

 <b>Estácio</b>	<p align="center"> <b>UNIVERSIDADE ESTÁCIO DE SÁ</b>          POLO INDAIATUBA – INDAIATUBA/SP          DESENVOLVIMENTO FULL STACK - 22.3          Relatório da Missão Prática   Nível 1   Mundo 3       </p>
Aluno:	SAULO HENRIQUE DOS SANTOS
Professor:	Rodrigo Dias
Repositório:	<a href="https://github.com/SauloHenriqueSantos/Mundo03Nivel01">https://github.com/SauloHenriqueSantos/Mundo03Nivel01</a>

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

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

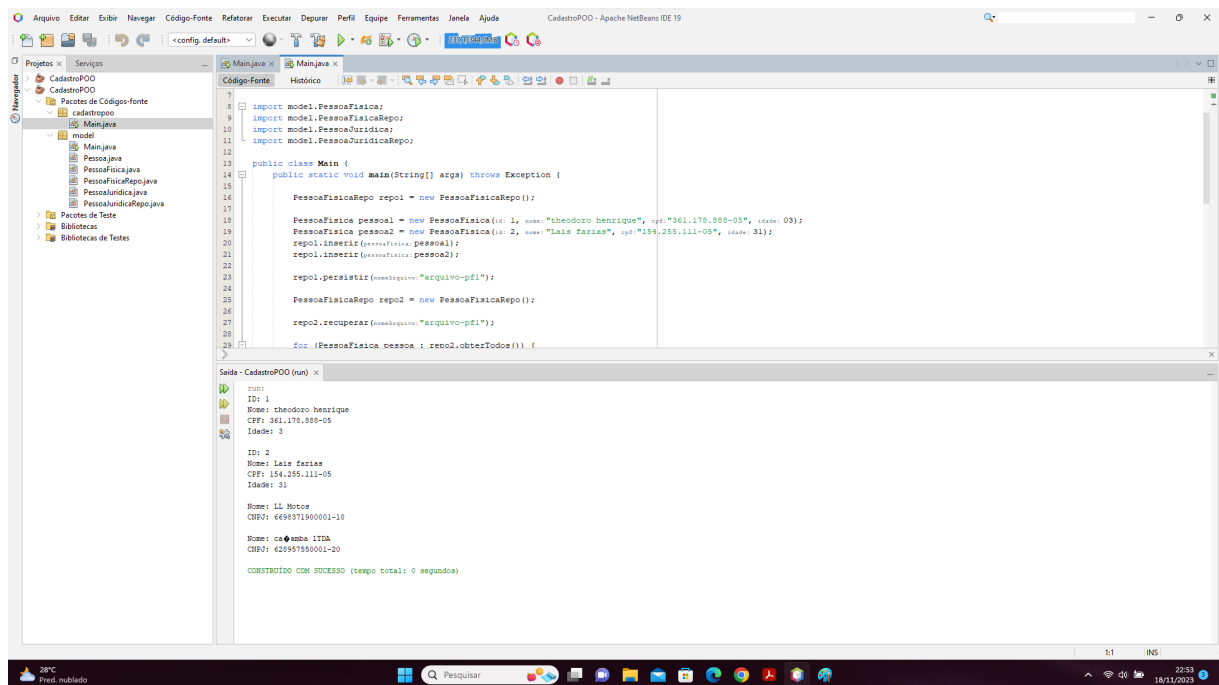
#### 1. Objetivos da prática:

- ☐ 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.

