



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

**André Luis Gonçalves Carvalho - Matrícula: 20220318540**

**Campus Barra World - Desenvolvimento Full-Stack  
Vamos manter as informações! – 9001 – 2023.3**

#### **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.**

#### **Link GitHub**

**<https://github.com/ANDREC1986/RPG0014-202203185403.git>**

## 1º Procedimento | Criação das Entidades e Sistema de Persistência

### 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 André Carvalho
 */
public class Pessoa implements Serializable {

    protected int id;
    protected String nome;

    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);}

    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 model;  
  
import java.io.Serializable;  
  
/*  
 * 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  
*/  
  
/**  
 *  
 * @author André Carvalhal  
 */  
public class PessoaFisica extends Pessoa implements Serializable {  
    protected String cpf;  
    protected int idade;  
  
    /**  
     *  
     * @param id  
     * @param nome  
     * @param cpf  
     * @param idade  
     */  
  
    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 void setCpf(String cpf) {
    this.cpf = cpf;
}

public int getIdade() {
    return idade;
}

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

```

## 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 André Carvalho
*/
public class PessoaJuridica extends Pessoa implements Serializable {
    protected String cnpj;

    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;
    }
}

```

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

```

```

/**
 *
 * @author andre
 */
public class PessoaFisicaRepo {
    ArrayList<PessoaFisica> Repositorio;

    public PessoaFisicaRepo() {
        this.Repositorio = new ArrayList<PessoaFisica>();
    };

    public void inserir(int id, String nome, String cpf, int idade){
        Repositorio.add(new PessoaFisica(id,nome,cpf,idade));
    };

    public void alterar(int id, String param, String value){
        for (PessoaFisica i : Repositorio){
            if (i.getId() == id) {
                switch(param) {
                    case "nome" -> i.setNome(value);
                    case "cpf" -> i.setCpf(value);
                    case "idade" -> i.setIdade(Integer.parseInt(value));
                }
                break;
            }
        }
    };

    public void excluir(int id){
        for(int i = 0; i < Repositorio.size(); i++) {
            if(Repositorio.get(i).getId() == id) {
                Repositorio.remove(i);
                break;
            }
        }
    };

    public int obter(int id){
        for (PessoaFisica i : Repositorio) {
            if (id == i.getId()) {
                i.exibir();
                return 1;
            }
        }
        System.out.println("Nao Encontrado");
        return 0;
    }
}

```

```

};

public void obterTodos(){
for (PessoaFisica i : Repositorio){
    System.out.println("-----");
    i.exibir();
    System.out.println("-----");
}
};

public int size(){
return Repositorio.size();
};

public void recuperar(String fileToLoad) throws FileNotFoundException, IOException,
ClassNotFoundException{
    FileInputStream fis = new FileInputStream(fileToLoad);
    ObjectInputStream ois;
    ois = new ObjectInputStream(fis);
    this.Repositorio = (ArrayList<model.PessoaFisica>) ois.readObject();
    System.out.println("Dados de Pessoa Fisica Recuperados.");

};

public void persistir(String fileToSaveTo) throws FileNotFoundException,
IOException{
    FileOutputStream fos = new FileOutputStream(fileToSaveTo);
    ObjectOutputStream out = new ObjectOutputStream(fos);
    out.writeObject(Repositorio);
    System.out.println("Dados de Pessoa Fisica Armazenados.");
};
}

```

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

/**
 *
 * @author andre
 */
public class PessoaJuridicaRepo {
    ArrayList<PessoaJuridica> Repositorio;

    public PessoaJuridicaRepo() {
        this.Repositorio = new ArrayList<>();
    };

    public void inserir(int id, String nome, String cnpj){
        Repositorio.add(new PessoaJuridica(id,nome,cnpj));
    };

    public void alterar(int id, String param, String value){
        for (PessoaJuridica i : Repositorio){
            if (i.getId() == id) {
                switch(param) {
                    case "nome" -> i.setNome(value);
                    case "cpf" -> i.setCnpj(value);
                }
                break;
            }
        }
    };

    public void excluir(int id){
        for(int i = 0; i < Repositorio.size(); i++) {
            if(Repositorio.get(i).getId() == id) {
                Repositorio.remove(i);
                break;
            }
        }
    };

    public int obter(int id){

