



## **RPG0014 - Iniciando o caminho pelo Java**

**Jéssica Maria de Carvalho Matrícula: 202209187939**

**POLO JARDIM SÃO BERNARDO - SÃO PAULO - SP**

**Nível 1: Iniciando o Caminho Pelo Java – 9003 – 3º**

**Endereço do Repositório GIT: <https://github.com/Jessicac30/CadastroPOO2>**

### **Objetivo da Prática**

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

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.

## 2º Procedimento | Criação do Cadastro em Modo Texto

### Organização dos arquivos

CadastroPOO/

├── src/

| ├── cadastro/

| | ├── Main.java

| ├── model/

| | ├── Pessoa.java

| | ├── PessoaFisica.java

| | ├── PessoaFisicaRepo.java

| | ├── PessoaJuridica.java

| | ├── PessoaJuridicaRepo.java

## Arquivo Main.java

```
package cadastro;

import java.util.Scanner;
import model.PessoaFisica;
import model.PessoaFisicaRepo;
import model.PessoaJuridica;
import model.PessoaJuridicaRepo;

public class Main {

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        PessoaFisicaRepo repoPessoaFisica = new PessoaFisicaRepo();

        PessoaJuridicaRepo repoPessoaJuridica = new PessoaJuridicaRepo();

        while (true) {

            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 Dados");

            System.out.println("0 - Finalizar Programa");

            System.out.println("=====");

            System.out.print("Escolha uma opção: ");

            int opcao = scanner.nextInt();

            scanner.nextLine();

            switch (opcao) {

                case 1:

                    System.out.print("F - Pessoa Fisica | J - Pessoa Juridica: ");

                    String tipo = scanner.nextLine().toUpperCase();
```

```

        if (tipo.equals("F")) {

            System.out.print("Digite o Id da pessoa: ");

            int id = scanner.nextInt();

            scanner.nextLine();

            System.out.print("Nome: ");

            String nome = scanner.nextLine();

            System.out.print("CPF: ");

            String cpf = scanner.nextLine();

            System.out.print("Idade: ");

            int idade = scanner.nextInt();

            scanner.nextLine();

            PessoaFisica pessoaFisica = new PessoaFisica(id, nome, cpf, idade);

            repoPessoaFisica.inserir(pessoaFisica);

            System.out.println("Pessoa Fisica incluída com sucesso.");
        } else if (tipo.equals("J")) {

            System.out.print("Digite o Id da pessoa: ");

            int id = scanner.nextInt();

            scanner.nextLine();

            System.out.print("Nome: ");

            String nome = scanner.nextLine();

            System.out.print("CNPJ: ");

            String cnpj = scanner.nextLine();

            PessoaJuridica pessoaJuridica = new PessoaJuridica(id, nome, cnpj);

            repoPessoaJuridica.inserir(pessoaJuridica);

            System.out.println("Pessoa Jurídica incluída com sucesso.");
        } else {

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

        break;

    case 2:

        System.out.print("F - Pessoa Fisica | J - Pessoa Juridica: ");

        String tipoAlteracao = scanner.nextLine().toUpperCase();

        if (tipoAlteracao.equals("F")) {

            System.out.print("Digite o Id da pessoa a ser alterada: ");

```

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

scanner.nextLine();

PessoaFisica pessoaFisicaExistente = repoPessoaFisica.obter(id);
if (pessoaFisicaExistente == null) {
    System.out.println("Pessoa Fisica não encontrada.");
} else {
    System.out.println("Dados atuais da Pessoa Fisica:");
    System.out.println(pessoaFisicaExistente.toString());

    System.out.print("Novo nome: ");
    String novoNome = scanner.nextLine();

    System.out.print("Novo CPF: ");
    String novoCpf = scanner.nextLine();

    System.out.print("Nova idade: ");
    int novaIdade = scanner.nextInt();
    scanner.nextLine();

    pessoaFisicaExistente.setNome(novoNome);
    pessoaFisicaExistente.setCpf(novoCpf);
    pessoaFisicaExistente.setIdade(novaIdade);