#### 2. Todos os códigos solicitados neste roteiro de aula:

<https://github.com/SauloHenriqueSantos/Mundo03Nivel01>

### 3. Resultados da execução dos códigos PT1:



The screenshot shows an IDE window titled 'CadastroPOO - Apache NetBeans IDE 19'. The main editor displays the source code for 'Main.java'. The code imports classes from the 'model' package and implements a 'Main' class with a 'main' method. The 'main' method creates two 'PessoaFisica' objects, inserts them into a repository, persists the repository to a file, and then retrieves and displays the data. The output window at the bottom shows the execution results, including the IDs, names, CPFs, and ages of the two individuals, followed by a confirmation message and execution time.

```
7 import model.PessoaFisica;
8 import model.PessoaFisicaRepo;
9 import model.PessoaJuridica;
10 import model.PessoaJuridicaRepo;
11
12 public class Main {
13     public static void main(String[] args) throws Exception {
14
15         PessoaFisicaRepo repol = new PessoaFisicaRepo();
16
17         PessoaFisica pessoa1 = new PessoaFisica(id: 1, nome: "theodoro henrique", cpf: "361.178.888-05", idade: 03);
18         PessoaFisica pessoa2 = new PessoaFisica(id: 2, nome: "Lais faria", cpf: "156.255.111-05", idade: 31);
19         repol.inserir(pessoa1);
20         repol.inserir(pessoa2);
21
22         repol.persistir(conteinerio: "arquivo-pfi");
23
24         PessoaFisicaRepo repo2 = new PessoaFisicaRepo();
25
26         repo2.recuperar(conteinerio: "arquivo-pfi");
27
28         for (PessoaFisica pessoa : repo2.obterTodos()) {
29
```

Seide - CadastroPOO (run) x

```
run:
ID: 1
Nome: theodoro henrique
CPF: 361.178.888-05
Idade: 3

ID: 2
Nome: Lais faria
CPF: 156.255.111-05
Idade: 31

Nome: LL Motow
CNPJ: 669371940001-10

Nome: G&mba LTDA
CNPJ: 629957550001-20

CONSTRUIDO COM SUCESSO (tempo total: 0 segundos)
```

### Análise e Conclusão:

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

Vantagens:

Reutilização de código

Extensibilidade, ou seja, criação de novas classes baseadas em classes existentes.

Desvantagens:

Código menos flexível e mais difícil de manter, por conta do acoplamento.

Fragilidade da hierarquia

Complexidade

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

A interface `Serializable` é necessária ao efetuar persistência em arquivos binários em Java pois permite que os objetos de uma classe sejam convertidos em uma sequência de bytes, tornando-os serializáveis, o que facilita sua gravação em um arquivo binário.

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

A API Stream no Java utiliza o paradigma funcional ao permitir operações de transformação e filtragem de dados em coleções de forma declarativa, usando funções lambda e expressões funcionais. Isso promove código mais conciso e legível, seguindo os princípios do paradigma funcional, como imutabilidade e composição de funções.

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

Em Java, um padrão comum de desenvolvimento para a persistência de dados em arquivos é o uso do padrão de projeto "Serialização". A serialização permite que objetos Java sejam convertidos em uma sequência de bytes e, em seguida, gravados em arquivos binários. Isso facilita a persistência e a recuperação de objetos e seus dados em arquivos, tornando-os portáteis e eficientes para armazenamento e transporte. Para implementar a serialização, é comum utilizar as interfaces **`Serializable`** e **`ObjectInputStream/ObjectOutputStream`**.

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

### 1. Resultados da execução dos códigos PT2:

```
25 System.out.println("1 - Incluir pessoa");
26 System.out.println("2 - Alterar Pessoa");
27 System.out.println("3 - Excluir Pessoa");
28 System.out.println("4 - Buscar pelo ID");
29 System.out.println("5 - Exibir Todos");
30 System.out.println("6 - Persistir Dados");
31 System.out.println("7 - Recuperar Dados");
32 System.out.println("0 - Finalizar Programa");
33 System.out.println("=====");
34 System.out.print("Digite a opção desejada: ");
35
36 try {
37     opcao = reader.readLine();
38
39     switch (opcao) {
40         case "1":
41             System.out.println("1 - Pessoa Fisica | 2 - Pessoa Juridica");
42             String tipoPessoa = reader.readLine();
43             switch (tipoPessoa) {
44                 case "1":
45                     PessoaFisica pf = lerDadosPessoaFisica(reader);
46                     repoFisica.inserir(pessoaFisica pf);
47                     System.out.println("((( Pessoa inserida com sucesso. )))");
```

