[1] Missão Prática | Nível 1 | Mundo 3

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

[3] Códigos solicitados:

obs.: apenas a classe Main foi alterada

//Main

```
package model;
import java.io.IOException;
import java.util.ArrayList;
import java.util.Scanner;

public class Main {
   public static String f_ou_j(){
            Scanner entrada = new Scanner(System.in);
            System.out.println("Insira o tipo de pessoa:");
            String x = entrada.nextLine();
            x.toLowerCase();
            return x;
         }

   public static void main(String args[]) {

      //instânciação de repositórios
      PessoaFisicaRepo repoF = new PessoaFisicaRepo();
      PessoaJuridicaRepo repoJ = new PessoaJuridicaRepo();

      boolean controle = true;
      while (controle){

         //menu
```

```
System.out.println("=====================================");
System.out.println("1 - Incluir Pessoa");
System.out.println("2 - Alterar Pessoa");
System.out.println("3 - Excluir Pessoa");
System.out.println("4 - Buscar pelo ID");
System.out.println("5 - Exibir Todos");
System.out.println("6 - Persistir Dados");
System.out.println("7 - Recuperar Dados");
System.out.println("0 - Finalizar Programa");
System.out.println("=====================================");
System.out.print("Escolha uma opção: ");

//instância da classe Scanner (possibilita ao usuário digitar dados)
Scanner entrada = new Scanner(System.in);
int op = entrada.nextInt();

//escolha o menu
switch(op){
   case 1: //incluir
      System.out.println("F - Pessoa Física | J - Pessoa Jurídica");
      String pes = entrada.next();
      if (pes.toLowerCase().equals("f")){ //insira dados da pessoa Física
         System.out.println("Digite o ID da pessoa:");
         int id = entrada.nextInt();
         System.out.println("Digite o nome da pessoa:");
         String nome = entrada.next();
         System.out.println("Digite a idade da pessoa:");
         int idade = entrada.nextInt();
         System.out.println("Digite o cpf da pessoa(XXX.XXX.XXX-XX):");
         String cpf = entrada.next();

         //adiciona pessoa ao repositório
         repoF.inserir(new PessoaFisica(id, nome, idade, cpf));

      }else if (pes.toLowerCase().equals("j")){ //insira dados de pessoa Jurídica
         System.out.println("Digite o ID da pessoa:");
         int id = entrada.nextInt();
         System.out.println("Digite o nome da pessoa:");
         String nome = entrada.next();
         System.out.println("Digite o cnpj da pessoa(XX.XXX.XXX/YYYY-XX):");
         String cnpj = entrada.next();

         //adiciona pessoa ao repositório
         repoJ.inserir(new PessoaJuridica(id, nome, cnpj));
      }else{
```

```
            System.out.println("Opção inválida. Reiniciando...");
        }
    break;
    case 2: //alterar
        String tipo = f_ou_j();
        System.out.println("Digite o id da pessoa que sofrerá alterações:");
        int id = entrada.nextInt();
        if (tipo.equals("f")){
            repoF.alterar(id);
        }else if (tipo.equals("j")){
            repoJ.alterar(id);
        }else{
            System.out.println("Opção inválida. Reiniciando...");
        }
    break;
    case 3: //excluir
        tipo = f_ou_j();
        System.out.println("Digite o id da pessoa a ser excluída:");
        id = entrada.nextInt();
        if (tipo.equals("f")){
            repoF.excluir(id);
        }else if (tipo.equals("j")){
            repoJ.excluir(id);
        }else{
            System.out.println("Opção inválida. Reiniciando...");
        }
    break;
    case 4: //buscar
        tipo = f_ou_j();
        System.out.println("Digite o id da pessoa buscada:");
        id = entrada.nextInt();
        if (tipo.equals("f")){
            repoF.obter(id);
        }else if (tipo.equals("j")){
            repoJ.obter(id);
        }else{
            System.out.println("Opção inválida. Reiniciando...");
        }
    break;
    case 5: //exibir
        tipo = f_ou_j();
        if (tipo.equals("f")){
            ArrayList<PessoaFisica> pessoas = repoF.obterTodos();
            for (PessoaFisica pessoa : pessoas){
                pessoa.exibir();
```

```
                }
            }else if (tipo.equals("j")){
                ArrayList<PessoaJuridica> pessoas = repoJ.obterTodos();
                for (PessoaJuridica pessoa : pessoas){
                    pessoa.exibir();
                }
            }else{
                System.out.println("Opção inválida. Reiniciando...");
            }
            break;
            case 6: //persisitr
                System.out.print("Digite o prefixo para os arquivos de persistência: ");
                String prefixoPersistencia = entrada.next();
                try {
                    repoF.persistir(prefixoPersistencia);
                    repoJ.persistir(prefixoPersistencia);
                } catch (IOException e) {
                    System.err.println("Erro ao persistir os dados: " + e.getMessage());
                }
                break;

            case 7: //recuperar
                System.out.print("Digite o prefixo para os arquivos de recuperação: ");
                String prefixoRecuperacao = entrada.next();
                try {
                    repoF.recuperar(prefixoRecuperacao);
                    repoJ.recuperar(prefixoRecuperacao);
                } catch (IOException | ClassNotFoundException e) {
                    System.err.println("Erro ao recuperar os dados: " + e.getMessage());
                }
                break;

            case 0: //fim programa
                controle = false;
                break;
        }
    }
  }
}
```

