UNIVERSIDADE ESTÁCIO DE SÁ

CAMPUS POLO CENTRO - MARICÁ - RJ

ALUNO: LUIZ CARLOS MARINHO JUNIOR

MATRÍCULA: 2023.11.17557-1

CURSO: DESENVOLVIMENTO FULL STACK

SEMESTRE LETIVO: 2024.4

DISCIPLINA: NÍVEL 1: INICIANDO O CAMINHO PELO JAVA

TÍTULO: 2º PROCEDIMENTO | CRIAÇÃO DO CADASTRO EM MODO TEXTO

RIO DE JANEIRO

2024

LUIZ CARLOS MARINHO JUNIOR

2º PROCEDIMENTO | CRIAÇÃO DO CADASTRO EM MODO TEXTO

Trabalho prático para aprovação na disciplina de Nível 1: Iniciando o Caminho Pelo Java.

Tutora: Prof. Maria B.

RIO DE JANEIRO

2024

SUMÁRIO

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, terá implementado um sistema cadastral em Java, utilizando os recursos da programação orientada a objetos e a persistência em arquivos binários.

2. CÓDIGOS SOLICITADOS

2.1. APRESENTAÇÃO

Todos os códigos que serão aqui apresentados, estão disponíveis de forma completa no repositório do Github. O desenvolvimento do código está separado por branch, e no caso desse 2ª Procedimento, o código se encontra na branch SegundoProcedimento.

O **link para o repositório do Github** desse 2ª Procedimento se encontra logo abaixo:
https://github.com/luizmarinhojr/cadastro-poo/tree/SegundoProcedimento

2.2. PRINCIPAIS INSTRUÇÕES DO 2ª PROCEDIMENTO

1. Alterar o método main da classe principal do projeto, para implementação do cadastro em modo texto:

a. Apresentar as opções do programa para o usuário, sendo 1 para incluir, 2 para alterar, 3 para excluir, 4 para exibir pelo id, 5 para exibir todos, 6 para salvar dados, 7 para recuperar dados e 0 para finalizar a execução.

Implementando o procedimento:

```
public class CadastroPOO {
```

```
/**
 * @param args the command line arguments
 */

static final Scanner LEITOR = new Scanner(System.in);

static final PessoaFisicaRepo FISICAREPO = new PessoaFisicaRepo();
static final PessoaJuridicaRepo JURIDICAREPO = new PessoaJuridicaRepo();

public static void main(String[] args) {

   Integer escolhaMenu = -1;

   String opcoes = """

                MENU PRINCIPAL
   ======================================
   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
   ======================================""";
```

```
do {
    boolean escolhaValida = false;
    System.out.println(opcoes);
    System.out.print("Digite o número da opção -> ");
    do {
        try {
            escolhaMenu = Integer.parseInt(LEITOR.nextLine());
            escolhaValida = true;
        } catch(NumberFormatException ex) {
            System.out.println("\nEntrada inválida, tente novamente");
        }
    } while (!escolhaValida);


    switch (escolhaMenu) {
        case 1 -> incluirPessoa();

        case 2 -> alterarPessoa();

        case 3 -> excluirPessoa();

        case 4 -> buscarPeloId();

        case 5 -> obterTodos();

        case 6 -> persistirDados();

        case 7 -> recuperarDados();
```

```
            case 0 -> System.out.println("\nFinalizando o programa...");

            default -> System.out.println("\nOpção inválida! Tente novamente.");

        }

    } while (escolhaMenu != 0);
}
```

b. Selecionada a opção incluir, escolher o tipo (Física ou Jurídica), receber os dados a partir do teclado e adicionar no repositório correto.

