Estácio	**Universidade Estácio** **Polo São Lourenço da Mata** **Desenvolvimento Full Stack** **Semestre 2024.1**	Disciplina: **Iniciando o Caminho Pelo Java.** Aluno: **Manoel José** Matrícula: 202301361117 Turma: 2023.1

Criação das Entidades e Sistema de Persistência.

Objetivos da Prática:

1. Utilizar herança e polimorfismo na definição de entidades.
2. Utilizar persistência de objetos em arquivos binários.
3. Implementar uma interface cadastral em modo texto.
4. Utilizar o controle de exceções da plataforma Java.
5. No final do projeto, o aluno terá implementado um sistema cadastral em Java, utilizando os recursos da programação orientada a objetos e a persistência em arquivos binários.

Todos os códigos solicitados neste roteiro de aula:

1- Código da Classe Principal (CadastroPOO.java):

```
import java.io.IOException;
import java.util.Scanner;
import model.PessoaFisica;
import model.PessoaFisicaRepo;
import model.PessoaJuridica;
import model.PessoaJuridicaRepo;

public class CadastroPOO {
    public static void main(String[] args) {
        try (Scanner scanner = new Scanner(System.in)) {
            PessoaFisicaRepo repoFisica = new PessoaFisicaRepo();
            PessoaJuridicaRepo repoJuridica = new PessoaJuridicaRepo();

            int opcao = -1;
            while (opcao != 0) {
                System.out.println("===== Menu =====");
```

```
System.out.println("1. Incluir");
System.out.println("2. Alterar");
System.out.println("3. Excluir");
System.out.println("4. Exibir pelo ID");
System.out.println("5. Exibir todos");
System.out.println("6. Salvar dados");
System.out.println("7. Recuperar dados");
System.out.println("0. Finalizar");
System.out.print("Escolha uma opção: ");
opcao = scanner.nextInt();
scanner.nextLine(); // Consumir a quebra de linha

switch (opcao) {
    case 1 -> {
        System.out.println("Selecionada a opção Incluir.");
        System.out.println("Escolha o tipo (1 para Física, 2 para Jurídica): ");
        int tipo = scanner.nextInt();
        scanner.nextLine(); // Consumir a quebra de linha
        switch (tipo) {
            case 1 -> {
                System.out.println("Incluir Pessoa Física:");
                PessoaFisica pessoaFisica = new PessoaFisica();
                pessoaFisica.lerDados(scanner);
                repoFisica.inserir(pessoaFisica);
            }
            case 2 -> {
                System.out.println("Incluir Pessoa Jurídica:");
                PessoaJuridica pessoaJuridica = new PessoaJuridica();
                pessoaJuridica.lerDados(scanner);
                repoJuridica.inserir(pessoaJuridica);
            }
            default -> System.out.println("Opção inválida.");
        }
    }
    case 2 -> {
        System.out.println("Selecionada a opção Alterar.");
        System.out.println("Escolha o tipo (1 para Física, 2 para Jurídica): ");
        int tipo = scanner.nextInt();
        scanner.nextLine(); // Consumir a quebra de linha
        System.out.println("Digite o ID: ");
        int id = scanner.nextInt();
        scanner.nextLine(); // Consumir a quebra de linha
        switch (tipo) {
            case 1 -> {
```

```
                PessoaFisica pessoaFisica = repoFisica.obter(id);
                if (pessoaFisica != null) {
                    System.out.println("Dados atuais:");
                    pessoaFisica.exibir();
                    System.out.println("Digite os novos dados:");
                    pessoaFisica.lerDados(scanner);
                    repoFisica.alterar(pessoaFisica);
                } else {
                    System.out.println("Pessoa física não encontrada.");
                }
            }
            case 2 -> {
                PessoaJuridica pessoaJuridica = repoJuridica.obter(id);
                if (pessoaJuridica != null) {
                    System.out.println("Dados atuais:");
                    pessoaJuridica.exibir();
                    System.out.println("Digite os novos dados:");
                    pessoaJuridica.lerDados(scanner);
                    repoJuridica.alterar(pessoaJuridica);
                } else {
                    System.out.println("Pessoa jurídica não encontrada.");
                }
            }
            default -> System.out.println("Opção inválida.");
        }
    }
    case 3 ->  {
        System.out.println("Selecionada a opção Excluir.");
        System.out.println("Escolha o tipo (1 para Física, 2 para Jurídica): ");
        int tipo = scanner.nextInt();
        scanner.nextLine(); // Consumir a quebra de linha
        System.out.println("Digite o ID: ");
        int id = scanner.nextInt();
        scanner.nextLine(); // Consumir a quebra de linha
        switch (tipo) {
            case 1 -> {
                repoFisica.excluir(id);
                System.out.println("Pessoa física excluída com sucesso.");
            }
            case 2 -> {
                repoJuridica.excluir(id);
                System.out.println("Pessoa jurídica excluída com sucesso.");
            }
            default -> System.out.println("Opção inválida.");
```

```
    }
}
case 4 ->  {
    System.out.println("Selecionada a opção Exibir pelo ID.");
    System.out.println("Escolha o tipo (1 para Física, 2 para Jurídica): ");
    int tipo = scanner.nextInt();
    scanner.nextLine(); // Consumir a quebra de linha
    System.out.println("Digite o ID: ");
    int id = scanner.nextInt();
    scanner.nextLine(); // Consumir a quebra de linha
    switch (tipo) {
        case 1 -> {
            PessoaFisica pessoaFisica = repoFisica.obter(id);
            if (pessoaFisica != null) {
                System.out.println("Dados da pessoa física:");
                pessoaFisica.exibir();
            } else {
                System.out.println("Pessoa física não encontrada.");
            }
        }
        case 2 -> {
            PessoaJuridica pessoaJuridica = repoJuridica.obter(id);
            if (pessoaJuridica != null) {
                System.out.println("Dados da pessoa jurídica:");
                pessoaJuridica.exibir();
            } else {
                System.out.println("Pessoa jurídica não encontrada.");
            }
        }
        default -> System.out.println("Opção inválida.");
    }
}
case 5 ->  {
    System.out.println("Selecionada a opção Exibir todos.");
    System.out.println("Escolha o tipo (1 para Física, 2 para Jurídica): ");
    int tipo = scanner.nextInt();
    scanner.nextLine(); // Consumir a quebra de linha
    switch (tipo) {
        case 1 -> {
            System.out.println("Pessoas físicas:");
            for (PessoaFisica pessoa : repoFisica.obterTodos()) {
                pessoa.exibir();
            }
        }
```

```
                    case 2 -> {
                        System.out.println("Pessoas jurídicas:");
                        for (PessoaJuridica pessoa : repoJuridica.obterTodos()) {
                            pessoa.exibir();
                        }
                    }
                    default -> System.out.println("Opção inválida.");
                }
            }
            case 6 ->  {
                try {
                    System.out.println("Selecionada a opção Salvar dados.");
                    System.out.print("Digite o prefixo dos arquivos: ");
                    String prefixo = scanner.nextLine();
                    repoFisica.persistir(prefixo + ".fisica.bin");
                    repoJuridica.persistir(prefixo + ".juridica.bin");
                    System.out.println("Dados salvos com sucesso.");
                } catch (IOException e) {
                    System.err.println("Erro ao salvar os dados: " + e.getMessage());
                }
            }
            case 7 ->  {
                try {
                    System.out.println("Selecionada a opção Recuperar dados.");
                    System.out.print("Digite o prefixo dos arquivos: ");
                    String prefixo = scanner.nextLine();
                    repoFisica.recuperar(prefixo + ".fisica.bin");
                    repoJuridica.recuperar(prefixo + ".juridica.bin");
                    System.out.println("Dados recuperados com sucesso.");
                } catch (IOException | ClassNotFoundException e) {
                    System.err.println("Erro ao recuperar os dados: " + e.getMessage());
                }
            }
            case 0 -> System.out.println("Finalizando...");
            default -> System.out.println("Opção inválida.");
        }
    }
  }
}
```

