

Nome do Aluno: Matheus Oliveira de Jesus

Matrícula: 2023.10.10087-8

Disciplina: Iniciando o Caminho Pelo Java

Turma: RPG0014

Semestre Letivo: 1º/2024

Este relatório tem como objetivo apresentar códigos e resultados referente aos exercícios solicitados em: Missão Prática | Nível 1 | Mundo 3.

Respostas: 2º Procedimento | Criação do Cadastro em Modo Texto

1 – Título da Prática: Iniciando o caminho pelo Java

2 – Objetivo da Prática:

- 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;
- 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.

3 – Códigos gerados:

1) 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 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;

    }

    public void exibir() {

        System.out.println("ID: " + id);

    }

}
```

```
        System.out.println("Nome: " + nome);
    }
}
```

2) GeradorID.java

```
package model;

public class GeradorID {

    private static int currentId = 1;

    public static int getNextId() {

        return currentId++;

    }

}
```

3) PessoaFisica.java

```
package model;

public class PessoaFisica extends Pessoa {

    private String cpf;

    private int idade;

    public PessoaFisica() {

        super(GeradorID.getNextId(), "");

    }

    public PessoaFisica(String nome, String cpf, int idade) {

        super(GeradorID.getNextId(), nome);

        this.cpf = cpf;

        this.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;
    }

    @Override
    public void exibir() {
        super.exibir();

        System.out.println("CPF: " + cpf);
        System.out.println("Idade: " + idade);
    }
}
```

4) PessoaJuridica.java

```
package model;

public class PessoaJuridica extends Pessoa {

    private String cnpj;

    public PessoaJuridica() {

        super(GeradorID.getNextId(), "");

    }

    public PessoaJuridica(int id, String nome, String cnpj) {

        super(GeradorID.getNextId(), nome);

        this.cnpj = cnpj;

    }

    public String getCnpj() {

        return cnpj;

    }

    public void setCnpj(String cnpj) {

        this.cnpj = cnpj;

    }

    @Override

    public void exibir() {

        super.exibir();

        System.out.println("CNPJ: " + cnpj);

    }

}
```

5) PessoaFisicaRepo.java

```
package model;

import java.io.*;
import java.util.ArrayList;
import java.util.Iterator;

public class PessoaFisicaRepo {

    private ArrayList<PessoaFisica> pessoasFisicas = new ArrayList<>();

    public void inserir(PessoaFisica pessoaFisica) {

        pessoasFisicas.add(pessoaFisica);
    }

    public void alterar(PessoaFisica pessoaFisica) {

        for (int i = 0; i < pessoasFisicas.size(); i++) {

            if (pessoasFisicas.get(i).getId() == pessoaFisica.getId()) {

                pessoasFisicas.set(i, pessoaFisica);

                return;
            }
        }
    }

    public void excluir(int id) {

        Iterator<PessoaFisica> iterator = pessoasFisicas.iterator();

        while (iterator.hasNext()) {

            if (iterator.next().getId() == id) {

                iterator.remove();

                return;
            }
        }
    }
}
```

```
}
```

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

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

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

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

6) PessoaJuridicaRepo.java

```
package model;

import java.io.*;
import java.util.ArrayList;
import java.util.Iterator;

public class PessoaJuridicaRepo {

    private ArrayList<PessoaJuridica> pessoasJuridicas = new ArrayList<>();

    public void inserir(PessoaJuridica pessoaJuridica) {

        pessoasJuridicas.add(pessoaJuridica);

    }

    public void alterar(PessoaJuridica pessoaJuridica) {

        for (int i = 0; i < pessoasJuridicas.size(); i++) {

            if (pessoasJuridicas.get(i).getId() == pessoaJuridica.getId()) {

                pessoasJuridicas.set(i, pessoaJuridica);

                return;

            }

        }

    }

    public void excluir(int id) {

        Iterator<PessoaJuridica> iterator = pessoasJuridicas.iterator();

        while (iterator.hasNext()) {

            if (iterator.next().getId() == id) {

                iterator.remove();

                return;

            }

        }

    }

}
```



```

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

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

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

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

```

7) Main.java

```
package model;
```

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

```
import java.util.Scanner;
```

```
public class Main {
```

```
    public static void main(String[] args) {
```

```
        Scanner scanner = new Scanner(System.in);
```

```
        PessoaFisicaRepo repoFisica = new PessoaFisicaRepo();
```

```
        PessoaJuridicaRepo repoJuridica = new PessoaJuridicaRepo();
```

```
        boolean running = true;
```

```
        while (running) {
```

```
            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("Selecione uma opção: ");
```

