Estácio

Polo Tijuca

Aluno	Luiz Felippe Sarmento Bonet
Matrícula	2023.09.34276-6
Curso	Desenvolvimento Full Stack
Disciplina	RPG0014 - Iniciando o caminho pelo Java
Turma	9001
Período	2024 – 2º Semestre

Missão Prática: Mundo 3 - Nível 1

Título da Prática 1: Criação das Entidades e Sistema de Persistência

Título da Prática 2: Criação do Cadastro em Modo Texto

Objetivos:

- Utilizar herança e polimorfismo na definição de entidades;
- Utilizar persistência de objetos em arquivos binários;
- Implementar uma interface cadastral em modo texto;
- Utilizar o controle de exceções da plataforma Java;
- Implementar um sistema cadastral em Java utilizando os recursos da programação orientada a objetos e persistência em arquivos binários.

Resultados de Execução – Prática 1

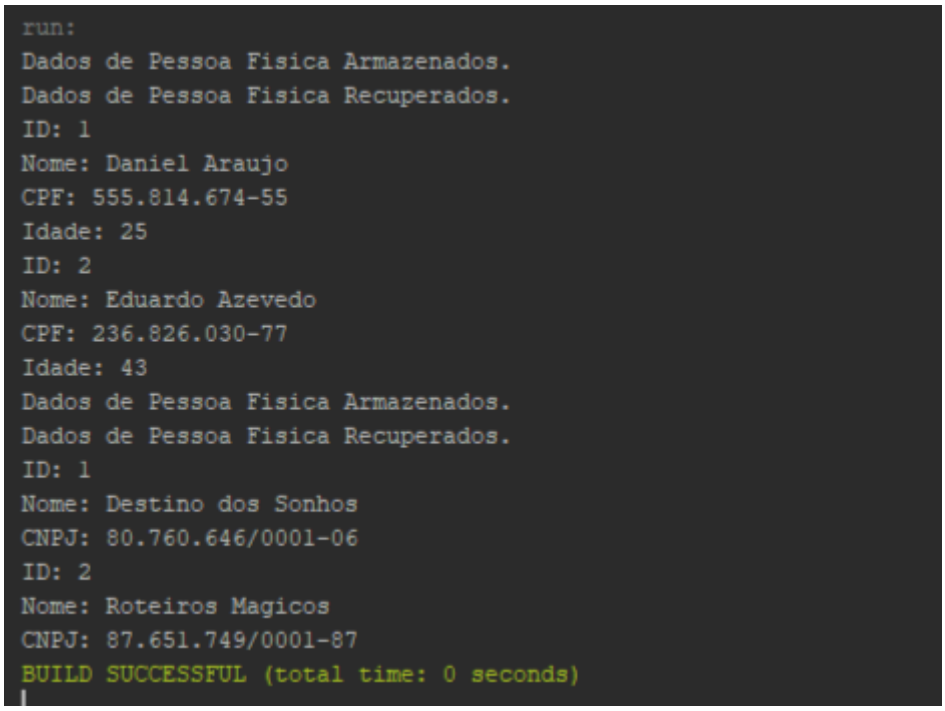

Figura 1. A figura mostra o resultado da execução do código, onde os dados são salvos em repositórios, armazenados, recuperados e então exibidos.

Resultados de Execução – Prática 2

```
run:
========================
1 - Incluir Pessoa
2 - Alterar Pessoa
3 - Excluir Pessoa
4 - Buscar pelo ID
5 - Exibir Todos
6 - Persistir Dados
7 - Recuperar Todos
0 - Finalizar Programa
========================
1
F - Pessoa Fisica | J - Pessoa Juridica
F
Digite o ID da Pessoa Fisica:
1
Digite o Nome da Pessoa Fisica:
Mateus Fernandes
Digite o CPF da Pessoa Fisica:
291.537.366-30
Digite a Idade da Pessoa Fisica:
33
Pessoa Fisica cadastrada com sucesso.
```

```
1
F - Pessoa Fisica | J - Pessoa Juridica
J
Digite o ID da Pessoa Juridica:
1
Digite o Nome da Pessoa Juridica:
Gravadora GRAVE
Digite o CNPJ da Pessoa Juridica:
48.327.886/0001-09
Pessoa Juridica cadastrada com sucesso.
```

