



Estácio

Estácio - Unidade São Pedro

RJ 140 Km 2, 512 loja 1, São Pedro da Aldeia - RJ, 28941-182

Desenvolvimento Full Stack

Classe: Missão Prática | Nível 1 | Mundo 3

3º Semestre

Marvin de Almeida Costa

Título da Prática: 2º Procedimento | Criação do Cadastro em Modo Texto

Objetivo 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:

- **CadastroPOO.java**

```
package cadastropoo;
```

```
/**
```

```
 *
```

```
 * @author Marvin
```

```
 */
```

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

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

```

import java.util.List;
import java.io.File;
import java.io.ObjectOutputStream;
import java.io.FileOutputStream;
import java.io.ObjectInputStream;
import java.io.FileInputStream;

public class CadastroPOO {

    private static Scanner scanner = new Scanner(System.in);
    private static PessoaFisicaRepo repoFisica = new PessoaFisicaRepo();
    private static PessoaJuridicaRepo repoJuridica = new PessoaJuridicaRepo();

    public static void main(String[] args) {
        try {
            repoFisica.recuperar("pessoasFisicas.dat");
        } catch (IOException | ClassNotFoundException e) {

        }

        try {
            repoJuridica.recuperar("pessoasJuridicas.dat");
        } catch (IOException | ClassNotFoundException e) {

        }

        int opcao;

        do {
            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("=====");
            opcao = scanner.nextInt();

            switch (opcao) {
                case 1:
                    incluirPessoa();

```

```

        break;
    case 2:
        alterarPessoa();
        break;
    case 3:
        excluirPessoa();
        break;
    case 4:
        buscarPeloid();
        break;
    case 5:
        exibirTodos();
        break;
    case 6:
        persistirDados();
        break;
    case 7:
        recuperarDados();
        break;
    case 0:
        System.out.println("Saindo...");
        break;
    default:
        System.out.println("Opção inválida!");
    }
} while (opcao != 0);
}

private static void incluirPessoa() {
    String tipo;
    int id;
    String nome;
    String cpf;
    int idade;
    String cnpj;

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

    tipo = scanner.next().toUpperCase();

    if (!tipo.equals("J") &&!tipo.equals("F")) {
        System.out.println("Opção inválida!");
        return;
    }
}

```

```

System.out.println("Digite o id da Pessoa:");

id = scanner.nextInt();

System.out.println("Insira os dados...");
System.out.println("Nome:");

nome = scanner.next();

if (tipo.equals("J")) {
    System.out.println("CNPJ:");
    cnpj = scanner.next();

    PessoaJuridica pessoaJ = new PessoaJuridica(id, nome, cnpj);
    repoJuridica.inserir(pessoaJ);

    saveData("J");
}

if (tipo.equals("F")) {
    System.out.println("CPF:");
    cpf = scanner.next();

    System.out.println("Idade:");
    idade = scanner.nextInt();

    PessoaFisica pessoaF = new PessoaFisica(id, nome, cpf, idade);
    repoFisica.inserir(pessoaF);

    saveData("F");
}
}

private static void saveData(String tipo) {
    if (tipo.equals("J")) {
        try {
            repoJuridica.persistir("pessoasJuridicas.dat");
        } catch (IOException e) {

        }
    }
}

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

```

```

        try {
            repoFisica.persistir("pessoasFisicas.dat");
        } catch (IOException e) {

        }
    }
}

private static void exibirTodos() {
    System.out.println("Dados de Pessoas Fisicas:");
    List<PessoaFisica> pessoasFisicas = repoFisica.obterTodos();
    if (!pessoasFisicas.isEmpty()) {
        for (PessoaFisica pf : pessoasFisicas) {
            pf.exibir();
            System.out.println("");
        }
    } else {
        System.out.println("Nenhuma Pessoa Fisica");
    }

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

    System.out.println("Dados de Pessoas Juridicas:");

    List<PessoaJuridica> pessoaJuridicas = repoJuridica.obterTodos();
    if (!pessoaJuridicas.isEmpty()) {
        for (PessoaJuridica pf : pessoaJuridicas) {
            pf.exibir();
            System.out.println("");
        }
    } else {
        System.out.println("Nenhuma Pessoa Juridica");
    }
}

private static void buscarPeloid() {
    String tipo;
    int id;

    System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");
    tipo = scanner.next().toUpperCase();

    if (!tipo.equals("J") && !tipo.equals("F")) {
        System.out.println("Opção inválida!");
    }
}

