

Campus: 1197 - POLO CENTRO - SÃO LOURENÇO DA MATA - PE

Curso: Desenvolvimento Full Stack - Graduação Tecnóloga Disciplina: RPG0014 - Iniciando o caminho pelo Java

Turma: 9001 Semestre: 2024.1

Matrícula: 2023.01.53256-6 Aluno: Gilvan Pereira de Oliveira Repositório GitHub: GilvanPOliveira/CadastroPoo (github.com)

Relatório discente de acompanhamento

1º Procedimento - Criação das Entidades e Sistema de Persistência

Objetivo da prática:

Nesta etapa o objetivo foi criar um sistema de persistência, armazenar e recuperar, informações de entidades para um cadastro de pessoas físicas e jurídicas em Java, utilizando POO, herança, polimorfismo e manipulação de arquivos. O sistema permitirá operações de inserção, alteração, exclusão, recuperação e obtenção das entidades armazenadas, proporcionando uma maneira eficiente e segura de gerenciar e recuperar informações.

Prática:

- Criação das Entidades: Pessoa, PessoaFisica, PessoaJuridica;
- Criação dos gerenciadores: PessoaFisicaRepo, PessoaJuridicaRepo;
- Adição dos métodos: inserir, alterar, excluir, obter e obterTodos, além de pesistir e recuperar aos gerenciadores para armazenagem dos dados no disco;
- Alterar a classe principal (main) para testar os repositórios.
- Executar e verificar as funcionalidades implementadas e os arquivos gerados.

Códigos e resultados obtidos:

Resultado da execução dos códigos:

```
run-single:
Dados de Pessoas Fisica Armazenados.
Dados de Pessoas Fisica Recuperados.
ID: 1, Nome: Ana, CPF: 111.111.111-11, Idade: 25
ID: 2, Nome: Carlos, CPF: 222.222.222-22, Idade: 52
Dados de Pessoas Juridica Armazenados.
Dados de Pessoas Juridica Recuperados.
ID: 3, Nome: XPTO Sales, CNPJ: 33.333.333/3333-33
ID: 4, Nome: XPTO Solutions, CNPJ: 44.444.444/4444-44
BUILD SUCCESSFUL (total time: 0 seconds)
```