    System.out.println("Pessoa Fisica alterada com sucesso.");
}
} else if (tipoAlteracao.equals("J")) {
    System.out.print("Digite o Id da pessoa a ser alterada: ");
    int id = scanner.nextInt();
    scanner.nextLine(); // Limpar a quebra de linha

    PessoaJuridica pessoaJuridicaExistente = repoPessoaJuridica.obter(id);
    if (pessoaJuridicaExistente == null) {
        System.out.println("Pessoa Jurídica não encontrada.");
    } else {
        System.out.println("Dados atuais da Pessoa Jurídica:");
        System.out.println(pessoaJuridicaExistente.toString());

        System.out.print("Novo nome: ");
        String novoNome = scanner.nextLine();

        System.out.print("Novo CNPJ: ");
```

```

        String novoCnpj = scanner.nextLine();

        pessoaJuridicaExistente.setNome(novoNome);
        pessoaJuridicaExistente.setCnpj(novoCnpj);

        System.out.println("Pessoa Jurídica alterada com sucesso.");
    }
} else {
    System.out.println("Opção inválida.");
}

break;

case 3:

    System.out.print("F - Pessoa Física | J - Pessoa Jurídica: ");
    String tipoExclusao = scanner.nextLine().toUpperCase();

    if (tipoExclusao.equals("F")) {
        System.out.print("Digite o Id da pessoa a ser excluída: ");
        int id = scanner.nextInt();
        scanner.nextLine();

        boolean removido = repoPessoaFisica.remover(id);
        if (removido) {
            System.out.println("Pessoa Física removida com sucesso.");
        } else {
            System.out.println("Pessoa Física não encontrada.");
        }
    } else if (tipoExclusao.equals("J")) {
        System.out.print("Digite o Id da pessoa a ser excluída: ");
        int id = scanner.nextInt();
        scanner.nextLine();

        boolean removido = repoPessoaJuridica.remover(id);
        if (removido) {
            System.out.println("Pessoa Jurídica removida com sucesso.");
        } else {
            System.out.println("Pessoa Jurídica não encontrada.");
        }
    } else {

```

```

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

    break;

case 4:

    System.out.print("F - Pessoa Fisica | J - Pessoa Juridica: ");
    String tipoBusca = scanner.nextLine().toUpperCase();

    if (tipoBusca.equals("F")) {

        System.out.print("Digite o Id da pessoa a ser buscada: ");
        int id = scanner.nextInt();
        scanner.nextLine();

        PessoaFisica pessoaFisicaBusca = repoPessoaFisica.obter(id);

        if (pessoaFisicaBusca != null) {

            System.out.println("Pessoa Fisica encontrada:");
            System.out.println(pessoaFisicaBusca.toString());

        } else {

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

        }

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

        System.out.print("Digite o Id da pessoa a ser buscada: ");
        int id = scanner.nextInt();
        scanner.nextLine();

        PessoaJuridica pessoaJuridicaBusca = repoPessoaJuridica.obter(id);

        if (pessoaJuridicaBusca != null) {

            System.out.println("Pessoa Jurídica encontrada:");
            System.out.println(pessoaJuridicaBusca.toString());

        } else {

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

        }

    } else {

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

    }

    break;

case 5:

    System.out.print("F - Pessoa Fisica | J - Pessoa Juridica: ");

```

```

String tipoExibicao = scanner.nextLine().toUpperCase();

if (tipoExibicao.equals("F")) {

    System.out.println("Pessoas Físicas no Repositório:");

    for (PessoaFisica pessoa : repoPessoaFisica.obterTodos()) {

        System.out.println(pessoa.toString());

    }

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

    System.out.println("Pessoas Jurídicas no Repositório:");

    for (PessoaJuridica pessoa : repoPessoaJuridica.obterTodos()) {

        System.out.println(pessoa.toString());

    }

} else {

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

}

break;

case 6:

    System.out.print("Digite o prefixo para os arquivos de persistência: ");

    String prefixo = scanner.nextLine();

    try {

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

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

        System.out.println("Dados persistidos com sucesso.");

    } catch (Exception e) {

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

    }

    break;

case 7:

    System.out.print("Digite o prefixo para os arquivos de recuperação: ");

    String prefixoRecuperacao = scanner.nextLine();

    try {

        repoPessoaFisica.recuperar(prefixoRecuperacao + ".fisica.bin");

        repoPessoaJuridica.recuperar(prefixoRecuperacao + ".juridica.bin");

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

    } catch (Exception e) {

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

    }

}

```



```
        break;

    case 0:

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

        return;

    default:

        System.out.println("Opção inválida. Tente novamente.");

    }

}

}
```

## Arquivo Pessoa.java

```
package model;

import java.io.Serializable;

public class Pessoa implements Serializable {

    private int id;

    private String nome;

    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;

    }

    @Override

    public String toString() {

        return "ID: " + id + ", Nome: " + nome;

    }

}
```

## Arquivo PessoaFisica.java

```
package model;

import java.io.Serializable;

public class PessoaFisica extends Pessoa implements Serializable {

    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 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 String toString() {

        return super.toString() + ", CPF: " + cpf + ", Idade: " + idade;

    }

}
```

## Arquivo PessoaFisicaRepo.java

```
package model;

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

public class PessoaFisicaRepo {

    private ArrayList<PessoaFisica> pessoasFisicas;

    public PessoaFisicaRepo() {

        pessoasFisicas = new ArrayList<>();

    }

    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();

        }

    }

    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 boolean remover(int id) {  
    for (int i = 0; i < pessoasFisicas.size(); i++) {  
        if (pessoasFisicas.get(i).getId() == id) {  
            pessoasFisicas.remove(i);  
            return true;  
        }  
    }  
    return false;  
}  
  
public PessoaFisica obter(int id) {  
    for (PessoaFisica pessoaFisica : pessoasFisicas) {  
        if (pessoaFisica.getId() == id) {  
            return pessoaFisica;  
        }  
    }  
    return null;  
}  
  
public ArrayList<PessoaFisica> obterTodos() {  
    return pessoasFisicas;  
}  
}
```

## Arquivo PessoaJuridica.java

```
package model;

import java.io.Serializable;

public class PessoaJuridica extends Pessoa implements Serializable {

    private String cnpj;

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

        super(id, nome);

        this.cnpj = cnpj;

    }

    public String getCnpj() {

        return cnpj;

    }

    public void setCnpj(String cnpj) {

        this.cnpj = cnpj;

    }

    @Override

    public String toString() {

        return super.toString() + ", CNPJ: " + cnpj;

    }

}
```

## Arquivo PessoaJuridicaRepo.java

```
package model;

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

public class PessoaJuridicaRepo {

    private ArrayList<PessoaJuridica> pessoasJuridicas;

    public PessoaJuridicaRepo() {

        pessoasJuridicas = new ArrayList<>();

    }

    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();

        }

    }

    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;

