PONTIFÍCIA UNIVERSIDADE CATÓLICA DE GOIÁS

ESCOLA DE CIÊNCIAS EXATAS E DA COMPUTAÇÃO.

ANÁLISE E DESENVOLVIMENTO DE SISTEMAS.



**AGENDA EM LISTA ENCADEADA E ORDENADA**

GOIÂNIA, 12 DE MAIO DE 2020

Bruno Camargo Manso

**Agenda em Lista Encadeada e Ordenada**

Trabalho para composição das notas de N2

Orientador: José Olímpio Ferreira

GOIÂNIA,GO

2020

**Resumo**

O presente trabalho tem o objetivo de desenvolver uma agenda cujos dados são guardados em uma lista simplesmente encadeada. A lista deve ser ordenada seguindo a lógica de ordenação de uma lista encadeada. O programa deverá, além de mostrar os dados de forma ordenada (crescente e decrescente) para o usuário, salvar em um arquivo de texto, ou backup, para que este seja devidamente reaproveitado na próxima vez que o programa for executado. O trabalho a seguir contém códigos em Java tanto do programa em modo texto (terminal) quanto em interface gráfica (Jframe).

**Palavras-chave:** Agenda. Lista simplesmente encadeada. Ordenada. Backup. Códigos em Java. Programa. Modo texto. Interface gráfica.

**Sumário**

1. **Classe Lista Encadeada** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 5
2. **Classe Pessoa** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 7
3. **Classe Gerar Arquivo** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 8
4. **Interface Gráfica** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .8

4.1.  **Classe Programa** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 9

4.2. **Classe FrmContatos** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 13

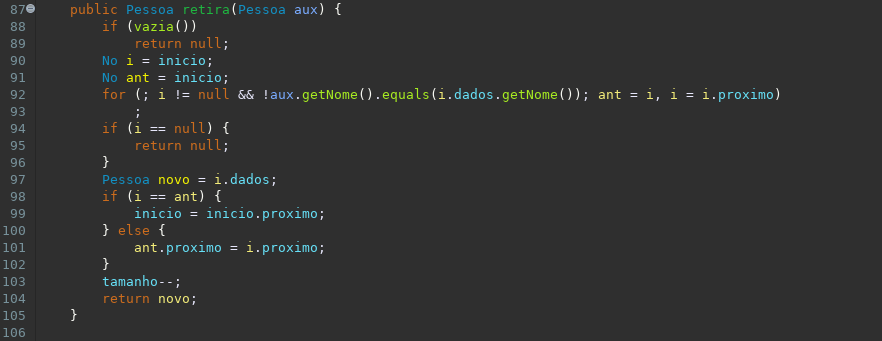
1. **Menu (modo terminal/texto)** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 16
2. **Exemplo do arquivo gerado** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 19
3. **Classe Lista Encadeada**

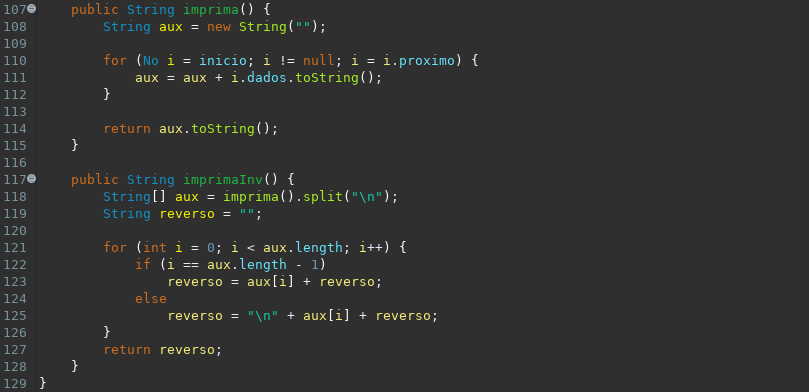
Essa classe contém o algoritmo necessário para guardar Objetos do tipo Pessoa de maneira ordenada. A ordenação dependerá do método ‘Inserir’ que sempre irá comparar valores antes da inserção, garantindo que tal pessoa seja inserida de forma alfabeticamente correta.