Figura 2. A imagem mostra o resultado da execução do comando Incluir Pessoa. Foram cadastradas 3 pessoas físicas e 2 pessoas jurídicas.

```
=========================
1 - Incluir Pessoa
2 - Alterar Pessoa
3 - Excluir Pessoa
4 - Buscar pelo ID
5 - Exibir Todos
6 - Persistir Dados
7 - Recuperar Todos
0 - Finalizar Programa
=========================
2
F - Pessoa Fisica | J - Pessoa Juridica
F
Digite o ID da Pessoa a alterar:
2
Nome atual: Luan Almeida
Novo nome: Victor Rocha
CPF atual: 502.606.339-09
Novo CPF: 112.247.848-80
Idade atual: 43
Nova idade: 37
Cadastro de Pessoa Fisica alterado.
```

Figura 3. A imagem mostra o resultado da execução do comando Alterar Pessoa. No exemplo, foram alteradas a informações da pessoa física com ID 2.

```
=========================
1 - Incluir Pessoa
2 - Alterar Pessoa
3 - Excluir Pessoa
4 - Buscar pelo ID
5 - Exibir Todos
6 - Persistir Dados
7 - Recuperar Todos
0 - Finalizar Programa
=========================
3
F - Pessoa Fisica | J - Pessoa Juridica
F
Digite o ID da Pessoa a deletar:
3
Cadastro de Pessoa Fisica excluido.
```

Figura 4. A imagem mostra o resultado da execução do comando Excluir Pessoa. Neste exemplo, foi excluída a pessoa física com ID 3.

```
0 - Finalizar Programa
========================
4
F - Pessoa Fisica | J - Pessoa Juridica
F
Digite o ID da Pessoa a buscar:
1
ID: 1
Nome: Mateus Fernandes
CPF: 291.537.366-30
Idade: 33
Pessoa Fisica encontrada.
```

```
0 - Finalizar Programa
========================
4
F - Pessoa Fisica | J - Pessoa Juridica
J
Digite o ID da Pessoa a buscar:
1
ID: 1
Nome: Gravadora GRAVE
CNPJ: 48.327.886/0001-09
Pessoa Juridica encontrada.
```

Figura 5. A imagem mostra o resultado da execução do comando Buscar Pessoa. O exemplo mostra apenas uma busca de cada tipo de pessoa.

```
5
F - Pessoa Fisica | J - Pessoa Juridica
F
ID: 1
Nome: Mateus Fernandes
CPF: 291.537.366-30
Idade: 33
ID: 2
Nome: Victor Rocha
CPF: 112.247.848-80
Idade: 37
Todas as Pessoa Fisicas foram listadas.
```

```
5
F - Pessoa Fisica | J - Pessoa Juridica
J
ID: 1
Nome: Gravadora GRAVE
CNPJ: 48.327.886/0001-09
ID: 2
Nome: Elek-Tech
CNPJ: 92.169.596/001-76
Todas as Pessoas Juridicas foram listadas.
```

Figura 6. A imagem mostra o resultado da execução do comando Exibir Todos, mostrando todos os cadastros de pessoas físicas e pessoas jurídicas.

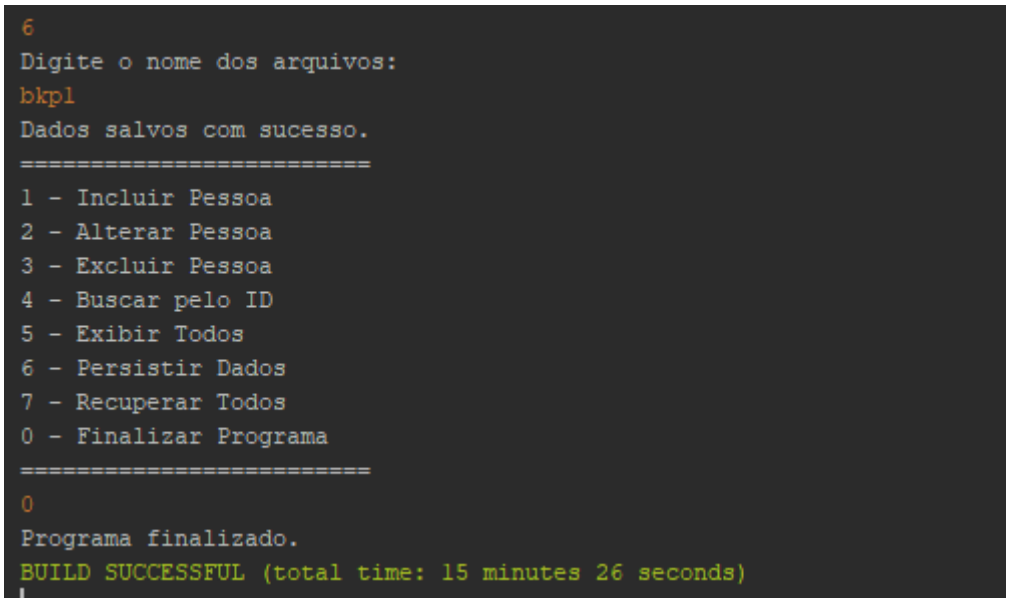

Figura 7. A imagem mostra o resultado da execução dos comandos Persistir Dados e Finalizar Programa. Neste exemplo, os dados dos repositórios foram salvos sob o nome “bkp1” e o programa foi encerrado.