```
            int opcao = scanner.nextInt();
```

```
            scanner.nextLine();
```

```
            switch (opcao) {
```

case 1:

incluir(scanner, repoFisica, repoJuridica);

break;

case 2:

alterar(scanner, repoFisica, repoJuridica);

break;

case 3:

excluir(scanner, repoFisica, repoJuridica);

break;

case 4:

exibirPeloid(scanner, repoFisica, repoJuridica);

break;

case 5:

exibirTodos(scanner, repoFisica, repoJuridica);

break;

case 6:

salvarDados(scanner, repoFisica, repoJuridica);

break;

case 7:

recuperarDados(scanner, repoFisica, repoJuridica);

break;

case 0:

running = false;

System.out.println("Finalizando o programa.");

break;

default:

System.out.println("Opção inválida!");

break;

```
    }  
}  
scanner.close();  
}
```

```
private static void incluir(Scanner scanner, PessoaFisicaRepo repoFisica,  
PessoaJuridicaRepo repoJuridica) {
```

```
    System.out.println("Incluir: (1 - Pessoa Física, 2 - Pessoa Jurídica)");  
    int tipo = scanner.nextInt();  
    scanner.nextLine();
```

```
    if (tipo == 1) {  
        System.out.print("Nome: ");  
        String nome = scanner.nextLine();  
        System.out.print("CPF: ");  
        String cpf = scanner.nextLine();  
        System.out.print("Idade: ");  
        int idade = scanner.nextInt();  
        PessoaFisica pessoa = new PessoaFisica(nome, cpf, idade);  
        repoFisica.inserir(pessoa);  
        System.out.println("Pessoa física incluída com sucesso!");  
    } else if (tipo == 2) {  
        System.out.print("Nome: ");  
        String nome = scanner.nextLine();  
        System.out.print("CNPJ: ");  
        String cnpj = scanner.nextLine();  
        PessoaJuridica pessoa = new PessoaJuridica(nome, cnpj);  
        repoJuridica.inserir(pessoa);
```

```
        System.out.println("Pessoa jurídica incluída com sucesso!");  
    } else {  
        System.out.println("Opção inválida!");  
    }  
}
```

```
private static void alterar(Scanner scanner, PessoaFisicaRepo repoFisica,  
PessoaJuridicaRepo repoJuridica) {
```

```
    System.out.println("Alterar: (1 - Pessoa Física, 2 - Pessoa Jurídica)");  
    int tipo = scanner.nextInt();  
    scanner.nextLine();
```

```
    if (tipo == 1) {  
        System.out.print("ID da Pessoa Física: ");  
        int id = scanner.nextInt();  
        scanner.nextLine();  
        PessoaFisica pessoa = repoFisica.obter(id);  
        if (pessoa != null) {  
            System.out.println("Pessoa Física Atual:");  
            pessoa.exibir();  
            System.out.print("Novo nome: ");  
            String nome = scanner.nextLine();  
            System.out.print("Novo CPF: ");  
            String cpf = scanner.nextLine();  
            System.out.print("Nova idade: ");  
            int idade = scanner.nextInt();  
            pessoa.setNome(nome);  
            pessoa.setCpf(cpf);
```

```
        pessoa.setIdade(idade);
        repoFisica.alterar(pessoa);
        System.out.println("Pessoa física alterada com sucesso!");
    } else {
        System.out.println("Pessoa física não encontrada.");
    }
} else if (tipo == 2) {
    System.out.print("ID da Pessoa Jurídica: ");
    int id = scanner.nextInt();
    scanner.nextLine();
    PessoaJuridica pessoa = repoJuridica.obter(id);
    if (pessoa != null) {
        System.out.println("Pessoa Jurídica Atual:");
        pessoa.exibir();
        System.out.print("Novo nome: ");
        String nome = scanner.nextLine();
        System.out.print("Novo CNPJ: ");
        String cnpj = scanner.nextLine();
        pessoa.setNome(nome);
        pessoa.setCnpj(cnpj);
        repoJuridica.alterar(pessoa);
        System.out.println("Pessoa jurídica alterada com sucesso!");
    } else {
        System.out.println("Pessoa jurídica não encontrada.");
    }
} else {
    System.out.println("Opção inválida!");
}
```

```
}
```

```
private static void excluir(Scanner scanner, PessoaFisicaRepo repoFisica,  
PessoaJuridicaRepo repoJuridica) {
```