Abaixo sua implementação:







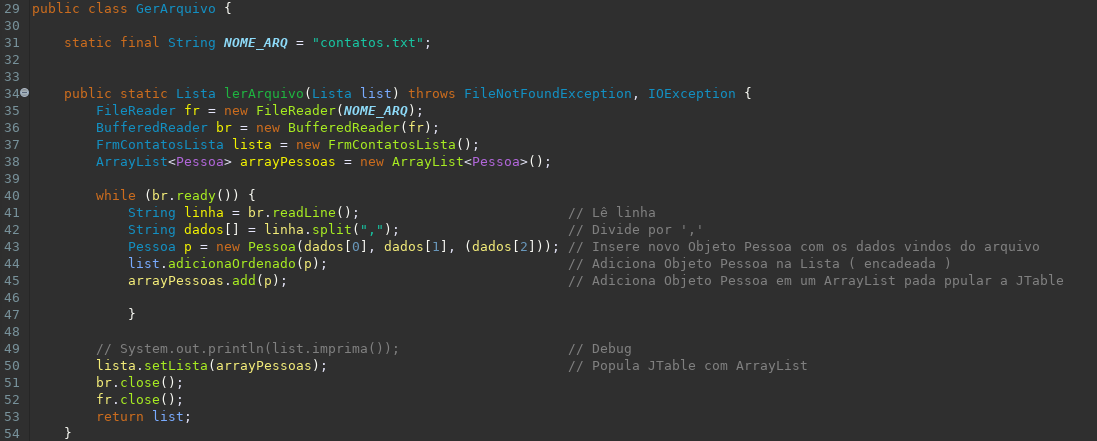
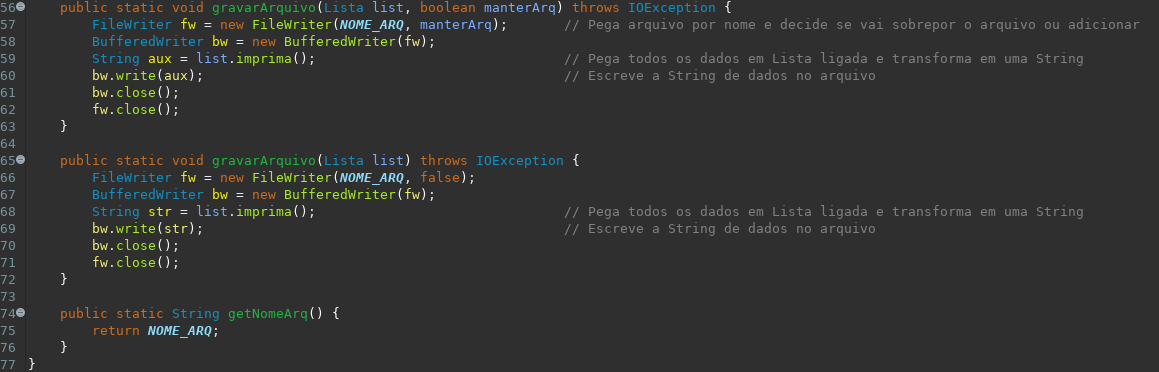


1. **Classe Pessoa**

A classe define os atributos que serão armazenados dentro de um Objeto do tipo Pessoa. Os atributos são do tipo String: nome, telefone e endereço.