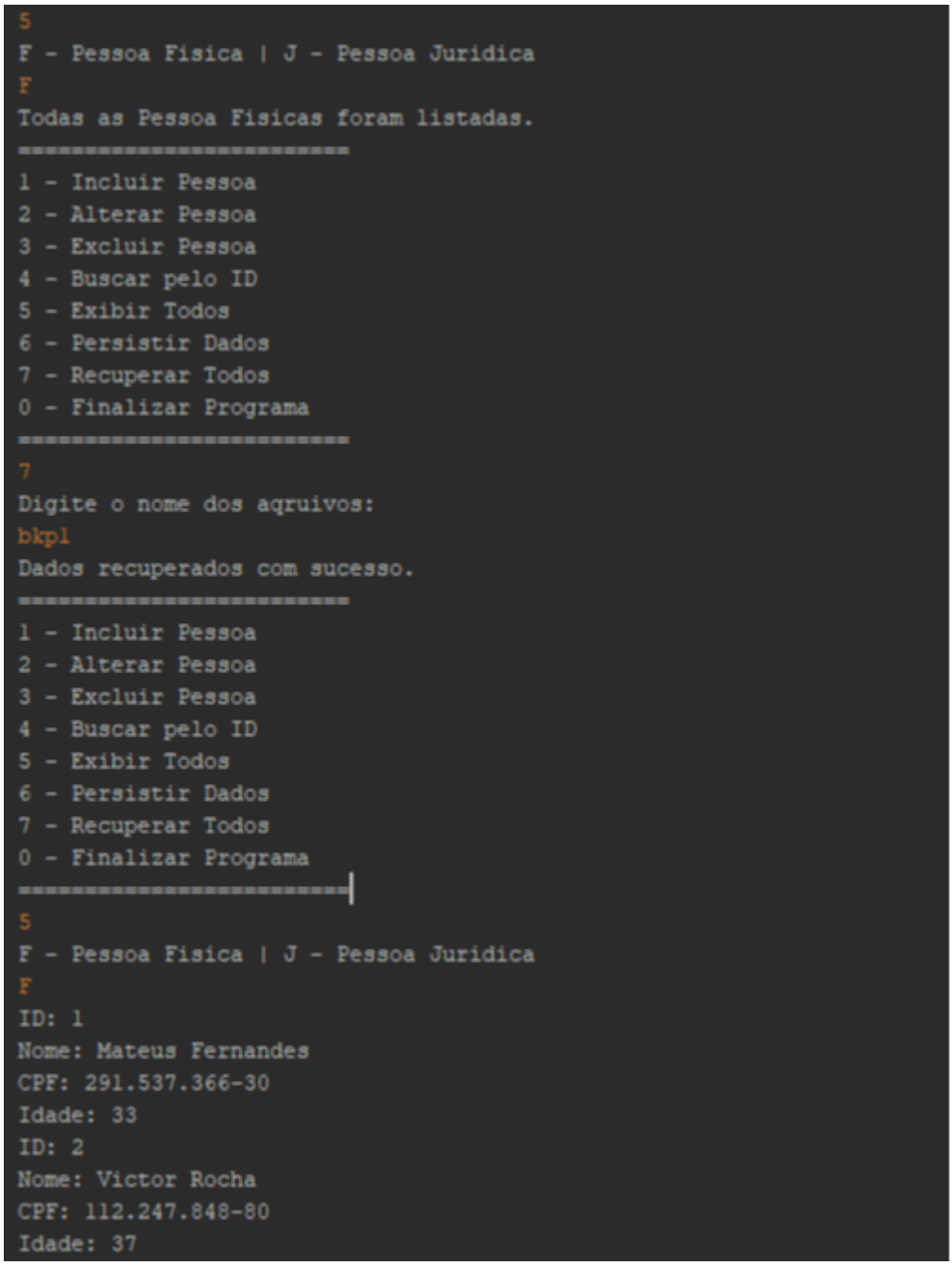

Figura 8. A imagem mostra o resultado da execução do comando Recuperar Todos. Neste exemplo, observa-se que, após iniciar nova instância do programa, não há nenhum cadastro para ser mostrado através do comando Exibir Todos. Depois da recuperação dos dados armazenados previamente, através do comando Recuperar Todos, é possível ver que agora existem cadastros para serem exibidos.

