

 Estácio	<p align="center">UNIVERSIDADE ESTÁCIO DE SÁ POLO CENTRO – SANTA ROSA – RS</p> <p align="center">DESENVOLVIMENTO FULL STACK Relatório da Missão Prática Nível 1 – Mundo 3</p>
Disciplina:	RPG0014 - Iniciando o caminho pelo Java
Aluno/Matrícula:	Anderson Rech - 202304442215
Repositório:	https://github.com/4nderech/Mundo3-Missao-Pratica-N1.git

RPG0014 - Iniciando o caminho pelo Java

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

2. Objetivos 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.

3. Todos os códigos solicitados neste roteiro da aula:

→ <https://github.com/4nderech/Mundo3-Missao-Pratica-N1.git>

4. 1º Procedimento | Criação das Entidades e Sistema de Persistência

Resultados da execução do código do 1º procedimento:



```

etBeans IDE 22
Search (Ctrl+F)

Output - CadastroPOO (run) x
RUN:
Dados de Pessoa Fisica Armazenados.
Dados de Pessoa Fisica Recuperados.
ID: 1
Nome: Ana
CPF: 111111111111
Idade: 25
ID: 2
Nome: Carlos
CPF: 222222222222
Idade: 52
Dados de Pessoa Juridica Armazenados.
Dados de Pessoa Juridica Recuperados.
ID: 3
Nome: XPTO Sales
CPF: 333333333333
ID: 4
Nome: XPTO Solutions
CPF: 444444444444
BUILD SUCCESSFUL (total time: 0 seconds)

```

5. Análise e Conclusão

a. Quais as vantagens e desvantagens do uso de herança?

Vantagens da Herança

- ✓ **Reuso de Código:** Evita repetição, aproveitando métodos e atributos da superclasse.
- ✓ **Hierarquia de Classes:** Estrutura o código de forma intuitiva, refletindo relações do mundo real.
- ✓ **Polimorfismo:** Permite que objetos de subclasses sejam tratados como objetos da superclasse.
- ✓ **Extensibilidade:** Facilita a criação de novas classes baseadas em classes existentes.
- ✓ **Manutenção:** Mudanças na superclasse propagam-se para as subclasses, facilitando a manutenção.

Desvantagens da Herança

- ✗ **Acoplamento Aumentado:** Dependência entre superclasse e subclasses, complicando mudanças.
- ✗ **Herança Múltipla:** Java não suporta herança múltipla de classes, limitando o design.
- ✗ **Complexidade:** Hierarquias profundas tornam o código difícil de entender e manter.
- ✗ **Substituição de Métodos:** Modificar métodos da superclasse pode causar bugs.

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

A interface **Serializable** em Java é crucial para a persistência de objetos em arquivos binários porque permite converter um objeto em uma sequência de bytes, facilitando seu armazenamento e posterior recuperação. Esse processo de serialização e desserialização garante que a estrutura e o estado do objeto sejam preservados, permitindo que ele seja armazenado em arquivos ou transferido através de redes de forma consistente.

Implementando **Serializable**, os objetos podem ser facilmente salvos em arquivos, enviados através de streams e recuperados sem perder suas propriedades originais. Isso é especialmente útil para aplicações que exigem persistência de dados e comunicação entre diferentes partes do sistema.

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

A API de streams no Java utiliza o paradigma funcional para processar coleções de dados de forma declarativa e eficiente. Ela permite especificar operações sobre os dados, como filtragem, mapeamento e redução, sem modificar a coleção original, promovendo imutabilidade. As funções de alta

ordem, como expressões lambda, são usadas extensivamente, facilitando o uso de funções como argumentos. Além disso, o encadeamento de operações torna o código mais legível e expressivo, permitindo um fluxo de dados claro e contínuo. Isso resulta em código mais conciso, fácil de entender e manter, e possibilita processamento paralelo de maneira simples e eficaz.