```

```

        return;
    }

    System.out.println("Digite o id da Pessoa:");
    id = scanner.nextInt();

    if (tipo.equals("J")) {
        PessoaJuridica pessoa = repoJuridica.obter(id);
        if (pessoa == null) {
            System.out.println("Pessoa não encontrada");
        } else {
            System.out.println("Pessoa Juridica:");
            pessoa.exibir();
        }
    }

    if (tipo.equals("F")) {
        PessoaFisica pessoa = repoFisica.obter(id);
        if (pessoa == null) {
            System.out.println("Pessoa não encontrada");
        } else {
            System.out.println("Pessoa Fisica:");
            pessoa.exibir();
        }
    }
}

private static void excluirPessoa() {
    String tipo;
    int id;

    System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");
    tipo = scanner.next().toUpperCase();

    if (!tipo.equals("J") &&!tipo.equals("F")) {
        System.out.println("Opção inválida!");
        return;
    }

    System.out.println("Digite o id da Pessoa:");
    id = scanner.nextInt();

    if (tipo.equals("J")) {
        PessoaJuridica pessoa = repoJuridica.obter(id);

```

```

        if (pessoa == null) {
            System.out.println("Pessoa não encontrada");
        } else {
            repoJuridica.excluir(id);
            System.out.printf("A pessoa com a id:%d e o nome: %s foi excluída%n",
                pessoa.getId(), pessoa.getNome());
            saveData("J");
        }
    }
}

```

```

        if (tipo.equals("F")) {
            PessoaFisica pessoa = repoFisica.obter(id);
            if (pessoa == null) {
                System.out.println("Pessoa não encontrada");
            } else {
                repoFisica.excluir(id);
                System.out.printf("A pessoa com a id:%d e o nome: %s foi excluída%n",
                    pessoa.getId(), pessoa.getNome());
                saveData("F");
            }
        }
    }
}

```

```

private static void alterarPessoa() {
    String tipo;
    int id;
    String nome;
    String cpf;
    int idade;
    String cnpj;
    PessoaJuridica pessoaJuridica = null;
    PessoaFisica pessoaFisica = null;

    System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");
    tipo = scanner.next().toUpperCase();

    if (!tipo.equals("J") && !tipo.equals("F")) {
        System.out.println("Opção inválida!");
        return;
    }

    System.out.println("Digite o id da Pessoa:");
    id = scanner.nextInt();
}

```

```

if (tipo.equals("J")) {
    pessoaJuridica = repoJuridica.obter(id);
    if (pessoaJuridica == null) {
        System.out.println("Pessoa não encontrada");
        return;
    }
}

if (tipo.equals("F")) {
    pessoaFisica = repoFisica.obter(id);
    if (pessoaFisica == null) {
        System.out.println("Pessoa não encontrada");
        return;
    }
}

System.out.println("Insira os dados...");
if (tipo.equals("J")) {
    System.out.printf("Nome (%s):", pessoaJuridica.getNome());
    System.out.println("");
}

if (tipo.equals("F")) {
    System.out.printf("Nome (%s):", pessoaFisica.getNome());
    System.out.println("");
}

nome = scanner.next();

if (tipo.equals("J")) {
    System.out.printf("CNPJ (%s):", pessoaJuridica.getCnpj());
    System.out.println("");
    cnpj = scanner.next();

    PessoaJuridica pessoaJ = new PessoaJuridica(id, nome, cnpj);
    repoJuridica.alterar(id, pessoaJ);

    saveData("J");
}

if (tipo.equals("F")) {
    System.out.printf("CPF (%s):", pessoaFisica.getCpf());
    System.out.println("");
    cpf = scanner.next();
}

```



```

        System.out.printf("Idade (%d):", pessoaFisica.getIdade());
        System.out.println("");
        idade = scanner.nextInt();

        PessoaFisica pessoaF = new PessoaFisica(id, nome, cpf, idade);
        repoFisica.alterar(id, pessoaF);

        saveData("F");
    }
}

private static void persistirDados() {
    String tipo;
    String prefixo;
    File file;

    System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");
    tipo = scanner.next().toUpperCase();

    if (!tipo.equals("J") &&!tipo.equals("F")) {
        System.out.println("Opção inválida!");
        return;
    }

    System.out.println("Insira o prefixo do arquivo:");
    prefixo = scanner.next();

    if (tipo.equals("J")) {
        file = new File(prefixo + ".juridica.bin");
    } else {
        file = new File(prefixo + ".fisica.bin");
    }

    try (ObjectOutputStream oos = new ObjectOutputStream(new FileOutputStream(file))) {
        if (tipo.equals("J")) {
            List<PessoaJuridica> pessoas = repoJuridica.obterTodos();
            oos.writeObject(pessoas);
        } else {
            List<PessoaFisica> pessoas = repoFisica.obterTodos();
            oos.writeObject(pessoas);
        }
    } catch (IOException e) {
        System.err.println("Erro ao salvar dados: " + e.getMessage());
    }
}