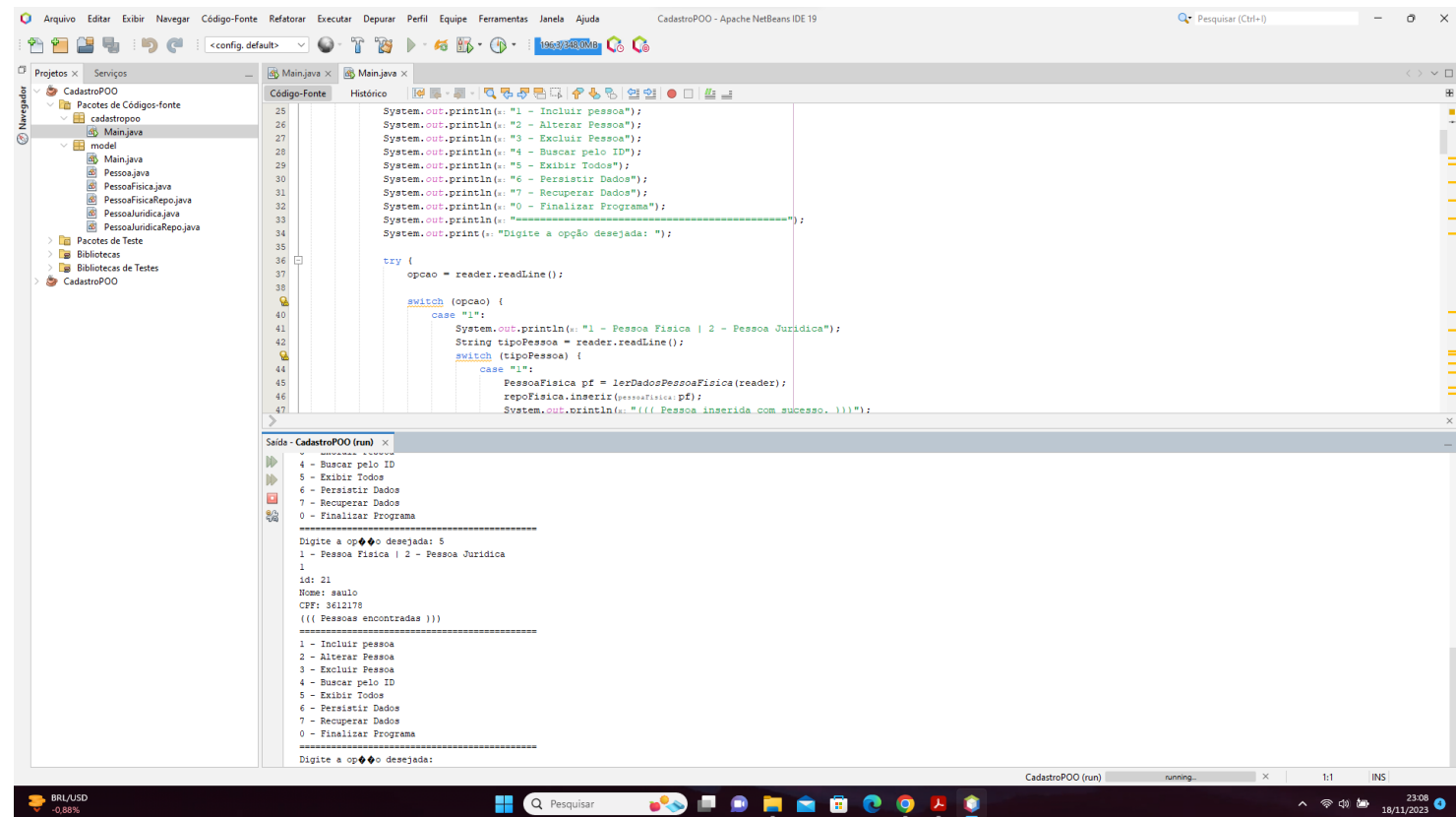