• Códigos:

```
Pessoa.java ×
package model;
3 - /**
    *
4
    * @author gilvan
6
7
8 - import java.io.Serializable;
public class Pessoa implements Serializable {
      private int id;
11
12
      private String nome;
13
14 🖃
      public Pessoa() {
15
       1
16
17 🖃
      public Pessoa(int id, String nome) {
18
       this.id = id;
19
          this.nome = nome;
20
       }
21
22 🖃
      public int getId() {
23
       return id;
24
25
26 =
      public void setId(int id) {
27
       this.id = id;
28
29
30 🖃
      public String getNome() {
31
       return nome;
32
33
34 -
      public void setNome(String nome) {
35
       this.nome = nome;
36
37
public void exibir() {
       System.out.println("ID: " + id + ", Nome: " + nome);
39
40
41
```

```
Pessoafisica.java [-/M] ×
package model:
3 日 /**
    * #author gilven
B import java.io.Serializable;
10 public class PessoaFisica extends Pessoa implements Serializable (
       private String opf;
11
12
       private int idade:
13
14 日
       public PessoaFisica() (
15
16
17 📮
       public PessoaFisica(int id, String nome, String opf, int idade) (
        super(id, nome);
this.cpf = cpf;
10
          this.idade = idade:
23
       public String getCpf() (
24
         return cpf;
25
27 E
       public void setCpf(String cpf) (
       this.cpf = cpf;
29
29
31
       public int getIdade() (
32
          return idade;
33
34
35 F
       public void setIdade(int idade) (
36
          this.idade = idade;
37
38
29
       @Override
(a)
       public word exibir() (
          System.out.println("ID: " + getId() + ", Nome: " + getNome() + ", CFF: " + opf + ", Idade: " + idade);
41
42
🔊 PessoaJuridica.java 🗴
Source History 🖟 👼 🕶 🔻 🔻 🖓 🖧 🐉 🔒 🔯 💇 🔵 🗆 🖺 🕌
    package model;
3 - /**
   * @author gilvan
 5
 6
 10
     public class PessoaJuridica extends Pessoa implements Serializable {
11
        private String cnpj;
12
13 🖃
        public PessoaJuridica() {
14
        }
15
16 =
         public PessoaJuridica(int id, String nome, String cnpj) {
17
            super(id, nome);
18
             this.cnpj = cnpj;
19
        }
20
21 =
         public String getCnpj() {
22
         return cnpj;
23
         }
24
25 📮
         public void setCnpj(String cnpj) {
26
          this.cnpj = cnpj;
27
28
29
         @Override
o -
         public void exibir() {
31
         System.out.println("ID: " + getId() + ", Nome: " + getNome() + ", CNPJ: " + cnpj);
32
33
     1
```

```
Source History | [6] | [7] | [7] | [7] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | [8] | 
          package model;
  3 - /**
              * @author gilvan
  5
  6
 8 [ import java.io.*;
         import java.util.ArrayList;
 10
11
           public class PessoaFisicaRepo {
 12
                   private ArrayList<PessoaFisica> pessoas = new ArrayList<>();
13
14 -
                    public void inserir(PessoaFisica pessoa) {
                           pessoas.add(pessoa);
15
16
17
 18 🖃
                     public void alterar(PessoaFisica pessoa) {
19
                             for (int i = 0; i < pessoas.size(); i++) {
20
                                       if (pessoas.get(i).getId() == pessoa.getId()) {
21
                                                pessoas.set(i, pessoa);
22
                                                 break:
23
                                        }
24
25
26
27 🖃
                      public void excluir(int id) {
28 =
                              for (int i = 0; i < pessoas.size(); i++) {</pre>
 29 =
                                        if (pessoas.get(i).getId() == id) {
30
                                                pessoas.remove(i):
31
                                                 break;
32
                                        }
33
 34
35
                     public PessoaFisica obter(int id) {
36
37
                               for (PessoaFisica pessoa : pessoas) {
38
                                      if (pessoa.getId() == id) {
 39
                                                return pessoa;
 40
 41
 42
                              return null;
 43
44
45 🖃
                     public ArrayList<PessoaFisica> obterTodos() {
 46
                              return pessoas;
47
 48
 49 📮
                     public void persistir(String nomeArquivo) throws IOException {
 50
                               FileOutputStream fileOut = new FileOutputStream(nomeArquivo);
 51 🚊
                                try (ObjectOutputStream objectOut = new ObjectOutputStream(fileOut)) {
 52
                                       objectOut.writeObject(pessoas);
53
54
55
56 =
                      public void recuperar(String nomeArquivo) throws IOException, ClassNotFoundException {
 57
                               FileInputStream fileIn = new FileInputStream(nomeArquivo);
58 😑