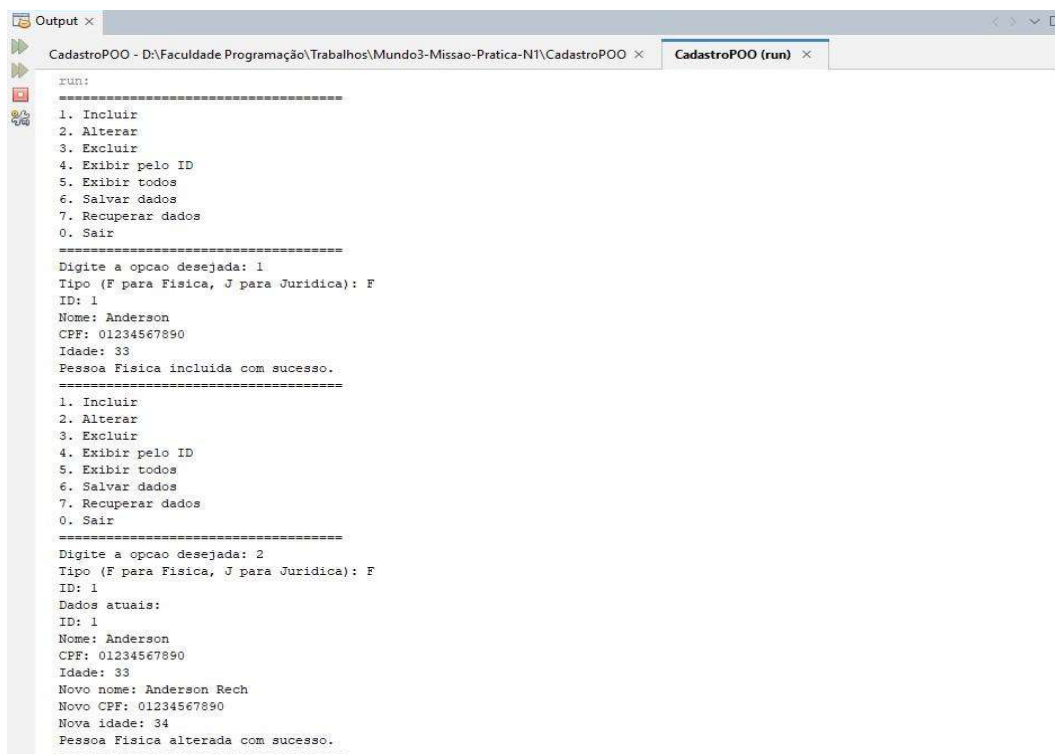
d. Quando trabalhamos com Java, qual padrão de desenvolvimento é adotado na persistência de dados em arquivos?

Um padrão comum de desenvolvimento para a persistência de dados em arquivos é o uso do padrão de projeto "Serialização". Para implementar a serialização em Java, é comum utilizar as interfaces `Serializable` e as classes `ObjectInputStream/ObjectOutputStream`. 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.

6. 2º Procedimento | Criação do Cadastro em Modo Texto

Resultados da execução do código do 2º Procedimento:

a. Inclusão e alteração do cadastro:



```
run:
=====
1. Incluir
2. Alterar
3. Excluir
4. Exibir pelo ID
5. Exibir todos
6. Salvar dados
7. Recuperar dados
0. Sair
=====
Digite a opcao desejada: 1
Tipo (F para Fisica, J para Juridica): F
ID: 1
Nome: Anderson
CPF: 01234567890
Idade: 33
Pessoa Fisica incluida com sucesso.
=====
1. Incluir
2. Alterar
3. Excluir
4. Exibir pelo ID
5. Exibir todos
6. Salvar dados
7. Recuperar dados
0. Sair
=====
Digite a opcao desejada: 2
Tipo (F para Fisica, J para Juridica): F
ID: 1
Dados atuais:
ID: 1
Nome: Anderson
CPF: 01234567890
Idade: 33
Novo nome: Anderson Rech
Novo CPF: 01234567890
Nova idade: 34
Pessoa Fisica alterada com sucesso.
=====
```