Análise e Conclusão – Prática 1

A) Quais as vantagens e desvantagens do uso de herança?

Vantagens: Herança em Java permite que classes filhas herdem métodos e atributos de classes mães. Essa ferramenta pode deixar o código mais rápido e fácil de escrever, economizando tempo do programador na implementação e na manutenção do código. Quando bem utilizada, também é possível criar classes funcionalmente mais complexas, mas cujo código é mais simples.

Desvantagens: O uso inconsequente de herança pode gerar um acoplamento muito forte entre classes, de modo que alterações simples na classe mãe podem gerar consequências adversas nas classes filhas, assim como também pode gerar sobrecarga de métodos. Tudo isso pode impactar negativamente o código, especialmente sua manutenção.

B) Porque a interface *Serializable* é necessária ao efetuar persistência em arquivos binários?

O uso da interface *Serializable* é necessário pois ela permite que objetos sejam transformados em sequências (de onde vem o nome) de *bytes*. O inverso também é verdade, permitindo a conversão de sequências de *bytes* em objetos. Uma vez convertida, a sequência pode ser armazenada para uso posterior, ou transmitida por *stream*.

C) Como o paradigma funcional é utilizado pela *API Stream* no Java?

O paradigma funcional, ou seja, o foco em aplicação de funções matemáticas na programação, se dá através de funções anônimas, funções lambda no Java. Com essa ferramenta é possível processar sequências de elementos e encadear operações em sequências de objetos. Isso facilita tanto a compreensão do código quanto sua escrita.

D) Quando trabalhamos com Java, qual o padrão de desenvolvimento é adotado na persistência de dados em arquivos?

Além da técnica de serialização, como abordada na questão sobre *Serializable*, existem outros possíveis padrões de desenvolvimento. Por exemplo, *Data Access Object*, que separa a lógica de acesso da lógica de dados de negócio. Também é possível utilizar bibliotecas como *Java Persistence API*, que facilita a interação com bancos de dados, ou mesmo o uso direto de *streams* de saída e entrada.

Análise e Conclusão – Prática 2

A) O que são elementos estáticos e qual o motivo para o método *main* adotar esse modificador?

Elementos estáticos são aqueles que, em vez de pertencerem a instâncias individuais de uma classe, pertencem à classe em si. Dessa forma, estão associados à classe como um todo. Este tipo de elemento é marcado pelo modificador *static*.

O *main* utiliza este método para que, em vez de haver a necessidade de instanciar a classe para rodar o programa, ele possa ser acessado diretamente pela JVM.

B) Para que serve a classe *Scanner*?

A classe *Scanner* é uma ferramenta simples que permite a entrada de dados a partir do console durante a execução do programa.

C) Como o uso de classes de repositório impactou na organização do código?

As classes repositório foram criadas para servirem como intermediárias entre os dados armazenados e o programa, em vez de deixarem alguma outra classe desnecessariamente “inchada”. A criação destas classes deixa o código como um todo mais organizado. Dessa forma, o impacto da utilização de classes repositório foi positivo, até mesmo deixando o projeto em conformidade com as boas práticas de programação

Anexo – Códigos do Projeto

Arquivo – Pessoa.java

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package model;


import java.io.Serializable;


/**
 *
 * @author Felippe
 */


public class Pessoa implements Serializable {
    private static final long serialVersionUID = 1L;
    private int id;
    private String nome;


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


    public void setID(int id) {
        this.id = id;
```

```
    }

    public int getID() {
        return id;
    }

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

    public String getNome() {
        return nome;
    }

    public void exibir() {
        System.out.println("ID: " + getID());
        System.out.println("Nome: " + getNome());
    }
}
```

Arquivo PessoaFisica.java

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package model;

import java.io.Serializable;
```

```
/**
 *
 * @author Felippe
 */

public class PessoaFisica extends Pessoa implements Serializable {
    private static final long serialVersionUID = 1L;
    private String cpf;
    private int idade;

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

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

    public String getCPF() {
        return cpf;
    }

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

