



{ **COTI INFORMÁTICA** }
ESCOLA DE NERDS



Java WebDeveloper - Aula 04a - Noite Seg, Qua, Sex - Noite



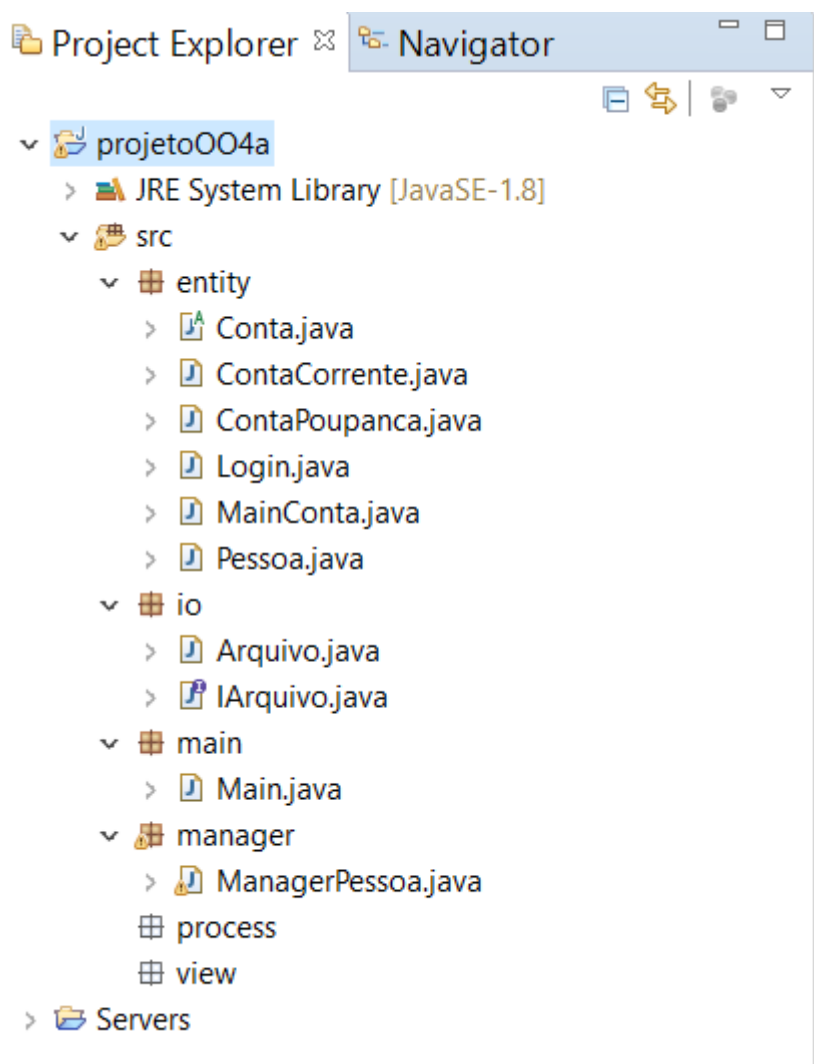
Tema da aula:

Orientação a Objetos, Interface, Gravação e leitura de arquivo TXT, Classe abstrata, Metodos abstratos, Calculo, Comparable, Ordenação, Lista, Filter, Map, Collect

— Prof Edson Belém - profedsonbelem@gmail.com (mailto:profedsonbelem@gmail.com) Segunda, Mar 30, 2020



Estrutura do projeto depois de finalizado:





Login.java

```
1 package entity;
2
3 public class Login {
4
5     private Integer id;
6     private String login;
7     private String senha;
8     private String nome;
9
10    public Login() {
11    }
12
13    public Login(Integer id, String login, String senha, String nome) {
14        this.id = id;
15        this.login = login;
16        this.senha = senha;
17        this.nome = nome;
18    }
19
20    public String toString() {
21        // posicao 4 (posicoes) 0 a 3
22        return this.id + "," + this.login + "," + this.nome
23            + "," + this.senha;
24    }
25
26    public Integer getId() {
27        return id;
28    }
29
30    public void setId(Integer id) {
31        this.id = id;
32    }
33
34    public String getLogin() {
35        return login;
36    }
37
38    public void setLogin(String login) {
39        this.login = login;
40    }
41
42    public String getSenha() {
43        return senha;
44    }
45
46    public void setSenha(String senha) {
47        this.senha = senha;
48    }
49
50    public String getNome() {
51        return nome;
52    }
53
54    public void setNome(String nome) {
```

```
55         this.nome = nome;
56     }
57
58 }
```



IArquivo.java

```
1  package io;
2
3  import entity.Login;
4
5  public interface IArquivo {
6
7      // Virtual (abstrato)
8      public void registrar(Login lg) throws Exception;
9
10     // Virtual (abstrato)
11     public String logar() throws Exception;
12
13 }
14
15 //Classe que irá Implementar a Interface ...
```



Arquivo.java

```
1 package io;
2
3 import java.io.BufferedReader;
4 import java.io.FileReader;
5 import java.io.FileWriter;
6
7 import entity.Login;
8
9 //Arquivo é uma Interface
10 //is a
11 //A Classe implementa A ação da regra de Negocio ...
12
13 public class Arquivo implements IArquivo {
14
15     FileWriter fw;
16
17     @Override
18     public void registrar(Login lg) throws Exception {
19         // c:\temp\dados.txt (false)
20         //ATENÇÃO PARA O CAMINHO NO WINDOWS
21         fw = new FileWriter("/tmp/dados.txt", false);
22         fw.write(lg.toString());
23         fw.flush();
24         fw.close();
25     }
26
27     FileReader fr;
28
29     @Override
30     public String logar() throws Exception {
31         fr = new FileReader("/tmp/dados.txt");
32         BufferedReader bf = new BufferedReader(fr);
33         String linha = "";
34         if ((linha = bf.readLine()) != null);
35         return linha;
36     }
37 }
```



