EAD - Polo Santa Luiza – Vitória – ES

DESENVOLVIMENTO FULL STACK

RELATÓRIO DE PRÁTICA
MUNDO 3 – NÍVEL 1 – MISSÃO PRÁTICA
RPG0014 - Iniciando o caminho pelo Java

CARLOS ALTOMARE CATÃO

Semestre Letivo: 2025/1 - 3º Período

Data: 2025/04

PROCEDIMENTO 1

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.

Objetivos do Procedimento 1: Criação das Entidades e Sistema de Persistência

O objetivo do primeiro procedimento é **iniciar a estruturação do projeto** Java com foco na programação orientada a objetos. Isso inclui a criação das **entidades básicas** (*Pessoa*, *PessoaFisica* e *PessoaJuridica*) e a implementação de um **sistema de persistência** utilizando arquivos binários. Esse sistema garantirá que os dados sejam **armazenados e recuperados mesmo após o encerramento da aplicação**, simulando um banco de dados simples.

Além disso, no desenvolvimento deste procedimento a utilização da POO permite gerar um código de forma organizada, promovendo a reutilização de código e facilitando a manutenção do sistema.

Códigos:

Os códigos relativos a esta prática se encontram em reservatório no GitHub e podem ser acessados pelo link: https://github.com/CarlosCatao/Mundo_3_Nivel_1_Missao_Pratica/tree/main/Procedimento_1/CadastroPOO.

Resultados:

Os resultados da execução dos códigos se encontram no arquivo PDF OUTPUT.pdf armazenado em repositório GitHub e pode ser acessado pelo link: https://github.com/CarlosCatao/Mundo_3_Nivel_1_Missao_Pratica/blob/main/Procedimento_1/OUTPUT.pdf.

Análise e Conclusão:

Quais as vantagens e desvantagens do uso de herança:

Entre as principais vantagens do uso da herança podemos citar a reutilização de código evitando a redundância e facilitando as manutenções futuras. Outro aspecto positivo reside na organização hierárquica do código.

Com relação às desvantagens podemos citar o fato do aumento da complexidade em caso de uma hierarquia muito profunda, bem como a alta dependência entre classes.

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

A interface *Serializable* do Java permite converter um dado objeto em uma sequência de bytes em formato binário, processo este que é chamado de serialização. Isso possibilita salvar o objeto em um arquivo, enviá-lo por uma rede ou armazená-lo em um banco de dados.

Essa interface permite manipular objetos de forma mais fácil, Sem essa interface, o Java não consegue serializar (salvar) o estado dos objetos.

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

A API *Stream* aplica conceitos funcionais como *map*, *filter*, *reduce* para processar coleções de forma declarativa e com menos código, utilizando expressões *lambda* e evitando laços explícitos.

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

O padrão utilizado é a serialização de objetos, seguindo o paradigma de gravação binária com as classes *ObjectOutputStream* e *ObjectInputStream*.