



Estácio

Missão Prática | Nível 1 | Mundo 3

RPG0014 - Iniciando o caminho pelo Java.

Aluna: Simone Ramos de Jesus.

Matricula: 202208290965.

Campus: Polo Prado – Belo Horizonte – MG.

Curso: Desenvolvimento Full Stack

Turma: 2023.3

GITHUB:

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.
7. persistência em arquivos binários.

Model-Entidades-Pessoa:

```
Pessoa.java X
model > entidades > Pessoa.java

1 + package model.entidades;
2
3 import java.io.Serializable;
4
5 public abstract class Pessoa implements Serializable {
6     private String nome;
7     private int id;
8
9     public Pessoa() {
10    }
11
12    public Pessoa(String nome, int id) {
13        this.nome = nome;
14        this.id = id;
15    }
16
17    public int getId() {
18        return id;
19    }
20
21    public void setId(int id) {
22        this.id = id;
23    }
24
25    public String getNome() {
26        return nome;
27    }
28
29    public void setNome(String nome) {
30        this.nome = nome;
31    }
32
33    public void exibir() throws Exception {
34        System.out.println("Nome: " + nome);
35        System.out.println("ID: " + id);
36    }
37 }
```

Model- Entidades- Pessoa Física:

```
PessoaFisica.java X
model > entidades > PessoaFisica.java

1+ package model.entidades;
2
3 import java.io.Serializable;
4 import java.time.LocalDate;
5
6 public class PessoaFisica extends Pessoa implements Serializable {
7     private String cpf;
8     private int idade;
9
10    public PessoaFisica() {
11    }
12
13    public PessoaFisica(String nome, String cpf, int idade, int id) {
14        super(nome, id);
15        this.idade = idade;
16        this.cpf = cpf;
17    }
18
19    public PessoaFisica(String string, String string2, LocalDate localDate) {
20    }
21
22    public String getCpf() {
23        return cpf;
24    }
25
26    public void setCpf(String cpf) {
27        this.cpf = cpf;
28    }
29
30    public int getIdade() {
31        return idade;
32    }
33
34    public void setIdade(int idade) {
35        this.idade = idade;
36    }
37
38    @Override
39    public void exibir() {
40        System.out.println("***Pessoa Física***");
41        System.out.println("ID: " + getId());
42        System.out.println("Nome: " + getNome());
43        System.out.println("CPF: " + getCpf());
44        System.out.println("Idade: " + getIdade());
45    }
46 }
```

Model – Entidades – Pessoa Jurídica:

```
PessoaJuridica.java X
model > entidades > PessoaJuridica.java

1 + package model.entidades;
2
3 import java.io.Serializable;
4
5 public class PessoaJuridica extends Pessoa implements Serializable {
6     private String cnpj;
7
8     public PessoaJuridica() {
9     }
10
11     public PessoaJuridica(String nome, String cnpj, int id) {
12         super(nome, id);
13         this.cnpj = cnpj;
14     }
15
16     public PessoaJuridica(String nomeCompleto, int id, Object cnpj2) {
17     }
18
19     public String getCnpj() {
20         return cnpj;
21     }
22
23     public void setCnpj(String cnpj) {
24         this.cnpj = cnpj;
25     }
26
27     @Override
28     public void exibir() {
29         System.out.println("****Pessoa Jurídica****");
30         System.out.println("ID: " + getId());
31         System.out.println("Nome: " + getNome());
32         System.out.println("CNPJ: " + getCnpj());
33     }
34
35 }
36
```

Model – Gerenciador – Pessoa Física Repo:

```
PessoaFisicaRepo.java X
model > gerenciador > PessoaFisicaRepo.java

1  package model.gerenciador;
2
3  import java.io.File;
4  import java.util.ArrayList;
5  import java.util.List;
6
7  import model.entidades.PessoaFisica;
8
9  import java.io.FileInputStream;
10 import java.io.FileOutputStream;
11 import java.io.IOException;
12 import java.io.ObjectInputStream;
13 import java.io.ObjectOutputStream;
14
15 public class PessoaFisicaRepo {
16     private List<PessoaFisicaRepo> pessoasFisicasLista = new ArrayList<>();
17     public PessoaFisicaRepo pessoasFisicas;
18
19     public void inserir(PessoaFisica pf) {
20         getPessoasFisicasLista().add(pf);
21     }
22
23     public List<PessoaFisicaRepo> getPessoasFisicasLista() {
24         return pessoasFisicasLista;
25     }
26
27     public void setPessoasFisicasLista(List<PessoaFisicaRepo> list) {
28         this.pessoasFisicasLista = list;
29     }
30
31     public void alterar(PessoaFisicaRepo pessoaFisicaExcluir) {
32 +     int index = getPessoasFisicasLista().indexOf(pessoaFisicaExcluir);
33         if (index != -1) {
34             getPessoasFisicasLista().set(index, pessoaFisicaExcluir);
35         }
36     }
37 }
```

```

PessoaFisicaRepo.java
model > gerenciador > PessoaFisicaRepo.java

37  /**
38   * @param id
39   */
40  public void excluir(int id) {
41      getPessoasFisicasLista().removeIf(pessoa -> pessoa.getId() == id);
42  }
43
44  int getId() {
45      return 0;
46  }
47
48  public PessoaFisicaRepo obter(int id) {
49      for (PessoaFisicaRepo pessoa : getPessoasFisicasLista()) {
50          if (pessoa.getId() == id) {
51              return pessoa;
52          }
53      }
54      return null;
55  }
56
57  public List<PessoaFisicaRepo> obterTodos() {
58      return this.getPessoasFisicasLista();
59  }
60
61  public void persistir(String nameArquivo) throws IOException {
62      try (ObjectOutputStream oos = new ObjectOutputStream(new FileOutputStream(nameArquivo))) {
63          oos.writeObject(getPessoasFisicasLista());
64      }
65  }

```

```

PessoaFisicaRepo.java
model > gerenciador > PessoaFisicaRepo.java

66  /**
67   * @param nameArquivo
68   * @throws IOException
69   * @throws ClassNotFoundException
70   */
71  public void recuperar(String nameArquivo) throws IOException, ClassNotFoundException {
72      try (ObjectInputStream ois = new ObjectInputStream(new FileInputStream(nameArquivo))) {
73          setPessoasFisicasLista((List<PessoaFisicaRepo>) ois.readObject());
74      }
75  }
76  public void setIdade(int parseInt) {}
77  public void setCpf(String readLine) {}
78  public String getIdade() {
79      return null;
80  }
81  public String getNome() {
82      return null;
83  }
84  public void setNome(String readLine) {}
85  public String getCnpj() {
86      return null;
87  }
88  public void setCnpj(String readLine) {}
89  public String getNomeEmpresa() {
90      return null;
91  }
92  public void setNomeEmpresa(String readLine) {}
93  public PessoaFisica[] obterTodos(int i) {
94      return null;
95  }
96  public String getCpf() {
97      return null;
98  }
99  }

```

Model – Gerenciador – Pessoa Jurídica Repo:

```
PessoaJuridicaRepo.java X
model > gerenciador > PessoaJuridicaRepo.java
1  package model.gerenciador;
2
3  import java.io.*;
4  import java.util.HashMap;
5  import java.util.Map;
6  import java.util.List;
7
8  import model.entidades.PessoaFisica;
9  import model.entidades.PessoaJuridica;
10
11 public class PessoaJuridicaRepo {
12     private Map<Integer, PessoaJuridica> PessoaJuridicaMap;
13
14     public PessoaJuridicaRepo() {
15         PessoaJuridicaMap = new HashMap<>();
16     }
17
18     public void inserir(PessoaJuridica pj) {
19         PessoaJuridicaMap.put(pj.getId(), pj);
20     }
21
22     public void alterar(PessoaJuridica pessoaJuridica) {
23         this.PessoaJuridicaMap.put(pessoaJuridica.getId(), pessoaJuridica);
24     }
25
26     public void excluir(int id) {
27         PessoaJuridicaMap.remove(id);
28     }
29
30     public PessoaJuridica obter(int id) {
31         return PessoaJuridicaMap.get(id);
32     }
33
34     public void persistir(String arquivo) throws IOException {
35         try (ObjectOutputStream oos = new ObjectOutputStream(new FileOutputStream(arquivo))) {
36             oos.writeObject(PessoaJuridicaMap);
37         }
```

PessoaJuridicaRepo.java X

model > gerenciador > PessoaJuridicaRepo.java

```
38     }
39
40     public void recuperar(String arquivo) throws IOException, ClassNotFoundException {
41         try (ObjectInputStream ois = new ObjectInputStream(new FileInputStream(arquivo))) {
42             PessoaJuridicaMap = (Map<Integer, PessoaJuridica>) ois.readObject();
43         }
44     }
45
46     public void alterar(PessoaFisicaRepo pj) {
47     }
48
49     public String getId() {
50         return null;
51     }
52
53     public String getNome() {
54         return null;
55     }
56
57     public String getCnpj() {
58         return null;
59     }
60
61     public PessoaJuridicaRepo[] obterTodos() {
62         return null;
63     }
64 }
65
```


Model – Gerenciador – Principal:

```
Principal.java X
model > gerenciador > Principal.java

1  package model.gerenciador;
2  +
3  import java.io.IOException;
4  import java.time.LocalDate;
5  import java.util.ArrayList;
6  import java.util.List;
7
8  import System.out.Repositorio;
9  import model.entidades.Pessoa;
10 import model.entidades.PessoaFisica;
11 import model.entidades.PessoaJuridica;
12 import model.gerenciador.PessoaFisicaRepo;
13 import model.gerenciador.PessoaJuridicaRepo;
14 import model.gerenciador.Principal;
15
16 public class Principal{
17
18     private static final String Repositorio = null;
19
20     public static void main(String[] args) throws IOException{
21
22         Repositorio<PessoaFisica> repo1 = new Repositorio <>(){
23
24             @Override
25             public void salvarEmArquivo(String string) {
26                 // TODO Auto-generated method stub
27                 throw new UnsupportedOperationException("Unimplemented method 'salvarEmArquivo'");
28             }
29
30             @Override
31             public void adicionar(PessoaFisica pessoaFisica) {
32                 // TODO Auto-generated method stub
33                 throw new UnsupportedOperationException("Unimplemented method 'adicionar'");
34             }
35
36         }
```

```
Principal.java X
model > gerenciador > Principal.java
35
36 + @Override
37   public void recuperarArquivo(String string) {
38     // TODO Auto-generated method stub
39     throw new UnsupportedOperationException("Unimplemented method 'recuperarArquivo'");
40   }
41
42   @Override
43   public PessoaJuridica[] obterTodos() {
44     // TODO Auto-generated method stub
45     throw new UnsupportedOperationException("Unimplemented method 'obterTodos'");
46   }
47
48   @Override
49   public void recuperarDeArquivo(String string) {
50     // TODO Auto-generated method stub
51     throw new UnsupportedOperationException("Unimplemented method 'recuperarDeArquivo'");
52   }
53   Repositorio<PessoaFisica> repo2 = new Repositorio <>(){
54
55     @Override
56     public void salvarEmArquivo(String string) {
57       // TODO Auto-generated method stub
58       throw new UnsupportedOperationException("Unimplemented method 'salvarEmArquivo'");
59     }
60
61     @Override
62     public void adicionar(PessoaFisica pessoaFisica) {
63       // TODO Auto-generated method stub
64       throw new UnsupportedOperationException("Unimplemented method 'adicionar'");
65     }
66
67     @Override
68     public void recuperarArquivo(String string) {
69       // TODO Auto-generated method stub
70       throw new UnsupportedOperationException("Unimplemented method 'recuperarArquivo'");
71     }
72   }
```

Principal.java X

odel > gerenciador > Principal.java

```
73      @Override
74      + public PessoaJuridica[] obterTodos() {
75          // TODO Auto-generated method stub
76          throw new UnsupportedOperationException("Unimplemented method 'obterTodos'");
77      }
78
79      @Override
80      public void recuperarDeArquivo(String string) {
81          // TODO Auto-generated method stub
82          throw new UnsupportedOperationException("Unimplemented method 'recuperarDeArquivo'");
83      };
84
85      repo1.adicionar(new PessoaFisica("Simone", "01234", LocalDate.of(1985, 11, 18)));
86      repo1.adicionar(new PessoaFisica("Paula", "98765", LocalDate.of(1987, 04, 15)));
87
88      repo1.salvarEmArquivo("teste.fisica.bin");
89
90      try{
91          repo2.recuperarDeArquivo("teste.fisica.bin");
92      }
93      catch (IOException|ClassNotFoundException e){
94          e.printStackTrace();
95      }
96
97      for (PessoaFisicaRepo pf : ((PessoaFisicaRepo) repo2).obterTodos()){
98          System.out.println(pf);
99      }
100
101      Repositorio<PessoaJuridica> repo3 = new Repositorio<>() {
102
103          @Override
104          public void salvarEmArquivo(String string) {
105              // TODO Auto-generated method stub
106              throw new UnsupportedOperationException("Unimplemented method 'salvarEmArquivo'");
107          }
108      }
```

Principal.java X

model > gerenciador > Principal.java

```
108
109     @Override
110     public void adicionar(PessoaJuridica pessoaJuridica) {
111         // TODO Auto-generated method stub
112         throw new UnsupportedOperationException("Unimplemented method 'adicionar'");
113     }
114
115     @Override
116     public void recuperarArquivo(String string) {
117         // TODO Auto-generated method stub
118         throw new UnsupportedOperationException("Unimplemented method 'recuperarArquivo'");
119     }
120
121     @Override
122     public PessoaJuridica[] obterTodos() {
123         // TODO Auto-generated method stub
124         throw new UnsupportedOperationException("Unimplemented method 'obterTodos'");
125     }
126
127     @Override
128     public void adicionar(PessoaFisica pessoaFisica) {
129         // TODO Auto-generated method stub
130         throw new UnsupportedOperationException("Unimplemented method 'adicionar'");
131     }
132
133     @Override
134     public void recuperarDeArquivo(String string) {
135         // TODO Auto-generated method stub
136         throw new UnsupportedOperationException("Unimplemented method 'recuperarDeArquivo'");
137     }
138     Repositorio<PessoaJuridica> repo4 = new Repositorio<>() {
139
140     @Override
141     public void salvarEmArquivo(String string) {
142         // TODO Auto-generated method stub
143         throw new UnsupportedOperationException("Unimplemented method 'salvarEmArquivo'");
144     }
```

```
Principal.java X
model > gerenciador > Principal.java

145 +
146     @Override
147     public void adicionar(PessoaFisica pessoaFisica) {
148         // TODO Auto-generated method stub
149         throw new UnsupportedOperationException("Unimplemented method 'adicionar'");
150     }
151
152     @Override
153     public void recuperarArquivo(String string) {
154         // TODO Auto-generated method stub
155         throw new UnsupportedOperationException("Unimplemented method 'recuperarArquivo'");
156     }
157
158     @Override
159     public PessoaJuridica[] obterTodos() {
160         // TODO Auto-generated method stub
161         throw new UnsupportedOperationException("Unimplemented method 'obterTodos'");
162     }
163
164     @Override
165     public void recuperarDeArquivo(String string) {
166         // TODO Auto-generated method stub
167         throw new UnsupportedOperationException("Unimplemented method 'recuperarDeArquivo'");
168     };
169
170     repo3.adicionar(new PessoaJuridica("Empresa On", 9870, LocalDate.of(0, 0, 0)));
171     repo4.adicionar(new PessoaJuridica("Empresa Off", 5678, LocalDate.of(0, 0, 0)));
172
173     repo3.salvarEmArquivo("teste.juridica.bin");
174
175     repo4.recuperarArquivo("teste.juridica.bin");
176
177     for(PessoaJuridica pj : repo4.obterTodos()){
178         System.out.println(pj);
179     }
180 }
181 }
```

Model – Gerenciador – Main:

```
main.java X
model > gerenciador > main.java

3     import java.io.BufferedReader;
4     import java.io.IOException;
5     import java.io.InputStreamReader;
6     import java.time.LocalDate;
7     import java.util.ArrayList;
8     import java.util.List;
9
10    import model.entidades.Pessoa;
11    import model.entidades.PessoaFisica;
12    import model.entidades.PessoaJuridica;
13    import model.gerenciador.PessoaFisicaRepo;
14    import model.gerenciador.PessoaJuridicaRepo;
15    import model.gerenciador.Principal;
16
17    public class main {
18        private static BufferedReader reader;
19        private static String nome;
20        private static Object cnpj;
21
22        /**
23         * @param args
24         */
25        public static void main(String[] args){
26            BufferedReader reader = new BufferedReader(new InputStreamReader(System.in));
27            String opcao = " ";
28            PessoaFisicaRepo repoFisica = new PessoaFisicaRepo();
29            PessoaJuridicaRepo repoJuridica = new PessoaJuridicaRepo();
```

main.java X

model > gerenciador > main.java

```
30
31 while ( !"x".equalsIgnoreCase(opcao)){
32     System.out.println("*****");
33     System.out.println("Selecione a opção desejada: ");
34     System.out.println("01 - Inserir Pessoa");
35     System.out.println("02 - Alterar Pessoa");
36     System.out.println("03 - Excluir Pessoa");
37     System.out.println("04 - Pesquisar Por ID");
38     System.out.println("05 - Exibir Todos");
39     System.out.println("06 - Persistir Dados");
40     System.out.println("07 - Recuperar Dados");
41     System.out.println("00 - Exit");
42     System.out.println("*****");
43 }
44
45 try{
46     opcao = reader.readLine();
47     switch (opcao){
48         case "01":
49             System.out.println("1 - Pessoa Física / 2 - Pessoa Jurídica");
50             String tipoPessoa = reader.readLine();
51             switch (tipoPessoa){
52                 case "1":
53                     PessoaFisica pf = lerDadosPessoaFisica(reader);
54                     repoFisica.inserir(pf);
55                     System.out.println(" Pessoa Física inserida com sucesso. ");
56                     break;
57                 case "2":
58                     PessoaJuridica pj = lerDadosPessoaJuridica(reader);
59                     repojuridica.inserir(pj);
60                     System.out.println(" Pessoa Jurídica inserida com sucesso. ");
61                     break;
62                 default:
63                     System.out.println("Ooops! Opção inválida.Tente novamente");
64                     break;
65             }
66         }
67     }
68 }
```

```
main.java X
model > gerenciador > main.java
66
67 + case "02":
68     System.out.println("1 - Pessoa Física / 2 - Pessoa Jurídica");
69     String tipoPessoaAlterar = reader.readLine();
70     switch (tipoPessoaAlterar){
71         case "1":
72             alterarPessoa(repoFisica, reader);
73             System.out.println("Dados cadastrais, alterado com sucesso. ");
74             break;
75         case "2":
76             alterarPessoa(repoJuridica, reader);
77             System.out.println("Dados cadastrais, alterado com sucesso.");
78             break;
79         default:
80             System.out.println("Ooops! Opção inválida. Tente novamente!!!");
81             break;
82     case "03":
83         System.out.println("1 - Pessoa Física / 2 - Pessoa Jurídica");
84         String tipoPessoaExcluir = reader.readLine();
85         switch (tipoPessoaExcluir){
86             case "1":
87                 excluirPessoa(repoFisica, reader);
88                 System.out.println("Dados cadastrais, excluído com sucesso.");
89                 break;
90             case "2":
91                 excluirPessoa(repoJuridica, reader);
92                 System.out.println("Dados cadastrais, excluído com sucesso.");
93                 break;
94             default:
95                 System.out.println("Ooops! Opção inválida. Tente novamente!!!");
96                 break;
97         }
98     }
```

```
main.java X
model > gerenciador > main.java
99 + case "04":
100     System.out.println("1 - Pessoa Física / 2 - Pessoa Jurídica");
101     String tipoPessoaPesquisar = reader.readLine();
102     switch (tipoPessoaPesquisar){
103         case "1":
104             pesquisarPessoa(repoFisica, reader);
105             System.out.println("Pessoa Física encontrada com sucesso. ");
106             break;
107         case "2":
108             pesquisarPessoa(repoJuridica, reader);
109             System.out.println("Empresa encontrada com sucesso.");
110             break;
111         default:
112             System.out.println("Ooops! Opção inválida. Tente novamente!!!");
113             break;
114     }
115
116 case "05":
117     System.out.println("1 - Pessoa Física / 2 - Pessoa Jurídica");
118     String tipoPessoaExibirTodos = reader.readLine();
119     switch (tipoPessoaExibirTodos){
120         case "1":
121             exibirTodasPessoas(repoFisica, reader);
122             System.out.println("Exibição de todos os dados cadastrais, pessoas físicas. ");
123             break;
124         case "2":
125             exibirTodasPessoas(repoJuridica, reader);
126             System.out.println("Exibição de todos os dados cadastrais, pessoas jurídicas.");
127             break;
128         default:
129             System.out.println("Ooops! Opção inválida. Tente novamente!!!");
130             break;
131     }
132 }
```

```
main.java X
model > gerenciador > main.java
133 case "06":
134     System.out.println("Informe o prefixo dos arquivos");
135     String arquivoX = reader.readLine();
136     try {
137         repoFisica.persistir(arquivoX + ".fisica.bin");
138         repoJuridica.persistir(arquivoX + ".juridica.bin");
139         System.out.println("Dados cadastrais, salvos com sucesso.");
140     }
141     catch(IOException e){
142         System.out.println( "[x] Erro ao salvar os dados cadastrais: "+ e.getMessage());
143         break;
144     }
145 case "07":
146     System.out.println("Informe o prefixo dos arquivos: ");
147     String arquivoG = reader.readLine();
148     try {
149         repoFisica.recuperar(arquivoG + ".fisica.bin");
150         repoJuridica.recuperar(arquivoG + ".juridica.bin");
151         System.out.println("Dados cadastrais, recuperados com sucesso!");
152     }
153     catch(IOException|ClassNotFoundException e) {
154         System.out.println( "[x] Erro ao recuperar os dados cadastrais: "+ e.getMessage());
155         break;
156     }
157
158 case "00":
159     System.out.println("Fechando o sistema, gentileza aguarde! ");
160     break;
161
162 default:
163     System.out.println("Ooops! Opção inválida. Tente novamente!!!");
164     break;
165
166 catch (IOException e){
167     final Throwable eThrowable;
168     System.out.println( "[x] Erro ao recuperar os dados cadastrais: "+ e.getMessage());
169 }
```



```
main.java X
model > gerenciador > main.java
172 + }
173
174     private static PessoaJuridica lerDadosPessoaJuridica(BufferedReader reader2) {
175     return null;
176     }
177
178
179     private static PessoaFisica lerDadosPessoaFisica(BufferedReader reader ) throws IOException{
180     System.out.println("Digite o nome completo do usuário: ");
181     String nomeCompleto = reader.readLine();
182     System.out.println("Digite o CPF do usuário: ");
183     String cpf = reader.readLine();
184     System.out.println("Digite a Idade do usuário: ");
185     int idade = Integer.parseInt(reader.readLine());
186     System.out.println("Digite o ID do usuário: ");
187     int id = Integer.parseInt(reader.readLine());
188     return new PessoaFisica(nome, cpf, id, idade);
189     }
190
191
192     private static PessoaJuridica lerDadosPessoaJuridica() throws IOException{
193     System.out.println("Digite o nome completo da empresa: ");
194     String nomeCompleto = reader.readLine();
195     System.out.println("Digite o CNPJ do usuário: ");
196     String cnpj = reader.readLine();
197     System.out.println("Digite o ID da empresa: ");
198     int id = Integer.parseInt(reader.readLine());
199     return new PessoaJuridica(nomeCompleto, id, cnpj);
200     }
201
202     private static void alterarPessoa(Object repo, BufferedReader reader) throws IOException {
203     System.out.println("Digite o ID que deseja alterar: ");
204     int id = Integer.parseInt(reader.readLine());
205     if (repo instanceof PessoaFisicaRepo) {
206     PessoaFisicaRepo pf = ((PessoaFisicaRepo) repo).obter(id);
207     if(pf!=null){
208     System.out.println("Digite o CPF antigo: " + pf.getCpf());
```

```

main.java X
model > gerenciador > main.java
202 private static void alterarPessoa(Object repo, BufferedReader reader) throws IOException {
203     System.out.println("Digite o ID que deseja alterar: ");
204     int id = Integer.parseInt(reader.readLine());
205     if (repo instanceof PessoaFisicaRepo) {
206         PessoaFisicaRepo pf = ((PessoaFisicaRepo) repo).obter(id);
207         if(pf!=null){
208             System.out.println("Digite o CPF antigo: " + pf.getCpf());
209             System.out.println("Digite o novo CPF do usuário: ");
210             pf.setCpf(reader.readLine());
211             System.out.println("Digite a Idade antiga: " + pf.getIdade());
212             System.out.println("Digite a nova Idade do usuário: ");
213             pf.setIdade(Integer.parseInt(reader.readLine()));
214             System.out.println("Digite o Nome Antigo: " + pf.getNome());
215             System.out.println("Digite o novo Nome do usuário: ");
216             pf.setNome(reader.readLine());
217             ((PessoaFisicaRepo)repo).alterar(pf);
218             System.out.println("Usuário alterado com sucesso.");
219         }else{
220             System.out.println("Desculpe.Usuário não encontrado.Tente novamente!!!");}
221     }
222     else if (repo instanceof PessoaJuridicaRepo){
223         PessoaFisicaRepo pj = ((PessoaFisicaRepo)repo).obter(id);
224         if(pj != null){
225             System.out.println("CNPJ Antigo: " + pj.getCnpj());
226             System.out.println("Digite o novo CNPJ do usuário: ");
227             pj.setCnpj(reader.readLine());
228             System.out.println("Digite o nome da empresa antiga" + pj.getNomeEmpresa());
229             System.out.println("Digite o novo nome da empresa: ");
230             pj.setNomeEmpresa(reader.readLine());
231             ((PessoaJuridicaRepo)repo).alterar(pj);
232             System.out.println("Usuário Empresa, foi alterado com sucesso.");
233         }
234         else{
235             System.out.println("Desculpe. Usuário ou empresa não encontrada. Tente novamente!");
236         }
237     }
238 }

```

```

main.java X
model > gerenciador > main.java
239 +
240
241     private static void excluirPessoa(Object repo, BufferedReader reader) throws IOException {
242         System.out.println(" Digite o Id para procurar o usuário: ");
243         int id = Integer.parseInt(reader.readLine());
244         if (repo instanceof PessoaFisicaRepo) {
245             ((PessoaFisicaRepo)repo).excluir(id);
246             System.out.println("Usuário excluído com sucesso.");
247         }
248         else if(repo instanceof PessoaJuridicaRepo) {
249             ((PessoaFisicaRepo)repo).excluir(id);
250             System.out.println("Usuário Empresa foi excluído com sucesso.");
251         }
252     }
253
254     private static void buscarPessoa(Object repo, BufferedReader reader) throws IOException {
255         System.out.println(" Digite o Id para procurar o usuário: ");
256         int id = Integer.parseInt(reader.readLine());
257         if (repo instanceof PessoaFisicaRepo){
258             PessoaFisicaRepo pf = ((PessoaFisicaRepo)repo).obter(id);
259             if(pf != null){
260                 System.out.println("Id: " + pf.getId() );
261                 System.out.println("CPF: " + pf.getCpf());
262                 System.out.println("Idade: " + pf.getIdade());
263                 System.out.println("Nome: " + pf.getNome());
264             }
265             else{System.out.println("Desculpe! Usuário não encontrado. Tente novamente.");}
266         }
267         else if (repo instanceof PessoaJuridicaRepo){
268             int id;
269             PessoaJuridicaRepo repo;
270             PessoaJuridica pj = ((PessoaJuridicaRepo)repo).obter(id);
271             System.out.println("Id: " + pj.getId());
272             System.out.println("CNPJ: " + pj.getCnpj());
273             System.out.println("Nome: " + pj.getNome());
274         }
275     }

```

```

main.java X
model > gerenciador > main.java
274
275     }
276     else {
277         System.out.println("Desculpe. Usuário Empresa não encontrado. Tente novamente.");
278     }
279 +
280     private static void exibirTodasPessoas(Object repo){
281         if(repo instanceof PessoaFisicaRepo){
282             for(PessoaFisicaRepo pf : ((PessoaFisicaRepo)repo). obterTodos()){
283                 System.out.println("id: " + pf.getId());
284                 System.out.println("Nome: " + pf.getNome());
285                 System.out.println("CPF: " + pf.getCpf());
286             }
287         }
288         else if(repo instanceof PessoaJuridicaRepo){
289             for(PessoaJuridicaRepo pj : ((PessoaJuridicaRepo)repo).obterTodos()){
290                 System.out.println("id: " + pj.getId());
291                 System.out.println("Nome: " + pj.getNome());
292                 System.out.println("CNPJ: " + pj.getCnpj());
293             }
294         }
295     }
296
297     private static void exibirTodasPessoas(PessoaJuridicaRepo repojuridica, BufferedReader reader)
298     private static void exibirTodasPessoas(PessoaFisicaRepo repoFisica, BufferedReader reader) {}
299     private static void pesquisarPessoa(PessoaJuridicaRepo repojuridica, BufferedReader reader) {}
300     private static void pesquisarPessoa(PessoaFisicaRepo repoFisica, BufferedReader reader) {}
301     private static void excluirPessoa(PessoaJuridicaRepo repojuridica, BufferedReader reader) {}
302     private static void excluirPessoa(PessoaFisicaRepo repoFisica, BufferedReader reader) {}
303     private static void alterarPessoa(PessoaJuridicaRepo repojuridica, BufferedReader reader) {}
304     private static void alterarPessoa(PessoaFisicaRepo repoFisica, BufferedReader reader) {}

```

```
main.java X
model > gerenciador > main.java
300 private static void excluirPessoa(PessoaFisicaRepo repofisica, BufferedReader reader) {}
301 private static void alterarPessoa(PessoaJuridicaRepo repojuridica, BufferedReader reader) {}
302 private static void alterarPessoa(PessoaFisicaRepo repofisica, BufferedReader reader) {}
303 +
304
305 public static BufferedReader getReader() {
306     return reader;}
307 public static void setReader(BufferedReader reader) {}
308     main.reader = reader;
309 public static String getNome() {
310     return nome;}
311 public static void setNome(String nome) {
312     main.nome = nome;}
313 public static Object getCnpj() {
314     return cnpj;}
315 public static void setCnpj(Object cnpj) {
316     main.cnpj = cnpj;}
317
318 @Override
319 public String toString() {
320     return "main [";
321 }
322 }
323
```

Resultado da execução dos códigos:

Incluir os resultados obtidos ao executar o programa, como a interação com o usuário, os dados persistidos e recuperados, e qualquer saída relevante do sistema.

```
run:
Dados de Pessoa Fisica Armazenados.
Dados de Pessoa Fisica Recuperados.
Id:01234
Nome: Simone
CPF: 00000000000
Idade:38
Id: 98765
Nome: Paula
CPF: 00000000000
Idade: 36
Dados de Pessoa Juridica Armazenados.
Dados de Pessoa Juridica Recuperados.
Id:9870
Nome: Empresa On
CNPJ: 000000000000000
Id: 5678
Nome: Empresa Of
CNPJ: 000000000000000
BUILD SUCCESSFUL (total time: 1 second)
```

```
=====
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
=====
1
F - Pessoa Fisica | J - Pessoa Juridica
F
Digite o id da pessoa:
01234
Insira os dados...
Nome: Simone
```

Análise:

Os elementos estáticos em Java são relação à classe. O modificador Static é usado para declarar elementos estáticos, incluindo variáveis, métodos e blocos.

Variáveis estáticas fazem parte da classe e são compartilhadas por todas as instâncias dessa classe.

Métodos estáticos pertencem à classe e não a instâncias específicas.

Scanner é utilizada para facilitar a leitura de dados da entrada padrão, comumente associada ao teclado. Ela pertence ao pacote java.util, principal função é simplificar a interação do programa com o usuário, fornece métodos que permitem a leitura de diferentes tipos de dados, como Strings, inteiros e vários outras tipos interação com usuário.

As classes dos repositórios, tem a característica de organizar e modular, deixando o código mais intuitivo e fácil de dar manutenção e entendimento.

Conclusão:

A utilização dos elementos estáticos, classes de repositórios e classe Scanner, proporciona um código organizado, interativo, modular e com logica de persistência e uma ótima estrutura e desenvolvimento de sistemas mais complexos e eficientes em Java.