*atributos, métodos e funções da classe Pessoa*

3. **Classe Gerar Arquivo**

Responsável por fazer ações de Ler e Salvar os dados computados no programa em um arquivo de texto. Isso irá garantir que os dados não se percam a após finalização do programa e que sejam lidos assim que o programa reinicia. Abaixo temos a implementação dos métodos:

4. **Interface Gráfica**

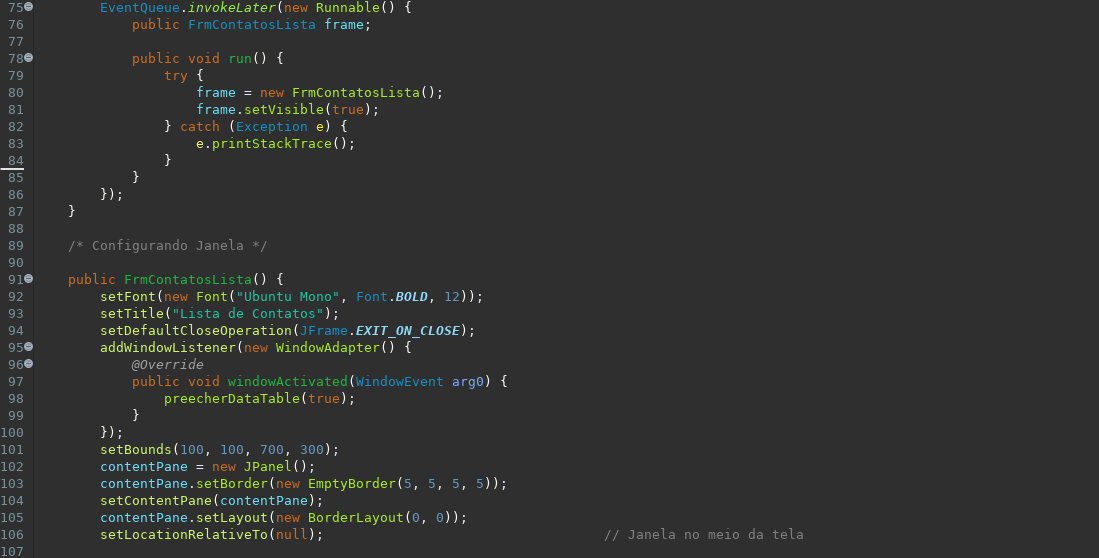
Implementada em Jframe podendo também exibir mensagens em JOptionPane, a interface gráfica melhora a experiência do usuário distanciando-o de processos do tipo terminal, facilitando assim o entendimento e a visualização do produto do programa, tornando este bem mais amistoso, intuitivo e mais dinâmico.

4.1.  **Classe Programa**

É a classe principal que deve ser iniciada para lançar o código, implementa a janela principal do programa. Ao iniciar, deverá carregar em uma tabela (JTable) os dados do arquivo. A tabela, devidamente preenchida, permite a ordenação dos elementos, por atributos e de forma crescente/decrescente. Os botões Adicionar, Editar, Excluir e Sair estão contidos nessa tabela, garantindo o devido CRUD das informações. Então para efetivar uma deleção (ou alteração), deve-se clicar sobre a linha desejada e utilizar os botões. Perceba que o aqui existe o uso do ArrayList somente para o preenchimento da tabela JFrame, nada mais. O ArrayList servirá apenas para popular os dados na tabela (JTable). O código abaixo está devidamente comentado ressaltando tanto esse último detalhe, quanto outros métodos implementados no código.



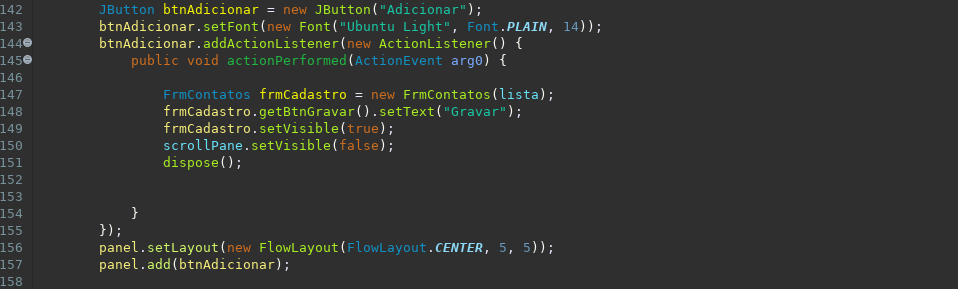
*iniciando variáveis e configurando o tema do visual da interface gráfica.*