                                try (ObjectInputStream objectIn = new ObjectInputStream(fileIn)) {
 59
                                       pessoas = (ArrayList<PessoaFisica>) objectIn.readObject();
 60
61
 62
63
```

```
package model;
3 🖵 /**
      * @author gilvan
5
6
8 - import java.io.*;
   import java.util.ArrayList;
10
11
    public class PessoaJuridicaRepo {
12
       private ArrayList<PessoaJuridica> pessoas = new ArrayList<>();
13
14 -
        public void inserir(PessoaJuridica pessoa) {
15
           pessoas.add(pessoa);
16
17
18 📮
         public void alterar(PessoaJuridica pessoa) {
19
            for (int i = 0; i < pessoas.size(); i++) {</pre>
20 🗀
                if (pessoas.get(i).getId() == pessoa.getId()) {
21
                    pessoas.set(i, pessoa);
22
                    break;
23
                1
24
25
26
27 =
         public void excluir(int id) {
28 🖨
             for (int i = 0; i < pessoas.size(); i++) {</pre>
29
                if (pessoas.get(i).getId() == id) {
30
                    pessoas.remove(i);
31
                    break;
32
                1
33
34
35
36 📮
         public PessoaJuridica obter(int id) {
37
             for (PessoaJuridica pessoa : pessoas) {
38 =
                if (pessoa.getId() == id) {
39
                    return pessoa;
40
41
            1
42
             return null;
43
44
45 =
        public ArrayList<PessoaJuridica> obterTodos() {
46
           return pessoas;
47
48
49 🖃
         public void persistir(String nomeArquivo) throws IOException {
            FileOutputStream fileOut = new FileOutputStream(nomeArquivo);
50
51 🚊
             try (ObjectOutputStream objectOut = new ObjectOutputStream(fileOut)) {
52
                objectOut.writeObject(pessoas);
53
             1
54
55
         public void recuperar(String nomeArquivo) throws IOException, ClassNotFoundException {
56 🖃
            FileInputStream fileIn = new FileInputStream(nomeArquivo);
57
58 😑
             try (ObjectInputStream objectIn = new ObjectInputStream(fileIn)) {
59
                pessoas = (ArrayList<PessoaJuridica>) objectIn.readObject();
60
61
62
63
    }
```

```
Main_01.java ×
Source History 🔯 🕟 - 💹 - 🔍 🛼 🍣 🕾 📮 🗳 🍖 🖭 💇 🔞 🗆 🏄
      package model;
 3 日 /**
   * @author gilvan
 8 E import java.io.*;
     public class Main_01 (
11
12 E
         public static void main(String[] args) {
13
              try (
                  //Instanciando repol
14
15
                  PessoaFisicaRepo repol = new PessoaFisicaRepo();
16
17
                  //Adicionando duas pessoas fisicas
                  PessoaFisica pessoaFisical = new PessoaFisica(1, "Ana", "111.111.111-11", 25);
18
                  PessoaFisica pessoaFisica2 = new PessoaFisica(2, "Carlos", "222.222.222-22", 52);
19
20
                  repol.inserir(pessoaFisical);
21
                  repol.inserir (pessoaFisica2);
22
23
                  //Persistindo os dados em repol
                  repol.persistir("pessoasFisicas.dat");
24
                  System.out.println("Dados de Pessoas Fisica Armazenados.");
25
26
27
                  //Instanciando repo2
                  PessoaFisicaRepo repo2 = new PessoaFisicaRepo():
28
29
30
                  //Recuperando os dados em repol
31
                  repo2.recuperar("pessoasFisicas.dat");
32
33
                  //Exibindo os dados recuperados das pessoas fisicas
                  System.out.println("Dados de Pessoas Fisica Recuperados.");
34
35
                  for (PessoaFisica pessoa : repo2.obterTodos()) {
36
                      pessoa.exibir();
37
38
39
                  //Instanciando repo3
40
                  PessoaJuridicaRepo repo3 = new PessoaJuridicaRepo();
41
42
                  //Adicionando duas pessoas juridicas
43
                  PessoaJuridica pessoaJuridical = new PessoaJuridica(3, "XPTO Sales", "33.333.333/3333-33");
                  PessoaJuridica pessoaJuridica2 = new PessoaJuridica(4, "XPTO Solutions", "44.444.444/4444-44");
44
45
                  repo3.inserir(pessoaJuridical);
46
                  repo3.inserir(pessoaJuridica2);
47
48
                  //Persistindo os dados em repo3
49
                  repo3.persistir("pessoasJuridicas.dat");
50
                  System.out.println("Dados de Pessoas Juridica Armazenados.");
51
52
                  //Instanciando repo4
53
                  PessoaJuridicaRepo repof = new PessoaJuridicaRepo();
54
55
                  //Recuperando os dados em repo4
56
                  repo4.recuperar ("pessoas Juridicas.dat");
57
58
                  //Exibindo os dados recuperados das pessoas juridicas
                  System.out.println("Dados de Pessoas Juridica Recuperados.");
59
60
                  for (PessoaJuridica pessoa : repo4.obterTodos()) (
61
                      pessoa.exibir();
62
63
              ) catch (IOException | ClassNotFoundException e) {
64
65
```

