Estácio

Missão Prática | Nível 1 | Mundo 3

{ DESENVOLVIMENTO FULL-STACK }

2023.1

Desenvolvedor: Felipe Adriano

RPG0014 – Iniciando o caminho pelo Java

Objetivo:

Implementação de um cadastro de clientes em modo texto com persistência em arquivos, baseado na tecnologia Java.

1° Procedimento | Criação das Entidades e Sistema de Persistência

Estrutura

- ➔ procedimento1
  - dominio
    - src
      - Pessoa
      - PessoaFisica
      - PessoaFisicaRepo
      - PessoaJuridica
      - PessoaJuridicaRepo
  - teste
    - src
      - CadastroPOO

Pessoa

```
Import java.io.Serializable;

public class Pessoa implements Serializable {
    private int id;
    private String nome;

    public Pessoa(String nome) {
        this.nome = nome;
    }

    public Pessoa(int id) {
        this.id = id;
    }

    public Pessoa(int id, String nome) {
        this.id = id;
        this.nome = nome;
    }

    public void exibir(){
        System.out.println(this.id);
        System.out.println(this.nome);
    }


    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getNome() {
        return nome;
    }

    public void setNome(String nome) {
        this.nome = nome;
    }
}
```

PessoaFisica

```java
import java.io.Serializable;

public class PessoaFisica extends Pessoa implements Serializable {
    private String cpf;
    private int idade;

    public void exibir() {
        System.out.println("ID: " + getId());
        System.out.println("NOME: " + getNome());
        System.out.println("CPF: " + this.cpf);
        System.out.println("IDADE: " + this.idade);
    }

    public String getCpf() {
        return cpf;
    }

    public void setCpf(String cpf) {
        this.cpf = cpf;
    }

    public int getIdade() {
        return idade;
    }

    public void setIdade(int idade) {
        this.idade = idade;
    }

    public PessoaFisica(int id, String nome, String cpf, int idade) {
        super(id, nome);
        this.cpf = cpf;
        this.idade = idade;


    }
}
```

PessoaFisicaRepo

```
import java.io.*;
import java.util.ArrayList;
import java.util.List;

public class PessoaFisicaRepo implements Serializable {
    private List<PessoaFisica> listaPessoaFisica;

    public PessoaFisicaRepo(List<PessoaFisica> listaPessoaFisica) {

        this.listaPessoaFisica = listaPessoaFisica;
    }

    public PessoaFisicaRepo() {

        this.listaPessoaFisica = new ArrayList<>();
    }

    public void inserir(PessoaFisica pessoa) {

        listaPessoaFisica.add(pessoa);

    }

    public void alterar(int indice, PessoaFisica pessoa) {
        if (indice >= 0 && indice < listaPessoaFisica.size()) {
            listaPessoaFisica.set(indice, pessoa);
        } else {
            System.out.println("Erro no Alterar");
        }
    }

    public void excluir(int id) {
        for (PessoaFisica pessoa : listaPessoaFisica) {
            if (pessoa.getId() == id) {
                listaPessoaFisica.remove(pessoa);
                System.out.println("Pessoa com ID " + id + " excluída
com sucesso");
                return;
            }
        }
        System.out.println("Pessoa com o ID " + id + " não encontrada
para excluir");
    }

    public PessoaFisica obter(int id) {
        for (PessoaFisica pessoa : listaPessoaFisica) {
            if (pessoa.getId() == id) {
                return pessoa;
            }
        }
        return null;
    }

    public ArrayList<PessoaFisica> obterTodos() {
        return (ArrayList<PessoaFisica>) listaPessoaFisica;
    }


    public void persistir(String persistirArquivo) {
```

```
        try (FileOutputStream filePer = new
FileOutputStream(persistirArquivo);
             ObjectOutputStream saida = new
ObjectOutputStream(filePer)) {
            saida.writeObject(listaPessoaFisica);
            System.out.println("Dados de Pessoas Fisicas
Armazenados");
        } catch (IOException e) {
            throw new RuntimeException(e);
        }
    }

    public void recuperar(String recuperarArquivo) {
        try (FileInputStream fileRec = new
FileInputStream(recuperarArquivo);
             ObjectInputStream entrada = new
ObjectInputStream(fileRec)) {
            listaPessoaFisica = (ArrayList<PessoaFisica>)
entrada.readObject();
            System.out.println("Dados de Pessoas Fisicas
Recuperados");
        } catch (IOException | ClassNotFoundException e) {
            throw new RuntimeException(e);
        }
    }

}
```

PessoaJuridica

```
public class PessoaJuridica extends Pessoa {
    private String cnpj;

    public PessoaJuridica(int id, String nome, String cnpj) {
        super(id, nome);
        this.cnpj = cnpj;
    }

    public void exibir() {
        System.out.println("ID: " + getId());
        System.out.println("NOME: " + getNome());
        System.out.println("CNPJ: " + this.cnpj);
    }

    public String getCnpj() {
        return cnpj;
    }

    public void setCnpj(String cnpj) {
        this.cnpj = cnpj;
    }
}
```

PessoaJuridicaRepo

```
import java.io.*;
import java.util.ArrayList;
import java.util.List;

public class PessoaJuridicaRepo {
    private List<PessoaJuridica> listaPessoaJuridica;

    public PessoaJuridicaRepo(List<PessoaJuridica>
listaPessoaJuridica) {

        this.listaPessoaJuridica = listaPessoaJuridica;
    }

    public PessoaJuridicaRepo() {

        this.listaPessoaJuridica = new ArrayList<>();
    }

    public void inserir(PessoaJuridica pessoa) {

        listaPessoaJuridica.add(pessoa);
    }

    public void alterar(int indice, PessoaJuridica pessoa) {
        if (indice >= 0 && indice < listaPessoaJuridica.size()) {
            listaPessoaJuridica.set(indice, pessoa);
        } else {
            System.out.println("Erro no Alterar");
        }
    }

    public void excluir(int id) {
        for (PessoaJuridica pessoa : listaPessoaJuridica) {
            if (pessoa.getId() == id) {
                listaPessoaJuridica.remove(pessoa);
                System.out.println("Pessoa com ID " + id + " excluída
com sucesso");
                return;
            }
        }
        System.out.println("Pessoa com o ID " + id + " não encontrada
para excluir");
    }

    public PessoaJuridica obter(int id) {
        for (PessoaJuridica pessoa : listaPessoaJuridica) {
            if (pessoa.getId() == id) {
                return pessoa;
            }
        }
        return null;
    }

    public ArrayList<PessoaJuridica> obterTodos() {
        return (ArrayList<PessoaJuridica>) listaPessoaJuridica;
    }

    public void persistir(String persistirArquivo) {
        try (FileOutputStream filePer = new
```

```
FileOutputStream(persistirArquivo);
             ObjectOutputStream saida = new
ObjectOutputStream(filePer)) {
            saida.writeObject(listaPessoaJuridica);
            System.out.println("Dados de Pessoas Juridicas
Armazenados");
        } catch (IOException e) {
            throw new RuntimeException(e);
        }
    }

    public void recuperar(String recuperarArquivo) {
        try (FileInputStream fileRec = new
FileInputStream(recuperarArquivo);
             ObjectInputStream entrada = new
ObjectInputStream(fileRec)) {
            listaPessoaJuridica = (ArrayList<PessoaJuridica>)
entrada.readObject();
            System.out.println("Dados de Pessoas Juridicas
Recuperados");
        } catch (IOException | ClassNotFoundException e) {
            throw new RuntimeException(e);
        }
    }


}
```