*iniciando JFrame, gerando janela, configurando módulos da janela.*



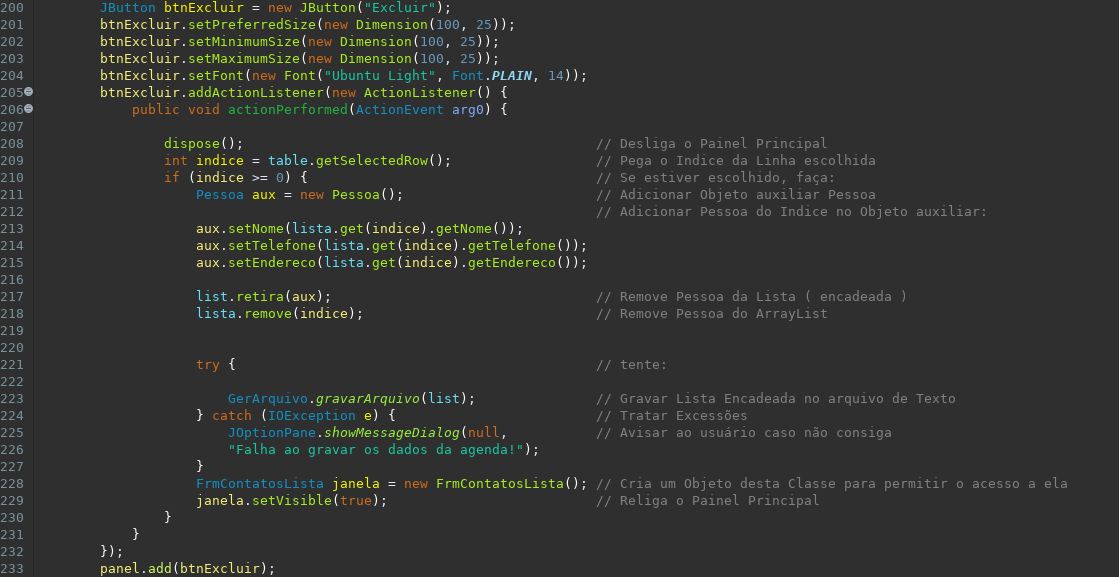
*ainda configurando módulos da janela principal.*



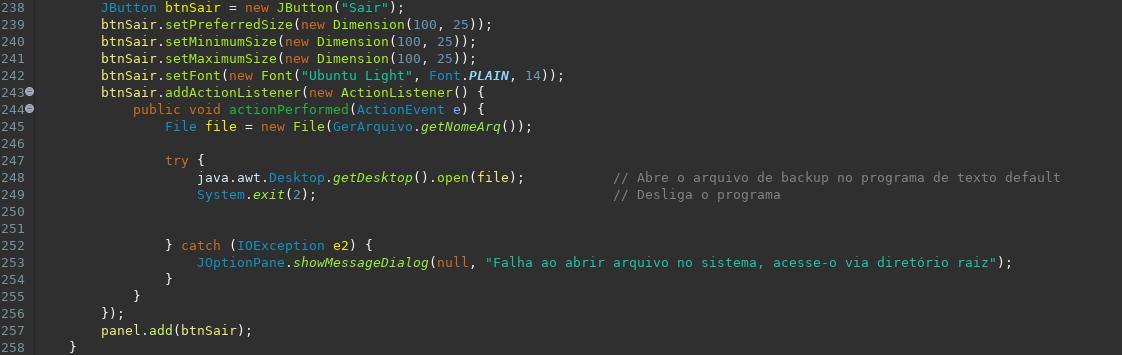
*configurando métodos para o botão ‘Adicionar’*