Conclusão:

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

Reutilização de código, para assim evitar redundâncias; Polimorfismo, para melhor manipulação de objetos de diferentes tipos;

Organização da estrutura de forma hierárquica.

Desvantagens:

Hierarquia complexa, dependendo da complexidade do código pode dificultar o entendimento;

Acoplamento entre as classes, tornando mais difícil modificar e até compreender;

Alterações na classe base podem afetar todas as classes derivadas, causando efeitos colaterais indesejados.

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

Para que permita que objetos Java sejam transformados em sequências de bytes, facilitando a sua gravação e leitura em arquivos binários de forma eficiente e consistente.

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

Permite operações de processamento de dados de forma funcional, expressões lambda e pipelines de dados, ou seja, permite escrever códigos mais conciso, legível e eficiente para as operações solicitadas. Tornando assim o código mais declarativo, facilitando o desenvolvimento e manutenção do mesmo.

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

O padrão comum para persistência de dados é o de projeto DAO (Data Access Object), que separa a lógica de acesso a dados da lógica de negócios, proporcionando uma abstração limpa e modular para a manipulação dos dados armazenados.

Objetivo da prática:

Nesta etapa o objetivo é criar um sistema simples de cadastro em texto, onde o usuário pode realizar operações como adicionar, alterar, excluir e visualizar dados de entidades (pessoas físicas ou jurídicas). Essas operações serão feitas através de um menu de opções, onde o usuário digitará números para escolher o que deseja fazer. Além disso, o sistema permite salvar e recuperar os dados em arquivos.

Prática:

- Alterar a classe principal (main) para implementação do cadastro em modo texto;
- Criar e apresentar um meu com opções do programa para o usuário: 1 Incluir, 2
 Alterar, 3 Excluir, 4 Exibir pelo Id, 5 Exibir Todos, 6 Salvar Dados, 7 Recuperar Dados e 0 Finalizar a Execução;
- Criar estrutura (código) para as opções do menu;
- Executar e verificar as funcionalidades implementadas e os arquivos gerados.

Códigos e resultados obtidos:

Resultado inicial (exemplo):

