Campus: Polo Ingleses

Curso: Desenvolvimento Full Stack

Disciplina: Nível 1 - Iniciando o Caminho Pelo Java

Turma: 9001

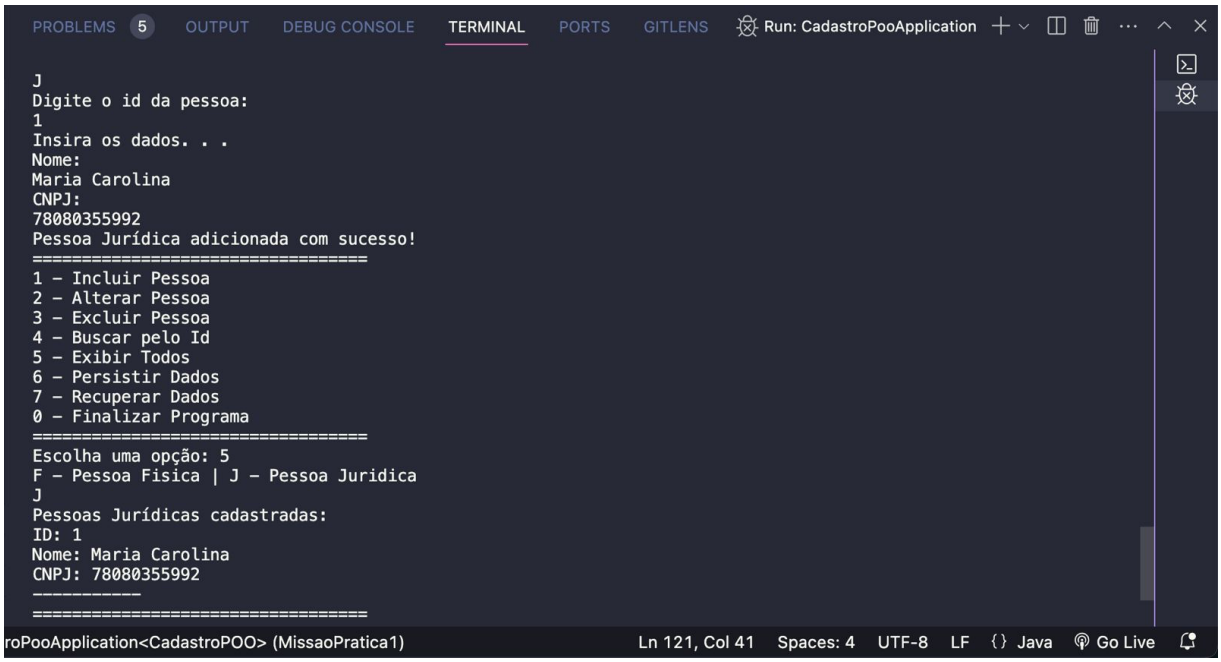
Semestre: 23.3

Aluna: Maria Carolina Knudsen Boabaid

1º Procedimento | Criação das Entidades e Sistema de Persistência

Objetivo da prática: Realizar uma carga inaugural de informações nas categorias 'Indivíduo' e 'Entidade Empresarial' durante a operação do aplicativo por intermédio do procedimento principal da classe primordial.

Resultados da execução dos códigos:



Análise e Conclusão:

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

A herança, na programação orientada a objetos, apresenta benefícios como a utilização recorrente de código, em que as classes filhas herdam características e procedimentos da classe mãe, minimizando redundâncias e simplificando a administração. Adicionalmente, as alterações realizadas na classe mãe são automaticamente propagadas para as filhas, facilitando a coesão e a uniformidade do sistema. Em última análise, a herança favorece o polimorfismo, possibilitando a abordagem uniforme de objetos provenientes de classes diversas.

Contudo, as desvantagens englobam o possível acoplamento substancial entre as classes mães e filhas, intensificando a sensibilidade do sistema a alterações e complicando a tarefa de manutenção. A herança também pode introduzir complicações nos testes unitários, uma vez que é imperativo considerar o comportamento integrado das classes mães e filhas.