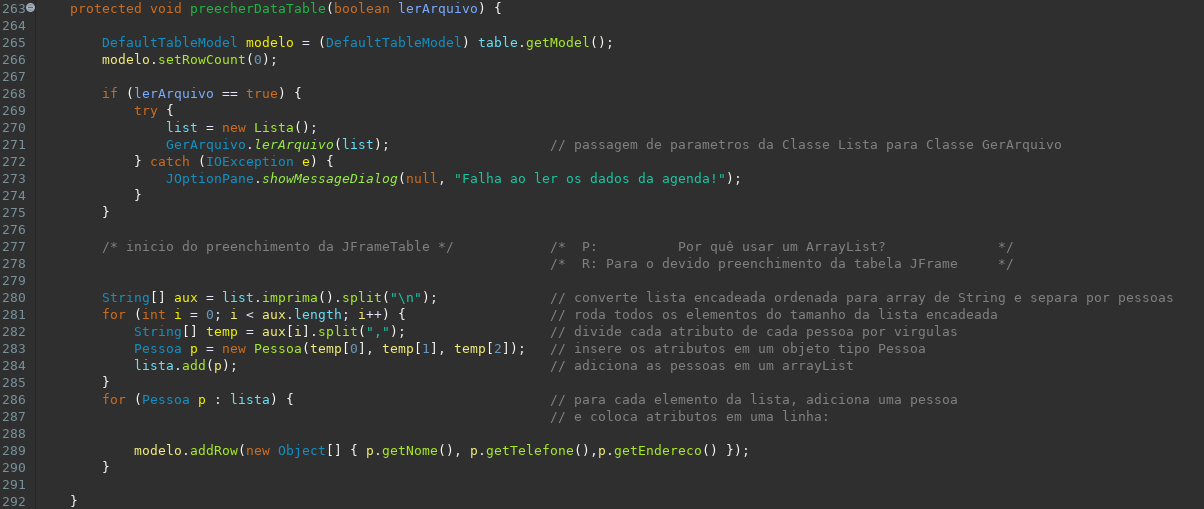
*configurando métodos para o botão ‘Alterar’*



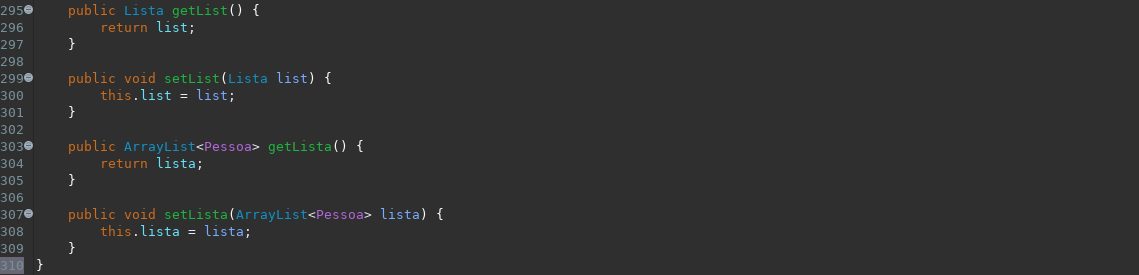
*configurando métodos para botão ‘Excluir’*