CadastroPOO

```
import java.util.ArrayList;


public class CadastroPOO {
    public static void main(String[] args) {
        PessoaFisicaRepo repo1 = new PessoaFisicaRepo();
        PessoaFisicaRepo repo2 = new PessoaFisicaRepo();
        PessoaJuridicaRepo repo3 = new PessoaJuridicaRepo();
        PessoaJuridicaRepo repo4 = new PessoaJuridicaRepo();
        PessoaFisica pessoaf1 = new PessoaFisica(1, "Joãozin", "111-
222-333-45", 14);
        PessoaFisica pessoaf2 = new PessoaFisica(2, "Cleberzin", "222-
333-444-56", 15);
        PessoaJuridica pessoaj1 = new PessoaJuridica(3, "Casa dos
doces", "132-456-546-4");
        PessoaJuridica pessoaj2 = new PessoaJuridica(4, "Casa dos
salgados", "665-498-546-5");
        repo1.inserir(pessoaf1);
        repo1.inserir(pessoaf2);
        repo3.inserir(pessoaj1);
        repo3.inserir(pessoaj2);


        repo1.persistir("pessoa_fisica.txt");
        repo2.recuperar("pessoa_fisica.txt");
        ArrayList<PessoaFisica> listaPessoaFisica =
repo2.obterTodos();
        listaPessoaFisica.forEach(PessoaFisica::exibir);

        repo3.persistir("pessoa_juridica.txt");
        repo4.recuperar("pessoa_juridica.txt");
        ArrayList<PessoaJuridica> listaPessoaJuridica =
repo4.obterTodos();
        listaPessoaJuridica.forEach(PessoaJuridica::exibir);

    }
}
```

Resultado do Procedimento

```
Dados de Pessoas Fisicas Armazenados
Dados de Pessoas Fisicas Recuperados
ID: 1
NOME: Joãozin
CPF: 111-222-333-45
IDADE: 14
ID: 2
NOME: Cleberzin
CPF: 222-333-444-56
IDADE: 15
Dados de Pessoas Juridicas Armazenados
Dados de Pessoas Juridicas Recuperados
ID: 3
NOME: Casa dos doces
CNPJ: 132-456-546-4
ID: 4
NOME: Casa dos salgados
CNPJ: 665-498-546-5

Process finished with exit code 0
```

Análise e Conclusão da primeira etapa.

O uso adequado de herança pode trazer vantagens significativas, como reutilização e a organização do código, porém, também pode gerar uma certa complexidade entre as classes, como por exemplo, citando as vantagens do uso da herança, temos reutilização de códigos, que nos permite reduzir a quantidade de códigos duplicados, melhorando assim a organização e a estrutura do seu código. Mas assim como apresenta suas vantagens, também temos as desvantagens do uso, como por exemplo a limitação da reutilização dos códigos em certos casos, pois o Java não suporta a herança múltipla, outro caso também é a complexidade que vai aumentando, conforme o crescimento da hierarquia das classes, começa a se tornar complicado entender como as classes estão conectadas.

Um ponto forte a ser destacado aqui é a interface “Serializable”, responsável por efetuar a persistência em arquivos binários, sinalizando que os objetos de uma certa classe podem ser convertidos em uma sequência de bytes, permitindo assim que sejam armazenados em um arquivo e posteriormente podendo ser recuperados.

A API Stream no Java utiliza o paradigma funcional para realizar operações em coleções de forma mais eficiente e concisa. Alguns conceitos do paradigma funcional presentes nesta API Sream são as **Expressões lambda**, que permitem passar comportamentos como parâmetros de métodos, permitindo assim a criação de um código mais flexível e legível. **Imutabilidade** o que significa que são geralmente imutáveis, ou seja, significa que elas não modificam a coleção original, mas criam uma nova coleção com os resultados das operações.

Por fim, temos o padrão de desenvolvimento mais comum para a persistência de dados em arquivos Java, o padrão DAO (Data Access Object) que separa a lógica de negócios da lógica de acesso aos dados, fornecendo uma interface para acessar os dados que podem ser implementadas de varias maneiras, tornando o código mais modular e facilitando a manutenão e teste do sistema.

Segue abaixo para o segundo procedimento

2° Procedimento | Criação do Cadastro em Modo Texto

Estrutura

- ➔ Procedimento2
  - dominio2
    - src
      - Pessoa
      - PessoaFisica
      - PessoaFisicaRepo
      - PessoaJuridica
      - PessoaJuridicaRepo
  - teste2
    - src
      - CadastroPOO

Pessoa

```java
import java.io.Serializable;

public class Pessoa implements Serializable {
    private int id;
    private String nome;

    public Pessoa(String nome) {
        this.nome = nome;
    }

    public Pessoa(int id) {
        this.id = id;
    }

    public Pessoa(int id, String nome) {
        this.id = id;
        this.nome = nome;
    }

    public void exibir(){
        System.out.println(this.id);
        System.out.println(this.nome);
    }


    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getNome() {
        return nome;
    }

    public void setNome(String nome) {
        this.nome = nome;
    }
}
```

PessoaFisica

```
import java.io.Serializable;

public class PessoaFisica extends Pessoa implements Serializable {
    private String cpf;
    private int idade;

    public void exibir() {
        System.out.println("ID: " + getId());
        System.out.println("NOME: " + getNome());
        System.out.println("CPF: " + this.cpf);
        System.out.println("IDADE: " + this.idade);
    }


    public String getCpf() {
        return cpf;
    }

    public void setCpf(String cpf) {
        this.cpf = cpf;
    }

    public int getIdade() {
        return idade;
    }

    public void setIdade(int idade) {
        this.idade = idade;
    }

    public PessoaFisica(int id, String nome, String cpf, int idade) {
        super(id, nome);
        this.cpf = cpf;
        this.idade = idade;


    }
}
```