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

A utilização da interface Serializable é essencial ao realizar a persistência em arquivos binários em Java, uma vez que desempenha um papel fundamental no processo de serialização de objetos. A serialização, que envolve a conversão de um objeto em uma sequência de bytes, possibilita o armazenamento desse objeto em arquivos binários ou a transmissão pela rede.

Quando um objeto de uma classe que implementa a interface Serializable é persistido em um arquivo binário, o Java encarrega-se de converter os dados do objeto em uma forma serializada. Essa representação serializada pode ser facilmente armazenada e, posteriormente, reconstruída conforme necessário.

Assim, ao efetuar a persistência em arquivos binários, a presença da interface Serializable garante que os objetos possam ser transformados em bytes e, posteriormente, restaurados à sua forma original quando exigido.

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

O paradigma funcional é uma abordagem de programação que se concentra na composição de funções puras. A API Stream do Java é uma biblioteca que fornece uma API declarativa para processar coleções de dados. Ela é baseada no paradigma funcional e utiliza operações de alto nível, como map, filter e reduce.

A utilização do paradigma funcional na API Stream oferece uma série de benefícios, incluindo:

- Concisão: O uso de operações de alto nível permite expressar operações complexas de forma legível e concisa, sem a necessidade de loops explícitos.
- Clareza: O paradigma funcional incentiva a imutabilidade e evita efeitos colaterais, resultando em código mais legível, modular e fácil de entender.
- Paralelismo: O uso de operações puras facilita a exploração eficiente de operações paralelas.
-

A API Stream com o paradigma funcional no Java oferece uma abordagem poderosa e expressiva para lidar com operações em coleções de dados. Ela oferece benefícios em termos de concisão, clareza e potencial de paralelismo.

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

O padrão de persistência de dados em Java depende do tipo de dados a serem persistidos.

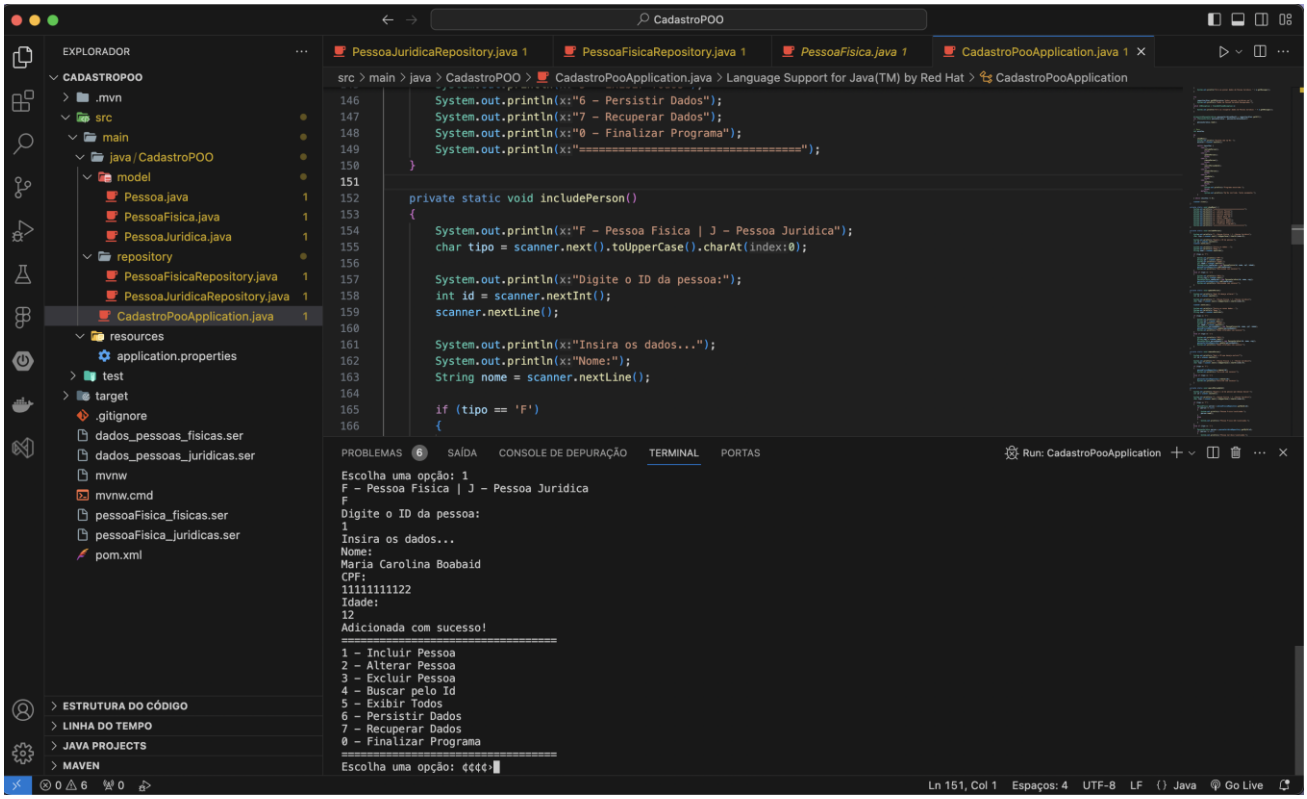
Para bancos de dados relacionais, o uso de frameworks ORM é comum. Esses frameworks fornecem uma camada de abstração entre objetos Java e tabelas de banco de dados, facilitando o desenvolvimento de aplicações que acessam dados relacionais.