//Pessoa

```
package model;
import java.io.Serializable;

public class Pessoa implements Serializable{
```

```
    //atributos
    int id;
    String nome;

    //construtor
    public Pessoa(int id, String nome) {
        this.id = id;
        this.nome = nome;
    }

    //métodos
    public void exibir(){

    }
    //acessores
    public int getId() {
        return id;
    }
    public void setId(int inteiro) {
        this.id = inteiro;
    }

    public String getNome() {
        return nome;
    }
    public void setNome(String nome) {
        this.nome = nome;
    }

}
```

//Pessoa Física

```
package model;
import java.io.Serializable;

public class PessoaFisica extends Pessoa implements Serializable{
    //atributos
    String cpf;
    int idade;

    //construtor
    public PessoaFisica(int id, String nome, int idade, String cpf) {
        super(id, nome);
        this.cpf = cpf;
        this.idade = idade;
```

```
    }

    //métodos
    @Override
    public void exibir(){
        System.out.println("ID: " + getId() + ", Nome: " + getNome() + ", CPF: " + getCpf() +
", Idade: " + getIdade());
    }

    //acessores
    public String getCpf() {
        return cpf;
    }
    public void setCpf(String cpf) {
        this.cpf = cpf;
    }

    public int getIdade() {
        return idade;
    }
    public void setIdade(int idade) {
        this.idade = idade;
    }

}
```

//Pessoa Jurídica

```
package model;
import java.io.Serializable;

public class PessoaJuridica extends Pessoa implements Serializable{
    //atributos
    String cnpj;

    //construtor
    public PessoaJuridica(int id, String nome, String cnpj) {
        super(id, nome);
        this.cnpj = cnpj;
    }

    //métodos
    @Override
    public void exibir(){
        System.out.println("ID: " + getId() + ", Nome: " + getNome() + ", CNPJ: " + getCnpj());
    }
```

```
    //acessores
    public String getCnpj() {
        return cnpj;
    }
    public void setCnpj(String cnpj) {
        this.cnpj = cnpj;
    }

}
```