```
    public int getIdade() {
        return idade;
    }

    @Override
    public void exibir() {
        super.exibir();
        System.out.println("CPF: " + getCPF());
        System.out.println("Idade: " + getIdade());
    }
}
```

Arquivo PessoaJuridica.java

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package model;

import java.io.Serializable;

/**
 *
 * @author Felippe
 */

public class PessoaJuridica extends Pessoa implements Serializable {
    private static final long serialVersionUID = 1L;
```

```
    private String cnpj;

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

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

    public String getCNPJ() {
        return cnpj;
    }

    @Override
    public void exibir() {
        super.exibir();
        System.out.println("CNPJ: " + getCNPJ());
    }
}
```

Arquivo PessoaFisicaRepo.java

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package model;
```

```
import java.util.ArrayList;
import java.util.List;
import java.io.Serializable;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.ObjectInputStream;
import java.io.ObjectOutputStream;
import java.io.IOException;

/**
 *
 * @author Felippe
 */

public class PessoaFisicaRepo implements Serializable {
    private static final long serialVersionUID = 1L;
    private final List<PessoaFisica> listaPF = new ArrayList<>();

    public void inserir(PessoaFisica pF) {
        listaPF.add(pF);
    }

    public PessoaFisica obter(int id) {
        for (PessoaFisica pF : listaPF) {
            if (pF.getID() == id) {
                return pF;
            }
        }
```

```
        return null;
    }

    public void alterar(PessoaFisica pF) {
        for (int i = 0; i < listaPF.size(); i++) {
            if (pF.getID() == listaPF.get(i).getID()) {
                listaPF.set(i, pF);
                return;
            }
        }
    }

    public void excluir(int id) {
        for (int i = 0; i < listaPF.size(); i++) {
            if (listaPF.get(i).getID() == id) {
                listaPF.remove(i);
                return;
            }
        }
    }

    public List<PessoaFisica> obterTodos() {
        return listaPF;
    }

    public void persistir(String filename) throws IOException {
        try (ObjectOutputStream OOS = new ObjectOutputStream (new
FileOutputStream (filename))) {
            OOS.writeObject(listaPF);
```

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


    public void recuperar(String filename) throws IOException,
ClassNotFoundException {
        try (ObjectInputStream OIS = new ObjectInputStream(new
FileInputStream(filename))) {
            listaPF.clear();
            List<PessoaFisica> listaRec = (List<PessoaFisica>) OIS.readObject();
            listaPF.addAll(listaRec);
        }
        catch (IOException e) {
            throw e;
        }
    }
}
```

Arquivo PessoaJuridicaRepo.java

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt
to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit
this template
 */
package model;


import java.util.ArrayList;
```

```
import java.util.List;
import java.io.Serializable;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.ObjectInputStream;
import java.io.ObjectOutputStream;
import java.io.IOException;

/**
 *
 * @author Felippe
 */

public class PessoaJuridicaRepo implements Serializable {
    private static final long serialVersionUID = 1L;
    private final List<PessoaJuridica> listaPJ = new ArrayList<>();

    public void inserir(PessoaJuridica pJ) {
        listaPJ.add(pJ);
    }

    public PessoaJuridica obter(int id) {
        for (PessoaJuridica pJ : listaPJ) {
            if (pJ.getID() == id) {
                return pJ;
            }
        }
        return null;
    }
```

```
    public void alterar(PessoaJuridica pJ) {
        for (int i = 0; i < listaPJ.size(); i++) {
            if (pJ.getID() == listaPJ.get(i).getID()) {
                listaPJ.set(i, pJ);
                return;
            }
        }
    }

    public void excluir(int id) {
        for (int i = 0; i < listaPJ.size(); i++) {
            if (listaPJ.get(i).getID() == id) {
                listaPJ.remove(i);
                return;
            }
        }
    }

    public List<PessoaJuridica> obterTodos() {
        return listaPJ;
    }

    public void persistir(String filename) throws IOException {
        try (ObjectOutputStream OOS = new ObjectOutputStream (new
FileOutputStream (filename))) {
            OOS.writeObject(listaPJ);
        }
        catch (IOException e) {
```