PessoaFisicaRepo

```
import javax.swing.plaf.synth.SynthOptionPaneUI;
import java.io.*;
import java.util.ArrayList;
import java.util.List;

public class PessoaFisicaRepo implements Serializable {
    private List<PessoaFisica> listaPessoaFisica;

    public PessoaFisicaRepo(List<PessoaFisica> listaPessoaFisica) {

        this.listaPessoaFisica = listaPessoaFisica;
    }

    public PessoaFisicaRepo() {

        this.listaPessoaFisica = new ArrayList<>();
    }

    public void inserir(PessoaFisica pessoa) {
        listaPessoaFisica.add(pessoa);


    }

    public void alterar(int id, PessoaFisica pessoaFisicaAlterada) {
        for (int i = 0; i < this.listaPessoaFisica.size(); i++) {
            PessoaFisica pessoaFisica = this.listaPessoaFisica.get(i);
            if (pessoaFisica.getId() == id) {
                pessoaFisica.setNome(pessoaFisicaAlterada.getNome());
                pessoaFisica.setCpf(pessoaFisicaAlterada.getCpf());

pessoaFisica.setIdade(pessoaFisicaAlterada.getIdade());

            }
        }
    }

    public void excluir(int id) {
        for (PessoaFisica pessoa : listaPessoaFisica) {
            if (pessoa.getId() == id) {
                listaPessoaFisica.remove(pessoa);
                return;
            }
        }
        System.out.println("Pessoa com o ID " + id + " não encontrada
para excluir");
    }

    public PessoaFisica obter(int id) {
        boolean encontrado = false;

        for (PessoaFisica pessoaFisica : listaPessoaFisica) {
            if (pessoaFisica.getId() == id) {
                System.out.println("╠ ID: " + pessoaFisica.getId());
                System.out.println("╠ NOME: " +
pessoaFisica.getNome());
                System.out.println("╠ CPF: " + pessoaFisica.getCpf());
```

```
                System.out.println("╠ IDADE: " +
pessoaFisica.getIdade());
                System.out.println("╠═══════════");
                encontrado = true;
            }
        }
        if (!encontrado) {
            System.out.println("Sem Dados neste ID");
        }
        return null;
    }

    public ArrayList<PessoaFisica> obterTodos() {
        if (listaPessoaFisica.isEmpty()) {
            System.out.println("Lamento, mas sua lista está vazia.");
        }

        for (PessoaFisica pessoaFisica : listaPessoaFisica) {
            System.out.println("╠ ID: " + pessoaFisica.getId());
            System.out.println("╠ NOME: " + pessoaFisica.getNome());
            System.out.println("╠ CPF: " + pessoaFisica.getCpf());
            System.out.println("╠ IDADE: " + pessoaFisica.getIdade());
            System.out.println("╠═══════════");
        }
        return null;
    }


    public void persistir(String persistirArquivo) {
        try (FileOutputStream filePer = new
FileOutputStream(persistirArquivo);
             ObjectOutputStream saida = new
ObjectOutputStream(filePer)) {
            saida.writeObject(listaPessoaFisica);
            System.out.println("Dados de Pessoas Fisicas
Armazenados");
        } catch (IOException e) {
            throw new RuntimeException(e);
        }
    }

    public void recuperar(String recuperarArquivo) {
        try (FileInputStream fileRec = new
FileInputStream(recuperarArquivo);
             ObjectInputStream entrada = new
ObjectInputStream(fileRec)) {
            listaPessoaFisica = (ArrayList<PessoaFisica>)
entrada.readObject();
            System.out.println("Dados de Pessoas Fisicas
Recuperados");
        } catch (IOException | ClassNotFoundException e) {
            throw new RuntimeException(e);
        }
    }

}
```

PessoaJuridica

```
public class PessoaJuridica extends Pessoa {
    private String cnpj;

    public PessoaJuridica(int id, String nome, String cnpj) {
        super(id, nome);
        this.cnpj = cnpj;
    }

    public void exibir() {
        System.out.println("ID: " + getId());
        System.out.println("NOME: " + getNome());
        System.out.println("CNPJ: " + this.cnpj);
    }

    public String getCnpj() {
        return cnpj;
    }

    public void setCnpj(String cnpj) {
        this.cnpj = cnpj;
    }
}
```

PessoaJuridicaRepo

```
import java.io.*;
import java.util.ArrayList;
import java.util.List;

public class PessoaJuridicaRepo {
    private List<PessoaJuridica> listaPessoaJuridica;

    public PessoaJuridicaRepo(List<PessoaJuridica>
listaPessoaJuridica) {

        this.listaPessoaJuridica = listaPessoaJuridica;
    }

    public PessoaJuridicaRepo() {

        this.listaPessoaJuridica = new ArrayList<>();
    }

    public void inserir(PessoaJuridica pessoa) {

        listaPessoaJuridica.add(pessoa);

    }

    public void alterar(int id, PessoaJuridica pessoaJuridicaAlterada)
{
        for (int i = 0; i < this.listaPessoaJuridica.size(); i++) {
            PessoaJuridica pessoaJuridica =
this.listaPessoaJuridica.get(i);
```

```
            if (pessoaJuridica.getId() == id) {

pessoaJuridica.setNome(pessoaJuridicaAlterada.getNome());

pessoaJuridica.setCnpj(pessoaJuridicaAlterada.getCnpj());
            }
        }
    }

    public void excluir(int id) {
        for (PessoaJuridica pessoa : listaPessoaJuridica) {
            if (pessoa.getId() == id) {
                listaPessoaJuridica.remove(pessoa);
                return;
            }
        }
        System.out.println("Pessoa com o ID " + id + " não encontrada
para excluir");
    }

    public PessoaJuridica obter(int id) {
        boolean encontrado = false;
        for (PessoaJuridica pessoaJuridica : listaPessoaJuridica) {
            if (pessoaJuridica.getId() == id) {
                System.out.println("╠ ID: " + pessoaJuridica.getId());
                System.out.println("╠ NOME: " +
pessoaJuridica.getNome());
                System.out.println("╠ CNPJ: " +
pessoaJuridica.getCnpj());
                System.out.println("╠═══════════");
                encontrado = true;
            }
        }
        if (!encontrado) {
            System.out.println("Sem Dados neste ID");
        }
        return null;
    }

    public ArrayList<PessoaJuridica> obterTodos() {
        if (listaPessoaJuridica.isEmpty()) {
            System.out.println("Lamento, mas sua lista está vazia.");
        }

        for (PessoaJuridica pessoaJuridica : listaPessoaJuridica) {
            System.out.println("╠ ID: " + pessoaJuridica.getId());
            System.out.println("╠ NOME: " + pessoaJuridica.getNome());
            System.out.println("╠ CNPJ: " + pessoaJuridica.getCnpj());
            System.out.println("╠═══════════");
        }
        return null;
    }

    public void persistir(String persistirArquivo) {
        try (FileOutputStream filePer = new
FileOutputStream(persistirArquivo);
             ObjectOutputStream saida = new
ObjectOutputStream(filePer)) {
            saida.writeObject(listaPessoaJuridica);
            System.out.println("Dados de Pessoas Juridicas
Armazenados");
```

```
        } catch (IOException e) {
            throw new RuntimeException(e);
        }
    }

    public void recuperar(String recuperarArquivo) {
        try (FileInputStream fileRec = new
FileInputStream(recuperarArquivo);
             ObjectInputStream entrada = new
ObjectInputStream(fileRec)) {
            listaPessoaJuridica = (ArrayList<PessoaJuridica>)
entrada.readObject();
            System.out.println("Dados de Pessoas Juridicas
Recuperados");
        } catch (IOException | ClassNotFoundException e) {
            throw new RuntimeException(e);
        }
    }


}
```