Implementando o procedimento:

```
private static void incluirPessoa() {
    switch(selecionarTipoPessoa().toUpperCase()) {
        case "F" -> incluirPessoaFisica();

        case "J" -> incluirPessoaJuridica();
    }
    digiteParaVoltarAoMenu();

}
```

```
private static String selecionarTipoPessoa() {
    String escolhaMenu = "";

    do {
        System.out.println("\nF - Pessoa física    |    J - Pessoa Jurídica");
        System.out.print("Digite a letra da opção desejada -> ");
```

```
        escolhaMenu = LEITOR.nextLine();


    } while(!(escolhaMenu.equalsIgnoreCase("F") ||
        escolhaMenu.equalsIgnoreCase("J")));


    return escolhaMenu;
}
```

```
private static void incluirPessoaFisica() {
    Integer idPessoa = digitarIdPessoa("física");


    System.out.println("\nInsira os dados...");
    System.out.print("\nNome -> ");
    String nomePessoa = LEITOR.nextLine();


    System.out.print("\nCPF (Somente números) -> ");
    String cpfPessoa = LEITOR.nextLine();


    Integer idadePessoa = digitarIdadePessoa();


    FISICAREPO.inserir(new PessoaFisica(idPessoa, nomePessoa, cpfPessoa,
        idadePessoa));
}
```

```
private static void incluirPessoaJuridica() {
    Integer idPessoa = digitarIdPessoa("jurídica");



    System.out.println("\nInsira os dados...");
```

```
    System.out.print("\nNome -> ");
    String nomePessoa = LEITOR.nextLine();


    System.out.print("\nCNPJ (Somente números) -> ");
    String cnpjPessoa = LEITOR.nextLine();


    JURIDICAREPO.inserir(new PessoaJuridica(idPessoa, nomePessoa, cnpjPessoa));
}
```

c. Selecionada a opção alterar, escolher o tipo (Física ou Jurídica), receber o id a partir do teclado, apresentar os dados atuais, solicitar os novos dados e alterar no repositório correto.

Implementando o procedimento:

```
private static void alterarPessoa() {
    switch(selecionarTipoPessoa().toUpperCase()) {
        case "F" -> alterarPessoaFisica();


        case "J" -> alterarPessoaJuridica();
    }
    digiteParaVoltarAoMenu();
}
```

```
private static String selecionarTipoPessoa() {
    String escolhaMenu = "";
```

```
    do {
        System.out.println("\nF - Pessoa física    |   J - Pessoa Jurídica");
        System.out.print("Digite a letra da opção desejada -> ");
        escolhaMenu = LEITOR.nextLine();

    } while(!(escolhaMenu.equalsIgnoreCase("F") ||
       escolhaMenu.equalsIgnoreCase("J")));

     return escolhaMenu;
}
```

```
private static void alterarPessoaFisica() {
    String escolhaMenu = "";
    Optional<PessoaFisica> pessoaFisica = buscarPessoaFisicaPeloId();

    if (pessoaFisica.isPresent()) {
        do {
            System.out.println("Deseja alterar essa pessoa física? (S) Sim | (N) Não ");
            System.out.print("\nDigite a letra da opção desejada -> ");
            escolhaMenu = LEITOR.nextLine();
        } while(!(escolhaMenu.equalsIgnoreCase("S") ||
                escolhaMenu.equalsIgnoreCase("N")));

        switch (escolhaMenu.toUpperCase()) {
            case "S" -> {
                System.out.println("\nInsira os novos dados...");
                Integer idPessoa = digitarIdPessoa("física");
                System.out.print("\nNome -> ");
                String nomePessoa = LEITOR.nextLine();
```

```
            System.out.print("\nCPF (Somente números) -> ");
            String cpfPessoa = LEITOR.nextLine();
            Integer idadePessoa = digitarIdadePessoa();
            FISICAREPO.alterar(pessoaFisica.get(), new PessoaFisica(idPessoa,
              nomePessoa, cpfPessoa, idadePessoa));
        }

        case "N" -> System.out.println("\nPessoa física não alterada. Voltando para
      o menu...");
      }
   }
}
```

```
private static void alterarPessoaJuridica() {
   String escolhaMenu = "";
   Optional<PessoaJuridica> pessoaJuridica = buscarPessoaJuridicaPeloId();

   if (pessoaJuridica.isPresent()) {
      do {
         System.out.println("Deseja alterar essa pessoa jurídica? (S) Sim | (N) Não
               ");
         System.out.print("Digite a letra da opção desejada -> ");
         escolhaMenu = LEITOR.nextLine();
      } while(!(escolhaMenu.equalsIgnoreCase("S") ||
            escolhaMenu.equalsIgnoreCase("N")));

      switch (escolhaMenu.toUpperCase()) {
         case "S" -> {
            System.out.println("\nInsira os novos dados...");
            Integer idPessoa = digitarIdPessoa("jurídica");
```

```
            System.out.print("\nNome -> ");
            String nomePessoa = LEITOR.nextLine();
            System.out.print("\nCNPJ (Somente números) -> ");
            String cnpjPessoa = LEITOR.nextLine();
            JURIDICAREPO.alterar(pessoaJuridica.get(), new PessoaJuridica(idPessoa,
              nomePessoa, cnpjPessoa));
        }

        case "N" -> System.out.println("\nPessoa física não alterada. Voltando para
            o menu...");
      }
    }
}
```

