

Informações:

- Nome: Felipe de Souza Komatsu
- Semestre: 3°
- Turma:9001
- Campus: 147 POLO ALCÂNTARA - SÃO GONÇALO - RJ
- Curso: Desenvolvimento Full Stack
- Github: <https://github.com/Felppss>

2° procedimento: Criação do Cadastro em Modo Texto

Objetivos da prática:

Implementação de um cadastro de clientes em modo texto, com persistência em arquivos, baseado na tecnologia Java.

- Utilizar herança e polimorfismo na definição de entidades.
- Utilizar persistência de objetos em arquivos binários.
- Implementar uma interface cadastral em modo texto.
- Utilizar o controle de exceções da plataforma Java.
- 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.

Observação: Neste relatório está apenas incluído as modificações que foram requeridas para o arquivo principal CadastroPOO 2° Procedimento.java

Códigos:

CadastroPOO 2° Procedimento.java

```
/**  
 * @author Felipe komatsu  
 */  
package cadastrapoo;  
  
import java.io.IOException;  
import java.util.List;
```

```

import java.util.Scanner;
import model.entidades.PessoaFisica;
import model.entidades.PessoaJuridica;
import model.gerenciadores.PessoaFisicaRepo;
import model.gerenciadores.PessoaJuridicaRepo;

public class CadastroPOOParte2 {
    public static void main(String[] args) {
        PessoaFisicaRepo repoPF = new PessoaFisicaRepo();
        PessoaJuridicaRepo repoPJ = new PessoaJuridicaRepo();

        Scanner sc = new Scanner(System.in);
        boolean executando = true;

        while (executando) {
            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("=====");
            int opcao = sc.nextInt();

            switch (opcao) {
                case 0:
                    System.out.println("Finalizando programa...");
                    executando = false;
                    break;
                case 1:
                    System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");
                    char tipoPessoaInserir = sc.next().toUpperCase().charAt(0);
                    if (tipoPessoaInserir == 'F') {
                        System.out.println("Digite o ID da Pessoa: ");

```

```

        int id = sc.nextInt();
        sc.nextLine();
        System.out.println("Insira os dados...");
        System.out.println("Nome: ");
        String nome = sc.nextLine();
        System.out.println("CPF: ");
        String cpf = sc.nextLine();
        System.out.println("Idade: ");
        int idade = sc.nextInt();
        repoPF.inserir(new PessoaFisica(id, nome, cpf, idade));

    } else if (tipoPessoaInserir == 'J') {
        System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");
        System.out.println("Digite o ID da Pessoa: ");
        int id = sc.nextInt();
        sc.nextLine();
        System.out.println("Insira os dados...");
        System.out.println("Nome: ");
        String nome = sc.nextLine();
        System.out.println("CNPJ: ");
        String cnpj = sc.nextLine();
        repoPJ.inserir(new PessoaJuridica(id, nome, cnpj));
    } else {
        System.out.println("Erro: Escolha Invalida!");
    }
    break;
case 2:
    System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");
    char tipoPessoaAlterar = sc.next().toUpperCase().charAt(0);
    if (tipoPessoaAlterar == 'F') {
        System.out.println("Digite o ID da Pessoa: ");
        int id = sc.nextInt();
        sc.nextLine();
        PessoaFisica pessoa = repoPF.obter(id);
        if (pessoa != null) {
            System.out.println("Dados atuais:");
            System.out.println("Nome: " + pessoa.getNome());

```

```

        System.out.println("CPF: " + pessoa.getCpf());
        System.out.println("Idade: " + pessoa.getIdade());

        System.out.println("Insira os novos dados:");
        System.out.println("Nome: ");
        String novoNome = sc.nextLine();
        System.out.println("CPF: ");
        String novoCpf = sc.nextLine();
        System.out.println("Idade: ");
        int novaldade = sc.nextInt();

        pessoa.setNome(novoNome);
        pessoa.setCpf(novoCpf);
        pessoa.setIdade(novaldade);

        repoPF.alterar(pessoa);
    }
} else if (tipoPessoaAlterar == 'J') {
    System.out.println("Digite o ID da Pessoa: ");
    int id = sc.nextInt();
    sc.nextLine();
    PessoaJuridica pessoa = repoPJ.obter(id);
    if (pessoa != null) {
        System.out.println("Dados atuais:");
        System.out.println("Nome: " + pessoa.getNome());
        System.out.println("CNPJ: " + pessoa.getCnpj());

        System.out.println("Insira os novos dados:");
        System.out.println("Nome: ");
        String novoNome = sc.nextLine();
        System.out.println("CNPJ: ");
        String novoCnpj = sc.nextLine();

        pessoa.setNome(novoNome);
        pessoa.setCnpj(novoCnpj);

        repoPJ.alterar(pessoa);
    }
}