CadastroPOO

```
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;


public class CadastroPOO {
    public static void main(String[] args) {
        PessoaFisicaRepo repo1 = new PessoaFisicaRepo();
        PessoaFisicaRepo repo2 = new PessoaFisicaRepo();
        PessoaJuridicaRepo repo3 = new PessoaJuridicaRepo();
        Scanner scanner = new Scanner(System.in);
        int opcao = -1;
        String arquivoDeDadosPessoaFisica = "pessooa_fisica.bin";
        String arquivoDeDadosPessoaJuridica = "pessooa_Juridica.bin";
        while (opcao != 0) {
            System.out.println("╔═══════════════════════════════════╗");
            System.out.println("║RPG014 - Sistema de Cadastramento║");
            System.out.println("╚═══════════════════════════════════╝");
            System.out.println("╠ (\u001B[1m1\u001B[0m) - Incluir
Pessoa            ╣");
            System.out.println("╠══                                ══╣");
            System.out.println("╠ (\u001B[1m2\u001B[0m) - Alterar
Pessoa            ╣");
            System.out.println("╠══                                ══╣");
            System.out.println("╠ (\u001B[1m3\u001B[0m) - Excluir
Pessoa            ╣");
            System.out.println("╠══                                ══╣");
            System.out.println("╠ (\u001B[1m4\u001B[0m) - Buscar pelo
Id              ╣");
            System.out.println("╠══                                ══╣");
            System.out.println("╠ (\u001B[1m5\u001B[0m) - Exibir Todos
╣");
            System.out.println("╠══                                ══╣");
            System.out.println("╠ (\u001B[1m6\u001B[0m) - Persistir
Dados           ╣");
            System.out.println("╠══                                ══╣");
            System.out.println("╠ (\u001B[1m7\u001B[0m) - Recuperar
Dados           ╣");
            System.out.println("╠══                                ══╣");
            System.out.println("╠ (\u001B[1m0\u001B[0m) - Finalizar
Programa        ╣");
            System.out.println("╚═══════════════════════════════════╝");

            opcao = scanner.nextInt();
            String tipoPessoa;
            switch (opcao) {
                case 1:

System.out.println("╔═══════════════════════════════════╗");
                    System.out.println("╠ Opção (1)        Incluir
╠");

System.out.println("╠═══════════════════════════════════╣");
                    System.out.println("╠ (\u001B[1mF\u001B[0m) -
Pessoa Fisica           ╣");
                    System.out.println("╠══
══╣");
```

```
                    System.out.println("╠ (\u001B[1mJ\u001B[0m) -
Pessoa Juridica          ╣");

System.out.println("╚═══════════════════════════════════════╝");
                    tipoPessoa = scanner.next().toUpperCase();
                    if (tipoPessoa.equals("F")) {

System.out.println("╔═══════════════════════════════════════╗");
                        System.out.println("╣ Opção (F) selecionada:
Fisico   ╠");

System.out.println("╠═══════════════════════════════════════╣");
                        System.out.println("╠  Digite o ID da Pessoa
Fisica   ╣");
                        int idPessoaF = scanner.nextInt();

System.out.println("╠═══════════════════════════════════════╣");
                        System.out.println("╠ Digite o NOME da Pessoa
Fisica  ╣");
                        scanner.nextLine();
                        String nomePessoaF = scanner.nextLine();

System.out.println("╠═══════════════════════════════════════╣");
                        System.out.println("╠  Digite o CPF da Pessoa
Fisica  ╣");
                        String cpfPessoaF = scanner.next();

System.out.println("╠═══════════════════════════════════════╣");
                        System.out.println("╠ Digite a IDADE da Pessoa
Fisica ╣");
                        int idadePessoaF = scanner.nextInt();
                        PessoaFisica cadastroCompleto = new
PessoaFisica(idPessoaF, nomePessoaF, cpfPessoaF, idadePessoaF);
                        repo1.inserir(cadastroCompleto);
                        System.out.println("╠        CADASTRO COMPLETO
╣");

System.out.println("╚═══════════════════════════════════════╝");


                    } else if (tipoPessoa.equals("J")) {

System.out.println("╔═══════════════════════════════════════╗");
                        System.out.println("╣ Opção (J) selecionada:
Juridico ╠");

System.out.println("╠═══════════════════════════════════════╣");
                        System.out.println("╠ Digite o ID da Pessoa
Juridica  ╣");
                        int idPessoaJ = scanner.nextInt();

System.out.println("╠═══════════════════════════════════════╣");
                        System.out.println("╠ Digite o NOME da Pessoa
Juridica╣");
                        scanner.nextLine();
                        String nomePessoaJ = scanner.nextLine();

System.out.println("╠═══════════════════════════════════════╣");
                        System.out.println("╠ Digite o CNPJ da Pessoa
Juridica╣");
                        String cnpjPessoaJ = scanner.next();
```

```
                        PessoaJuridica cadastroCompleto = new
PessoaJuridica(idPessoaJ, nomePessoaJ, cnpjPessoaJ);
                        repo3.inserir(cadastroCompleto);
                        System.out.println("╠        CADASTRO COMPLETO
╣");

System.out.println("╚═══════════════════════════════════╝");

                    } else {
                        System.out.println("Opção Inválida");
                    }
                    break;
                case 2:

System.out.println("╔═══════════════════════════════════╗");
                    System.out.println("║ Opção (2)     Alterar
║");

System.out.println("╠═══════════════════════════════════╣");
                    System.out.println("╠ (\u001B[1mF\u001B[0m) -
Pessoa Fisica            ╣");
                    System.out.println("╠══
══╣");
                    System.out.println("╠ (\u001B[1mJ\u001B[0m) -