Para arquivos locais, como arquivos de texto, binários ou JSON, o uso de bibliotecas de manipulação de arquivos ou serialização/desserialização manual também é comum. Bibliotecas como Jackson para JSON ou BufferedWriter/BufferedReader para arquivos de texto são exemplos de abordagens nesse contexto.

2º Procedimento | Criação do Cadastro em Modo Texto

Objetivo da prática: Gerar um CRUD (create, read, update e delete) das classes 'Pessoa Física' e 'Pessoa Jurídica' e manipulá-las através do console, gerando persistência de dados em arquivo binário.

Resultados da execução dos códigos:



Análise e Conclusão:

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

Em Java, componentes estáticos, como variáveis e métodos, encontram-se vinculados à classe, e não a instâncias específicas da classe em questão. As variáveis estáticas são compartilhadas globalmente entre todas as instâncias, enquanto os métodos estáticos pertencem à classe e não requerem uma instância particular para serem invocados.

O método main é frequentemente designado como estático, uma vez que desempenha o papel de ponto de entrada para um programa Java. Ao ser estático, pode ser diretamente chamado pela JVM, dispensando a necessidade de uma instância específica da classe que o contém. Essa característica estática permite que o método main seja acessado diretamente pela classe, tornando-se essencial para a execução do programa, pois oferece à JVM um ponto de partida sem a obrigatoriedade de criar instâncias da classe.

b) Para que serve a classe Scanner?

A classe Scanner em Java desempenha um papel fundamental ao simplificar a leitura de dados provenientes de diferentes fontes, como teclado, arquivos ou strings. Pertencente ao pacote

java.util, essa classe oferece métodos que facilitam a leitura de uma variedade de tipos de dados, incluindo inteiros, ponto flutuante e caracteres.

A principal função do Scanner é tornar mais fácil a captura de dados fornecidos pelo usuário ou presentes em fluxos de entrada. Ao criar uma instância do objeto Scanner, é possível utilizar seus métodos para ler e converter dados de maneira conveniente, otimizando a interação com o usuário e a manipulação de informações em arquivos.

c) 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, proporcionando uma separação clara entre a lógica de negócios e as operações de persistência de dados.