```

```

    }
} else {
    System.out.println("Erro: Escolha Invalida!");
}
break;

case 3:
    System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");
    char tipoPessoaExcluir = sc.next().toUpperCase().charAt(0);

    switch (tipoPessoaExcluir) {
        case 'F':
            System.out.println("Digite o ID da Pessoa: ");
            int id = sc.nextInt();
            PessoaFisica pessoaFisica = repoPF.obter(id);
            if (pessoaFisica != null) {
                repoPF.excluir(id);
                System.out.println("Pessoa fisica excluida com sucesso!");
            } else {
                System.out.println("Pessoa fisica nao encontrada!");
            }
            break;
        case 'J':
            System.out.println("Digite o ID da Pessoa: ");
            id = sc.nextInt();
            PessoaJuridica pessoaJuridica = repoPJ.obter(id);
            if (pessoaJuridica != null) {
                repoPJ.excluir(id);
                System.out.println("Pessoa juridica excluida com sucesso!");
            } else {
                System.out.println("Pessoa juridica nao encontrada!");
            }
            break;
        default:
            System.out.println("Erro: Escolha Invalida!");
    }
}

```

```
break;
```

case 4:

```
System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");
```

```
char tipoPessoaObter = sc.next().toUpperCase().charAt(0);
```

```
switch (tipoPessoaObter) {
```

```
    case 'F':
```

```
        System.out.println("Digite o ID da pessoa fisica: ");
```

```
        int id = sc.nextInt();
```

```
        PessoaFisica pessoaFisica = repoPF.obter(id);
```

```
        if (pessoaFisica == null) {
```

```
            System.out.println("Pessoa fisica nao encontrada!");
```

```
        } else {
```

```
            System.out.println("Dados da pessoa fisica:");
```

```
            System.out.println(pessoaFisica);
```

```
        }
```

```
        break;
```

```
    case 'J':
```

```
        System.out.println("Digite o ID da pessoa juridica: ");
```

```
        id = sc.nextInt();
```

```
        PessoaJuridica pessoaJuridica = repoPJ.obter(id);
```

```
        if (pessoaJuridica == null) {
```

```
            System.out.println("Pessoa juridica nao encontrada!");
```

```
        } else {
```

```
            System.out.println("Dados da pessoa juridica:");
```

```
            System.out.println(pessoaJuridica);
```

```
        }
```

```
        break;
```

```
    default:
```

```
        System.out.println("Erro: Escolha invalida!");
```

```
}
```

```
break;
```

case 5:

```
System.out.println("F - Pessoas Fisicas | J - Pessoas Juridicas");
```

```
char tipoPessoaObterTodos = sc.next().toUpperCase().charAt(0);
```

```

if (tipoPessoaObterTodos == 'F') {
    System.out.println("Pessoas Fisicas:");
    List<PessoaFisica> pessoasFisicas = repoPF.obterTodos();

    if (pessoasFisicas.isEmpty()) {
        System.out.println("Nao existem pessoas fisicas cadastradas no sistema.");
    } else {
        for (PessoaFisica pf : pessoasFisicas) {
            System.out.println(pf.toString());
        }
    }
} else if (tipoPessoaObterTodos == 'J') {
    System.out.println("Pessoas Juridicas:");
    List<PessoaJuridica> pessoasJuridicas = repoPJ.obterTodos();

    if (pessoasJuridicas.isEmpty()) {
        System.out.println("Nao existem pessoas juridicas cadastradas no sistema.");
    } else {
        for (PessoaJuridica pj : pessoasJuridicas) {
            System.out.println(pj.toString());
        }
    }
} else {
    System.out.println("Erro: Escolha invalida!");
}
break;