```
    System.out.println("Excluir: (1 - Pessoa Física, 2 - Pessoa Jurídica)");
```

```
    int tipo = scanner.nextInt();
```

```
    scanner.nextLine();
```

```
    if (tipo == 1) {
```

```
        System.out.print("ID da Pessoa Física: ");
```

```
        int id = scanner.nextInt();
```

```
        repoFisica.excluir(id);
```

```
        System.out.println("Pessoa física excluída com sucesso!");
```

```
    } else if (tipo == 2) {
```

```
        System.out.print("ID da Pessoa Jurídica: ");
```

```
        int id = scanner.nextInt();
```

```
        repoJuridica.excluir(id);
```

```
        System.out.println("Pessoa jurídica excluída com sucesso!");
```

```
    } else {
```

```
        System.out.println("Opção inválida!");
```

```
    }
```

```
}
```

```
private static void exibirPeloid(Scanner scanner, PessoaFisicaRepo repoFisica,  
PessoaJuridicaRepo repoJuridica) {
```

```
    System.out.println("Exibir pelo ID: (1 - Pessoa Física, 2 - Pessoa Jurídica)");
```

```
    int tipo = scanner.nextInt();
```

```
    scanner.nextLine();
```

```

if (tipo == 1) {

    System.out.print("ID da Pessoa Física: ");

    int id = scanner.nextInt();

    PessoaFisica pessoa = repoFisica.obter(id);

    if (pessoa != null) {

        pessoa.exibir();

    } else {

        System.out.println("Pessoa física não encontrada.");

    }

} else if (tipo == 2) {

    System.out.print("ID da Pessoa Jurídica: ");

    int id = scanner.nextInt();

    PessoaJuridica pessoa = repoJuridica.obter(id);

    if (pessoa != null) {

        pessoa.exibir();

    } else {

        System.out.println("Pessoa jurídica não encontrada.");

    }

} else {

    System.out.println("Opção inválida!");

}

}

```

```

private static void exibirTodos(Scanner scanner, PessoaFisicaRepo repoFisica,
PessoaJuridicaRepo repoJuridica) {

    System.out.println("Exibir todos: (1 - Pessoas Físicas, 2 - Pessoas Jurídicas)");

    int tipo = scanner.nextInt();

    scanner.nextLine();

```



```

if (tipo == 1) {
    System.out.println("Pessoas Físicas:");
    for (PessoaFisica pessoa : repoFisica.obterTodos()) {
        pessoa.exibir();
    }
} else if (tipo == 2) {
    System.out.println("Pessoas Jurídicas:");
    for (PessoaJuridica pessoa : repoJuridica.obterTodos()) {
        pessoa.exibir();
    }
} else {
    System.out.println("Opção inválida!");
}
}

```

```

private static void salvarDados(Scanner scanner, PessoaFisicaRepo repoFisica,
PessoaJuridicaRepo repoJuridica) {

```

```

    System.out.print("Digite o prefixo do arquivo: ");

```

```

    String prefixo = scanner.nextLine();

```

```

    try {

```

```

        repoFisica.persistir(prefixo + ".fisica.bin");

```

```

        repoJuridica.persistir(prefixo + ".juridica.bin");

```

```

        System.out.println("Dados salvos com sucesso!");

```

```

    } catch (IOException e) {

```

```

        System.out.println("Erro ao salvar os dados: " + e.getMessage());

```

```

    }

```

```
}
```

```
private static void recuperarDados(Scanner scanner, PessoaFisicaRepo  
repoFisica, PessoaJuridicaRepo repoJuridica) {
```