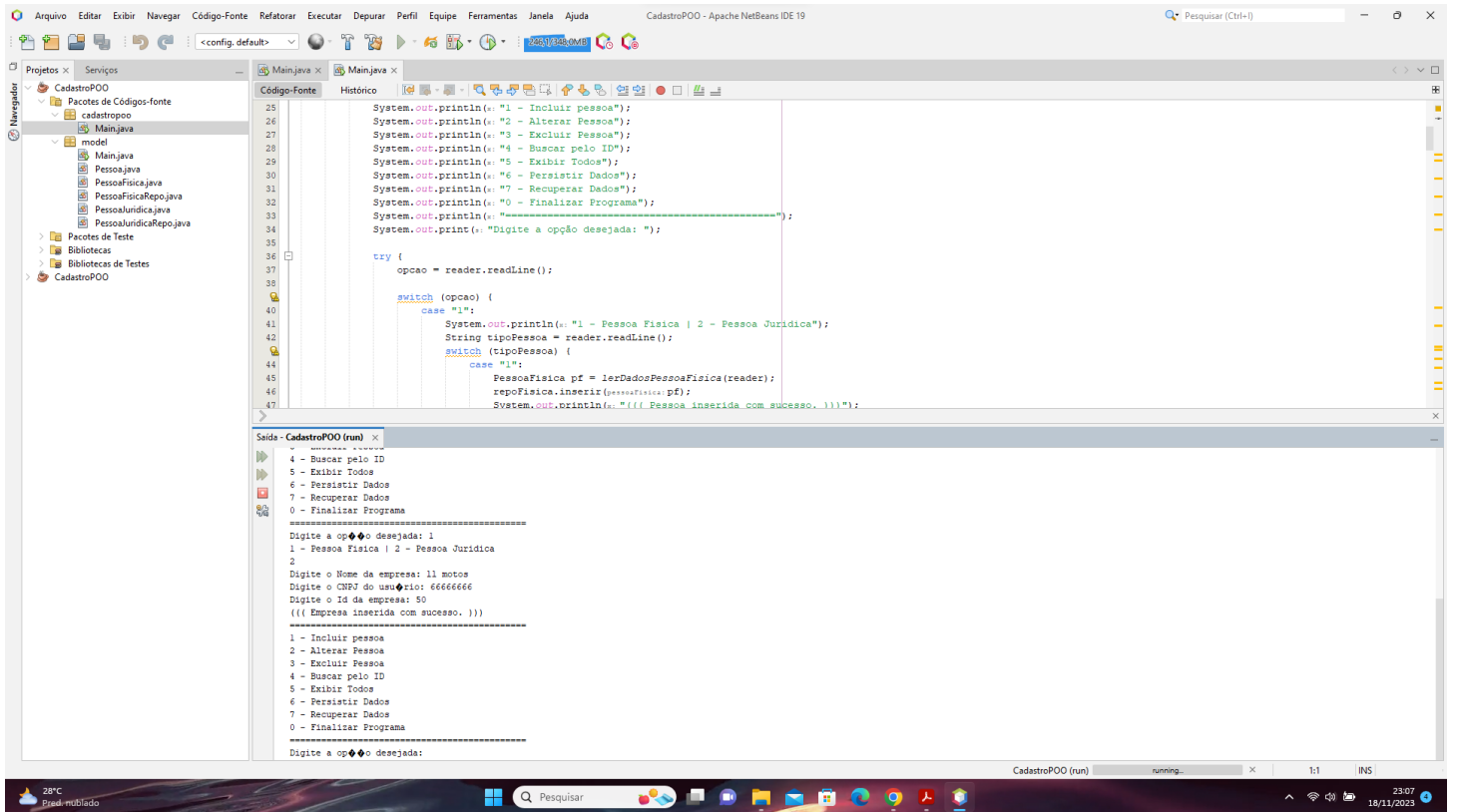
Saída - CadastroPOO (run)