```

```

        return;
    }

    System.out.println("Dados salvos com êxito");
}

private static void recuperarDados() {
    String tipo;
    String prefixo;
    File file;
    ObjectInputStream ois;
    List<PessoaFisica> pessoasFisicas = null;
    List<PessoaJuridica> pessoasJuridicas = null;

    System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");
    tipo = scanner.next().toUpperCase();

    if (!tipo.equals("J") && !tipo.equals("F")) {
        System.out.println("Opção inválida!");
        return;
    }

    System.out.println("Insira o prefixo do arquivo:");
    prefixo = scanner.next();

    try {
        if (tipo.equals("J")) {
            file = new File(prefixo + ".juridica.bin");
            ois = new ObjectInputStream(new FileInputStream(file));
            pessoasJuridicas = (List<PessoaJuridica>) ois.readObject();
            ois.close();
            repoJuridica.novosDados(pessoasJuridicas);
            saveData("J");
        }
        if (tipo.equals("F")) {
            file = new File(prefixo + ".fisica.bin");
            ois = new ObjectInputStream(new FileInputStream(file));
            pessoasFisicas = (List<PessoaFisica>) ois.readObject();
            ois.close();
            repoFisica.novosDados(pessoasFisicas);
            saveData("F");
        }
    } catch (IOException | ClassNotFoundException e) {
        System.err.println("Erro ao recuperar dados: " + e.getMessage());
    }
}

```

```

        return;
    }

    System.out.println("Dados recuperados com êxito");
}
}

```

- **Pessoa.java**

```

package cadastrapoo;

import java.io.Serializable;

/**
 *
 * @author Marvin
 */

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

    // Getters e Setters
    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**

```

package cadastrapoo;

/**
 *
 * @author Marvin
 */
import java.io.Serializable;

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

    public PessoaFisica() {
        super();
    }

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

```

```

    public String getCpf() {
        return cpf;
    }

    public int getIdade() {
        return idade;
    }
}

```

- **PessoaFisicaRepo.java**

```

package cadastropoo;

/**
 *
 * @author Marvin
 */
import java.io.*;
import java.util.ArrayList;
import java.util.List;

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

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

    public void alterar(int id, PessoaFisica novaPessoaFisica) {
        for (int i = 0; i < pessoasFisicas.size(); i++) {
            if (pessoasFisicas.get(i).getId() == id) {
                pessoasFisicas.set(i, novaPessoaFisica);
                break;
            }
        }
    }

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

```

```

public void novosDados(List<PessoaFisica> pessoas) {
    pessoasFisicas = pessoas;
}

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

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

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

public List<PessoaFisica> recuperar(String fileName) throws IOException,
ClassNotFoundException {
    try (ObjectInputStream ois = new ObjectInputStream(new FileInputStream(fileName))) {
        pessoasFisicas = (List<PessoaFisica>) ois.readObject();
    }
    return pessoasFisicas;
}
}

```

- **PessoaJuridica.java**

```

package cadastropoo;

/**
 *
 * @author Marvin
 */
import java.io.Serializable;

```

```

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

    public PessoaJuridica() {
        super();
    }

    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;
    }
}

```

- **PessoaJuridicaRepo.java**

```

package cadastrapoo;

/**
 *
 * @author Marvin
 */
import java.io.*;
import java.util.ArrayList;
import java.util.List;

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

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

```

```

public void alterar(int id, PessoaJuridica novaPessoaJuridica) {
    for (int i = 0; i < pessoasJuridicas.size(); i++) {
        if (pessoasJuridicas.get(i).getId() == id) {
            pessoasJuridicas.set(i, novaPessoaJuridica);
            break;
        }
    }
}

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

public void novosDados(List<PessoaJuridica> pessoas) {
    pessoasJuridicas = pessoas;
}