Códigos:

```
Main_02.java ×
package model;
  3 - /**
  4
  5
       * @author gilvan
  8
  9
   import java.io.*;
    import java.util.Scanner;
 10
 11
      public class Main 02 {
 12
 13 🖃
          public static void main(String[] args) {
              Scanner scanner = new Scanner(System.in);
 14
 15
 16
              PessoaFisicaRepo repoPessoaFisica = new PessoaFisicaRepo();
 17
               PessoaJuridicaRepo repoPessoaJuridica = new PessoaJuridicaRepo();
 18
 19
              boolean continuar = true:
 20 =
              while (continuar) {
 21
                  System.out.println("===
                  System.out.println("1 - Incluir Pessoa");
 22
                  System.out.println("2 - Alterar Pessoa");
 23
 24
                  System.out.println("3 - Excluir Pessoa");
                  System.out.println("4 - Buscar pelo Id");
 25
                  System.out.println("5 - Exibir Todos");
 26
                  System.out.println("6 - Persistir dados");
 27
                  System.out.println("7 - Recuperar dados");
 28
 29
                   System.out.println("0 - Finalizar Programa");
 30
                  System.out.println("===
 31
                  int opcao = scanner.nextInt();
 32
 33
                  switch (opcao) {
 34
                      case 1 -> incluir(scanner, repoPessoaFisica, repoPessoaJuridica);
 35
                      case 2 -> alterar(scanner, repoPessoaFisica, repoPessoaJuridica);
 36
                      case 3 -> excluir(scanner, repoPessoaFisica, repoPessoaJuridica);
 37
                      case 4 -> exibirPorId(scanner, repoPessoaFisica, repoPessoaJuridica);
 38
                      case 5 -> exibirTodos(scanner, repoPessoaFisica, repoPessoaJuridica);
 39
                      case 6 -> salvarDados(scanner, repoPessoaFisica, repoPessoaJuridica);
 40
                      case 7 -> recuperarDados(scanner, repoPessoaFisica, repoPessoaJuridica);
 41
                      case 0 -> continuar = false;
 42
                      default -> System.out.println("Opcao invalida. Tente novamente.");
 43
 44
 45
 46
          private static void incluir (Scanner scanner, Pessoa Fisica Repo repo Pessoa Fisica,
 47
 48 🖃
                  PessoaJuridicaRepo repoPessoaJuridica) {
 49
              System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");
 50
              String tipo = scanner.next();
 51
 52
              System.out.println("Digite o ID da pessoa:");
 53
              int id = scanner.nextInt();
 54
               System.out.println("Insira os Dados...");
 55
 56 😑
               if (tipo.equalsIgnoreCase("F")) {
 57
                  System.out.println("Nome:");
 58
                   String nome = scanner.next();
 59
                  System.out.println("CPF:"):
 60
                  String cpf = scanner.next();
                  System.out.println("Idade:");
 61
 62
                   int idade = scanner.nextInt();
                  PessoaFisica pessoaFisica = new PessoaFisica(id, nome, cpf, idade);
 63
 64
                  repoPessoaFisica.inserir(pessoaFisica);
 65
                  pessoaFisica.exibir();
 66
               } else if (tipo.equalsIgnoreCase("J")) {
 67
                  System.out.println("Nome:");
                  String nome = scanner.next();
 68
 69
                  System.out.println("CNPJ:");
 70
                  String cnpj = scanner.next();
                  PessoaJuridica pessoaJuridica = new PessoaJuridica(id, nome, cnpj);
 71
 72
                  repoPessoaJuridica.inserir(pessoaJuridica);
 73
                  pessoaJuridica.exibir();
 74
               } else {
 75
                  System.out.println("Opcao invalida.");
 76
 77
```

```
Source
           private static void alterar (Scanner scanner, PessoaFisicaRepo repoPessoaFisica,
 79
 80 🖃
                  PessoaJuridicaRepo repoPessoaJuridica) {
 81
               System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");
              String tipo = scanner.next();
 82
 83
 84
              System.out.println("Digite o ID da pessoa:");
 85
              int id = scanner.nextInt():
 86
 87
               switch (tipo.toUpperCase()) {
 88 😑
                  case "F" -> {
 89
                      PessoaFisica pessoaFisica = repoPessoaFisica.obter(id);
 90 -
                      if (pessoaFisica != null) {
 91
                          System.out.println("Dados atuais:");
 92
                          pessoaFisica.exibir();
 93
 94
                          scanner.nextLine();
 95
                          System.out.println("Digite o novo nome:");
 96
 97
                          String nome = scanner.nextLine();
 98
                          System.out.println("Digite o novo CPF:");
                          String cpf = scanner.nextLine();
99
100
                          System.out.println("Digite a nova idade:");
101
                          int idade = scanner.nextInt();
102
103
                          pessoaFisica.setNome(nome);
104
                          pessoaFisica.setCpf(cpf);
105
                          pessoaFisica.setIdade(idade);
106
107
                          repoPessoaFisica.alterar(pessoaFisica);
108
                          pessoaFisica.exibir();
109
                      } else {
                          System.out.println("Pessoa fisica nao encontrada.");
110
111