```
            throw e;
        }
    }


    public void recuperar(String filename) throws IOException,
ClassNotFoundException {
        try (ObjectInputStream OIS = new ObjectInputStream(new
FileInputStream(filename))) {
            listaPJ.clear();
            List<PessoaJuridica> listaRec = (List<PessoaJuridica>)
OIS.readObject();
            listaPJ.addAll(listaRec);
        }
        catch (IOException e) {
            throw e;
        }
    }
}
```

Arquivo CadastroPoo.java

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt
to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Main.java to edit
this template
 */
package cadastropoo;


import model.PessoaFisica;
import model.PessoaFisicaRepo;
import model.PessoaJuridica;
```

```
import model.PessoaJuridicaRepo;


/**
 *
 * @author Felippe
 */


public class CadastroPOO {


    /**
    * @param args the command line arguments
    * @throws java.lang.Exception
    */


public static void main(String[] args) throws Exception {
        PessoaFisicaRepo repo1 = new PessoaFisicaRepo();
        PessoaFisica fis1 = new PessoaFisica(1, "Daniel Araujo", "555.814.674-
55", 25);
        PessoaFisica fis2 = new PessoaFisica(2, "Eduardo Azevedo",
"236.826.030-77", 43);
        repo1.inserir(fis1);
        repo1.inserir(fis2);
        repo1.persistir("dadosPF");
        System.out.println("Dados de Pessoa Fisica Armazenados.");
        PessoaFisicaRepo repo2 = new PessoaFisicaRepo();
        repo2.recuperar("dadosPF");
        System.out.println("Dados de Pessoa Fisica Recuperados.");
        for (PessoaFisica pF : repo2.obterTodos()) {
            pF.exibir();
        }
```

```
        PessoaJuridicaRepo repo3 = new PessoaJuridicaRepo();
        PessoaJuridica jur1 = new PessoaJuridica(1, "Destino dos Sonhos",
"80.760.646/0001-06");
        PessoaJuridica jur2 = new PessoaJuridica(2, "Roteiros Magicos",
"87.651.749/0001-87");
        repo3.inserir(jur1);
        repo3.inserir(jur2);
        repo3.persistir("dadosPJ");
        System.out.println("Dados de Pessoa Fisica Armazenados.");
        PessoaJuridicaRepo repo4 = new PessoaJuridicaRepo();
        repo4.recuperar("dadosPJ");
        System.out.println("Dados de Pessoa Fisica Recuperados.");
        for  (PessoaJuridica pJ : repo4.obterTodos()) {
            pJ.exibir();
        }
    }

}
```

Arquivo CadastroPOO2.java

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt
to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Main.java to edit
this template
 */
package cadastropoo;


import java.io.BufferedReader;
import java.io.IOException;
```

```
import java.io.InputStreamReader;
import model.PessoaFisica;
import model.PessoaFisicaRepo;
import model.PessoaJuridica;
import model.PessoaJuridicaRepo;

/**
 *
 * @author Felippe
 */

public class CadastroPOO2 {

    /**
    * @param args the command line arguments
    */

   public static void main(String[] args) {
        BufferedReader reader = new BufferedReader(new
InputStreamReader(System.in));
        String option = "";
        PessoaFisicaRepo fisRepo = new PessoaFisicaRepo();
        PessoaJuridicaRepo jurRepo = new PessoaJuridicaRepo();

       while (!option.equals("0")) {
                optionMenu();
                try {
                        option = reader.readLine();
                        switch (option) {
```

```
                        case "1" ->  {
                                System.out.println("F - Pessoa Fisica | J -
Pessoa Juridica");
                                String pessoaType = reader.readLine();
                                switch (pessoaType) {
                                        case "F" ->  {
                                                PessoaFisica pF =
fisEnter(reader);
                                                fisRepo.inserir(pF);
                                                System.out.println("Pessoa
Fisica cadastrada com sucesso.");
                                        }
                                        case "J" ->  {
                                                PessoaJuridica pJ =
jurEnter(reader);
                                                jurRepo.inserir(pJ);
                                                System.out.println("Pessoa
Juridica cadastrada com sucesso.");
                                        }
                                        default -> {
                                                System.out.println("Tipo invalido
de cadastro.");
                                        }


                                }
                        }
                        case "2" ->  {
                                System.out.println("F - Pessoa Fisica | J -
Pessoa Juridica");
                                String pessoaType = reader.readLine();
                                switch (pessoaType) {
```

```
                                    case "F" ->  {
                                    pessoaAlter(fisRepo, reader);
                                            System.out.println("Cadastro de
Pessoa Fisica alterado.");
                                    }
                                    case "J" ->  {
                                            pessoaAlter(jurRepo, reader);
                                            System.out.println("Cadastro de
Pessoa Juridica alterado.");
                                    }
                                    default -> {
                                            System.out.println("Tipo invalido de
cadastro.");
                                    }
                            }
                            }
                            case "3" ->  {
                                    System.out.println("F - Pessoa Fisica | J -
Pessoa Juridica");
                                    String pessoaType = reader.readLine();
                                    switch (pessoaType) {
                                            case "F" ->  {
                                    pessoaDelete(fisRepo, reader);
                                                    System.out.println("Cadastro de
Pessoa Fisica excluido.");
                                            }
                                            case "J" ->  {
                                                    pessoaDelete(jurRepo, reader);
                                                    System.out.println("Cadastro de
Pessoa Juridica excluido.");
                                            }
```