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

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

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

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

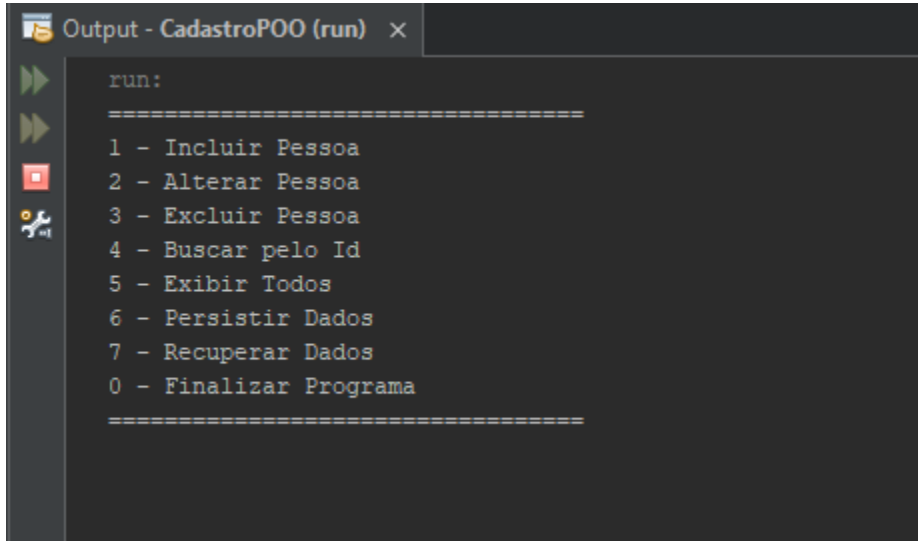
```



```
}  
}
```

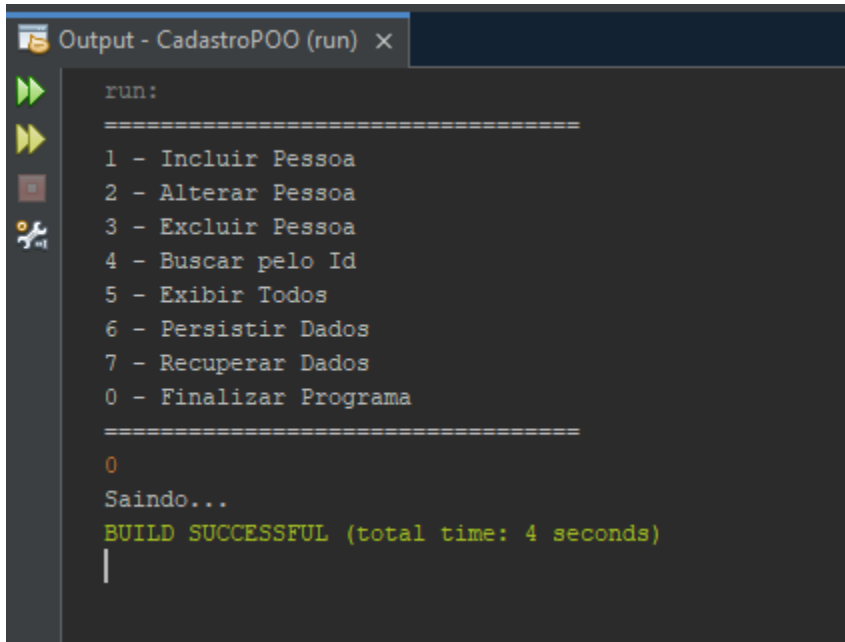
Os resultados da execução dos códigos também devem ser apresentados;

- Opções



```
Output - CadastroPOO (run) x  
run:  
=====  
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  
=====
```

- Finalizar programa



```
Output - CadastroPOO (run) x  
run:  
=====  
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  
=====  
0  
Saindo...  
BUILD SUCCESSFUL (total time: 4 seconds)  
|
```

- Incluir Pessoa Fisica e Exibir Todos

```
Output - CadastroPOO (run) x
run:
=====
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
=====

1
F - Pessoa Fisica | J - Pessoa Juridica
F
Digite o id da Pessoa:
2
Insira os dados...
Nome:
Marvin
CPF:
123456789
Idade:
33
=====

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
=====

5
Dados de Pessoas Fisicas:
Id: 1
Nome: chantily
CPF: 1234556
Idade: 24

Id: 2
Nome: Marvin
CPF: 123456789
Idade: 33

-----
Dados de Pessoas Juridicas:
Id: 1
Nome: marvinenterprise
CNPJ: 1234556
=====