Pessoa Juridica          ╣");

System.out.println("╚═══════════════════════════════════╝");
                    tipoPessoa = scanner.next().toUpperCase();
                    if (tipoPessoa.equals("F")) {

System.out.println("╔═══════════════════════════════════╗");
                        System.out.println("║ Opção (F) selecionada:
Fisico    ║");

System.out.println("╠═══════════════════════════════════╣");
                        System.out.println("╠   Digite o ID da Pessoa
Fisica  ╣");
                        int idPessoaF = scanner.nextInt();
                        repo1.obter(idPessoaF);
                        System.out.println("╠          DADOS ATUAIS
╣");

System.out.println("╠═══════════════════════════════════╣");
                        System.out.println("║
║");
                        System.out.println("║
║");

System.out.println("╠═══════════════════════════════════╣");
                        System.out.println("║         ALTERAR DADOS
║");

System.out.println("╠═══════════════════════════════════╣");
                        System.out.println("╠ Altere o NOME da Pessoa
Fisica  ╣");
                        scanner.nextLine();
                        String nomePessoaF = scanner.nextLine();


System.out.println("╠═══════════════════════════════════╣");
                        System.out.println("╠  Altere o CPF da Pessoa
```

```
Fisica  ╢");
                        String cpfPessoaF = scanner.next();

System.out.println("╟──────────────────────────────────╢");
                        System.out.println("╟ Altere a IDADE da Pessoa
Fisica ╢");
                        int idadePessoaF = scanner.nextInt();
                        PessoaFisica cadastroCompleto = new
PessoaFisica(idPessoaF, nomePessoaF, cpfPessoaF, idadePessoaF);
                        repo1.alterar(idPessoaF, cadastroCompleto);
                        System.out.println("╟        CADASTRO ALTERADO
╢");

System.out.println("╚══════════════════════════════════╝");


                    } else if (tipoPessoa.equals("J")) {

System.out.println("╔══════════════════════════════════╗");
                        System.out.println("╢ Opção (J) selecionada:
Juridico ╟");

System.out.println("╟──────────────────────────────────╢");
                        System.out.println("╟  Digite o ID da Pessoa
Juridica ╢");
                        int idPessoaJ = scanner.nextInt();
                        repo3.obter(idPessoaJ);
                        System.out.println("╟          DADOS ATUAIS
╢");

System.out.println("╟──────────────────────────────────╢");
                        System.out.println("╢
╟");
                        System.out.println("╢
╟");

System.out.println("╟──────────────────────────────────╢");
                        System.out.println("╢         ALTERAR DADOS
╟");

System.out.println("╟──────────────────────────────────╢");
                        System.out.println("╟ Altere o NOME da Pessoa
Juridica╢");
                        scanner.nextLine();
                        String nomePessoaJ = scanner.nextLine();


System.out.println("╟──────────────────────────────────╢");
                        System.out.println("╟ Altere o CNPJ da Pessoa
Juridica╢");
                        String cnpjPessoaJ = scanner.next();
                        PessoaJuridica cadastroCompleto = new
PessoaJuridica(idPessoaJ, nomePessoaJ, cnpjPessoaJ);
                        repo3.alterar(idPessoaJ, cadastroCompleto);
                        System.out.println("╟        CADASTRO ALTERADO
╢");

System.out.println("╚══════════════════════════════════╝");
                    } else {
                        System.out.println("Opção Inválida");
                    }
```

```
                    break;
                case 3:

System.out.println("╔═══════════════════════════════════════╗");
                    System.out.println("║ Opção (3)     Excluir
║");

System.out.println("╠═══════════════════════════════════════╣");
                    System.out.println("╠ (\u001B[1mF\u001B[0m) -
Pessoa Fisica           ╣");
                    System.out.println("╠══
══╣");
                    System.out.println("╠ (\u001B[1mJ\u001B[0m) -
Pessoa Juridica         ╣");

System.out.println("╚═══════════════════════════════════════╝");
                    tipoPessoa = scanner.next().toUpperCase();
                    if (tipoPessoa.equals("F")) {

System.out.println("╔═══════════════════════════════════════╗");
                        System.out.println("║ Opção (F) selecionada:
Fisico    ║");

System.out.println("╠═══════════════════════════════════════╣");
                        System.out.println("╠   Digite o ID da Pessoa
Fisica  ╣");
                        int idPessoaF = scanner.nextInt();
                        repo1.excluir(idPessoaF);
                        System.out.println("╠        EXCLUSÃO COMPLETA
╣");

System.out.println("╚═══════════════════════════════════════╝");


                    } else if (tipoPessoa.equals("J")) {

System.out.println("╔═══════════════════════════════════════╗");
                        System.out.println("║ Opção (J) selecionada:
Juridico ║");

System.out.println("╠═══════════════════════════════════════╣");
                        System.out.println("╠  Digite o ID da Pessoa
Juridica ╣");
                        int idPessoaJ = scanner.nextInt();
                        repo3.excluir(idPessoaJ);
                        System.out.println("╠        EXCLUSÃO COMPLETA
╣");

System.out.println("╚═══════════════════════════════════════╝");
                    } else {
                        System.out.println("Opção Inválida");
                    }
                    break;
                case 4:

System.out.println("╔═══════════════════════════════════════╗");
                    System.out.println("║ Opção (4)     Exibir Por ID
║");

System.out.println("╠═══════════════════════════════════════╣");
                    System.out.println("╠ (\u001B[1mF\u001B[0m) -
```