d. Selecionada a opção excluir, escolher o tipo (Física ou Jurídica), receber o id a partir do teclado e remover do repositório correto.

Implementando o procedimento:

```
private static void excluirPessoa() {
    switch(selecionarTipoPessoa().toUpperCase()) {
        case "F" -> excluirPessoaFisica();

        case "J" -> excluirPessoaJuridica();
    }
    digiteParaVoltarAoMenu();
}
```

```
private static String selecionarTipoPessoa() {
    String escolhaMenu = "";
```

```
    do {
        System.out.println("\nF - Pessoa física    |   J - Pessoa Jurídica");
        System.out.print("Digite a letra da opção desejada -> ");
        escolhaMenu = LEITOR.nextLine();

    } while(!(escolhaMenu.equalsIgnoreCase("F") ||
       escolhaMenu.equalsIgnoreCase("J")));

     return escolhaMenu;
}
```

```
private static void excluirPessoaFisica() {
    String escolhaMenu = "";
    Optional<PessoaFisica> pessoaFisica = buscarPessoaFisicaPeloId();
    if (pessoaFisica.isPresent()) {
        do {
            System.out.println("Deseja excluir " + pessoaFisica.get().getNome() + " do
                    registro? (S) Sim | (N) Não ");
            System.out.print("Digite a letra da opção desejada -> ");
            escolhaMenu = LEITOR.nextLine();
        } while(!(escolhaMenu.equalsIgnoreCase("S") ||
                escolhaMenu.equalsIgnoreCase("N")));

        switch (escolhaMenu.toUpperCase()) {
            case "S" -> FISICAREPO.excluir(pessoaFisica.get());

            case "N" -> System.out.println("\nPessoa física não excluída. Voltando para
                            o menu...");
        }
    } else {
```

```
            System.out.println("\nNão existe pessoa física cadastrada com esse ID");
        }
    }
```

```
    private static void excluirPessoaJuridica() {
        String escolhaMenu = "";
        Optional<PessoaJuridica> pessoaJuridica = buscarPessoaJuridicaPeloId();
        if (pessoaJuridica.isPresent()) {
            do {
                System.out.println("Deseja excluir " + pessoaJuridica.get().getNome() + " do
                    registro? (S) Sim | (N) Não ");
                System.out.print("Digite a letra da opção desejada -> ");
                escolhaMenu = LEITOR.nextLine();
            } while(!(escolhaMenu.equalsIgnoreCase("S") ||
escolhaMenu.equalsIgnoreCase("N")));

            switch (escolhaMenu.toUpperCase()) {
                case "S" -> JURIDICAREPO.excluir(pessoaJuridica.get());

                case "N" -> System.out.println("\nPessoa jurídica não excluída. Voltando
                            para o menu...");
            }
        } else {
            System.out.println("\nNão existe pessoa jurídica cadastrada com esse ID");
        }
    }
```

e. Selecionada a opção obter, escolher o tipo (Física ou Jurídica), receber o id a partir do teclado e apresentar os dados atuais para a entidade.

