("6 - Sair do Programa: ");
 85:                     System.out.println("");
 86:                     int option = 0;
 87:
 88:                     try { option = Integer.parseInt (enTeclado.readLine()); }
 89:                     catch (IOException e) { option = -1; }
 90:                     catch (NumberFormatException e) { option = -1; }
 91:
 92:                     switch (option) {
 93:                             case 1: buscaPorCPF();
 94:                             break;
 95:                             case 2: buscaPorNome();
 96:                             break;
 97:                             case 3: deletaRegistro();
 98:                             break;
 99:                             case 4: dataCPF.caminhaPreOrdem();
100:                             break;
101:                             case 5: dataNome.caminhaPreOrdem();
102:                             break;
103:                             case 6: System.exit(0);
104:                             break;
105:                             case -1: System.out.println("\nOpcao invalida, tente novamente\n");
106:                             break;
107:                             default: System.exit(0);
108:                     }
109:             }
110:     }
111:     static void buscaPorCPF() throws IOException {
112:             String cpf = panel.showInputDialog("Digite o CPF desejado:");
113:             HumanTree result = dataCPF.pesquisaCPF (cpf);
114:             if (result != null) result.imprime();
115:             else System.out.println("\nNenhuma ocorrencia para "+cpf+"\n");
116:     }
117:     static void buscaPorNome() throws IOException {
118:             String nome = panel.showInputDialog("Digite o Nome desejado:");
119:             HumanTree[] result = dataNome.pesquisaNome (nome);
120:             if (result[0] == null) {
121:                     System.out.println ("\nNao foi encontrada nenhuma ocorrencia para
"+nome+"\n");
122:             }
123:             else {
124:                     for (HumanTree x : result) {
125:                             if (x == null) break;
126:                             else x.imprime();
127:                     }
128:             }
129:     }
130:     static void deletaRegistro() throws IOException {
131:             String cpf = panel.showInputDialog("Digite o CPF que voce deseja remover");
132:             //Obtem o nodo a ser removido
133:             HumanTree deletar = dataCPF.pesquisaCPF(cpf);
134:             String nome = deletar.getName();
135:             //Remove o nodo da arvore de cpf
136:             dataCPF.remove(deletar);
137:             //Pesquisa os possiveis equivalentes na arvore de nomes
138:             HumanTree[] candidatos = dataNome.pesquisaNome(nome);
139:             for (int i=0; i< candidatos.length;i++) {
140:             //Remove o equivalente
141:                     if ( candidatos[i].getCPF().equals(cpf) ) dataNome.remove(candidatos[i]);
142:                     break;
143:             }
144:     }
145: }
```