Resultado da execução do código principal (CadastroPOO.java):

```
run:
===== Menu =====
```

```
1. Incluir
2. Alterar
3. Excluir
4. Exibir pelo ID
5. Exibir todos
6. Salvar dados
7. Recuperar dados
0. Finalizar
Escolha uma op��o: 1
Selecionada a op��o Incluir.
Escolha o tipo (1 para F�sica, 2 para Jur�dica):
1
Incluir Pessoa F�sica:
Digite o ID: 01
Digite o nome: Manoel Jose
Digite o CPF: 12345678900
Digite a idade: 20
===== Menu =====
1. Incluir
2. Alterar
3. Excluir
4. Exibir pelo ID
5. Exibir todos
6. Salvar dados
7. Recuperar dados
0. Finalizar
Escolha uma op��o: 0
Finalizando...
BUILD SUCCESSFUL (total time: 1 minute 0 seconds)
```

2- Código da Classe Pessoa (Pessoa.java.):

```
package model;

import java.io.Serializable;

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

    public Pessoa() {
    }
```

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

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

3- Código da Classe Pessoa Fisica (PessoaFisica.java):

```
package model;

import java.io.Serializable;
import java.util.Scanner;

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

    public PessoaFisica() {
    }

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

```
    this.idade = idade;
}

@Override
public void exibir() {
    super.exibir();
    System.out.println("CPF: " + cpf + ", Idade: " + 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 void lerDados(Scanner scanner) {
    System.out.print("Digite o ID: ");
    setId(scanner.nextInt());
    scanner.nextLine(); // Consumir a quebra de linha
    System.out.print("Digite o nome: ");
    setNome(scanner.nextLine());
    System.out.print("Digite o CPF: ");
    setCpf(scanner.nextLine());
    System.out.print("Digite a idade: ");
    setIdade(scanner.nextInt());
    scanner.nextLine(); // Consumir a quebra de linha
}

public static void main(String[] args) {
    PessoaFisica pessoa1 = new PessoaFisica(1, "João", "123.456.789-10", 30);
    pessoa1.exibir();