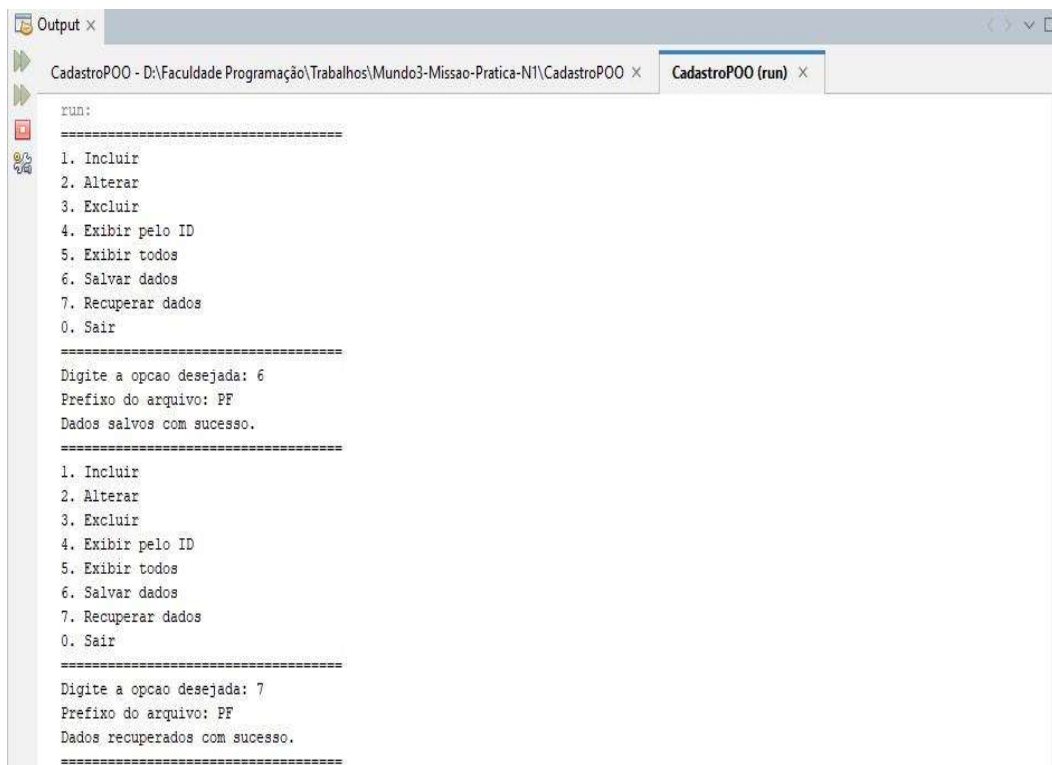
b. Exclusão de cadastro:

```
Output x
CadastroPOO - D:\Faculdade Programação\Trabalhos\Mundo3-Missao-Pratica-N1\CadastroPOO x CadastroPOO (run) x
1. Incluir
2. Alterar
3. Excluir
4. Exibir pelo ID
5. Exibir todos
6. Salvar dados
7. Recuperar dados
0. Sair
=====
Digite a opcao desejada: 2
Tipo (F para Fisica, J para Juridica): F
ID: 1
Dados atuais:
ID: 1
Nome: Anderson
CPF: 01234567890
Idade: 33
Novo nome: Anderson Rech
Novo CPF: 01234567890
Nova idade: 34
Pessoa Fisica alterada com sucesso.
=====
1. Incluir
2. Alterar
3. Excluir
4. Exibir pelo ID
5. Exibir todos
6. Salvar dados
7. Recuperar dados
0. Sair
=====
Digite a opcao desejada: 3
Tipo (F para Fisica, J para Juridica): F
ID: 1
Pessoa Fisica excluida com sucesso.
=====
```

c. Exibir por ID e Exibir Todos

```
Output x
CadastroPOO - D:\Faculdade Programação\Trabalhos\Mundo3-Missao-Pratica-N1\CadastroPOO x CadastroPOO (run) x
=====
Digite a opcao desejada: 1
Tipo (F para Fisica, J para Juridica): F
ID: 1
Nome: Anderson Rech
CPF: 01234567890
Idade: 34
Pessoa Fisica incluida com sucesso.
=====
1. Incluir
2. Alterar
3. Excluir
4. Exibir pelo ID
5. Exibir todos
6. Salvar dados
7. Recuperar dados
0. Sair
=====
Digite a opcao desejada: 4
Tipo (F para Fisica, J para Juridica): F
ID: 1
ID: 1
Nome: Anderson Rech
CPF: 01234567890
Idade: 34
=====
1. Incluir
2. Alterar
3. Excluir
4. Exibir pelo ID
5. Exibir todos
6. Salvar dados
7. Recuperar dados
0. Sair
=====
Digite a opcao desejada: 5
Tipo (F para Fisica, J para Juridica): F
ID: 1
Nome: Anderson Rech
CPF: 01234567890
Idade: 34
=====
```

d. Salvar e Recuperar Dados



```
run:
=====
1. Incluir
2. Alterar
3. Excluir
4. Exibir pelo ID
5. Exibir todos
6. Salvar dados
7. Recuperar dados
0. Sair
=====
Digite a opcao desejada: 6
Prefixo do arquivo: PF
Dados salvos com sucesso.
=====
1. Incluir
2. Alterar
3. Excluir
4. Exibir pelo ID
5. Exibir todos
6. Salvar dados
7. Recuperar dados
0. Sair
=====
Digite a opcao desejada: 7
Prefixo do arquivo: PF
Dados recuperados com sucesso.
=====
```

7. Análise e Conclusão:

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

Elementos estáticos (static) pertencem à classe e não a instâncias individuais e incluem métodos, atributos e blocos de inicialização que podem ser acessados sem criar um objeto da classe. O método main é “static” para permitir que a JVM (Java Virtual Machine) o chame diretamente e inicie a aplicação sem a necessidade de criar uma instância da classe. Isso define um ponto de entrada padrão para a execução do programa.

b. Para que serve a classe Scanner?

A classe Scanner em Java serve para ler e analisar entradas de diversas fontes de dados, como teclado e arquivos. Ela facilita a leitura de dados e a conversão para diferentes tipos primitivos, tornando a entrada de dados mais acessível e flexível em programas Java.

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

O uso de classes de repositório melhora a organização do código ao separar a lógica de persistência dos dados da lógica de negócios, encapsular o acesso a dados, facilitar testes, fornecer abstração, manter o código limpo e apoiar a escalabilidade. Resultando em um código mais modular, flexível e fácil de manter.