*configurando métodos para o botão ‘Sair’*

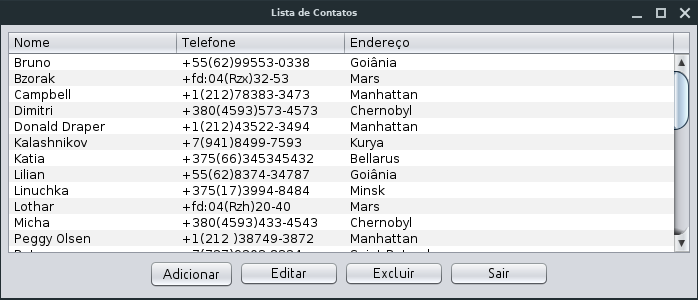


*configurando métodos de carregamento e preenchimento da tabela (JTable)*



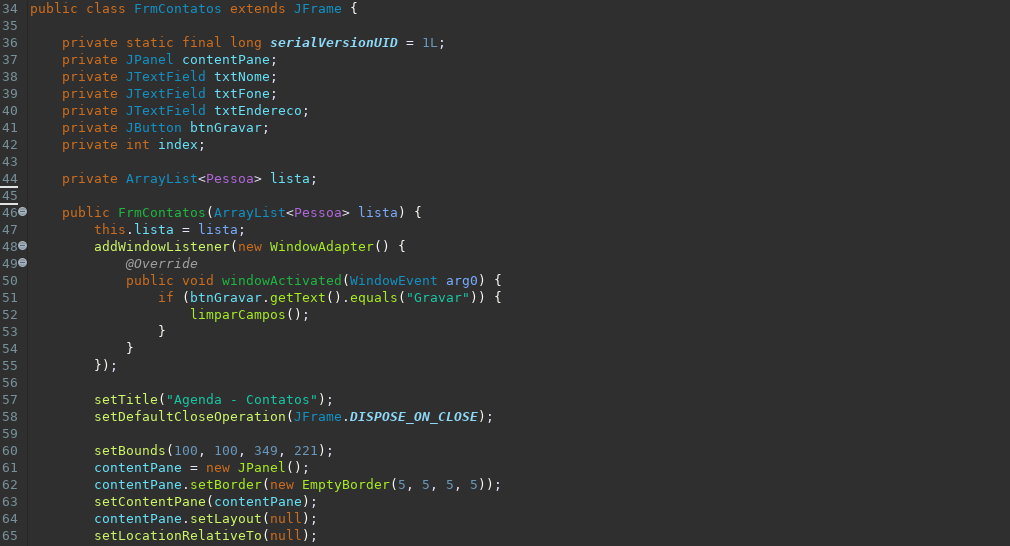
*getters e setters da classe*

A seguir, o resultado do código:



4.2. **Classe FrmContatos**

Quando os botões de adicionar e de excluir são clicados, uma segunda janela responsável pelo cadastro surge. Esta nova janela possuirá campos de preenchimento, e os botões gravar e cancelar. Abaixo, sua implementação:



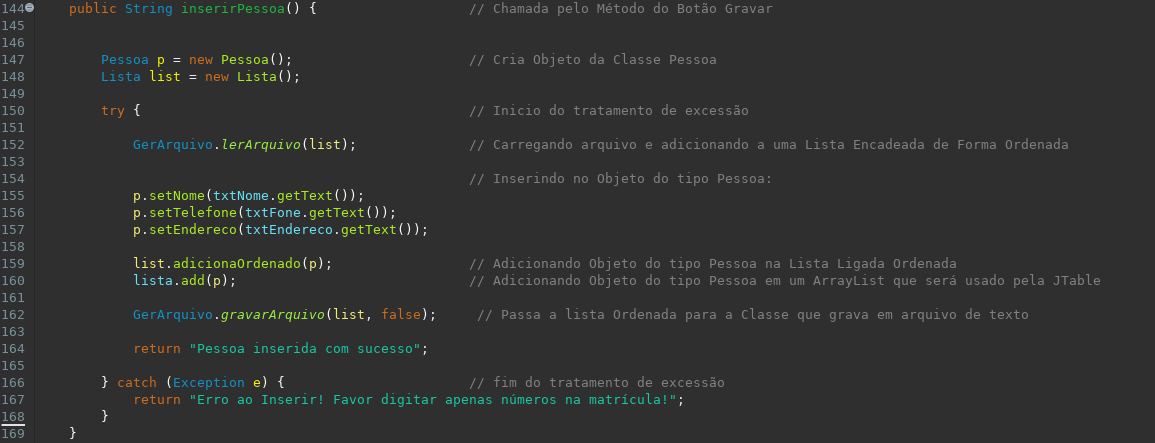
*iniciando nova janela, gerando janela, configurando seus módulos*