//Pessoa Física Repositório

```
package model;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.ObjectInputStream;
import java.io.ObjectOutputStream;
import java.util.ArrayList;
import java.util.Scanner;

public class PessoaFisicaRepo {
    Scanner entrada = new Scanner(System.in);

    private ArrayList <PessoaFisica> pessoasFisicas= new ArrayList<>();

    //métodos
    public void inserir(PessoaFisica pessoaf){
        pessoasFisicas.add(pessoaf);
        pessoaf.exibir();
        System.out.println("Pessoa adicionada com sucesso");
    }
    public void alterar(int id){
        boolean encontrada = false;
        for (PessoaFisica pes : pessoasFisicas){
            if (pes.getId() == id){

                encontrada = true;
                System.out.print("Pessoa encontrada: ");
                pes.exibir();
                System.out.println("Insira os novos dados:");
                System.out.print("Id: ");
                int ID = entrada.nextInt();
                System.out.print("Nome: ");
```

```
            String nome = entrada.next();
            System.out.print("Idade: ");
            int idade = entrada.nextInt();
            System.out.print("CPF: ");
            String cpf = entrada.next();

            pes.setId(ID);
            pes.setNome(nome);
            pes.setIdade(idade);
            pes.setCpf(cpf);

            System.out.println("Pessoa Física alterada com sucesso");
            pes.exibir();
        }
    }
    if (!encontrada){
        System.out.println("Pessoa Física não encontrada");
    }
}
public void excluir(int id){
    boolean encontrada = false;
    for (PessoaFisica pes: pessoasFisicas){
        if (pes.getId() == id){
            pessoasFisicas.remove(pessoasFisicas.indexOf(pes));
            System.out.println("Pessoa Física excluída com sucesso");
            encontrada = true;
            break;
        }
    }
    if (!encontrada){
        System.out.println("Pessoa Física não encontrada");
    }
}
public void obter(int id){
    boolean encontrada = false;
    for (PessoaFisica pes: pessoasFisicas){
        if (pes.getId() == id){
            pes.exibir();
            encontrada = true;
            break;
        }
    }
    if (!encontrada){
        System.out.println("Pessoa Física não encontrada");
    }
```

```
    }
    public ArrayList obterTodos(){
        return pessoasFisicas;
    }

    public void persistir(String prefixo) throws IOException{
    String arquivoFisica = prefixo + ".fisica.bin";
    try (FileOutputStream salvaFisica = new FileOutputStream(new File(arquivoFisica));
         ObjectOutputStream oosFisica = new ObjectOutputStream(salvaFisica)) {

        oosFisica.writeObject(pessoasFisicas);

        System.out.println("Dados de Pessoa Física Armazenados em " + arquivoFisica);
    } catch (IOException e) {
        throw new IOException("Erro ao persistir os dados de Pessoa Física: " +
e.getMessage());
    }
}

public void recuperar(String prefixo) throws IOException, ClassNotFoundException {
    String arquivoFisica = prefixo + ".fisica.bin";
    try (FileInputStream recuperaFisica = new FileInputStream(arquivoFisica);
         ObjectInputStream oisFisica = new ObjectInputStream(recuperaFisica)) {

        pessoasFisicas = (ArrayList<PessoaFisica>) oisFisica.readObject();

        System.out.println("Dados de Pessoa Física Recuperados de " + arquivoFisica);
    } catch (IOException | ClassNotFoundException e) {
        throw new IOException("Erro ao recuperar os dados de Pessoa Física: " +
e.getMessage());
    }
}
}
```

//Pessoa Jurídica Repositório

```
package model;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.ObjectInputStream;
import java.io.ObjectOutputStream;
import java.util.ArrayList;
import java.util.Scanner;
```

```
public class PessoaJuridicaRepo {
    Scanner entrada = new Scanner(System.in);

    private ArrayList <PessoaJuridica> pessoasJuridicas = new ArrayList<>();

    //métodos
    public void inserir(PessoaJuridica pessoaj){
        pessoasJuridicas.add(pessoaj);
        pessoaj.exibir();
        System.out.println("Pessoa adicionada com sucesso");
    }
    public void alterar(int id){
        boolean encontrada = false;
        for (PessoaJuridica pes : pessoasJuridicas){
            if (pes.getId() == id){

                encontrada = true;
                System.out.print("Pessoa encontrada: ");
                pes.exibir();
                System.out.println("Insira os novos dados:");
                System.out.print("Id: ");
                int ID = entrada.nextInt();
                System.out.print("Nome: ");
                String nome = entrada.next();
                System.out.print("CNPJ: ");
                String cnpj = entrada.next();

                pes.setId(ID);
                pes.setNome(nome);
                pes.setCnpj(cnpj);

                System.out.println("Pessoa Jurídica alterada com sucesso");
                pes.exibir();
            }
        }
        if (!encontrada){
            System.out.println("Pessoa Jurídica não encontrada");
        }
    }
    public void excluir(int id){
        boolean encontrada = false;
        for (PessoaJuridica pes: pessoasJuridicas){
            if (pes.getId() == id){
                pessoasJuridicas.remove(pessoasJuridicas.indexOf(pes));
                System.out.println("Pessoa Jurídica excluída com sucesso");
```

```
            encontrada = true;
            break;
         }
      }
      if (!encontrada){
         System.out.println("Pessoa Jurídica não encontrada");
      }
   }

   public void obter(int id){
      boolean encontrada = false;
      for (PessoaJuridica pes: pessoasJuridicas){
         if (pes.getId() == id){
            pes.exibir();
            encontrada = true;
            break;
         }
      }
      if (!encontrada){
         System.out.println("Pessoa Jurídica não encontrada");
      }
   }
   public ArrayList obterTodos(){
      return pessoasJuridicas;
   }

   public void persistir(String prefixo) throws IOException {
   String arquivoJuridica = prefixo + ".juridica.bin";
   try (FileOutputStream salvaJuridica = new FileOutputStream(new
File(arquivoJuridica));
       ObjectOutputStream oosJuridica = new ObjectOutputStream(salvaJuridica)) {

      oosJuridica.writeObject(pessoasJuridicas);

      System.out.println("Dados de Pessoa Jurídica Armazenados em " +
arquivoJuridica);
   } catch (IOException e) {
      throw new IOException("Erro ao persistir os dados de Pessoa Física: " +
e.getMessage());
   }
}

public void recuperar(String prefixo) throws IOException, ClassNotFoundException {
      String arquivoJuridica = prefixo + ".juridica.bin";
      try (FileInputStream recuperaJuridica = new FileInputStream(arquivoJuridica);
```

```
            ObjectInputStream oisJuridica = new ObjectInputStream(recuperaJuridica)) {

            pessoasJuridicas = (ArrayList<PessoaJuridica>) oisJuridica.readObject();

            System.out.println("Dados de Pessoa Jurídica Recuperados de " +
arquivoJuridica);
        } catch (IOException | ClassNotFoundException e) {
            throw new IOException("Erro ao recuperar os dados de Pessoa Física: " +
e.getMessage());
        }
    }
}
```