Implementando o procedimento:

```
private static void buscarPeloId() {
    switch(selecionarTipoPessoa().toUpperCase()) {
        case "F" -> buscarPessoaFisicaPeloId();

        case "J" -> buscarPessoaJuridicaPeloId();
    }
    digiteParaVoltarAoMenu();
}
```

```
private static String selecionarTipoPessoa() {
    String escolhaMenu = "";

    do {
        System.out.println("\nF - Pessoa física   |   J - Pessoa Jurídica");
        System.out.print("Digite a letra da opção desejada -> ");
        escolhaMenu = LEITOR.nextLine();

    } while(!(escolhaMenu.equalsIgnoreCase("F") ||
        escolhaMenu.equalsIgnoreCase("J")));

     return escolhaMenu;
}
```

```
private static Optional<PessoaFisica> buscarPessoaFisicaPeloId() {
    Integer idPessoa = digitarIdPessoa("física");
    Optional<PessoaFisica> pessoaFisica = FISICAREPO.obter(idPessoa);
    if (pessoaFisica.isPresent()) {
```

```
            System.out.println("\n**** Pessoa física ****\n" +
                    pessoaFisica.get().exibir());

            return pessoaFisica;

        }

        System.out.println("\nNão existe pessoa física cadastrada com esse ID");

        return pessoaFisica;

    }
```

```
    private static Optional<PessoaJuridica> buscarPessoaJuridicaPeloId() {

        Integer idPessoa = digitarIdPessoa("jurídica");

        Optional<PessoaJuridica> pessoaJuridica = JURIDICAREPO.obter(idPessoa);

        if (pessoaJuridica.isPresent()) {

            System.out.println("\n**** Pessoa jurídica ****\n" +
                    pessoaJuridica.get().exibir());

            return pessoaJuridica;

        }

        System.out.println("\nNão existe pessoa física cadastrada com esse ID");

        return pessoaJuridica;

    }
```

f. Selecionada a opção obterTodos, escolher o tipo (Física ou Jurídica) e apresentar os dados de todas as entidades do repositório correto.

Implementando o procedimento:

```
    private static void obterTodos() {

        switch(selecionarTipoPessoa().toUpperCase()) {

            case "F" -> System.out.println("\n**** Todas as pessoas físicas ****\n" +
                            FISICAREPO.obterTodos());
```

```
        case "J" -> System.out.println("\n**** Todas as pessoas jurídicas ****\n" +
                        JURIDICAREPO.obterTodos());
    }
    digiteParaVoltarAoMenu();
}
```

g. Selecionada a opção salvar, solicitar o prefixo dos arquivos e persistir os dados nos arquivos [prefixo].fisica.bin e [prefixo].juridica.bin.

Implementando o procedimento:

```
private static void persistirDados() {
    switch(selecionarTipoPessoa().toUpperCase()) {
        case "F" -> persistirPessoaFisica();

        case "J" -> persistirPessoaJuridica();
    }
    digiteParaVoltarAoMenu();
}
```

```
private static String selecionarTipoPessoa() {
    String escolhaMenu = "";

    do {
        System.out.println("\nF - Pessoa física   |   J - Pessoa Jurídica");
        System.out.print("Digite a letra da opção desejada -> ");
        escolhaMenu = LEITOR.nextLine();

    } while(!(escolhaMenu.equalsIgnoreCase("F") ||
       escolhaMenu.equalsIgnoreCase("J")));
```

```
        return escolhaMenu;
    }
```

```
    private static void persistirPessoaFisica() {
        System.out.print("\nDigite o prefixo do nome do arquivo -> ");
        String prefixo = LEITOR.nextLine();
        try {
            System.out.println();
            String nomeArquivo = FISICAREPO.persistir(prefixo);
            System.out.println("\nO nome do arquivo persistido é -> " + nomeArquivo);
        } catch (Exception ex) {
            System.out.println("\nHouve um erro ao armazenar os dados.\n" + "Erro: " +
                    ex.getMessage());
        }
    }
```

```
    private static void persistirPessoaJuridica() {
        System.out.print("\nDigite o prefixo do nome do arquivo -> ");
        String prefixo = LEITOR.nextLine();
        try {
            System.out.println();
            String nomeArquivo = JURIDICAREPO.persistir(prefixo);
            System.out.println("\nO nome do arquivo persistido é -> " + nomeArquivo);
        } catch (Exception ex) {
            System.out.println("\nHouve um erro ao armazenar os dados.\n" + "Erro: " +
                    ex.getMessage());
        }
    }
```

h. Selecionada a opção recuperar, solicitar o prefixo dos arquivos e obter os dados a partir dos arquivos [prefixo].fisica.bin e [prefixo].juridica.bin.