            }

        }

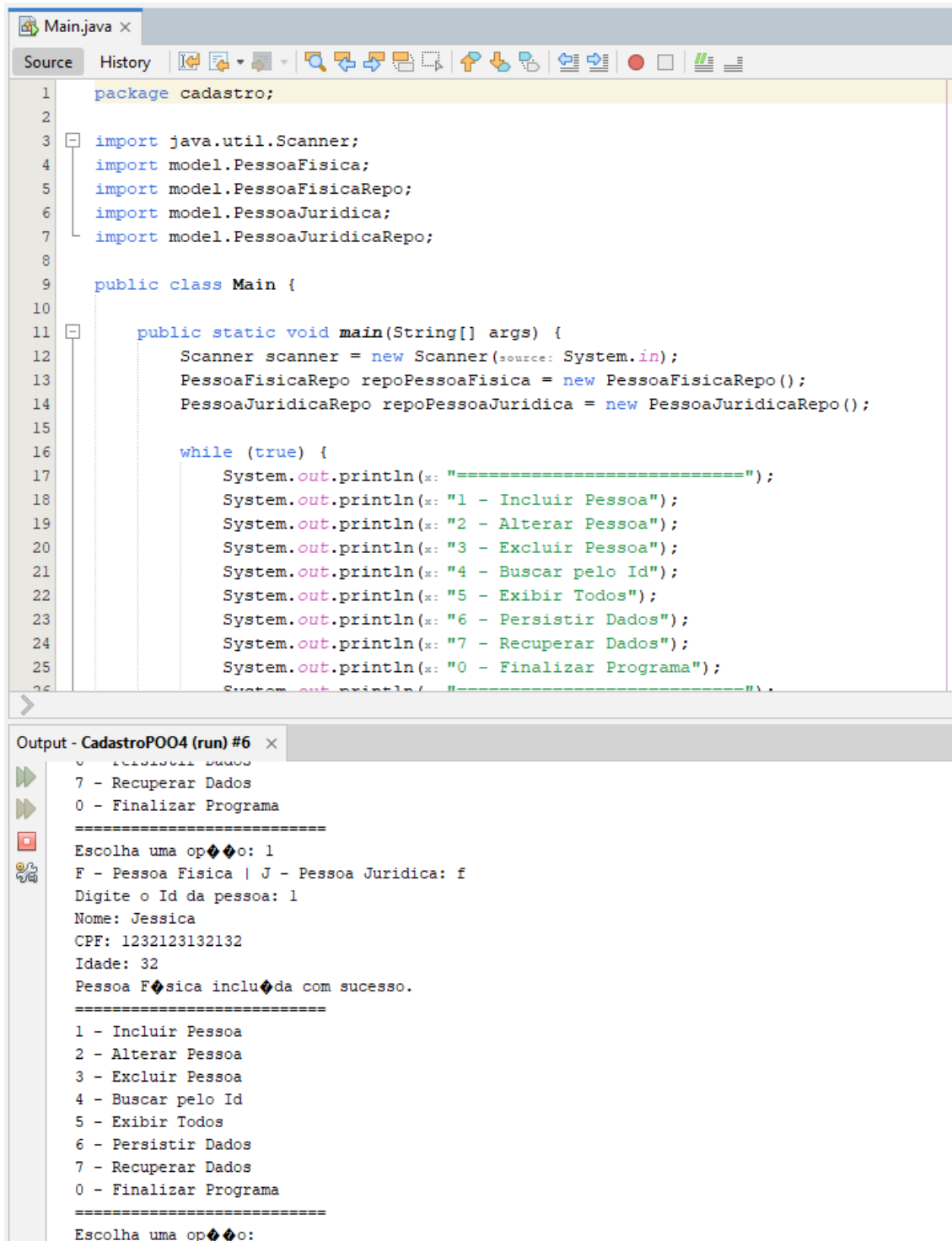
    }

}
```

```
    }  
}  
  
public boolean remover(int id) {  
    for (int i = 0; i < pessoasJuridicas.size(); i++) {  
        if (pessoasJuridicas.get(i).getId() == id) {  
            pessoasJuridicas.remove(i);  
            return true;  
        }  
    }  
    return false;  
}  
  
public PessoaJuridica obter(int id) {  
    for (PessoaJuridica pessoaJuridica : pessoasJuridicas) {  
        if (pessoaJuridica.getId() == id) {  
            return pessoaJuridica;  
        }  
    }  
    return null;  
}  
  
public ArrayList<PessoaJuridica> obterTodos() {  
    return pessoasJuridicas;  
}  
}
```



## Resultado



The image shows a screenshot of an IDE with two panels. The top panel displays the source code for a Java application named 'Main.java'. The code is organized into a package 'cadastro' and imports necessary classes from 'java.util' and 'model'. It defines a 'Main' class with a 'main' method that initializes a scanner and two repositories. A 'while' loop presents a menu of options to the user, including adding, altering, excluding, searching, displaying, persisting, and recovering data, as well as ending the program.

