```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.FileReader;
import java.io.IOException;
import java.io.FileNotFoundException;


/** Listener dos RadioButtons, seta o valor de uma variavel dependendo da opcao escolhida */
class RadListener implements ActionListener {

        static String choosen = "CPF";

        public void actionPerformed (ActionEvent evento) {
                if ( evento.getActionCommand().equals("changecpf") ){ choosen = "CPF"; }
                else if ( evento.getActionCommand().equals("changenome") ){ choosen = "Nome"; }
        }
}
public class GPeopleIndexer implements ActionListener {

        static TabelaHash data = new TabelaHash(13);

        static JOptionPane      error;
        static BufferedReader   enArquivo;
        static JTextField       queryText;
        static JTextArea        resultShow;
        static RadListener      radlist;

        public static void main(String[] arquivo ) throws IOException{

                error = new JOptionPane ();

                try {
                        //Verifica se foi passado o parametro com o nome do arquivo de entrada
                        if (arquivo.length == 0) {
                                error.showMessageDialog(null, "Voce deve passar o arquivo de entrada
como parametro",
                                                        "ERRO",
                                                        JOptionPane.ERROR_MESSAGE
                                );
                                System.exit(0);
                        } else enArquivo = new BufferedReader ( new FileReader(arquivo[0]) );
                }
                catch (FileNotFoundException erro) {
                        error.showMessageDialog(null, erro+"\n"+"Arquivo nao encontrado, confira o
nome do arquivo e tente novamente",
                                                        "ERRO",
                                                        JOptionPane.ERROR_MESSAGE
                        );
                }

                boolean endOfFile = false;
                int     regNum = 0;

                String linha;
                String[] dados = new String[5];


                //Le os dados do arquivo de entrada
                do {
                        for (int i = 0; i<5; i++) {
                                try {
                                        linha = enArquivo.readLine();
                                        if ( !linha.equals("endOfFile") ) {
                                                dados[i]=linha;
                                        } else {
                                                endOfFile = true;
                                                break;
                                        }
                                }
                                catch (IOException erro) {
                                        error.showMessageDialog(null,
erro,"ERRO",JOptionPane.ERROR_MESSAGE);
                                }
                        }
                        if ( !endOfFile ) {
```

```
                                data.insertRecord (dados[0], dados[1], dados[2], dados[3].charAt(0),
Integer.parseInt(dados[4]) );
                                regNum++;
                        }
                } while (!endOfFile);

                showGUI();


        }
        /** Retorna o nodo contendo o cpf passado */
        static String buscaPorCPF(String cpf) throws IOException {

                HumanNode result = data.getRegistroCPF (cpf);
                String retorno = "";
                //Se vazio, mostra dialog e resultados correspondentes
                if (result.getName() == null) {
                        error.showMessageDialog(null, "Nenhuma entrada encontrada para "+cpf,"Nao
Encontrado",JOptionPane.ERROR_MESSAGE);
                        retorno = "Nenhuma entrada encontrada para "+cpf;
                }
                else retorno = result.printMeG();
                return retorno;
        }
        /** Retorna uma lista de nodos correspondente ao nome passado */
        static String buscaPorNome(String nome) throws IOException {

                HumanList result = data.getRegistroNome (nome);
                String retorno = "";
                //Se vazio, mostra dialog e resultados correspondentes
                if (result.empty()) {
                        error.showMessageDialog(null, "Nenhuma entrada encontrada para "+nome,"Nao
Encontrado",JOptionPane.ERROR_MESSAGE);
                        retorno = "Nenhuma entrada encontrada para "+nome;
                }
                else retorno = result.printListG();
                return retorno;
        }
        /** Acao executada pelo botao de pesquisa */
        public void actionPerformed (ActionEvent evento) {

                String pesquisa = queryText.getText();

                try { fazerPesquisa (radlist.choosen, pesquisa); }
                catch (IOException erro)
{ error.showMessageDialog(null,erro,"Erro",JOptionPane.ERROR_MESSAGE); }
        }
        /** Executa a pesquisa dependendo do tipo de busca */
        public static void fazerPesquisa (String tipo, String string) throws IOException {
                //Ester egg, se nao entrar nada na busca, busca por void
                if (string.equals("")) string = "void";
                //Faz a busca e retorna
                if (tipo.equals("CPF")) resultShow.setText( buscaPorCPF(string) );
                else if (tipo.equals("Nome")) resultShow.setText( buscaPorNome(string) );
        }
        private static void showGUI () throws IOException {

                JFrame          mainFrame;
                JPanel          pesquisaPanel, resultPanel, bigPanel, enterTextPanel, radioPanel;
                JButton         openButton, searchButton;
                JRadioButton    cpfButton, nomeButton;
                ButtonGroup     choiceRadio;
                JScrollPane     scroller;
                GPeopleIndexer  listener;

                JFrame.setDefaultLookAndFeelDecorated(true);

                mainFrame       = new JFrame("People Indexer - Gráfico");

                bigPanel        = new JPanel();
                pesquisaPanel   = new JPanel();
                resultPanel     = new JPanel();
                enterTextPanel  = new JPanel();
                radioPanel      = new JPanel();

                searchButton    = new JButton("Pesquisar");
```

```
                resultShow      = new JTextArea(24,36);

                queryText       = new JTextField(20);

                cpfButton       = new JRadioButton("CPF");
                nomeButton      = new JRadioButton("Nome");

                choiceRadio     = new ButtonGroup();

                scroller        = new JScrollPane(resultShow, JScrollPane.VERTICAL_SCROLLBAR_ALWAYS,
JScrollPane.HORIZONTAL_SCROLLBAR_NEVER);

                listener        = new GPeopleIndexer();
                radlist         = new RadListener();

                bigPanel.setLayout( new BoxLayout( bigPanel, BoxLayout.PAGE_AXIS) );
                pesquisaPanel.setLayout( new BoxLayout( pesquisaPanel, BoxLayout.PAGE_AXIS) );

                pesquisaPanel.setPreferredSize(new Dimension(300, 100));
                resultPanel.setPreferredSize(new Dimension(450, 400));

                pesquisaPanel.setBorder( BorderFactory.createCompoundBorder(
                                        BorderFactory.createTitledBorder("Pesquisar"),
                                        BorderFactory.createEmptyBorder(0,0,0,0)
                                )
                        );

                resultPanel.setBorder( BorderFactory.createCompoundBorder(
                                        BorderFactory.createTitledBorder("Resultados da Pesquisa"),
                                        BorderFactory.createEmptyBorder(0,0,0,0)
                                )
                        );

                searchButton.addActionListener(listener);
                cpfButton.addActionListener(radlist);
                nomeButton.addActionListener(radlist);

                searchButton.setActionCommand("search");
                cpfButton.setActionCommand("changecpf");
                nomeButton.setActionCommand("changenome");

                enterTextPanel.add(queryText);
                enterTextPanel.add(searchButton);
                radioPanel.add(cpfButton);
                radioPanel.add(nomeButton);

                pesquisaPanel.add(enterTextPanel);
                pesquisaPanel.add(radioPanel);

                resultPanel.add(scroller);

                choiceRadio.add(cpfButton);
                choiceRadio.add(nomeButton);

                mainFrame.setContentPane(bigPanel);

                mainFrame.getContentPane().add(pesquisaPanel);
                mainFrame.getContentPane().add(resultPanel);

                cpfButton.setSelected(true);
                resultShow.setEditable(false);
                mainFrame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                mainFrame.setLocationRelativeTo(null);

                mainFrame.pack();
                mainFrame.setVisible(true);
        }
}
```