```
    System.out.print("Digite o prefixo do arquivo: ");
```

```
    String prefixo = scanner.nextLine();
```

```
    try {
```

```
        repoFisica.recuperar(prefixo + ".fisica.bin");
```

```
        repoJuridica.recuperar(prefixo + ".juridica.bin");
```

```
        System.out.println("Dados recuperados com sucesso!");
```

```
    } catch (IOException | ClassNotFoundException e) {
```

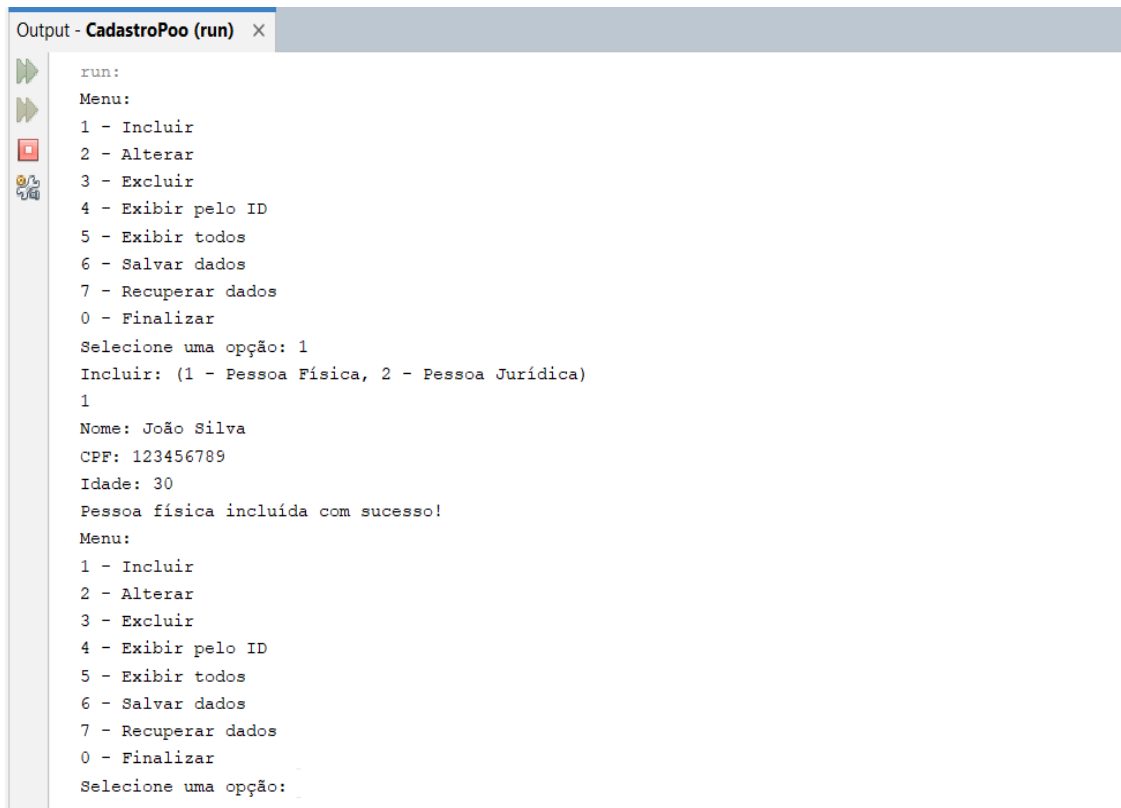
```
        System.out.println("Erro ao recuperar os dados: " + e.getMessage());
```

```
    }
```

```
}
```

```
}
```

4 – Resultado dos códigos executados:



```
Output - CadastroPoo (run) X
run:
Menu:
1 - Incluir
2 - Alterar
3 - Excluir
4 - Exibir pelo ID
5 - Exibir todos
6 - Salvar dados
7 - Recuperar dados
0 - Finalizar
Selecione uma opção: 1
Incluir: (1 - Pessoa Física, 2 - Pessoa Jurídica)
1
Nome: João Silva
CPF: 123456789
Idade: 30
Pessoa física incluída com sucesso!
Menu:
1 - Incluir
2 - Alterar
3 - Excluir
4 - Exibir pelo ID
5 - Exibir todos
6 - Salvar dados
7 - Recuperar dados
0 - Finalizar
Selecione uma opção: 
```

5 – Análise e Conclusão:

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

Elementos estáticos em Java são partes de uma classe (como atributos ou métodos) que não dependem de um objeto específico. Em vez de funcionar para cada objeto criado, eles pertencem à própria classe. Isso significa que você pode usá-los sem precisar criar um objeto.

O método main é estático porque é o primeiro que o Java executa ao rodar um programa. Como ele precisa funcionar sem que nenhum objeto tenha sido criado ainda, o modificador static permite que o Java execute o main diretamente pela classe.

b. Para que serve a classe Scanner?

A classe Scanner em Java serve para ler dados que o usuário digita no console (ou de outras fontes, como arquivos). Ela permite que o programa receba diferentes tipos de entrada, como números, textos ou palavras, de forma fácil.

Por exemplo, ao usar o Scanner, você pode pedir para o usuário digitar um número ou uma palavra, e o programa consegue capturar essa informação e usar no código.

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

O uso de classes de repositório ajuda a organizar o código ao separar a parte que lida com armazenamento de dados da lógica principal do programa. Isso torna o código mais limpo, mais fácil de manter e de reutilizar. Em resumo, ajuda a manter cada parte do programa com uma função específica e bem definida.