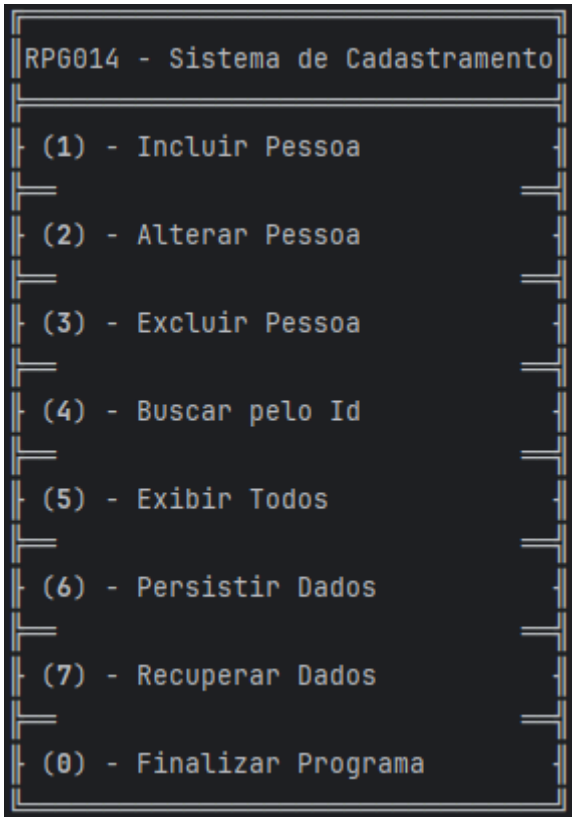
```
Pessoa Fisica             ╢");
                    System.out.println("╠══
══╢");
                    System.out.println("╠ (\u001B[1mJ\u001B[0m) -
Pessoa Juridica           ╢");

System.out.println("╚═══════════════════════════════════╝");
                    tipoPessoa = scanner.next().toUpperCase();
                    if (tipoPessoa.equals("F")) {

System.out.println("╔═══════════════════════════════════╗");
                        System.out.println("╢ Opção (F) selecionada:
Fisico    ╟");

System.out.println("╠═══════════════════════════════════╣");
                        System.out.println("╠   Digite o ID da Pessoa
Fisica  ╢");
                        int idPessoaF = scanner.nextInt();
                        repo1.obter(idPessoaF);
                        System.out.println("╠        EXIBIÇÃO COMPLETA
╢");

System.out.println("╚═══════════════════════════════════╝");


                    } else if (tipoPessoa.equals("J")) {

System.out.println("╔═══════════════════════════════════╗");
                        System.out.println("╢ Opção (J) selecionada:
Juridico ╟");

System.out.println("╠═══════════════════════════════════╣");
                        System.out.println("╠  Digite o ID da Pessoa
Juridica ╢");
                        int idPessoaJ = scanner.nextInt();
                        repo3.obter(idPessoaJ);
                        System.out.println("╠        EXIBIÇÃO COMPLETA
╢");

System.out.println("╚═══════════════════════════════════╝");
                    } else {
                        System.out.println("Opção Inválida");
                    }
                    break;

                case 5:

System.out.println("╔═══════════════════════════════════╗");
                    System.out.println("╢ Opção (5)      Exibir Todos
╟");

System.out.println("╠═══════════════════════════════════╣");
                    System.out.println("╠ (\u001B[1mF\u001B[0m) -
Pessoa Fisica             ╢");
                    System.out.println("╠══
══╢");
                    System.out.println("╠ (\u001B[1mJ\u001B[0m) -
Pessoa Juridica           ╢");

System.out.println("╚═══════════════════════════════════╝");
                    tipoPessoa = scanner.next().toUpperCase();
```

```
                    if (tipoPessoa.equals("F")) {

System.out.println("╔═══════════════════════════════════╗");
                        System.out.println("║ Opção (F) selecionada:
Fisico    ║");

System.out.println("╠═══════════════════════════════════╣");
                        System.out.println("╠═══  LISTA DE PESSOAS
FISICAS ═══╣");
                        repo1.obterTodos();
                        System.out.println("╠        EXIBIÇÃO COMPLETA
╣");

System.out.println("╚═══════════════════════════════════╝");


                    } else if (tipoPessoa.equals("J")) {

System.out.println("╔═══════════════════════════════════╗");
                        System.out.println("║ Opção (J) selecionada:
Juridico  ║");

System.out.println("╠═══════════════════════════════════╣");
                        System.out.println("╠═══LISTA DE PESSOAS
JURIDICAS═══╣");
                        repo3.obterTodos();
                        System.out.println("╠        EXIBIÇÃO COMPLETA
╣");

System.out.println("╚═══════════════════════════════════╝");
                    } else {
                        System.out.println("Opção Inválida");
                    }
                    break;
                case 6:

System.out.println("╔═══════════════════════════════════╗");
                    System.out.println("║ Opção (6)         Persistir
║");

System.out.println("╠═══════════════════════════════════╣");
                    System.out.println("╠ (\u001B[1mF\u001B[0m) -
Pessoa Fisica             ╣");
                    System.out.println("╠══
══╣");
                    System.out.println("╠ (\u001B[1mJ\u001B[0m) -
Pessoa Juridica           ╣");

System.out.println("╚═══════════════════════════════════╝");
                    tipoPessoa = scanner.next().toUpperCase();
                    if (tipoPessoa.equals("F")) {

System.out.println("╔═══════════════════════════════════╗");
                        System.out.println("║ Opção (F) selecionada:
Fisico    ║");

System.out.println("╠═══════════════════════════════════╣");
                        repo1.persistir(arquivoDeDadosPessoaFisica);
                        System.out.println("╠       SALVEMENTO
COMPLETO       ╣");
```

```
System.out.println("╚═══════════════════════════════════════╝");


                    } else if (tipoPessoa.equals("J")) {

System.out.println("╔═══════════════════════════════════════╗");
                        System.out.println("║ Opção (J) selecionada:
Juridico ║");

System.out.println("╠═══════════════════════════════════════╣");
                        repo3.persistir(arquivoDeDadosPessoaJuridica);
                        System.out.println("╠ Digite o ID da Pessoa
Juridica  ╣");
                        System.out.println("╠       SALVEMENTO
COMPLETO       ╣");

System.out.println("╚═══════════════════════════════════════╝");
                    } else {
                        System.out.println("Opção Inválida");
                    }
                    break;
                case 7:

System.out.println("╔═══════════════════════════════════════╗");
                    System.out.println("║ Opção (7)       Recuperar
║");

System.out.println("╠═══════════════════════════════════════╣");
                    System.out.println("╠ (\u001B[1mF\u001B[0m) -
Pessoa Fisica           ╣");
                    System.out.println("╠══
══╣");
                    System.out.println("╠ (\u001B[1mJ\u001B[0m) -
Pessoa Juridica           ╣");

System.out.println("╚═══════════════════════════════════════╝");
                    tipoPessoa = scanner.next().toUpperCase();
                    if (tipoPessoa.equals("F")) {

System.out.println("╔═══════════════════════════════════════╗");
                        System.out.println("║ Opção (F) selecionada:
Fisico    ║");

System.out.println("╠═══════════════════════════════════════╣");
                        System.out.println("╠═══  LISTA DE PESSOAS
FISICAS ═══╣");
                        repo1.recuperar(arquivoDeDadosPessoaFisica);
                        System.out.println("╠      RECUPERAÇÃO
COMPLETA       ╣");

System.out.println("╚═══════════════════════════════════════╝");


                    } else if (tipoPessoa.equals("J")) {

System.out.println("╔═══════════════════════════════════════╗");
                        System.out.println("║ Opção (J) selecionada:
Juridico ║");

System.out.println("╠═══════════════════════════════════════╣");
                        System.out.println("╠═══LISTA DE PESSOAS
```