```

case 6:

```

System.out.println("Salvar Dados");
sc.nextLine();
System.out.print("Informe o prefixo dos arquivos: ");
String prefixo = sc.nextLine();

try {
    repoPF.persistir(prefixo + ".fisica.bin");
    repoPJ.persistir(prefixo + ".juridica.bin");
}

```

```

    } catch (IOException e) {
        System.err.println("Erro ao salvar os dados: " + e.getMessage());
    }
    break;

case 7:
    System.out.println("Recuperar Dados");
    sc.nextLine();
    System.out.print("Informe o prefixo dos arquivos: ");
    String prefixoRec = sc.nextLine();

    try {
        repoPF.recuperar(prefixoRec + ".fisica.bin");
        repoPJ.recuperar(prefixoRec + ".juridica.bin");
    } catch (IOException | ClassNotFoundException e) {
        System.err.println("Erro ao recuperar os dados: " + e.getMessage());
    }
    break;

default:
    System.out.println("Erro: Escolha invalida!");
}
}

sc.close();
}
}

```

Resultado da Execução:

Output - CadastroPOO (run) #23

```

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:
190
Insira os dados...
Nome:
Camilla
CPF:
10000000
Idade:
24
=====
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
Digite o ID da Pessoa:
190
Dados atuais:
Nome: Camilla
CPF: 10000000
Idade: 24
Insira os novos dados:
Nome:
Camilla
CPF:
100000
Idade:
25
Pessoa Fisica alterada 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
=====
|

```

Nesta imagem, foram testados os métodos **Incluir** e **Alterar**.

```

=====
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
Digite o ID da pessoa fisica:
190
Dados da pessoa fisica:
ID = 190, Nome = Camilla, CPF = 100000, Idade = 25
=====
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
Digite o ID da Pessoa:
190
Pessoa fisica excluida 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
=====
0
Finalizando programa...
BUILD SUCCESSFUL (total time: 2 minutes 49 seconds)

```


Nesta imagem, foram testados os métodos **Buscar pelo Id**, **Excluir** e **Finalizar Programa**.

```

=====
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 - Pessoas Fisicas | J - Pessoas Juridicas
f
Pessoas Fisicas:
ID = 100, Nome = Alice, CPF = 10000000, Idade = 40
ID = 200, Nome = Bianca, CPF = 20000000, Idade = 25
=====
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 - Pessoas Fisicas | J - Pessoas Juridicas
j
Pessoas Juridicas:
ID = 300, Nome = Carlos, CNPJ = 30000000
ID = 400, Nome = Daniel, CNPJ = 40000000
=====
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
=====

```

Nesta imagem, foi testado o método **Exibir Todos**.



```
=====
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
Salvar Dados
Informe o prefixo dos arquivos: teste
Dados de Pessoa Fisica Armazenados.
Dados de Pessoa Juridica Armazenados.
=====

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
Recuperar Dados
Informe o prefixo dos arquivos: teste
Dados de Pessoa Fisica Recuperados
ID: 100
Nome: Alice
CPF: 10000000
Idade: 40
ID: 200
Nome: Bianca
CPF: 20000000
Idade: 25
Dados de Pessoa Juridica Recuperados:
ID: 300
Nome: Carlos
CNPJ: 30000000
ID: 400
Nome: Daniel
CNPJ: 40000000
=====

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
=====
.
```

Nesta imagem, foram testados os métodos **Persistir Dados** e **Recuperar Dados**.

Análise e Conclusão:

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

Elementos estáticos em Java são aqueles que pertencem à classe em si, em vez de pertencerem a instâncias específicas da classe. Isso significa que eles podem ser acessados sem a necessidade de criar um objeto da classe. Exemplos de elementos estáticos são métodos e variáveis estáticas.

O método main em Java é declarado como estático para que possa ser invocado sem a necessidade de instanciar a classe.

Para que serve a classe Scanner?

A classe Scanner em Java permite a leitura de diferentes tipos de dados diretamente do teclado, facilitando a interação com o usuário.

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

O uso de classes de repositório simplifica a organização do código, separando as operações de acesso aos dados das operações de negócios. Isso torna o código mais limpo e fácil de entender.