```



```

for (PessoaJuridica i : Repositorio) {
    if (id == i.getId()) {
        i.exibir();
        return 1;
    }
}

System.out.println("Nao Encontrado");
return 0;
};

public void obterTodos(){
for (PessoaJuridica i : Repositorio){
    System.out.println("-----");
    i.exibir();
    System.out.println("-----");
}
};

public int size(){
    return Repositorio.size();
};

public void recuperar(String fileToLoad) throws FileNotFoundException, IOException,
ClassNotFoundException{
    FileInputStream fis = new FileInputStream(fileToLoad);
    ObjectInputStream ois;
    ois = new ObjectInputStream(fis);
    this.Repositorio = (ArrayList<PessoaJuridica>)ois.readObject();
    System.out.println("Dados de Pessoa Juridica Recuperados.");

};

public void persistir(String fileToSaveTo) throws FileNotFoundException,
IOException{
    FileOutputStream fos = new FileOutputStream(fileToSaveTo);
    ObjectOutputStream out = new ObjectOutputStream(fos);
    out.writeObject(Repositorio);
    System.out.println("Dados de Pessoa Juridica Armazenados.");
};
}

```

## CadastroPOO.java (Do Teste)

```
/*
 * 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 cadastrapoo;

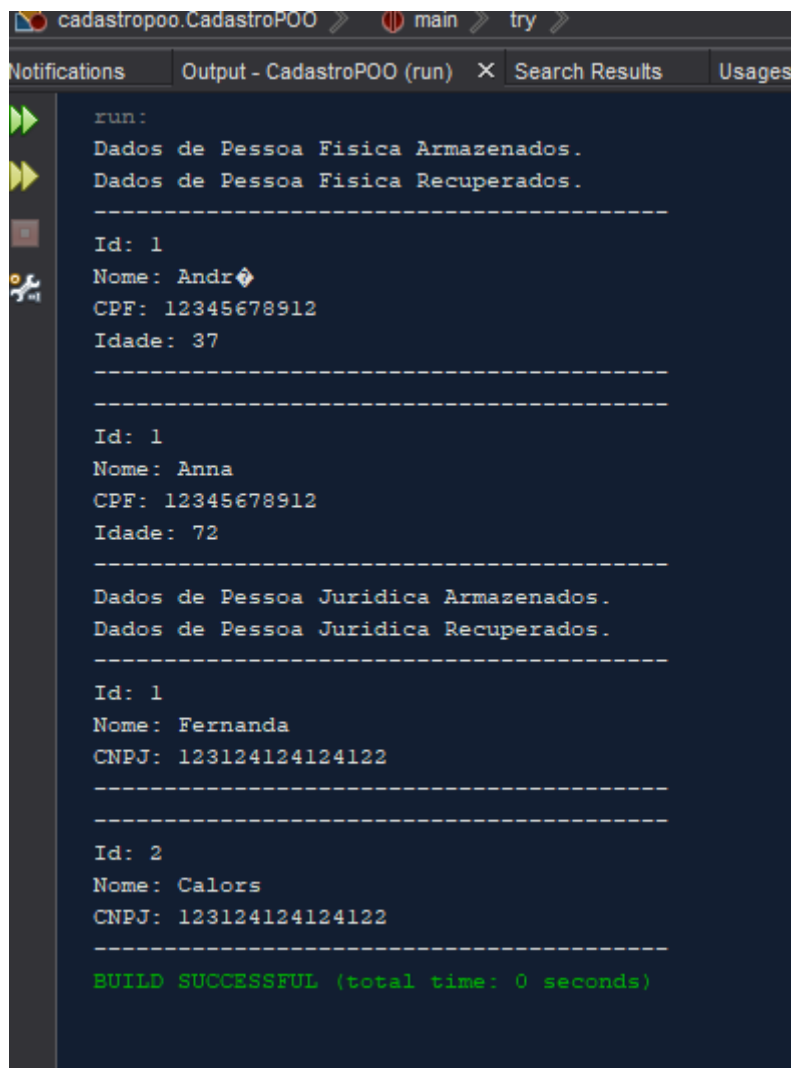
import java.io.FileNotFoundException;
import java.io.IOException;
import java.util.logging.Level;
import java.util.logging.Logger;
import model.PessoaFisicaRepo;
import model.PessoaJuridicaRepo;

/**
 *
 * @author André Carvalhal
 */
public class CadastroPOO {
    public static void main(String[] args) throws FileNotFoundException,
ClassNotFoundException {
        PessoaFisicaRepo repo1 = new PessoaFisicaRepo();
        repo1.inserir(1, "André", "12345678912", 37);
        repo1.inserir(1, "Anna", "12345678912", 72);
        try {
            repo1.persistir("testePF.bin");
            PessoaFisicaRepo repo2 = new PessoaFisicaRepo();
            repo2.recuperar("testePF.bin");
            repo2.obterTodos();
        } catch (IOException ex) {
            Logger.getLogger(CadastroPOO.class.getName()).log(Level.SEVERE, null, ex);
        }
        PessoaJuridicaRepo repo3 = new PessoaJuridicaRepo();
        repo3.inserir(1, "Fernanda", "123124124124122");
        repo3.inserir(2, "Calors", "123124124124122");
    }
}
```

```

try {
    repo3.persistir("testePJ.bin");
    PessoaJuridicaRepo repo4 = new PessoaJuridicaRepo();
    repo4.recuperar("testePJ.bin");
    repo4.obterTodos();
} catch (IOException ex) {
    Logger.getLogger(CadastroPOO.class.getName()).log(Level.SEVERE, null, ex);
}
}