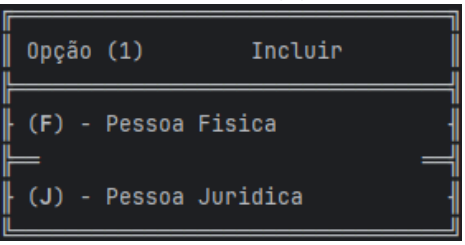
```
JURIDICAS═══╣");
                        repo3.recuperar(arquivoDeDadosPessoaJuridica);
                        System.out.println("╠      RECUPERAÇÃO
COMPLETA       ╣");

System.out.println("╚═══════════════════════════════════════╝");
                    } else {
                        System.out.println("Opção Inválida");
                    }

                    break;
                }

            }

        }
}
```

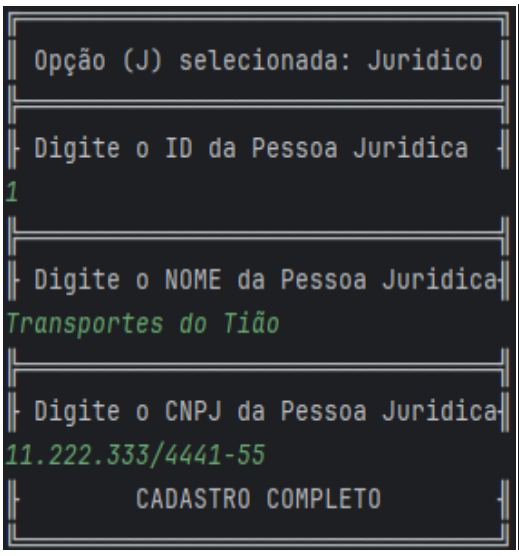
Resultados

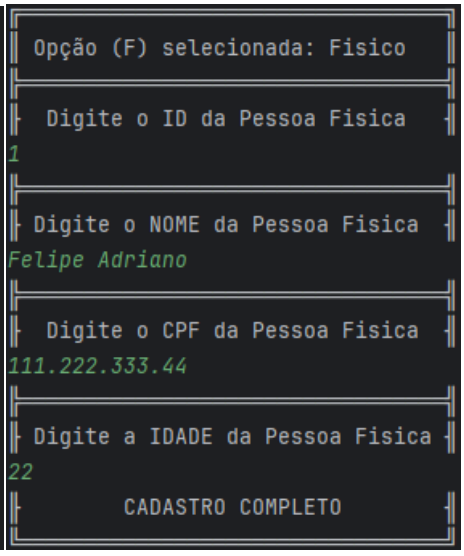


---

(1) Incluir Pessoa

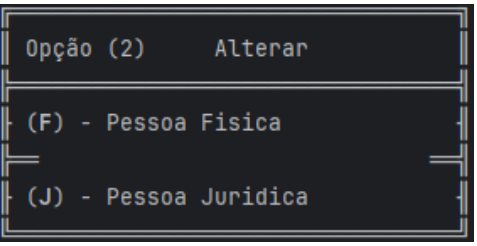


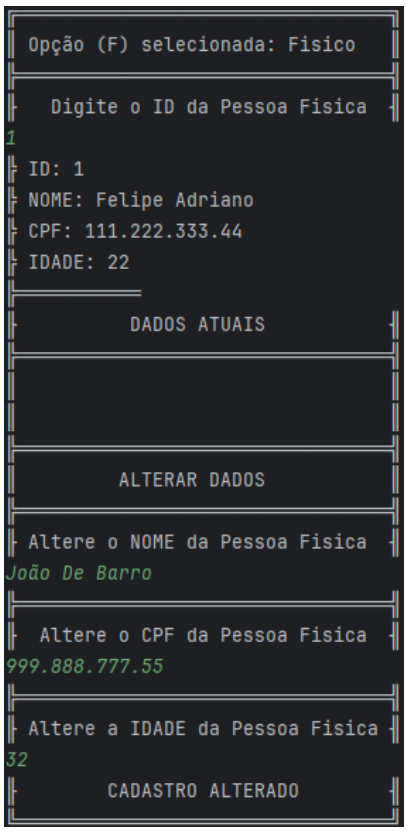


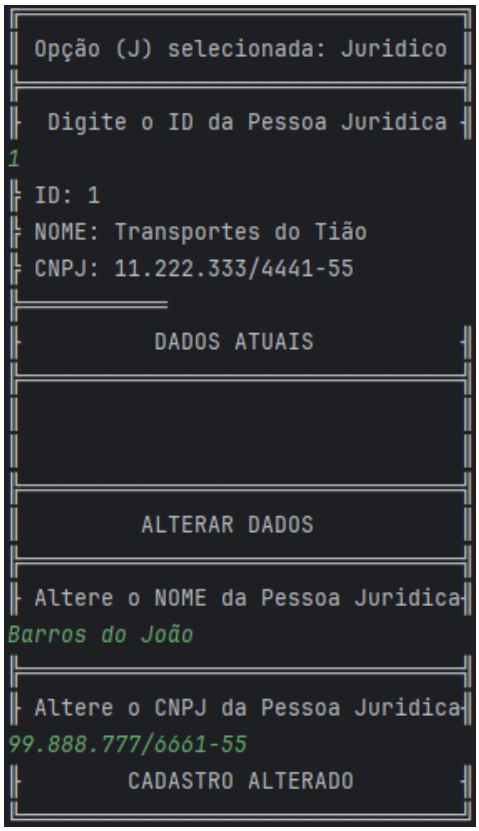


---

(2) Alterar Pessoa

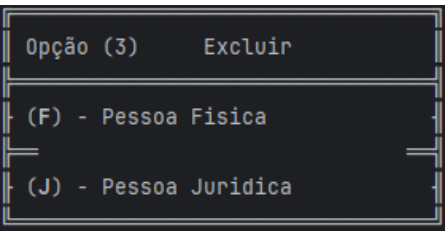


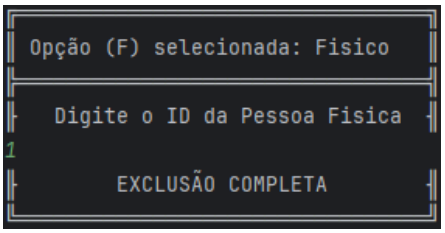


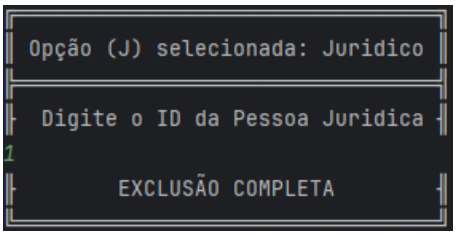


---

(3) Excluir Pessoa

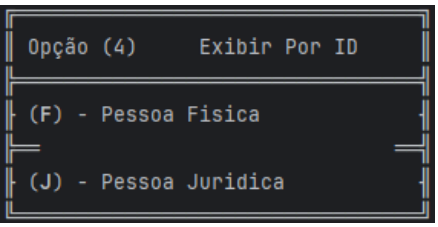


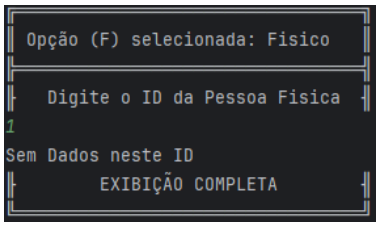


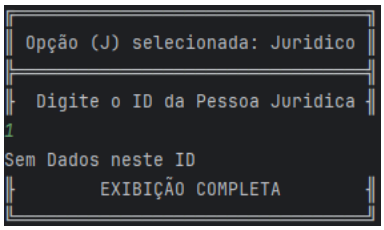


---

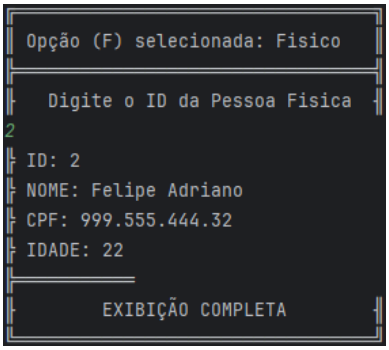
(4) Buscar pelo Id

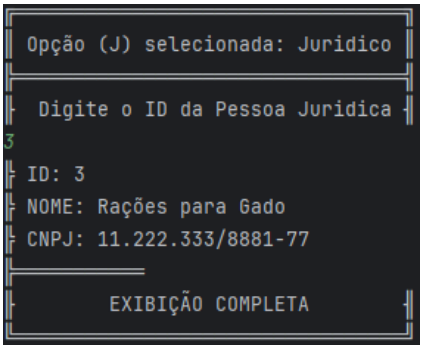






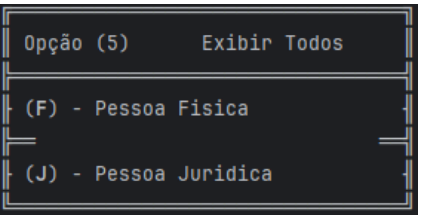
OBS: Como o passo 3 foi excluído os dados, aqui não havia mais, porém por conta disto, foi criado mais uma pessoa Física e mais uma Jurídica com outros IDs para melhorar a explicação.