```
                                        default -> {
                                                System.out.println("Tipo invalido
de cadastro.");
                                        }
                                }
                        }
                        case "4" ->  {
                                System.out.println("F - Pessoa Fisica | J -
Pessoa Juridica");
                                String pessoaType = reader.readLine();
                                switch (pessoaType) {
                                        case "F" ->  {
                                                pessoaSearch(fisRepo, reader);
                                                System.out.println("Pessoa
Fisica encontrada.");
                                        }
                                        case "J" ->  {
                                                pessoaSearch(jurRepo, reader);
                                                System.out.println("Pessoa
Juridica encontrada.");
                                        }
                                        default -> {
                                                System.out.println("Tipo invalido
de cadastro.");
                                        }
                                }
                        }
                        case "5" ->  {
                                System.out.println("F - Pessoa Fisica | J -
Pessoa Juridica");
                                String pessoaType = reader.readLine();
```

```
                                    switch (pessoaType) {
                                            case "F" ->  {
                                            pessoaShow(fisRepo);
                                                    System.out.println("Todas as
Pessoa Fisicas foram listadas.");
                                            }
                                            case "J" ->  {
                                                    pessoaShow(jurRepo);
                                                    System.out.println("Todas as
Pessoas Juridicas foram listadas.");
                                            }
                                            default -> {
                                                    System.out.println("Tipo invalido
de cadastro.");
                                            }
                                    }
                            }
                            case "6" ->  {
                                    System.out.println("Digite o nome dos
arquivos:");
                                    String filePersist = reader.readLine();
                                    try {
                            fisRepo.persistir(filePersist + ".fisica.bin");
                                            jurRepo.persistir(filePersist +
".juridica.bin");
                                            System.out.println("Dados salvos com
sucesso.");
                                    }
                                    catch (IOException e) {
                                            System.out.println("Ocorreu um erro ao
salvar os dados: " + e.getMessage());
```

```
                                        }
                                }
                                case "7" ->  {
                                        System.out.println("Digite o nome dos
aqruivos:");
                                        String fileRestore = reader.readLine();
                                        try {
                                                fisRepo.recuperar(fileRestore +
".fisica.bin");
                                                jurRepo.recuperar(fileRestore +
".juridica.bin");
                                                System.out.println("Dados recuperados
com sucesso.");
                                        }
                                        catch (IOException | ClassNotFoundException
e) {
                                                System.out.println("Ocorreu um erro ao
recuperar os dados: " + e.getMessage());
                                        }
                                }
                                case "0" ->  {
                                        System.out.println("Programa finalizado.");
                                }
                                default ->  {
                                        System.out.println("Opção invalida.");
                                }
                        }
                }
                catch (IOException e) {
                        System.out.println("Ocorreu um erro no processo: " +
e.getMessage());
```

```
                }
        }
    }

        private static PessoaFisica fisEnter(BufferedReader reader) throws
IOException {
                System.out.println("Digite o ID da Pessoa Fisica: ");
                int id = Integer.parseInt(reader.readLine());
                System.out.println("Digite o Nome da Pessoa Fisica: ");
                String nome = reader.readLine();
                System.out.println("Digite o CPF da Pessoa Fisica: ");
                String cpf = reader.readLine();
                System.out.println("Digite a Idade da Pessoa Fisica: ");
                int idade = Integer.parseInt(reader.readLine());
                return new PessoaFisica(id, nome, cpf, idade);
        }

        private static PessoaJuridica jurEnter(BufferedReader reader) throws
IOException {
                System.out.println("Digite o ID da Pessoa Juridica: ");
                int id = Integer.parseInt(reader.readLine());
                System.out.println("Digite o Nome da Pessoa Juridica: ");
                String nome = reader.readLine();
                System.out.println("Digite o CNPJ da Pessoa Juridica: ");
                String cnpj = reader.readLine();
                return new PessoaJuridica(id, nome, cnpj);
        }

        private static void pessoaAlter(Object repo, BufferedReader reader)
throws IOException {
```