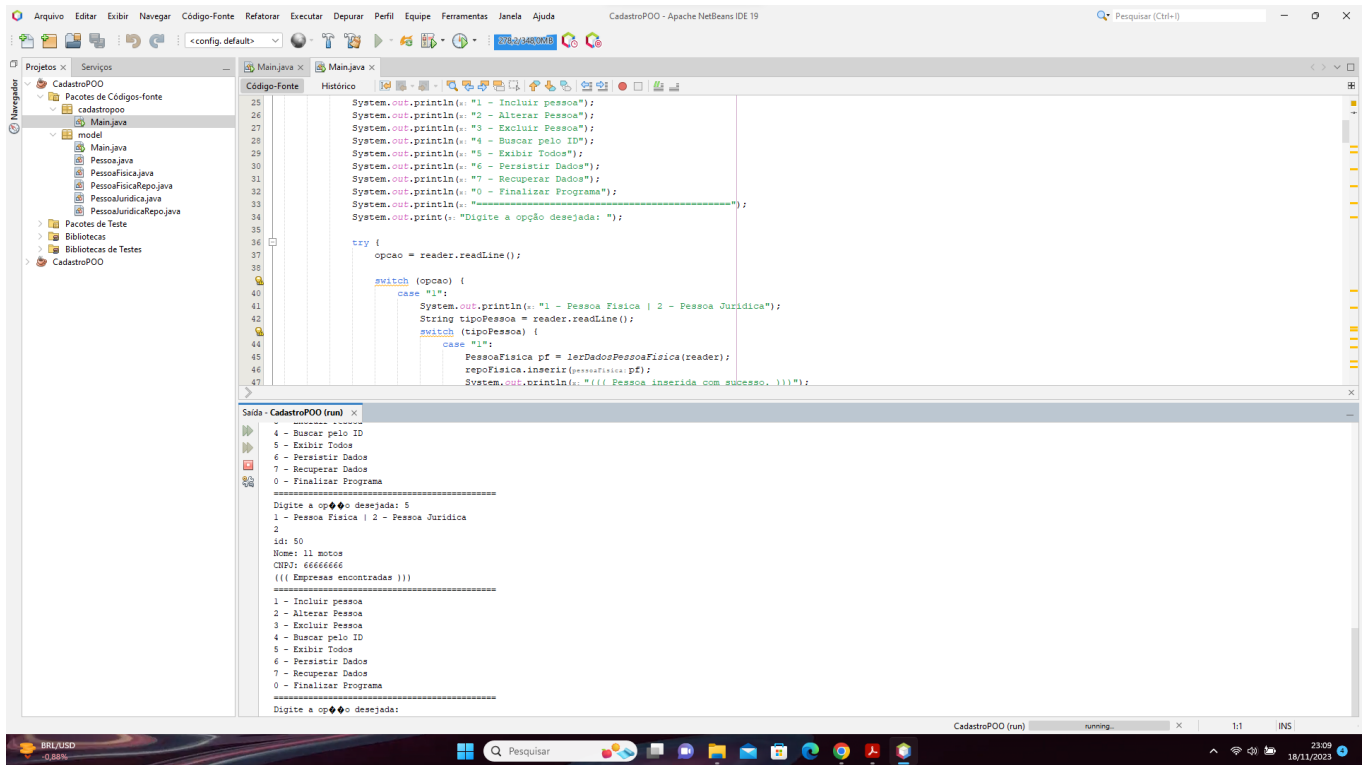
```
run:
1 - Incluir pessoa
2 - Alterar Pessoa
3 - Excluir Pessoa
4 - Buscar pelo ID
5 - Exibir Todos
6 - Persistir Dados
7 - Recuperar Dados
0 - Finalizar Programa
=====
Digite a opção desejada: 1
```

```
25 System.out.println("1 - Incluir pessoa");
26 System.out.println("2 - Alterar Pessoa");
27 System.out.println("3 - Excluir Pessoa");
28 System.out.println("4 - Buscar pelo ID");
29 System.out.println("5 - Exibir Todos");
30 System.out.println("6 - Persistir Dados");
31 System.out.println("7 - Recuperar Dados");
32 System.out.println("0 - Finalizar Programa");
33 System.out.println("=====");
34 System.out.print("Digite a opção desejada: ");
35
36 try {
37     opcao = reader.readLine();
38
39     switch (opcao) {
40         case "1":
41             System.out.println("1 - Pessoa Fisica | 2 - Pessoa Juridica");
42             String tipoPessoa = reader.readLine();
43             switch (tipoPessoa) {
44                 case "1":
45                     PessoaFisica pf = lerDadosPessoaFisica(reader);
46                     repoFisica.inserir(pessoaFisica pf);
47                     System.out.println("((( Pessoa inserida com sucesso. )))");
```

Saída - CadastroPOO (run)

```
7 - Recuperar Dados
0 - Finalizar Programa
=====
Digite a opção desejada: 1
1 - Pessoa Fisica | 2 - Pessoa Juridica
1
Digite o Nome do usuário: paulo
Digite o CPF do usuário: 3612178
Digite a Idade do usuário: 35
Digite o Id do usuário: 21
((( Pessoa inserida 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
=====
Digite a opção desejada: 1
1 - Pessoa Fisica | 2 - Pessoa Juridica
```





## 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 associados à classe em vez de instâncias individuais dessa classe. O método "main" é declarado como estático para que possa ser chamado sem a necessidade de criar um objeto da classe. Isso permite que seja o ponto de entrada do programa, sendo invocado diretamente pela JVM (Java Virtual Machine) durante a execução, sem a necessidade de criar uma instância da classe que contém o método.

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

A classe `Scanner` em Java é usada para ler entrada de dados do usuário ou de outros fluxos, como arquivos, de maneira simples e conveniente.

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

O uso de classes de repositório melhora a organização do código,

separando a lógica de acesso a dados da lógica de negócios,  
promovendo maior reutilização, testabilidade e clareza no código.