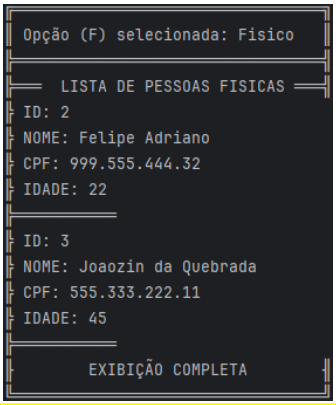


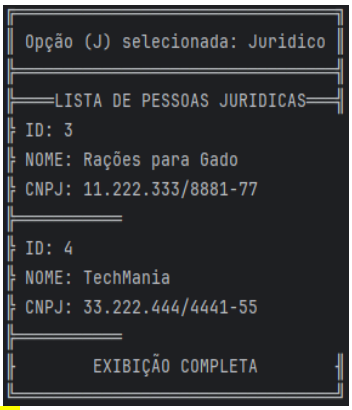


---

(5) Exibir Todos

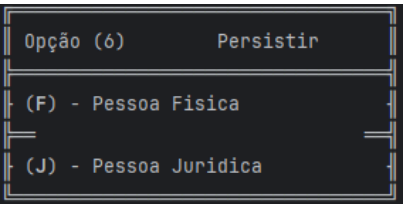


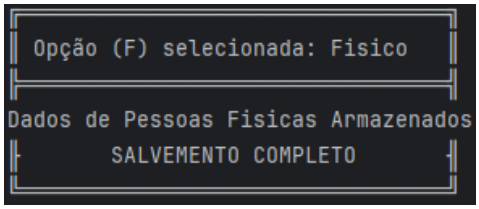


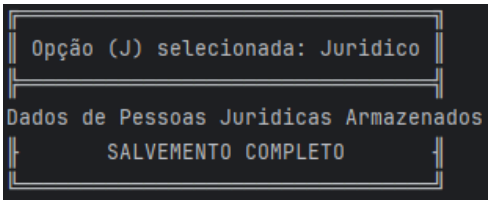


Foram acrescentados mais um em cada.

(6) Persistir Dados

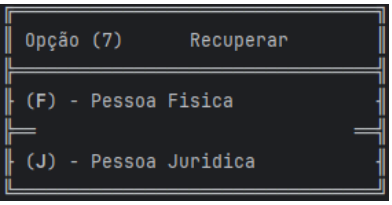


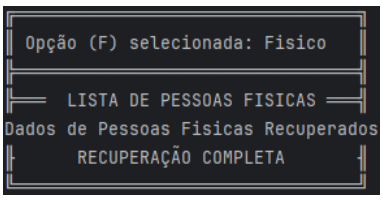


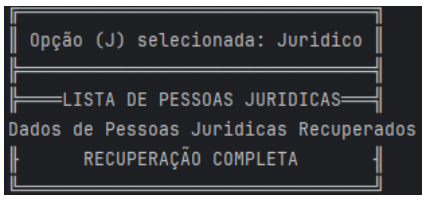


---

(7) Recuperar Dados







Análise e Conclusão da segunda etapa.

O uso de elementos estáticos como o método “CadastroPOO”, permite que o programa Java seja iniciado sem a necessidade de criar um objeto da classe, tornando ele um ponto de entrada para o programa Java. A classe Scanner implantada no método principal é uma ferramenta muito útil para obter a entrada do usuário a partir do próprio console.

O uso de classes de repositório para a organização do código auxiliou muito na organização, pois caso seja necessário a alteração de algo no projeto, o desenvolvedor já tem em mente que é ali que as coisas estão, em tal repositório ou módulo, tirando ele de um labirinto o orientando, facilitando a manutenção, organização e orientação do projeto.

Estácio

Aluno: Felipe Adriano

IDEA utilizada: IntelliJ IDEA Community Edition

Linguagem: Java

Nível 1: Iniciando o Caminho Pelo Java
RPG0014 – Iniciando o caminho pelo Java