112
113
                  case "J" -> {
114
                      PessoaJuridica pessoaJuridica = repoPessoaJuridica.obter(id);
115
                       if (pessoaJuridica != null) {
116
                          System.out.println("Dados atuais:");
117
                          pessoaJuridica.exibir();
118
119
                          scanner.nextLine();
120
                          System.out.println("Digite o novo nome:");
121
122
                          String nome = scanner.nextLine();
123
                          System.out.println("Digite o novo CNPJ:");
124
                          String cnpj = scanner.nextLine();
125
126
                          pessoaJuridica.setNome(nome);
127
                          pessoaJuridica.setCnpj(cnpj);
128
129
                          repoPessoaJuridica.alterar(pessoaJuridica);
130
                          pessoaJuridica.exibir();
131 🖨
                          System.out.println("Pessoa juridica nao encontrada.");
132
133
134
                  default -> System.out.println("Opcao invalida.");
135
136
137
138
139
          private static void excluir(Scanner scanner, PessoaFisicaRepo repoPessoaFisica,
140 🖵
                 PessoaJuridicaRepo repoPessoaJuridica) {
              System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");
141
142
              String tipo = scanner.next();
143
144
              System.out.println("Digite o ID da pessoa:");
145
              int id = scanner.nextInt();
146
147
              switch (tipo.toUpperCase()) {
148
                  case "F" -> repoPessoaFisica.excluir(id);
                  case "J" -> repoPessoaJuridica.excluir(id);
149
150
                  default -> System.out.println("Opcao invalida.");
151
152
```

```
private static void exibirPorId(Scanner scanner, PessoaFisicaRepo repoPessoaFisica,
154
155 📮
                 PessoaJuridicaRepo repoPessoaJuridica) {
156
              System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");
              String tipo = scanner.next();
157
158
              System.out.println("Digite o ID da pessoa:");
159
              int id = scanner.nextInt();
160
161
162
              switch (tipo.toUpperCase()) {
163
                  case "F" -> {
164
                      PessoaFisica pessoaFisica = repoPessoaFisica.obter(id);
165
                      if (pessoaFisica != null) {
166
                         pessoaFisica.exibir();
167
                      } else {
168
                         System.out.println("Pessoa fisica nao encontrada.");
169
170
171
                  case "J" -> {
172
                     PessoaJuridica pessoaJuridica = repoPessoaJuridica.obter(id);
173
                     if (pessoaJuridica != null) {
174
                         pessoaJuridica.exibir();
175
                      } else {
176
                         System.out.println("Pessoa juridica nao encontrada.");
177
178
179
                  default -> System.out.println("Opcao invalida.");
180
181
182
183
          private static void exibirTodos (Scanner scanner, PessoaFisicaRepo repoPessoaFisica,
184 🖵
                 PessoaJuridicaRepo repoPessoaJuridica) {
185
              System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");
186
              String tipo = scanner.next();
187
188
              switch (tipo.toUpperCase()) {
189 😑
                  case "F" -> {
                     System.out.println("Pessoas Fisicas:");
190
191 😑
                      for (PessoaFisica pessoa : repoPessoaFisica.obterTodos()) {
192
                         pessoa.exibir();
193
194
195
                  case "J" -> {
                     System.out.println("Pessoas Juridicas:");
196
197
                      for (PessoaJuridica pessoa : repoPessoaJuridica.obterTodos()) {
198
                         pessoa.exibir();
199
200
                  default -> System.out.println("Opcao invalida.");
201
202
203
204
          private static void salvarDados (Scanner scanner, PessoaFisicaRepo repoPessoaFisica,
205
206 🖃
                 PessoaJuridicaRepo repoPessoaJuridica) {
207
              try {
208
                  System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");
209
                 String prefixo = scanner.next();
                 repoPessoaFisica.persistir(prefixo + ".fisica.bin");
210
                 repoPessoaJuridica.persistir(prefixo + ".juridica.bin");
211
212
                  System.out.println("Dados salvos com sucesso.");
213
              } catch (IOException e) {
214
                  System.out.println("Erro ao salvar os dados: " + e.getMessage());
215
              1
216
217
218
          private static void recuperarDados(Scanner scanner, PessoaFisicaRepo repoPessoaFisica,
219 🖵
                 PessoaJuridicaRepo repoPessoaJuridica) {
220 🖹
              try {
221
                  System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");
222
                  String prefixo = scanner.next();
223
                 repoPessoaFisica.recuperar(prefixo + ".fisica.bin");
224
                  repoPessoaJuridica.recuperar(prefixo + ".juridica.bin");
225
                  System.out.println("Dados recuperados com sucesso.");
226
              } catch (IOException | ClassNotFoundException e) {
227
                  System.out.println("Erro ao recuperar os dados: " + e.getMessage());
228
229
```

230

• Resultado da execução dos códigos:

```
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
_____
F - Pessoa Fisica | J - Pessoa Juridica
Digite o ID da pessoa:
120
Insira os Dados...
                                     _____
Nome:
                                     1 - Incluir Pessoa
AAA
                                     2 - Alterar Pessoa
CPF:
                                     3 - Excluir Pessoa
111
Idade:
                                     4 - Buscar pelo Id
                                    5 - Exibir Todos
                                    6 - Persistir dados
ID: 120, Nome: AAA, CPF: 111, Idade: 11
                                     7 - Recuperar dados
_____
                                     0 - Finalizar Programa
1 - Incluir Pessoa
2 - Alterar Pessoa
                                     -----
3 - Excluir Pessoa
                                     F - Pessoa Fisica | J - Pessoa Juridica
4 - Buscar pelo Id
                                    F
5 - Exibir Todos
                                    Digite o ID da pessoa:
6 - Persistir dados
7 - Recuperar dados
                                    ID: 120, Nome: BBB, CPF: 222, Idade: 22
0 - Finalizar Programa
_____
                                     1 - Incluir Pessoa
                                    2 - Alterar Pessoa
F - Pessoa Fisica | J - Pessoa Juridica
                                    3 - Excluir Pessoa
                                     4 - Buscar pelo Id
Digite o ID da pessoa:
120
                                     5 - Exibir Todos
                                     6 - Persistir dados
Dados atuais:
                                    7 - Recuperar dados
ID: 120, Nome: AAA, CPF: 111, Idade: 11
                                     0 - Finalizar Programa
Digite o novo nome:
BBB
                                     ______
Digite o novo CPF:
                                     F - Pessoa Fisica | J - Pessoa Juridica
222
Digite a nova idade:
                                     Pessoas Fisicas:
```

```
1 - Incluir Pessoa
                                     1 - Incluir Pessoa
2 - Alterar Pessoa
                                     2 - Alterar Pessoa
3 - Excluir Pessoa
                                     3 - Excluir Pessoa
4 - Buscar pelo Id
                                     4 - Buscar pelo Id
5 - Exibir Todos
                                     5 - Exibir Todos
6 - Persistir dados
                                     6 - Persistir dados
7 - Recuperar dados
                                     7 - Recuperar dados
0 - Finalizar Programa
                                     0 - Finalizar Programa
_____
                                     _____
F - Pessoa Fisica | J - Pessoa Juridica
                                     F - Pessoa Fisica | J - Pessoa Juridica
Dados salvos com sucesso.
                                     Dados recuperados com sucesso.
                                     _____
1 - Incluir Pessoa
                                     1 - Incluir Pessoa
2 - Alterar Pessoa
                                      2 - Alterar Pessoa
3 - Excluir Pessoa
                                     3 - Excluir Pessoa
4 - Buscar pelo Id
                                     4 - Buscar pelo Id
5 - Exibir Todos
                                     5 - Exibir Todos
6 - Persistir dados
                                     6 - Persistir dados
7 - Recuperar dados
                                     7 - Recuperar dados
0 - Finalizar Programa
                                     0 - Finalizar Programa
_____
                                     _____
                                     5
F - Pessoa Fisica | J - Pessoa Juridica
                                     F - Pessoa Fisica | J - Pessoa Juridica
Digite o ID da pessoa:
                                     Pessoas Fisicas:
120
                                     ID: 120, Nome: BBB, CPF: 222, Idade: 22
_____
                                     _____
1 - Incluir Pessoa
                                     1 - Incluir Pessoa
2 - Alterar Pessoa
3 - Excluir Pessoa
                                     2 - Alterar Pessoa
4 - Buscar pelo Id
                                     3 - Excluir Pessoa
5 - Exibir Todos
                                     4 - Buscar pelo Id
6 - Persistir dados
                                     5 - Exibir Todos
7 - Recuperar dados
                                     6 - Persistir dados
0 - Finalizar Programa
                                     7 - Recuperar dados
_____
                                     0 - Finalizar Programa
                                     _____
F - Pessoa Fisica | J - Pessoa Juridica
                                     BUILD SUCCESSFUL (total time: 2 minutes 38 seconds)
```

Pessoas Fisicas:

Conclusão:

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

Elementos estáticos são aqueles que pertencem à classe em vez de instâncias individuais da classe. No contexto do método 'main', ele é estátco para ser acessado sem criar uma instância da classe, facilitando a execução do programa.

b) Para que serve a classe Scanner?

É utilizada para obter entrada do usuário a partir do teclado, permitindo ler diferentes tipos de dados de entrada, como inteiros, strings, etc. tornando-se fundamental para interações entre usuário e programa.

c) Como o uso de classes de repositório impactou na organização do código? O uso de classes promove uma organização modular e coesa para o código, separando as responsabilidades de gerenciamentos de tipos específicos de entidades, torando o código mais legível, de fácil manutenção e atualização, seguindo os princípios de encapsulamento e coesão, deixando a reutilização de código mais ágil, pois as operações relacionadas as entidades específicas estão contidas em suas próprias classes de repositório.