Implementando o procedimento:

```
private static void recuperarDados() {
    switch(selecionarTipoPessoa().toUpperCase()) {
        case "F" -> recuperarPessoaFisica();

        case "J" -> recuperarPessoaJuridica();
    }
    digiteParaVoltarAoMenu();
}
```

```
private static String selecionarTipoPessoa() {
    String escolhaMenu = "";

    do {
        System.out.println("\nF - Pessoa física   |   J - Pessoa Jurídica");
        System.out.print("Digite a letra da opção desejada -> ");
        escolhaMenu = LEITOR.nextLine();

    } while(!(escolhaMenu.equalsIgnoreCase("F") ||
        escolhaMenu.equalsIgnoreCase("J")));

     return escolhaMenu;
}
```

```
private static void recuperarPessoaFisica() {
    System.out.print("\nDigite o prefixo do nome do arquivo -> ");
```

```
        String prefixo = LEITOR.nextLine();
        try {
            System.out.println();
            FISICAREPO.recuperar(prefixo);
        } catch(Exception ex) {
            System.out.println("\nHouve um erro ao armazenar os dados.\n" + "Erro: " +
                    ex.getMessage());
        }
    }

    private static void recuperarPessoaJuridica() {
        System.out.print("\nDigite o prefixo do nome do arquivo -> ");
        String prefixo = LEITOR.nextLine();
        try {
            System.out.println();
            JURIDICAREPO.recuperar(prefixo);
        } catch(Exception ex) {
            System.out.println("\nHouve um erro ao armazenar os dados.\n" + "Erro: " +
ex.getMessage());
        }
    }
}
```

I. Nas opções salvar e recuperar devem ser tratadas as exceções.

Implementando o procedimento:

```
        try {
            System.out.println();
            FISICAREPO.recuperar(prefixo);
```

```
    } catch(Exception ex) {
        System.out.println("\nHouve um erro ao armazenar os dados.\n" + "Erro: " +
        ex.getMessage());
    }
```

```
    try {
        System.out.println();
        JURIDICAREPO.recuperar(prefixo);
    } catch(Exception ex) {
        System.out.println("\nHouve um erro ao armazenar os dados.\n" + "Erro: " +
        ex.getMessage());
    }
```

```
    try {
        System.out.println();
        String nomeArquivo = JURIDICAREPO.persistir(prefixo);
        System.out.println("\nO nome do arquivo persistido é -> " + nomeArquivo);
    } catch(Exception ex) {
        System.out.println("\nHouve um erro ao armazenar os dados.\n" + "Erro: " +
        ex.getMessage());
    }
```

```
    try {
        System.out.println();
        String nomeArquivo = FISICAREPO.persistir(prefixo);
        System.out.println("\nO nome do arquivo persistido é -> " + nomeArquivo);
    } catch(Exception ex) {
        System.out.println("\nHouve um erro ao armazenar os dados.\n" + "Erro: " +
        ex.getMessage());
```

```
    }
```

m. Selecionada a opção sair, finalizar a execução do sistema.

Implementando o procedimento:

```
do {
        boolean escolhaValida = false;
        System.out.println(opcoes);
        System.out.print("Digite o número da opção -> ");
        do {
           try {
              escolhaMenu = Integer.parseInt(LEITOR.nextLine());
              escolhaValida = true;
           } catch(NumberFormatException ex) {
              System.out.println("\nEntrada inválida, tente novamente");
           }
        } while (!escolhaValida);


        switch (escolhaMenu) {
           case 1 -> incluirPessoa();


           case 2 -> alterarPessoa();


           case 3 -> excluirPessoa();


           case 4 -> buscarPeloId();
```

```
            case 5 -> obterTodos();

            case 6 -> persistirDados();

            case 7 -> recuperarDados();

            case 0 -> System.out.println("\nFinalizando o programa...");

            default -> System.out.println("\nOpção inválida! Tente novamente.");

        }

    } while (escolhaMenu != 0);
```