```



```

run:
Dados de Pessoa Fisica Armazenados.
Dados de Pessoa Fisica Recuperados.
-----
Id: 1
Nome: Andr 
CPF: 12345678912
Idade: 37
-----
Id: 1
Nome: Anna
CPF: 12345678912
Idade: 72
-----
Dados de Pessoa Juridica Armazenados.
Dados de Pessoa Juridica Recuperados.
-----
Id: 1
Nome: Fernanda
CNPJ: 123124124124122
-----
Id: 2
Nome: Calors
CNPJ: 123124124124122
-----
BUILD SUCCESSFUL (total time: 0 seconds)

```

## **Conclusão:**

### **a) Quais as vantagens e desvantagens do uso de herança?**

As vantagens estão na reutilização do código, podendo desenvolver em uma superclasse e reutilizá-lo nas subclasses, você obtém extensibilidade ao poder estender a classe pai, criando novas subclasses, o código fica organizado de forma mais de compreender, pois estabelece uma hierarquia no código. As desvantagens são, o forte acoplamento entre a superclasse e as subclasses, tornando o sistema mais frágil, já que a alteração na superclasse afetará todos as subclasses relacionadas. As classes podem se tornar muito complexas, atrapalhando a reutilização do código.

### **b) Por que a interface Serializable é necessária ao efetuar persistência em arquivos binários?**

A interface “Serializable” permite ao mecanismo de serialização do java saber que determinada classe pode ser convertida em um forma binária, para posteriormente pode ser transmitida ou persistida em arquivo.

### **c) Como o paradigma funcional é utilizado pela API stream no Java?**

A API Stream no Java utilizando o paradigma funcional, assim como no Javascript que tivemos no mundo 2, ela permite a redução, ordenação, agrupamento, filtragem e mapeamento de dados em coleções de forma concisa.

### **d) Quando trabalhamos com Java, qual padrão de desenvolvimento é adotado na persistência de dados em arquivos?**

A Serialização, método que permite que classes sejam transformadas em bytes e operadas utilizando os ObjectOutputStream e ObjectInputStream para gravar e carregar os dados respectivamente.

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

### 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 cadastrapoo;

import java.io.IOException;
import java.util.Scanner;
import java.util.logging.Level;
import java.util.logging.Logger;
import model.PessoaFisicaRepo;
import model.PessoaJuridicaRepo;

/**
 *
 * @author André Carvalhal
 */
public class CadastroPOO {
    static Boolean running = true;
    static PessoaFisicaRepo repo1 = new PessoaFisicaRepo();
    static PessoaJuridicaRepo repo2 = new PessoaJuridicaRepo();
    static Scanner kb = new Scanner(System.in);
    public static void main(String[] args) {
        do {
            // TODO code application logic here
            Boolean navFlag = true;
            System.out.println("""
                =====
                1 - Incluir Pessoa
                2 - Alterar Pessoa1
                3 - Excluir Pessoa
                4 - Buscar pelo Id
                5 - Exibir Todos
                6 - Persistir Dados
            """);
        } while (navFlag);
    }
}
```

7 - Recuperar Dados  
0 - Finalizar Programa

=====

""");

String selected = kb.nextLine();

switch (selected) {

case "1" -> {

int tipo = tipo();

switch (tipo){

case 1 -> {

System.out.println("Insira o Nome:");

String nome = kb.nextLine();

System.out.println("Insira o CPF:");

String cpf = checked();

cpf = cpf.replace(".", "");

System.out.println("Insira a Idade:");

String idade = checked();

repo1.inserir(repo1.size() + 1, nome, cpf, Integer.parseInt(idade));

}

case 2 -> {

System.out.println("Insira o Nome:");

String nome = kb.nextLine();

System.out.println("Insira o CNPJ:");

String cnpj = checked();

repo2.inserir(repo2.size() + 1, nome, cnpj);

}

}

}

case "2" -> {

int tipo2 = tipo();

System.out.print("Informe a ID:");

int alterar = kb.nextInt();

switch(tipo2){

case 1 -> {

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

int pessoa = repo1.obter(alterar);

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

if (pessoa == 1) {

Boolean flag = false;

System.out.println("""

Deseja Editar:

N - Nome

C - CPF