[4] Resultados

```
run:
=======================================
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
=======================================
Escolha uma opção: 1
F - Pessoa Física | J - Pessoa Jurídica
f
Digite o ID da pessoa:
12
Digite o nome da pessoa:
Mantis
Digite a idade da pessoa:
34
Digite o cpf da pessoa(XXX.XXX.XXX-XX):
5555555
ID: 12, Nome: Mantis, CPF: 5555555, Idade: 34
Pessoa adicionada com sucesso
=======================================
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
==========
Escolha uma opção: 2
Insira o tipo de pessoa:
f
Digite o id da pessoa que sofrerá alterações:
12
Pessoa encontrada: ID: 12, Nome: Mantis, CPF: 5555555, Idade: 34
Insira os novos dados:
Id: 13
Nome: Nantiz
Idade: 34
CPF: 6666666
Pessoa Física alterada com sucesso
ID: 13, Nome: Nantiz, CPF: 6666666, Idade: 34
==========
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
==========
Escolha uma opção: 4
Insira o tipo de pessoa:
f
Digite o id da pessoa buscada:
13
ID: 13, Nome: Nantiz, CPF: 6666666, Idade: 34
==========
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
==========
Escolha uma opção: 6
Digite o prefixo para os arquivos de persistência: dados

Dados de Pessoa Física Armazenados em dados.fisica.bin
Dados de Pessoa Jurídica Armazenados em dados.juridica.bin
==========
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
==========
Escolha uma opção: 7
Digite o prefixo para os arquivos de recuperação: dados
Dados de Pessoa Física Recuperados de dados.fisica.bin
Dados de Pessoa Jurídica Recuperados de dados.juridica.bin
==========
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
==========
Escolha uma opção: 5
Insira o tipo de pessoa:
f
ID: 13, Nome: Nantiz, CPF: 6666666, Idade: 34
==========
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
==========
Escolha uma opção: 3
Insira o tipo de pessoa:
f
Digite o id da pessoa a ser excluída:
13

```
Pessoa Física excluída com sucesso
=====================================
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
=====================================
Escolha uma opção: 0
BUILD SUCCESSFUL (total time: 1 minute 32 seconds)
```

[5] Análise e Conclusão

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

São elementos que pertencem à classe e, portando não precisam ser instanciados. Por exemplo, uma variável estática de uma classe não será "recopiada" para cada objeto instanciada. Será apenas usada sua forma original. Quanto a métodos, como no caso de main, indica que ele pertence a classe, logo não precisa ser chamado através da instanciação de um objeto.

b. Para que serve a classe Scanner?

Para capturar a entrada de dados do usuário (uma sequência de caracteres qualquer) e separá-la em tipos primitivos (algo essencial em uma linguagem tipada como java) de forma simples e amigável.

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

Com elas, foi possível agrupar sequências de pessoas físicas/jurídicas de forma interna, mais automática e simples. Sem elas, o processo de exibir, salvar, recuperar, entre outros, ficaria a cargo da classe Main, o que tornaria o código mais complexo e "desorganizado".

[link]:https://github.com/duda6134/MissaoPratica1/tree/main/CadastroPOO/build/classes/model