```
            System.out.println("Digite o ID da Pessoa a alterar: ");
            int id = Integer.parseInt(reader.readLine());
switch (repo) {
    case PessoaFisicaRepo pessoaFisicaRepo -> {
        PessoaFisica pF = pessoaFisicaRepo.obter(id);
        if (pF != null) {
            System.out.println("Nome atual: " + pF.getNome());
            System.out.print("Novo nome: ");
            pF.setNome(reader.readLine());
            System.out.println("CPF atual: " + pF.getCPF());
            System.out.print("Novo CPF: ");
            pF.setCPF(reader.readLine());
            System.out.println("Idade atual: " + pF.getIdade());
            System.out.print("Nova idade: ");
            pF.setIdade(Integer.parseInt(reader.readLine()));
            pessoaFisicaRepo.alterar(pF);
        } else {
            System.out.println("Pessoa Física não encontrada.");
        }
    }          case PessoaJuridicaRepo pessoaJuridicaRepo -> {
        PessoaJuridica pJ = pessoaJuridicaRepo.obter(id);
        if (pJ != null) {
            System.out.println("Nome atual: " + pJ.getNome());
            System.out.print("Novo nome: ");
            pJ.setNome(reader.readLine());
            System.out.println("CNPJ atual: " + pJ.getCNPJ());
            System.out.print("Novo CNPJ: ");
            pJ.setCNPJ(reader.readLine());
            pessoaJuridicaRepo.alterar(pJ);
```

```
            } else {
                System.out.println("Pessoa Jurídica não encontrada.");
            }
        }
        default -> {
        }
    }
    }

    private static void pessoaDelete(Object repo, BufferedReader reader)
throws IOException {
            System.out.println("Digite o ID da Pessoa a deletar: ");
            int id = Integer.parseInt(reader.readLine());
    switch (repo) {
        case PessoaFisicaRepo pessoaFisicaRepo ->
pessoaFisicaRepo.excluir(id);
        case PessoaJuridicaRepo pessoaJuridicaRepo ->
pessoaJuridicaRepo.excluir(id);
        default -> {
        }
    }
      }

    private static void pessoaSearch(Object repo, BufferedReader reader)
throws IOException {
            System.out.println("Digite o ID da Pessoa a buscar: ");
            int id = Integer.parseInt(reader.readLine());
    switch (repo) {
        case PessoaFisicaRepo pessoaFisicaRepo -> {
            PessoaFisica pF = pessoaFisicaRepo.obter(id);
```

```
            if (pF != null) {
                System.out.println("ID: " + pF.getID());
                System.out.println("Nome: " + pF.getNome());
                System.out.println("CPF: " + pF.getCPF());
                System.out.println("Idade: " + pF.getIdade());
            } else {
                System.out.println("Pessoa Fisica não encontrada.");
            }
        }
        case PessoaJuridicaRepo pessoaJuridicaRepo -> {
            PessoaJuridica pJ = pessoaJuridicaRepo.obter(id);
            if (pJ != null) {
                System.out.println("ID: " + pJ.getID());
                System.out.println("Nome: " + pJ.getNome());
                System.out.println("CNPJ: " + pJ.getCNPJ());
            } else {
                System.out.println("Pessoa Juridica não encontrada.");
            }
        }
        default -> {
        }
    }
      }

    private static void pessoaShow(Object repo) {
    switch (repo) {
        case PessoaFisicaRepo pessoaFisicaRepo -> {
            for (PessoaFisica pF : pessoaFisicaRepo.obterTodos()) {
                System.out.println("ID: " + pF.getID());
```

```
                System.out.println("Nome: " + pF.getNome());
                System.out.println("CPF: " + pF.getCPF());
                System.out.println("Idade: " + pF.getIdade());
            }
        }
        case PessoaJuridicaRepo pessoaJuridicaRepo -> {
            for (PessoaJuridica pJ : pessoaJuridicaRepo.obterTodos()) {
                System.out.println("ID: " + pJ.getID());
                System.out.println("Nome: " + pJ.getNome());
                System.out.println("CNPJ: " + pJ.getCNPJ());
            }
        }
        default -> {
        }
    }
  }

  private static void optionMenu() {
            System.out.println("=========================");
            System.out.println("1 - Incluir Pessoa");
            System.out.println("2 - Alterar Pessoa");
            System.out.println("3 - Excluir Pessoa");
            System.out.println("4 - Buscar pelo ID");
            System.out.println("5 - Exibir Todos");
            System.out.println("6 - Persistir Dados");
            System.out.println("7 - Recuperar Todos");
            System.out.println("0 - Finalizar Programa");
            System.out.println("=========================");
  }
```

```
}
```