```
1 package cadastro;
2
3 import java.util.Scanner;
4 import model.PessoaFisica;
5 import model.PessoaFisicaRepo;
6 import model.PessoaJuridica;
7 import model.PessoaJuridicaRepo;
8
9 public class Main {
10
11     public static void main(String[] args) {
12         Scanner scanner = new Scanner(System.in);
13         PessoaFisicaRepo repoPessoaFisica = new PessoaFisicaRepo();
14         PessoaJuridicaRepo repoPessoaJuridica = new PessoaJuridicaRepo();
15
16         while (true) {
17             System.out.println("=====");
18             System.out.println("1 - Incluir Pessoa");
19             System.out.println("2 - Alterar Pessoa");
20             System.out.println("3 - Excluir Pessoa");
21             System.out.println("4 - Buscar pelo Id");
22             System.out.println("5 - Exibir Todos");
23             System.out.println("6 - Persistir Dados");
24             System.out.println("7 - Recuperar Dados");
25             System.out.println("0 - Finalizar Programa");
26             System.out.println("Escolha uma opção: ");
```

The bottom panel shows the output of the program. It displays the menu options and the user's input. The user has chosen option 1, selected 'F' for Pessoa Fisica, entered 'f' for Pessoa Juridica, and provided the ID '1'. The program then prompts for the name and CPF, which are 'Jessica' and '1232123132132' respectively. The output confirms that the 'Pessoa Fisica' was successfully included and displays the menu options again.

```
6 - Persistir Dados
7 - Recuperar Dados
0 - Finalizar Programa
=====
Escolha uma opção: 1
F - Pessoa Fisica | J - Pessoa Juridica: f
Digite o Id da pessoa: 1
Nome: Jessica
CPF: 1232123132132
Idade: 32
Pessoa Fisica incluída com sucesso.
=====
1 - Incluir Pessoa
2 - Alterar Pessoa
3 - Excluir Pessoa
4 - Buscar pelo Id
5 - Exibir Todos
6 - Persistir Dados
7 - Recuperar Dados
0 - Finalizar Programa
=====
Escolha uma opção:
```

## **Análise e Conclusão**

### **Elementos estáticos e motivo para o método main adotar esse modificador:**

Elementos estáticos em Java referem-se a membros de classe (variáveis e métodos) que pertencem à classe como um todo, em vez de instâncias específicas da classe. Eles são compartilhados por todas as instâncias da classe. O método main em Java é um método estático porque ele é chamado pela máquina virtual Java (JVM) para iniciar a execução do programa antes que qualquer objeto da classe seja criado. Ele é o ponto de entrada da aplicação Java e precisa ser estático para ser invocado sem a necessidade de criar uma instância da classe. Além disso, o método main deve ter uma assinatura específica que seja reconhecida pela JVM.

### **Classe Scanner:**

A classe Scanner em Java é usada para ler entradas de texto a partir de várias fontes, como teclado, arquivos ou outras fontes de entrada. Ela fornece métodos para analisar diferentes tipos de dados, como inteiros, ponto flutuante, strings, entre outros, tornando-a útil para interação do usuário com o programa. Ela é amplamente usada para coletar dados fornecidos pelo usuário, como as escolhas do menu em um programa interativo.

### **Impacto do uso de classes de repositório na organização do código:**

O uso de classes de repositório é uma prática comum para separar a lógica de persistência de dados do restante do código. Isso ajuda a manter o código organizado e facilita a manutenção e extensão do programa. Ao utilizar classes de repositório, você isola as operações de leitura, escrita, alteração e exclusão de dados em arquivos ou bancos de dados, tornando o código principal mais focado na lógica de negócios. Isso também permite uma maior flexibilidade para trocar o mecanismo de persistência de dados no futuro sem afetar o restante do programa.