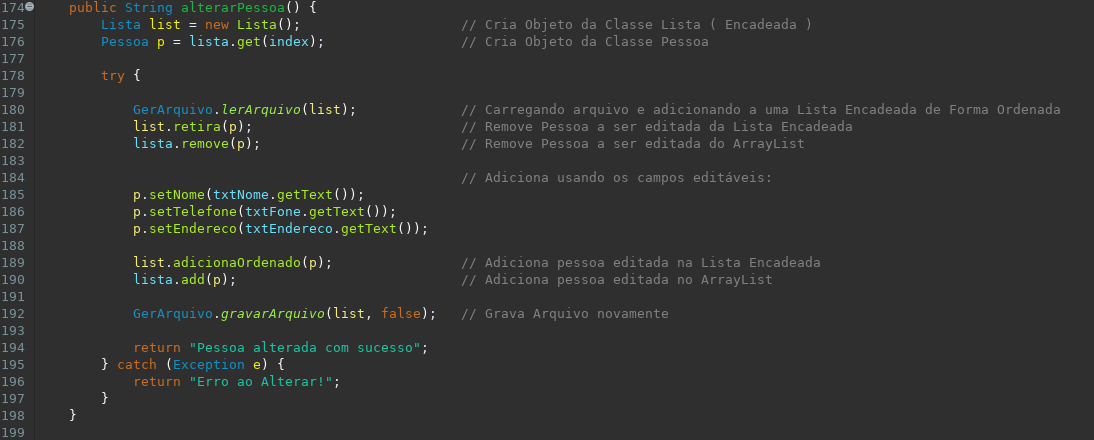
*ainda configurando módulos da janela*



*configurando métodos dos botões de cancelar e gravar*



*configurando função de inserção estendidos pelo botão de gravação*

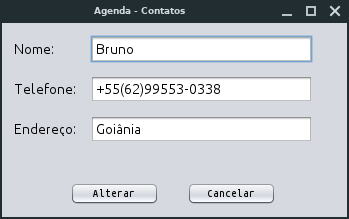


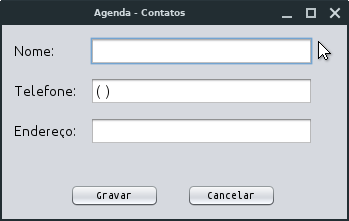
*configurando função que permite a alteração das pessoas*



*getters e setters da classe*

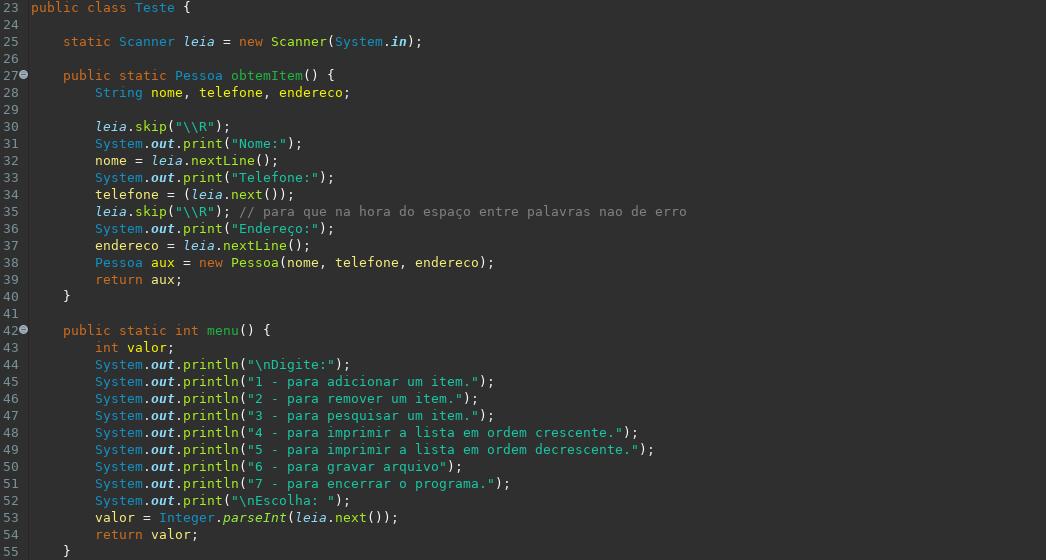
Abaixo temos os resultados da implementação:





5. **Menu (modo terminal/texto)**

Aqui encontra-se a implementação do menu em modo terminal (ou texto), pouco amistoso ao usuário e também menos dinâmico que a interface gráfica supracitado, porém com grande valia, uma vez que as devidas lógicas são testadas antes mesmo de serem implementadas em um ambiente gráfico. Nota-se que um novo método fora implementado na classe Lista para substituir a ação automática de ordenação no JTable do ambiente gráfico. Tal método chamado de imprima invertido (imprimaInv) tem apenas a função de inverter a ordem de impressão do método anterior ‘imprima’, função que pode ser acessada no menu de opções número 5. Abaixo a implementação do código:



*implementação das funções para obter item e o menu de acesso*

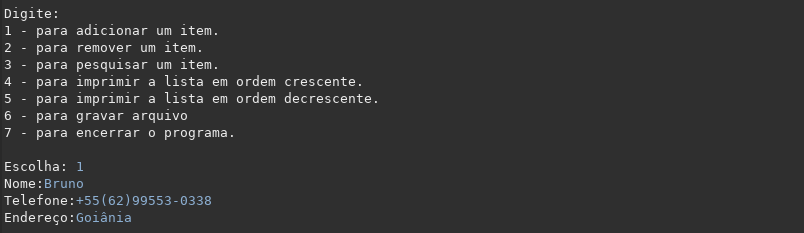


*início do switch de menu*

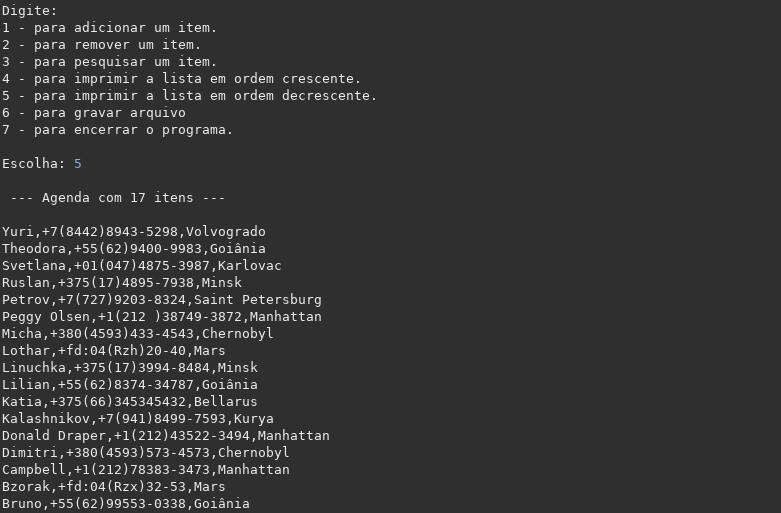


*fim do switch de menu*

Temos os seguintes resultados:



*menu de acesso opção de adição e cadastro*



*menu de acesso opção de imprimir a lista em ordem decrescente*

*6.* **Exemplo do arquivo gerado**

Enfim, no final deste trabalho, disponibilizo um exemplo de arquivo gerado pelo programa contendo os contatos adicionados pelo mesmo. O arquivo com nome de contatos.txt deve estar na mesma pasta do programa e seu conteúdo descrito em CSV (comma separated values), ou seja, os atributos de cada pessoa separados por vírgulas. A seguir:

