



Endereço do Campus/Polo:

Rua Moraes E Silva 40 - Maracanã - Rio de Janeiro - RJ - CEP:
20.271-030

Curso: Desenvolvimento Full Stack

Disciplina: Iniciando o Caminho Pelo Java

Turma: 9001

Semestre Letivo: Terceiro

Nome: Matheus Zimmer Moreira Martins

Endereço do Repositório:

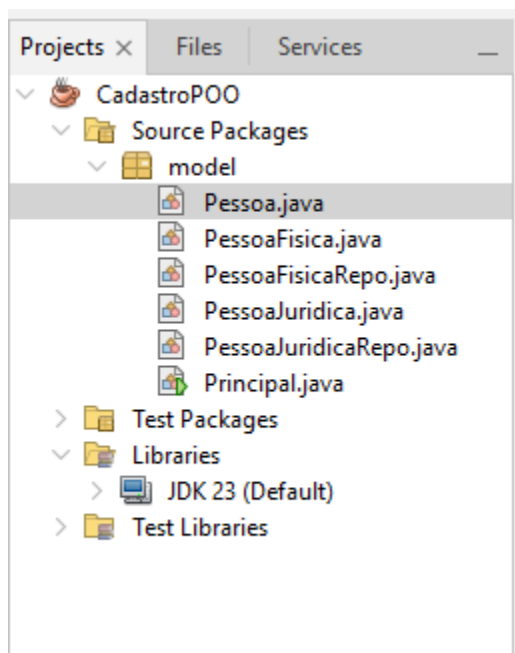
<https://github.com/Mazimo-Marts/Trabalho-Mundo3-Nvl1>

Criação de Cadastro em Modo Texto

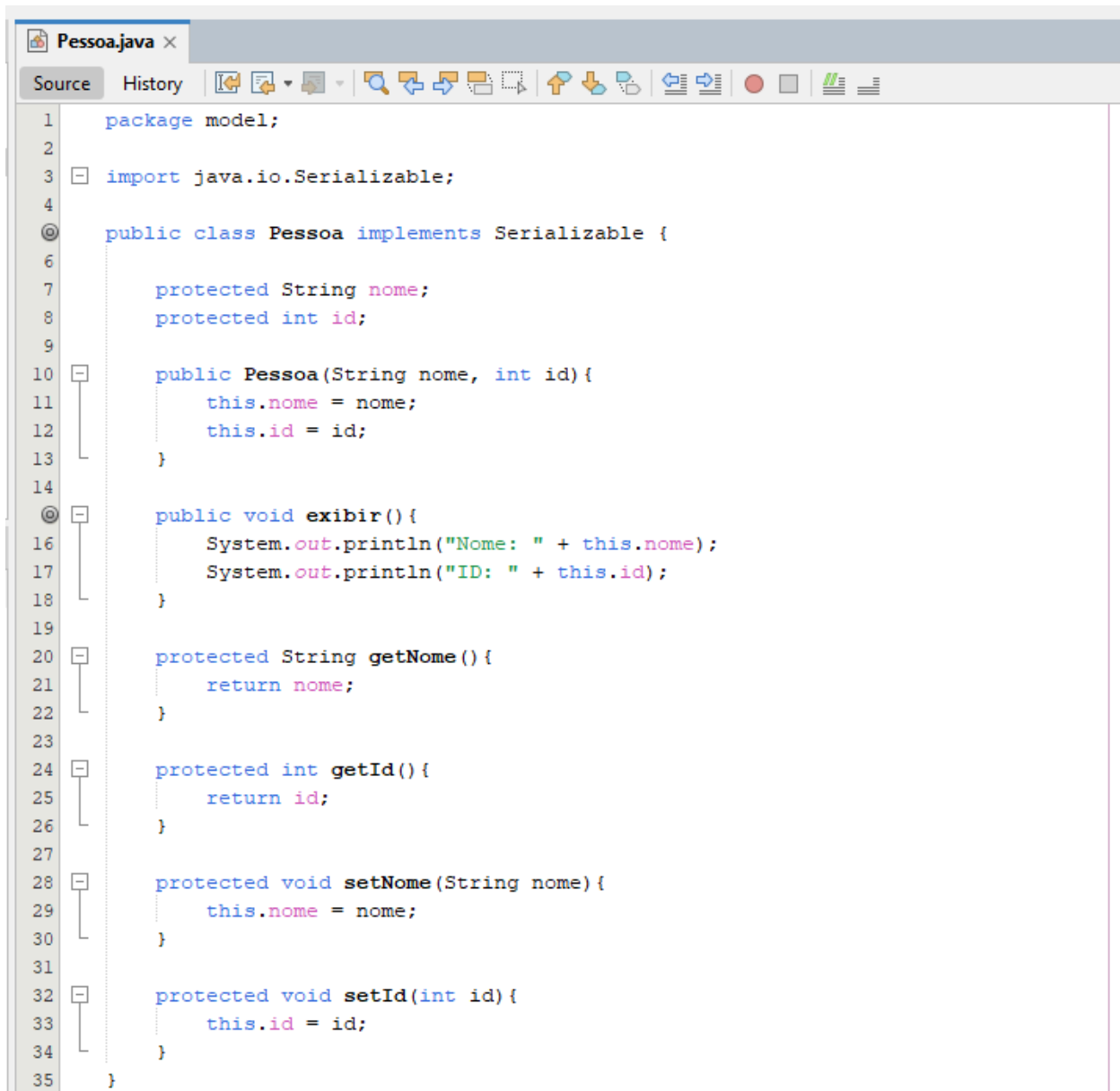
Objetivo:

Utilizar herança e polimorfismo na definição de entidades, utilizar persistência de objetos em arquivos binários, implementar uma interface cadastral em modo texto e utilizar o controle de exceções da plataforma java.

Códigos da Prática:

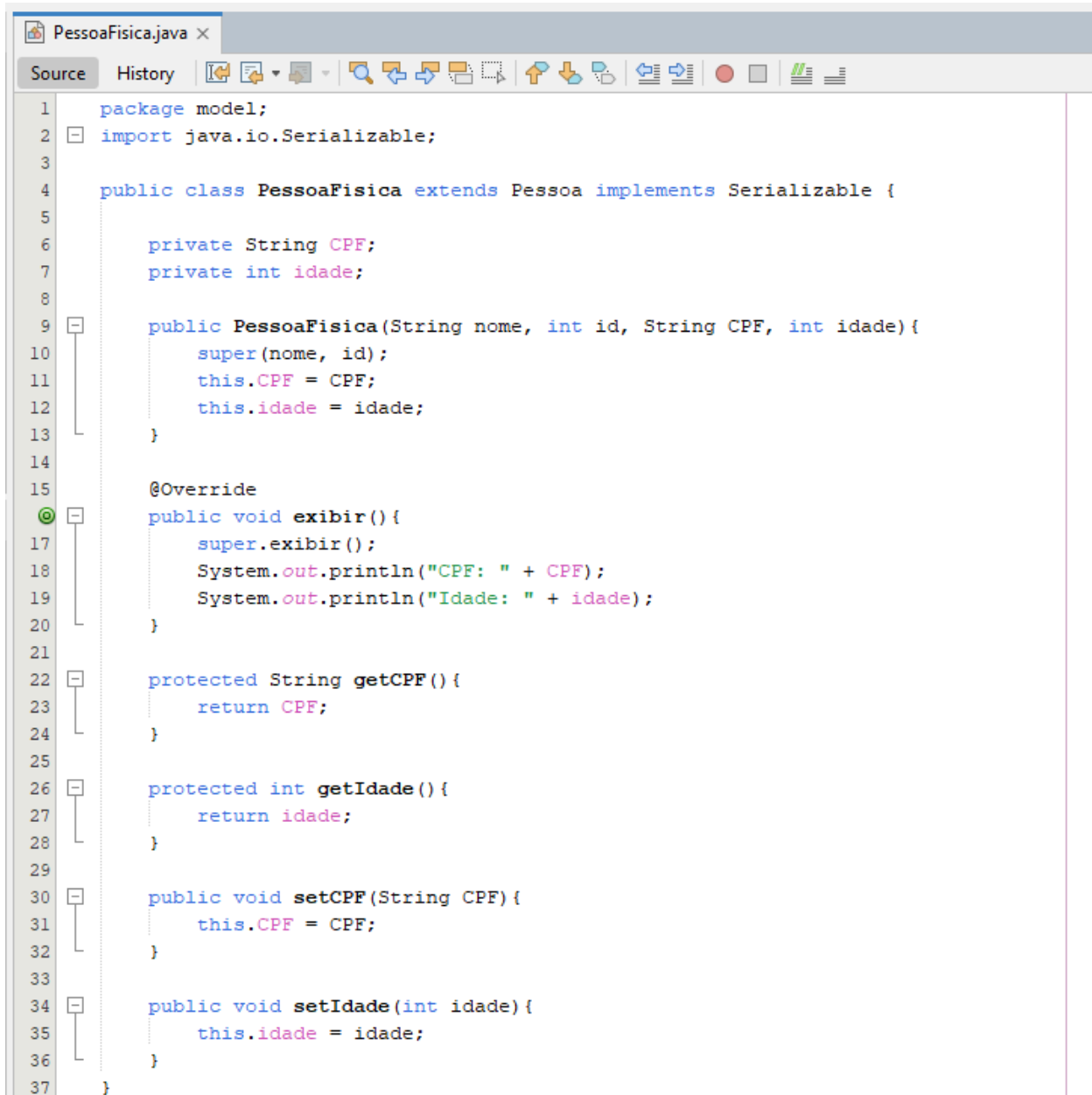


Classe Pessoa:



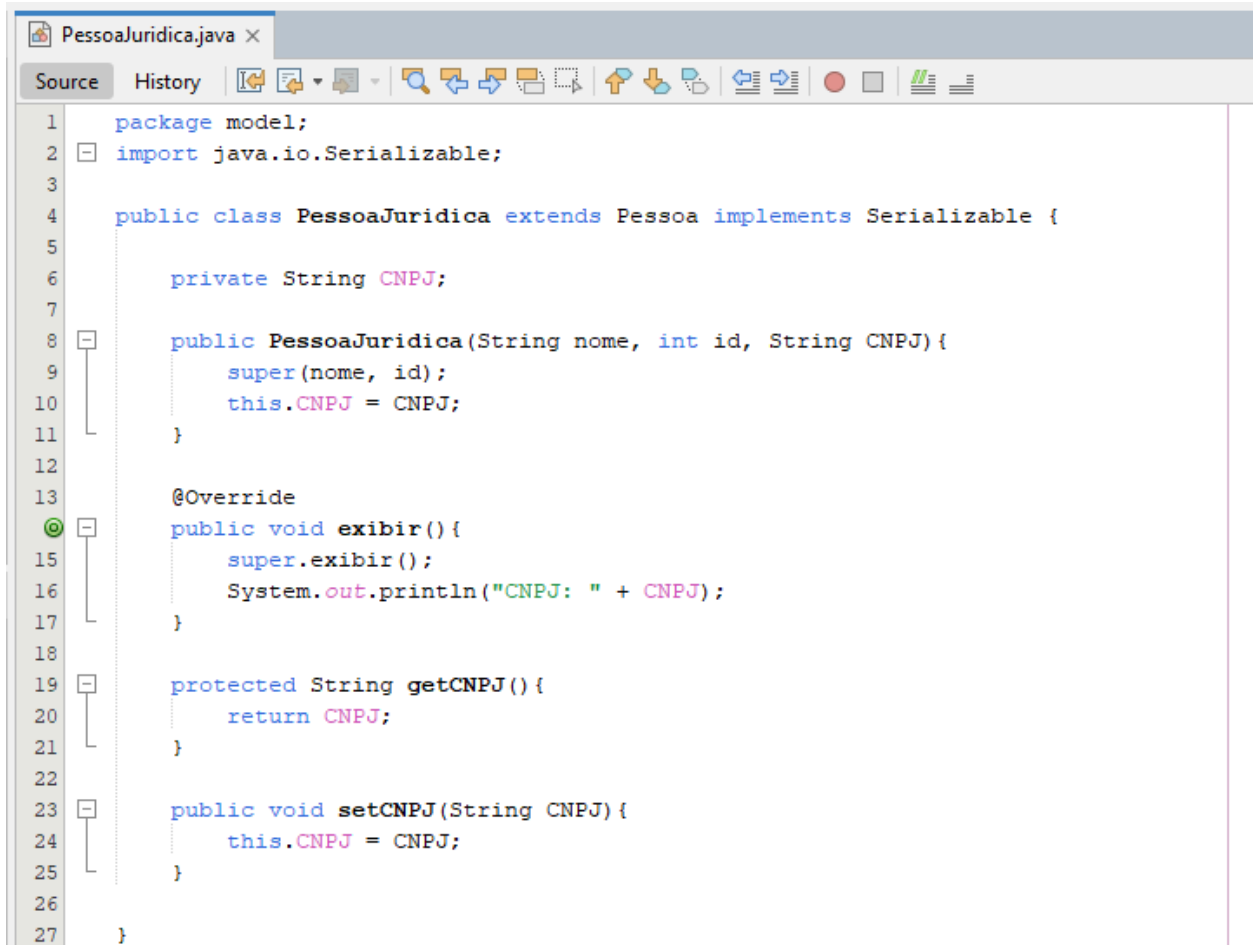
```
1 package model;
2
3 import java.io.Serializable;
4
5 public class Pessoa implements Serializable {
6
7     protected String nome;
8     protected int id;
9
10    public Pessoa(String nome, int id){
11        this.nome = nome;
12        this.id = id;
13    }
14
15    public void exibir(){
16    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

Classe Pessoa Física:



```
1 package model;
2 import java.io.Serializable;
3
4 public class PessoaFisica extends Pessoa implements Serializable {
5
6     private String CPF;
7     private int idade;
8
9     public PessoaFisica(String nome, int id, String CPF, int idade){
10         super(nome, id);
11         this.CPF = CPF;
12         this.idade = idade;
13     }
14
15     @Override
16     public void exibir(){
17         super.exibir();
18         System.out.println("CPF: " + CPF);
19         System.out.println("Idade: " + idade);
20     }
21
22     protected String getCPF(){
23         return CPF;
24     }
25
26     protected int getIdade(){
27         return idade;
28     }
29
30     public void setCPF(String CPF){
31         this.CPF = CPF;
32     }
33
34     public void setIdade(int idade){
35         this.idade = idade;
36     }
37 }
```

Classe Pessoa Jurídica:



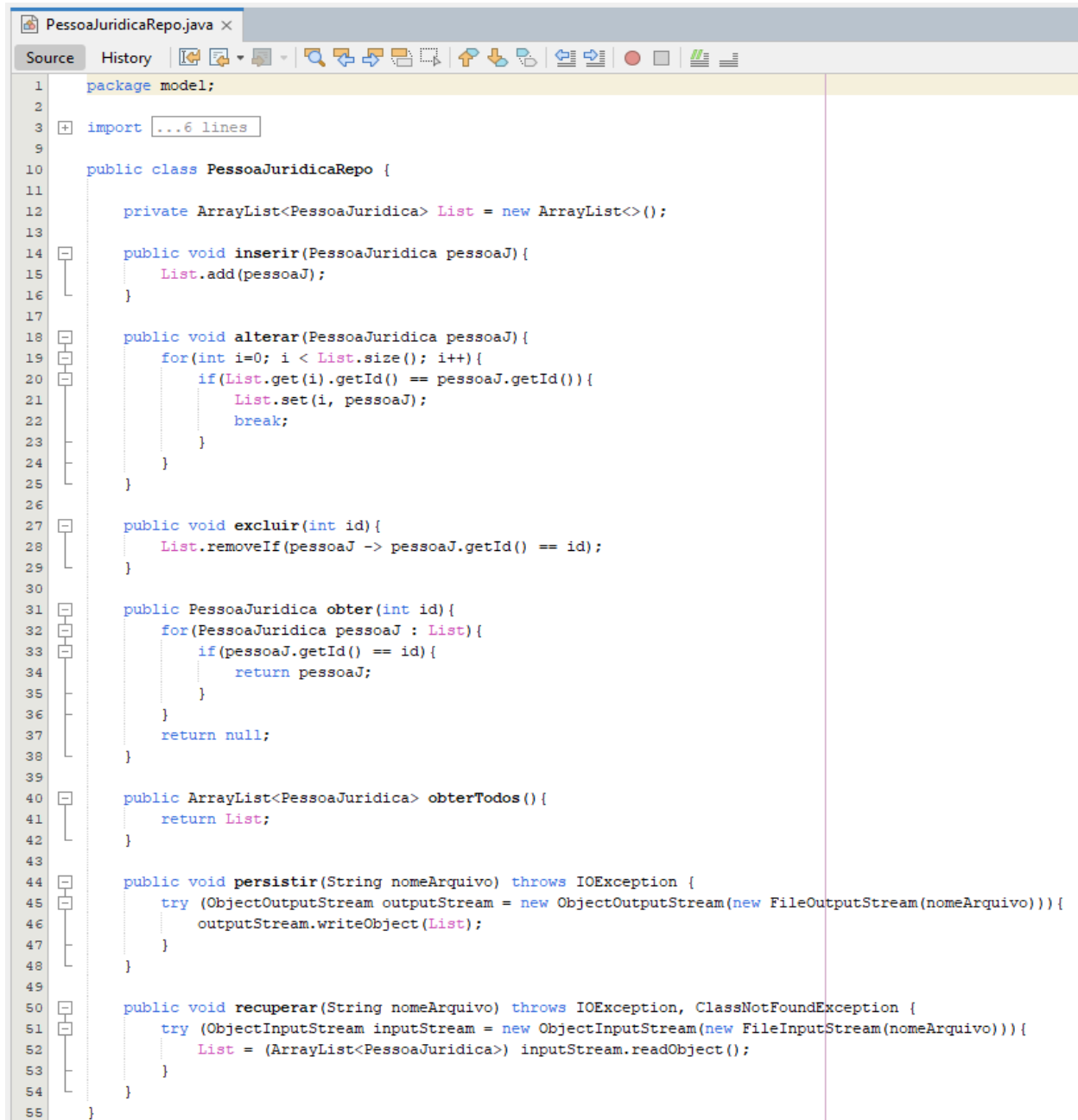
The screenshot shows an IDE window titled "PessoaJuridica.java". The code is written in Java and defines a class "PessoaJuridica" that extends "Pessoa" and implements "Serializable". The class has a private attribute "CNPJ" of type "String". It includes a constructor "PessoaJuridica(String nome, int id, String CNPJ)" that calls "super(nome, id)" and initializes "this.CNPJ". There is an overridden method "exibir()" that calls "super.exibir()" and prints the CNPJ value. Additionally, there are methods "getCNPJ()" and "setCNPJ(String CNPJ)" for accessing and modifying the CNPJ attribute. The code is formatted with standard Java syntax highlighting, and the IDE interface includes a toolbar with various editing and navigation tools.

```
1 package model;
2 import java.io.Serializable;
3
4 public class PessoaJuridica extends Pessoa implements Serializable {
5
6     private String CNPJ;
7
8     public PessoaJuridica(String nome, int id, String CNPJ){
9         super(nome, id);
10        this.CNPJ = CNPJ;
11    }
12
13    @Override
14    public void exibir(){
15        super.exibir();
16        System.out.println("CNPJ: " + CNPJ);
17    }
18
19    protected String getCNPJ(){
20        return CNPJ;
21    }
22
23    public void setCNPJ(String CNPJ){
24        this.CNPJ = CNPJ;
25    }
26
27 }
```

Classe Pessoa Física Repo:

```
PessoaFisicaRepo.java x
Source History
1 package model;
2
3 import ...6 lines
9
10 public class PessoaFisicaRepo {
11
12     private ArrayList<PessoaFisica> List = new ArrayList<>();
13
14     public void inserir(PessoaFisica pessoaF){
15         List.add(pessoaF);
16     }
17
18     public void alterar(PessoaFisica pessoaF){
19         for(int i=0; i < List.size(); i++){
20             if(List.get(i).getId() == pessoaF.getId()){
21                 List.set(i, pessoaF);
22                 break;
23             }
24         }
25     }
26
27     public void excluir(int id){
28         List.removeIf(pessoaF -> pessoaF.getId() == id);
29     }
30
31     public PessoaFisica obter(int id){
32         for(PessoaFisica pessoaF : List){
33             if(pessoaF.getId() == id){
34                 return pessoaF;
35             }
36         }
37         return null;
38     }
39
40     public ArrayList<PessoaFisica> obterTodos(){
41         return List;
42     }
43
44     public void persistir(String nomeArquivo) throws IOException {
45         try (ObjectOutputStream outputStream = new ObjectOutputStream(new FileOutputStream(nomeArquivo))) {
46             outputStream.writeObject(List);
47         }
48     }
49
50     public void recuperar(String nomeArquivo) throws IOException, ClassNotFoundException {
51         try (ObjectInputStream inputStream = new ObjectInputStream(new FileInputStream(nomeArquivo))) {
52             List = (ArrayList<PessoaFisica>) inputStream.readObject();
53         }
54     }
55 }
```

Classe Pessoa Jurídica Repo:



```
1 package model;
2
3 import ...6 lines
4
5
6
7
8
9
10 public class PessoaJuridicaRepo {
11
12     private ArrayList<PessoaJuridica> List = new ArrayList<>();
13
14     public void inserir(PessoaJuridica pessoaJ) {
15         List.add(pessoaJ);
16     }
17
18     public void alterar(PessoaJuridica pessoaJ) {
19         for(int i=0; i < List.size(); i++){
20             if(List.get(i).getId() == pessoaJ.getId()){
21                 List.set(i, pessoaJ);
22                 break;
23             }
24         }
25     }
26
27     public void excluir(int id){
28         List.removeIf(pessoaJ -> pessoaJ.getId() == id);
29     }
30
31     public PessoaJuridica obter(int id){
32         for(PessoaJuridica pessoaJ : List){
33             if(pessoaJ.getId() == id){
34                 return pessoaJ;
35             }
36         }
37         return null;
38     }
39
40     public ArrayList<PessoaJuridica> obterTodos(){
41         return List;
42     }
43
44     public void persistir(String nomeArquivo) throws IOException {
45         try (ObjectOutputStream outputStream = new ObjectOutputStream(new FileOutputStream(nomeArquivo))) {
46             outputStream.writeObject(List);
47         }
48     }
49
50     public void recuperar(String nomeArquivo) throws IOException, ClassNotFoundException {
51         try (ObjectInputStream inputStream = new ObjectInputStream(new FileInputStream(nomeArquivo))) {
52             List = (ArrayList<PessoaJuridica>) inputStream.readObject();
53         }
54     }
55 }
```

Classe Principal:

```
Principal.java x
Source History
1 package model;
2
3 import java.io.IOException;
4 import java.util.Scanner;
5
6 public class Principal {
7
8     public static void main(String[] args){
9
10         int opcao;
11         Scanner input = new Scanner(System.in);
12         PessoaFisicaRepo fisicaRepo = new PessoaFisicaRepo();
13         PessoaJuridicaRepo juridicaRepo = new PessoaJuridicaRepo();
14
15         do{
16             System.out.println("====Menu====");
17             System.out.println("1 - Incluir Pessoa");
18             System.out.println("2 - Alterar Pessoa");
19             System.out.println("3 - Excluir Pessoa");
20             System.out.println("4 - Buscar pelo Id");
21             System.out.println("5 - Exibir Todos");
22             System.out.println("6 - Persistir Dados");
23             System.out.println("7 - Recuperar Dados");
24             System.out.println("0 - Finalizar Programa");
25             System.out.println("====");
26             opcao = input.nextInt();
27
28
29             try{
30                 switch(opcao){
31                     case 1 -> incluir(input, fisicaRepo, juridicaRepo);
32                     case 2 -> alterar(input, fisicaRepo, juridicaRepo);
33                     case 3 -> excluir(input, fisicaRepo, juridicaRepo);
34                     case 4 -> obter(input, fisicaRepo, juridicaRepo);
35                     case 5 -> obterTodos(input, fisicaRepo, juridicaRepo);
36                     case 6 -> salvar(input, fisicaRepo, juridicaRepo);
37                     case 7 -> recuperar(input, fisicaRepo, juridicaRepo);
38                     case 0 -> System.out.println("Programa Finalizado.");
39                     default -> System.out.println("Opção inválida, tente novamente.");
40                 }
41             }catch(IOException | ClassNotFoundException e){
42                 System.out.println("Erro: " + e.getMessage());
43             }
44         }while(opcao != 0);
45         input.close();
46     }
```



```

47
48
49
50 private static void incluir(Scanner input, PessoaFisicaRepo fisicaRepo, PessoaJuridicaRepo juridicaRepo) {
51     System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");
52     String tipo = input.next();
53
54     switch (tipo) {
55         case "F":
56             {
57                 System.out.println("Nome: ");
58                 String nome = input.next();
59                 input.nextLine();
60                 System.out.println("Id: ");
61                 int id = input.nextInt();
62                 System.out.println("CPF: ");
63                 String CPF = input.next();
64                 System.out.println("Idade: ");
65                 int idade = input.nextInt();
66                 fisicaRepo.inserir(new PessoaFisica(nome, id, CPF, idade));
67                 break;
68             }
69         case "J":
70             {
71                 System.out.println("Nome: ");
72                 String nome = input.next();
73                 System.out.println("Id: ");
74                 int id = input.nextInt();
75                 System.out.println("CNPJ: ");
76                 String CNPJ = input.next();
77                 juridicaRepo.inserir(new PessoaJuridica(nome, id, CNPJ));
78                 break;
79             }
80         default:
81             System.out.println("Tipo inválido.");
82             break;
83     }
84 }

```

```

85 private static void alterar(Scanner input, PessoaFisicaRepo fisicaRepo, PessoaJuridicaRepo juridicaRepo) {
86
87     System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");
88     String tipo = input.next();
89     System.out.println("Id: ");
90     int id = input.nextInt();
91
92     switch (tipo) {
93         case "F":
94             {
95                 PessoaFisica pessoa = fisicaRepo.obter(id);
96                 if(pessoa != null){
97                     System.out.println("Dados atuais:");
98                     pessoa.exibir();
99                     System.out.println("Novo nome: ");
100                     String nome = input.next();
101                     System.out.println("Novo CPF: ");
102                     String CPF = input.next();
103                     System.out.println("Nova idade: ");
104                     int idade = input.nextInt();
105                     pessoa.setNome(nome);
106                     pessoa.setCPF(CPF);
107                     pessoa.setIdade(idade);
108                     fisicaRepo.alterar(pessoa);
109                     System.out.println("Pessoa Fisica alterada com sucesso.");
110                     break;
111                 }else{
112                     System.out.println("Pessoa Fisica não existente.");
113                     break;
114                 }
115             }
116         case "J":
117             {
118                 PessoaJuridica pessoa = juridicaRepo.obter(id);
119                 if(pessoa != null){
120                     System.out.println("Dados atuais: ");
121                     pessoa.exibir();
122                     System.out.println("Novo nome: ");
123                     String nome = input.next();
124                     System.out.println("Novo CNPJ: ");
125                     String CNPJ = input.next();
126                     pessoa.setNome(nome);
127                     pessoa.setCNPJ(CNPJ);
128                     juridicaRepo.alterar(pessoa);
129                     System.out.println("Pessoa Juridica alterada com sucesso.");
130                     break;
131                 }else{
132                     System.out.println("Pessoa Juridica não existe.");
133                     break;
134                 }
135             }
136         default:
137             System.out.println("Tipo inválido.");
138             break;
139     }

```

```

141
142
143 private static void excluir(Scanner input, PessoaFisicaRepo fisicaRepo, PessoaJuridicaRepo juridicaRepo) {
144
145     System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");
146     String tipo = input.next();
147     System.out.println("Id: ");
148     int id = input.nextInt();
149
150     switch (tipo) {
151         case "F":
152             {
153                 PessoaFisica pessoa = fisicaRepo.obter(id);
154                 if(pessoa != null){
155                     fisicaRepo.excluir(id);
156                     System.out.println("Pessoa Fisica excluida.");
157                     break;
158                 }else{
159                     System.out.println("Pessoa Fisica não existe.");
160                     break;
161                 }
162             }
163         case "J":
164             {
165                 PessoaJuridica pessoa = juridicaRepo.obter(id);
166                 if(pessoa != null){
167                     juridicaRepo.excluir(id);
168                     System.out.println("Pessoa Juridica excluida.");
169                     break;
170                 }else{
171                     System.out.println("Pessoa Juridica não existe.");
172                     break;
173                 }
174             }
175         default:
176             System.out.println("Tipo inválido.");
177             break;
178     }
179 }

```

```

178
179 private static void obter(Scanner input, PessoaFisicaRepo fisicaRepo, PessoaJuridicaRepo juridicaRepo) {
180
181     System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");
182     String tipo = input.next();
183     System.out.println("Id: ");
184     int id = input.nextInt();
185
186     switch (tipo) {
187         case "F":
188             {
189                 PessoaFisica pessoa = fisicaRepo.obter(id);
190                 if(pessoa != null){
191                     pessoa.exibir();
192                     break;
193                 }else{
194                     System.out.println("Pessoa Fisica não existe.");
195                     break;
196                 }
197             }
198         case "J":
199             {
200                 PessoaJuridica pessoa = juridicaRepo.obter(id);
201                 if(pessoa != null){
202                     pessoa.exibir();
203                     break;
204                 }else{
205                     System.out.println("Pessoa Juridica não existe.");
206                     break;
207                 }
208             }
209         default:
210             System.out.println("Tipo inválido.");
211             break;
212     }
213 }

```

```

214 private static void obterTodos(Scanner input, PessoaFisicaRepo fisicaRepo, PessoaJuridicaRepo juridicaRepo) {
215
216     System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");
217     String tipo = input.next();
218
219     switch (tipo) {
220     case "F":
221     {
222         for(PessoaFisica pessoa : fisicaRepo.obterTodos()){
223             pessoa.exibir();
224             System.out.println("=====");
225         }
226         break;
227     }
228     case "J":
229     {
230         for(PessoaJuridica pessoa : juridicaRepo.obterTodos()){
231             pessoa.exibir();
232             System.out.println("=====");
233         }
234         break;
235     }
236     default:
237         System.out.println("Tipo inválido.");
238         break;
239     }
240 }

```

```

241
242 private static void salvar(Scanner input, PessoaFisicaRepo fisicaRepo, PessoaJuridicaRepo juridicaRepo) throws IOException {
243     System.out.println("Digite o prefixo do arquivo: ");
244     String prefixo = input.next();
245
246     fisicaRepo.persistir(prefixo + ".fisica.bin");
247     juridicaRepo.persistir(prefixo + ".juridica.bin");
248
249     System.out.println("Arquivo salvo com sucesso.");
250 }
251
252 private static void recuperar(Scanner input, PessoaFisicaRepo fisicaRepo, PessoaJuridicaRepo juridicaRepo) throws IOException, ClassNotFoundException {
253     System.out.println("Digite o prefixo do arquivo: ");
254     String prefixo = input.next();
255
256     fisicaRepo.recuperar(prefixo + ".fisica.bin");
257     juridicaRepo.recuperar(prefixo + ".juridica.bin");
258
259     System.out.println("Arquivo recuperado com sucesso.");
260 }
261 }

```

Resultado da execução:

Output - CadastroPOO (run) #4 ×



run:

=====Menu=====

- 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

Nome:

Ana

Id:

1

CPF:

12345678890

Idade:

21

=====Menu=====

- 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

=====

2

F - Pessoa Fisica | J - Pessoa Juridica

F

Id:

1

Dados atuais:

Nome: Ana

ID: 1

CPF: 12345678890

Idade: 21

Novo nome:

Carlos

Novo CPF:

98776536521

Nova idade:

8

Pessoa Fisica alterada com sucesso.

=====Menu=====

- 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

=====

3

F - Pessoa Fisica | J - Pessoa Juridica

F

Id:

1

Pessoa Fisica exclu❖da.

=====Menu=====

- 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

=====

4

F - Pessoa Fisica | J - Pessoa Juridica

F

Id:

1

Nome: Ana

ID: 1

CPF: 1231480987

Idade: 22

=====Menu=====

- 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

=====

5

F - Pessoa Fisica | J - Pessoa Juridica

F

Nome: Ana

ID: 1

CPF: 1231480987

Idade: 22


```
=====Menu=====
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
=====
6
Digite o prefixo do arquivo:
pessoas
Arquivo salvo com sucesso.
=====Menu=====
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
=====
7
Digite o prefixo do arquivo:
pessoas
Arquivo recuperado com sucesso.
=====Menu=====
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
=====
0
Programa Finalizado.
BUILD SUCCESSFUL (total time: 57 seconds)
```

Análise

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

Elementos estáticos em Java são componentes que pertencem à classe propriamente dita, e não às instâncias (objetos) criadas a partir dessa classe. Eles são compartilhados por todos os objetos da classe e existem mesmo quando nenhuma instância foi criada. .

Main é estático porque precisa ser acessível antes da criação de qualquer objeto, permitindo que a JVM inicie a execução do programa de forma simples e direta.

b)Para que serve a classe Scanner?

É uma ferramenta para análise e leitura de dados de entrada do teclado.

c)Como o uso de classes de repositório impactou na organização do código?

Facilitou a compreensão do código como um todo ficando mais legível, fez com que a reutilização do código ficasse mais fácil e tem praticidade de manutenção pro futuro.