    // Utilizando os getters
    System.out.println("CPF: " + pessoa1.getCpf());
```

```
        System.out.println("Idade: " + pessoa1.getIdade());
    }
}
```

4- Código da Classe Pessoa Jurídica (PessoaJuridica.java):

```
package model;

import java.io.Serializable;
import java.util.Scanner;

public class PessoaJuridica extends Pessoa implements Serializable {
    private String cnpj;

    public PessoaJuridica() {
    }

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

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

    public String getCnpj() {
        return cnpj;
    }

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

    public void lerDados(Scanner scanner) {
        System.out.print("Digite o ID: ");
        setId(scanner.nextInt());
        scanner.nextLine(); // Consumir a quebra de linha
        System.out.print("Digite o nome: ");
        setNome(scanner.nextLine());
        System.out.print("Digite o CNPJ: ");
```

```
        setCnpj(scanner.nextLine());
    }

    public static void main(String[] args) {
        PessoaJuridica pessoa1 = new PessoaJuridica(1, "Empresa XYZ", "12.345.678/0001-90");
        pessoa1.exibir();

        // Utilizando o getter
        System.out.println("CNPJ: " + pessoa1.getCnpj());
    }
}
```

5- Código da Classe Pessoa Física Repo (PessoaFisicaRepo.java):

```
package model;

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

public class PessoaFisicaRepo {
    private ArrayList<PessoaFisica> pessoasFisicas = new ArrayList<>();

    public void inserir(PessoaFisica pessoa) {
        pessoasFisicas.add(pessoa);
    }

    public void alterar(PessoaFisica pessoa) {
        for (int i = 0; i < pessoasFisicas.size(); i++) {
            if (pessoasFisicas.get(i).getId() == pessoa.getId()) {
                pessoasFisicas.set(i, pessoa);
                return;
            }
        }
        throw new IllegalArgumentException("Pessoa não encontrada para alteração.");
    }

    public void excluir(int id) {
        pessoasFisicas.removeIf(pessoa -> pessoa.getId() == id);
    }
```

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

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

    public void persistir(String nomeArquivo) throws IOException {
        try (ObjectOutputStream outputStream = new ObjectOutputStream(new
FileOutputStream(nomeArquivo))) {
            outputStream.writeObject(pessoasFisicas);
        }
    }

    public void recuperar(String nomeArquivo) throws IOException, ClassNotFoundException {
        try (ObjectInputStream inputStream = new ObjectInputStream(new
FileInputStream(nomeArquivo))) {
            pessoasFisicas = (ArrayList<PessoaFisica>) inputStream.readObject();
        }
    }
}
```

6- Código da Classe Pessoa Jurídica Repo (PessoaJuridicaRepo.java):

```
package model;

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

public class PessoaJuridicaRepo {
    private ArrayList<PessoaJuridica> pessoasJuridicas = new ArrayList<>();
```

```
    public void inserir(PessoaJuridica pessoa) {
        pessoasJuridicas.add(pessoa);
    }

    public void alterar(PessoaJuridica pessoa) {
        for (int i = 0; i < pessoasJuridicas.size(); i++) {
            if (pessoasJuridicas.get(i).getId() == pessoa.getId()) {
                pessoasJuridicas.set(i, pessoa);
                return;
            }
        }
        throw new IllegalArgumentException("Pessoa não encontrada para alteração.");
    }

    public void excluir(int id) {
        pessoasJuridicas.removeIf(pessoa -> pessoa.getId() == id);
    }

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

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

    public void persistir(String nomeArquivo) throws IOException {
        try (ObjectOutputStream outputStream = new ObjectOutputStream(new
FileOutputStream(nomeArquivo))) {
            outputStream.writeObject(pessoasJuridicas);
        }
    }

    public void recuperar(String nomeArquivo) throws IOException, ClassNotFoundException {
        try (ObjectInputStream inputStream = new ObjectInputStream(new
FileInputStream(nomeArquivo))) {
            pessoasJuridicas = (ArrayList<PessoaJuridica>) inputStream.readObject();
        }
    }
```

}

Análise e Conclusão:

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

   R: Elementos estáticos são membros de uma classe que pertencem à classe em vez de pertencerem a instâncias individuais da classe. O método main é declarado como estático para que possa ser chamado sem a necessidade de instanciar a classe, o que é uma convenção do Java para o ponto de entrada do programa.

B. **Para que serve a classe Scanner?**

   R: A classe Scanner é utilizada para ler entradas do usuário a partir do console. Ela fornece métodos para ler diferentes tipos de dados, como inteiros, strings e outros tipos primitivos, facilitando a interação do programa com o usuário.

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

   R: O uso de classes de repositório separa a lógica de negócios da lógica de persistência de dados. Isso permite uma melhor organização do código, facilitando a manutenção e a reutilização. Além disso, as classes de repositório encapsulam as operações de acesso aos dados, tornando o código mais modular e mais fácil de testar.