1 - Incluir Pessoa
```

- Incluir Pessoa Juridica e Exibir Todos

```
Output - CadastroPOO (run) x
run:
=====
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
=====
1
F - Pessoa Fisica | J - Pessoa Juridica
J
Digite o id da Pessoa:
23
Insira os dados...
Nome:
Pizza
CNPJ:
213213123
=====
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
=====
5
Dados de Pessoas Fisicas:
Id: 1
Nome: chantily
CPF: 1234556
Idade: 24

-----
Dados de Pessoas Juridicas:
Id: 1
Nome: marvinenterprise
CNPJ: 1234556

Id: 23
Nome: Pizza
CNPJ: 213213123
=====
1 - Incluir Pessoa
```

- Alterar Pessoa e Buscar pelo Id

```
Output - CadastroPOO (run) x
=====
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
=====
2
F - Pessoa Fisica | J - Pessoa Juridica
F
Digite o id da Pessoa:
2
Insira os dados...
Nome (Marvin):
Marvilin
CPF (123456789):
1122223333444
Idade (33):
34
=====
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
=====
4
F - Pessoa Fisica | J - Pessoa Juridica
F
Digite o id da Pessoa:
2
Pessoa Fisica:
Id: 2
Nome: Marvilin
CPF: 1122223333444
Idade: 34
```

- Excluir Pessoa

```
Output - CadastroPOO (run) x
Dados de Pessoas Fisicas:
Id: 1
Nome: chantily
CPF: 1234556
Idade: 24

Id: 2
Nome: Marviline
CPF: 1122223333444
Idade: 34

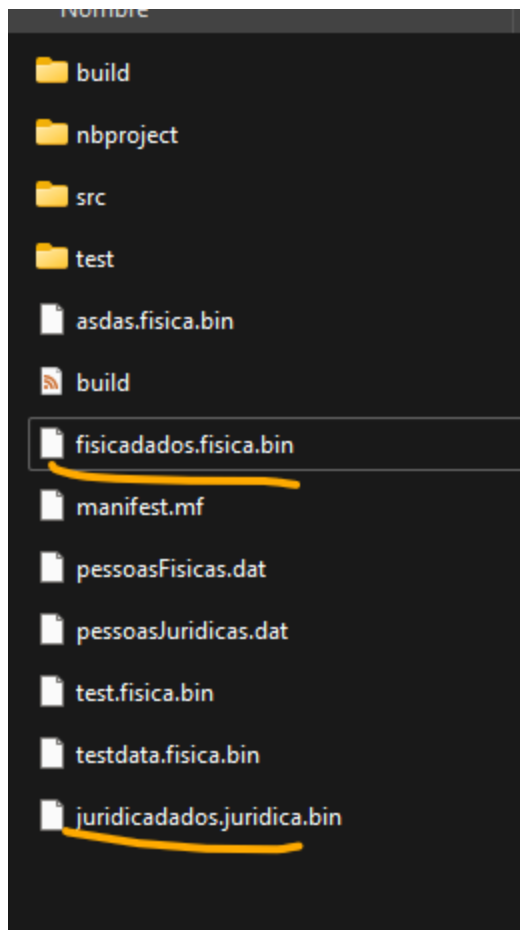
-----
Dados de Pessoas Juridicas:
Id: 1
Nome: marvinenterprise
CNPJ: 1234556

=====
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
=====
3
F - Pessoa Fisica | J - Pessoa Juridica
F
Digite o id da Pessoa:
2
A pessoa com a id:2 e o nome: Marviline foi excluida
=====
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
=====
5
Dados de Pessoas Fisicas:
Id: 1
Nome: chantily
CPF: 1234556
Idade: 24

-----
Dados de Pessoas Juridicas:
Id: 1
```

- Persistir dados

```
Output - CadastroPOO (run) x
run:
=====
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
=====
6
F - Pessoa Fisica | J - Pessoa Juridica
F
Insira o prefixo do arquivo:
fisicadados
Dados salvos com êxito
=====
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
=====
6
F - Pessoa Fisica | J - Pessoa Juridica
J
Insira o prefixo do arquivo:
juridicadados
Dados salvos com êxito
=====
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
=====
```



– Recuperar Dados (Primeiro, excluímos os dados)

```
=====
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
=====
3
F - Pessoa Fisica | J - Pessoa Juridica
j
Digite o id da Pessoa:
23
A pessoa com a id:23 e o nome: Pizza foi excluída
=====
```