```

        I - IDADE
        R - RETORNAR
        "");
while (navFlag == true){
    switch(kb.nextLine().toLowerCase()){
        case "n" -> {
            System.out.println("Novo nome:");
            String nome = kb.nextLine();
            repo1.alterar(alterar, "nome", nome);
            navFlag = false;
        }
        case "c" -> {
            System.out.println("Novo CPF:");
            String cpf = checked();
            repo1.alterar(alterar, "cpf", cpf);
            navFlag = false;
        }
        case "i" -> {
            System.out.println("Nova idade:");
            String idade = checked();
            repo1.alterar(alterar, "idade", idade);
            navFlag = false;
        }
        case "r" -> {
            navFlag = false;
        }
    }
} else {
    kb.nextLine();
}
case 2 -> {
    System.out.println("-----");
    int pessoa = repo2.obter(alterar);
    System.out.println("-----");
    if (pessoa == 1) {
        System.out.println("""
            Deseja Editar:
            N - Nome
            C - CNPJ
            R - RETORNAR
            """);
        while (navFlag == true){

```

```

switch(kb.nextLine().toLowerCase()){
    case "n" -> {
        System.out.println("Novo nome:");
        String nome = kb.nextLine();
        repo1.alterar(alterar, "nome", nome);
        navFlag = false;
    }
    case "c" -> {
        System.out.println("Novo cnpj:");
        String cpf = checked();
        repo2.alterar(alterar, "cnpj", cpf);
        navFlag = false;
    }

    case "r" -> {
        navFlag = false;
    }
}
}
}
}
}
}

case "3" -> {
    int tipo3 = tipo();
    System.out.println("Informe o ID:");
    int id = kb.nextInt();
    switch(tipo3) {
        case 1 -> {
            System.out.println("-----");
            int confirma = repo1.obter(id);
            System.out.println("-----");
            if(confirma == 1) { System.out.println("Excluido com Sucesso!");};
            repo1.excluir(id);
        }
        case 2 -> {
            System.out.println("-----");
            int confirma = repo2.obter(id);
            System.out.println("-----");
            if(confirma == 1) { System.out.println("Excluido com Sucesso!");};
            repo2.excluir(id);
        }
    }
}
}
}
}
}

```



```

    }
}
System.out.println("Insira R para retornar.");
do {
    String wait = kb.next().toLowerCase();
    if ("r".equals(wait)){
        navFlag = false;
    }
} while(navFlag == true);
}

case "4" -> {
int tipo4 = tipo();
System.out.println("Informe o ID:");
int id = kb.nextInt();
switch(tipo4) {
    case 1 -> {
        System.out.println("-----");
        repo1.obter(id);
        System.out.println("-----");
    }
    case 2 -> {
        System.out.println("-----");
        repo2.obter(id);
        System.out.println("-----");
    }
}
}
System.out.println("Insira R para retornar.");
do {
    String wait = kb.next().toLowerCase();
    if ("r".equals(wait)){
        navFlag = false;
    }
} while(navFlag == true);
}

case "5" -> {
int tipo5 = tipo();
switch(tipo5) {
    case 1 -> {
        repo1.obterTodos();
    }
    case 2 -> {
        repo2.obterTodos();
    }
}
System.out.println("Insira R para retornar.");
}

```

```

        do {
            String wait = kb.next().toLowerCase();
            if ("r".equals(wait)){
                navFlag = false;
            }
        } while(navFlag == true);
    }

    case "6" -> {
        int tipo6 = tipo();
        System.out.println("Nome do Arquivo a ser salvo:");
        String nome = kb.nextLine();
        switch(tipo6) {
            case 1 -> {
                try {
                    nome = nome + ".fisica.bin";
                    repo1.persistir(nome);
                } catch (IOException ex) {
                    System.out.print("Falha ao salvar o arquivo!");
                }
            }
            case 2 -> {
                try {
                    nome = nome + ".juridica.bin";
                    repo2.persistir(nome);
                } catch (IOException ex) {
                    System.out.print("Falha ao salvar o arquivo!");
                }
            }
        }
    }
}

case "7" -> {
    int tipo7 = tipo();
    System.out.println("Nome do arquivo a recuperar:");
    String arquivo = kb.nextLine();
    switch(tipo7) {
        case 1 -> {
            try {
                arquivo = arquivo + ".fisica.bin";
                repo1.recuperar(arquivo);
            } catch (IOException | ClassNotFoundException ex) {
                System.out.println("Nao foi possivel encontrar o arquivo!");
            }
        }
    }
}

```