3. RESULTADO DA EXECUÇÃO DOS CÓDIGOS

Link para o vídeo -> https://youtu.be/3Xbii-ZHBXQ

Abaixo, segue um breve resumo do resultado do código:

```
                    MENU PRINCIPAL

=======================================

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 o número da opção -> 1


F - Pessoa física   |   J - Pessoa Jurídica

Digite a letra da opção desejada -> f


Digite o ID da pessoa física -> 1


Insira os dados...
```

```
Nome -> Luiz

CPF (Somente números) -> 11111111111

Idade -> 25

**** Pessoa Física inserida com sucesso ****

Digite 1 para voltar ao Menu Principal ->
```

4. ANÁLISE E CONCLUSÃO

4.1. O QUE SÃO ELEMENTOS ESTÁTICOS E QUAL O MOTIVO PARA O MÉTODO MAIN ADOTAR ESSE MODIFICADOR?

Em Java, elementos estáticos (variáveis ou métodos) pertencem à classe e não a instâncias específicas de objetos. Isso significa que eles existem independentemente de qualquer objeto ser criado.

O método main é o ponto de partida da execução de um programa Java, então para que a JVM possa encontrar e executar o método main sem a necessidade de criar uma instância da classe, ele é declarado como estático.

O método main não depende de nenhum objeto específico, apenas dos argumentos passados pela linha de comando.

Graças a estaticidade do método main, a JVM é capaz de executar o programa diretamente, sem a necessidade de instanciar uma classe.

4.2. PARA QUE SERVE A CLASSE SCANNER?

A classe Scanner em Java é utilizada para ler e processar entradas de dados de diversas fontes, como teclado, arquivos, strings ou streams. Ela está localizada no pacote java.util e oferece uma maneira conveniente de analisar e dividir a entrada em tokens utilizando delimitadores, que por padrão são espaços em branco.

- **Funcionalidades principais:**
  1. Leitura de Dados do Teclado: A classe é frequentemente usada para receber entrada do usuário pelo console.
  2. Suporte a Diversos Tipos de Dados: O Scanner pode processar diferentes tipos de dados, como:
     a. Inteiros (nextInt())
     b. Números de ponto flutuante (nextDouble())
     c. Strings (nextLine() ou next())
  3. Leitura de Arquivos e Strings: Pode ser usado para ler arquivos de texto e processar strings diretamente.
  4. Personalização de Delimitadores: É possível alterar o delimitador padrão para dividir a entrada em tokens de acordo com caracteres específicos, como vírgulas ou pontos.

4.3. COMO O USO DE CLASSES DE REPOSITÓRIO IMPACTOU NA ORGANIZAÇÃO DO CÓDIGO?

O uso de classes de repositório são extremamente importantes para manter a organização do código de um projeto. Essas classes são responsáveis por centralizar a lógica de acesso e manipulação de dados, promovendo uma melhor organização, separação de responsabilidades e manutenibilidade.

REFERÊNCIAS

ORACLE, Java™ Platform, Standard Edition 8 API Specification. Disponível em: <https://docs.oracle.com/javase/8/docs/api/>. Acesso em 09 de dezembro de 2024.

GEEKS FOR GEEKS, Java main() Method – public static void main(String[] args). Disponível em: <https://www.geeksforgeeks.org/java-main-method-public-static-void-main-string-args/>. Acesso em 09 de dezembro de 2024.

DIGITAL OCEAN, Scanner Class in Java. Disponível em: <https://www.digitalocean.com/community/tutorials/scanner-class-in-java>. Acesso em 09 de dezembro de 2024.

JAVA DESIGN PATTERNS, Repository Pattern in Java: Simplifying Data Access with Abstracted Persistence. Disponível em: <https://java-design-patterns.com/patterns/repository/>. Acesso em 09 de dezembro de 2024.