```
=====
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
=====
3
F - Pessoa Fisica | J - Pessoa Juridica
f
Digite o id da Pessoa:
1
A pessoa com a id:1 e o nome: chantily foi excluida
```

```
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
=====
3
F - Pessoa Fisica | J - Pessoa Juridica
j
Digite o id da Pessoa:
1
A pessoa com a id:1 e o nome: marvinenterprise foi excluida
```

```
=====
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
=====
5
Dados de Pessoas Fisicas:
Nenhuma Pessoa Fisica
-----
Dados de Pessoas Juridicas:
Nenhuma Pessoa Juridica
=====
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
=====
1
```


– Recuperar Dados Pessoas Fisicas

```
Output - CadastroPOO (run) x
run:
=====
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
=====
5
Dados de Pessoas Fisicas:
Nenhuma Pessoa Fisica
-----
Dados de Pessoas Juridicas:
Nenhuma Pessoa Juridica
=====
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
=====
7
F - Pessoa Fisica | J - Pessoa Juridica
f
Insira o prefixo do arquivo:
fisicadados
Dados recuperados com êxito
=====
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
=====
5
Dados de Pessoas Fisicas:
Id: 1
Nome: chantily
CPF: 1234556
Idade: 24
```

– Recuperar Dados Pessoas Juridicas

```
=====
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
=====
7
F - Pessoa Fisica | J - Pessoa Juridica
j
Insira o prefixo do arquivo:
juridicadados
Dados recuperados com êxito
=====
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
=====
5
Dados de Pessoas Fisicas:
Id: 1
Nome: chantily
CPF: 1234556
Idade: 24

-----
Dados de Pessoas Juridicas:
Id: 1
Nome: marvinenterprise
CNPJ: 1234556

Id: 23
Nome: Pizza
CNPJ: 213213123
```

Análise e Conclusão:

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

Os elementos estáticos em Java são aqueles que pertencem à classe e não a uma instância específica dessa classe. Isso significa que você pode acessá-los sem criar um objeto da classe. Os elementos estáticos incluem variáveis estáticas, métodos estáticos e blocos estáticos. As variáveis estáticas são compartilhadas por todas as instâncias de uma classe, enquanto os métodos estáticos podem ser chamados sem a criação de uma instância da classe. Os blocos estáticos são usados para inicializar variáveis estáticas.

O motivo pelo qual o método “main” adota o modificador “static” é porque ele precisa ser acessível sem a criação de uma instância da classe. Em Java, o ponto de entrada de qualquer aplicativo é o método “main”, e esse método deve poder ser executado antes que qualquer instância da classe seja criada. Ao marcar o método “main” como “static” (estático), você está dizendo ao compilador Java que esse método pertence à própria classe e não a qualquer instância específica dessa classe. Isso permite que o programa seja iniciado e executado mesmo antes de qualquer objeto da classe ser criado.

Para que serve a classe Scanner?

A classe “Scanner” em Java faz parte do pacote “java.util” e foi introduzida na versão 1.5 do Java. É usada principalmente para receber entrada do usuário e analisá-la em tipos de dados primitivos, como “int”, “double” ou o tipo padrão “String”. É uma classe utilitária para analisar dados usando expressões regulares por meio da geração de tokens.

A classe “Scanner” oferece vários construtores que permitem a leitura de diferentes fontes de entrada, sendo as mais comuns:

- **InputStream:** normalmente, passamos “System.in” para receber a entrada do usuário.
- **File ou Path:** podemos ler dados de arquivos e trabalhar com os valores obtidos a partir deles.
- **String:** também podemos criar um scanner para uma fonte de string e analisar valores a partir dela.

Para criar um objeto da classe “Scanner”, geralmente passamos o objeto predefinido “System.in”, que representa o fluxo de entrada padrão. Se quisermos ler a entrada de um arquivo, podemos passar um objeto da classe “File”. A classe “Scanner” fornece métodos específicos para ler valores numéricos de determinados tipos de dados (“nextInt()”, “nextDouble()” etc.) e para ler cadeias de caracteres (“nextLine()”, bem como caracteres

individuais (`next().charAt(0)`). Ele divide cada linha de entrada em tokens, que são pequenos elementos significativos para o compilador Java.

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

O uso de classes de repositório teve um impacto significativo na organização do código, especialmente em projetos de software que envolvem operações CRUD (Create, Read, Update, Delete) com bancos de dados.

O uso de classes de repositório transformou a maneira como o código é organizado em projetos de software modernos, promovendo clareza, modularidade, reutilização, testabilidade e escalabilidade.