```

        case 2 -> {
            try {
                arquivo = arquivo + ".juridica.bin";
                repo1.recuperar(arquivo);
            } catch (IOException | ClassNotFoundException ex) {
                System.out.println("Nao foi possivel encontrar o arquivo!");
            }
        }

    }

}

case "0" -> {
    running = false;}
}
} while (running == true);
}

```

```

public static String checked(){
    String value = "null";
    Scanner valor = new Scanner(System.in);
    while (value == "null"){
        value = valor.nextLine();
        value = value.replace(".", "");
        try {
            Long.parseLong(value);
        }
        catch (NumberFormatException e) {
            System.out.println("Valor deve ser numerico");
            value = "null";
        }
    }
    return value;
};

```

```

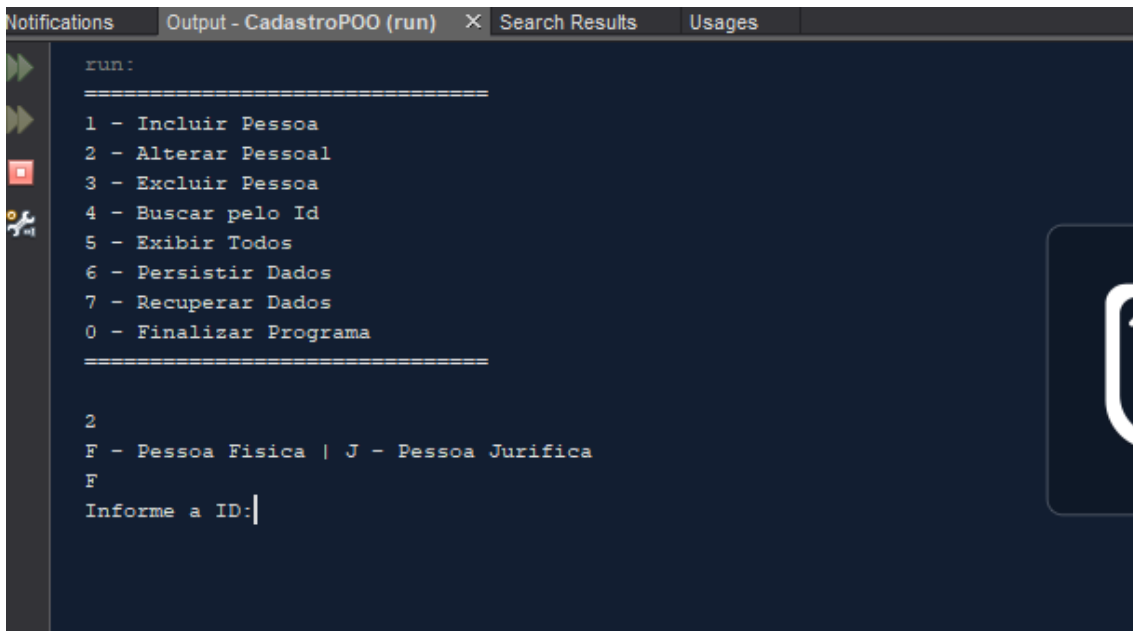
public static int tipo(){
    int selecionado = 0;
    Scanner selecionar = new Scanner(System.in);
    while (selecionado == 0) {
        System.out.println("F - Pessoa Fisica | J - Pessoa Jurifica");
        String tipo = selecionar.nextLine().toLowerCase();
        switch(tipo){
            case "f" -> {
                selecionado = 1;
            }
        }
    }
}

```

```

    }
    case "j" -> {
        selecionado = 2;
    }
    default -> {
        System.out.println("Tipo Invalido");
    }
}
}
return selecionado;
};
}

```



```

Notifications  Output - CadastroPOO (run)  X  Search Results  Usages
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
=====

2
F - Pessoa Fisica | J - Pessoa Jurifica
F
Informe a ID:|

```

## Conclusão:

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

Métodos estáticos são os métodos que não dependem do estado de um objeto, podendo ser invocados diretamente da classe, sem a necessidade de se instanciar um objeto. O Main utiliza pois o mesmo será acessado direto pela JVM, não tendo a necessidade de ser instanciado como um objeto

**b) Para que serve a classe Scanner?**

A classe Scanner é uma classe usada para ler dados, podendo ler dados de diversas fontes, como teclado, arquivos, streams, strings.

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

Permitiu a melhor leitura do código, separando o modelo do controle, permitindo a implementação mais rápida do método main.