Main.java

```

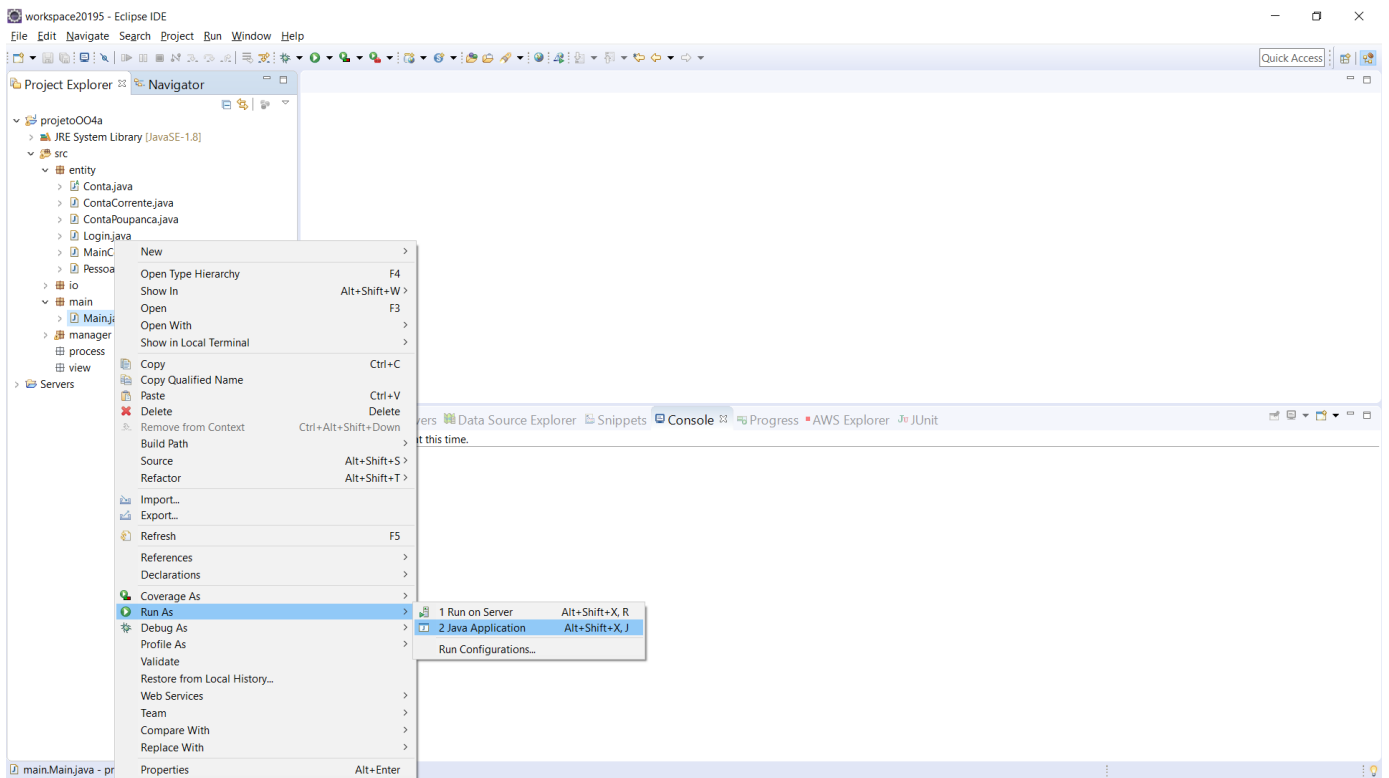
1  package main;
2
3  import entity.Login;
4  import io.Arquivo;
5  import io.IArquivo;
6
7  public class Main {
8
9      public static void main(String[] args) {
10
11          Login l = new Login(11, "jose@gmail.com", "123", "jose");
12          try {
13              IArquivo arq = new Arquivo();
14              arq.registrar(l);
15              System.out.println("Dados Ok ...");
16
17              String dad = "casa com pipoca e assistindo java ";
18              String divisao[] = dad.split("e");
19
20              System.out.println("=====");
21              System.out.println(divisao[0]);
22              System.out.println(divisao[1]);
23
24              String dados[] = arq.logar().split(",");
25
26              Login loga = new Login();
27
28              loga.setLogin("belem@gmail.com");
29              loga.setSenha("123");
30
31              System.out.println("comparando login:" +
32                  loga.getLogin().equals(dados[1]));
33              System.out.println("comprando senha:" +
34                  loga.getSenha().equals(dados[3]));
35
36          } catch (Exception ex) {
37              ex.printStackTrace();
38          }
39      }
40  }

```

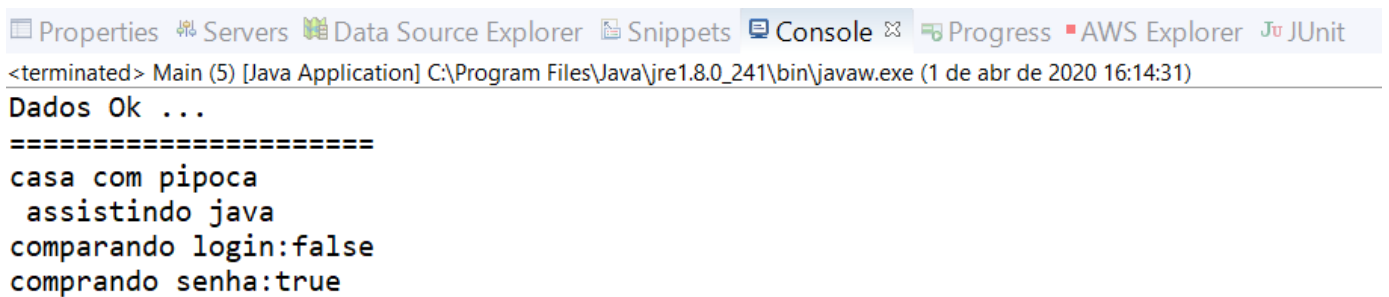


Para rodar a classe:

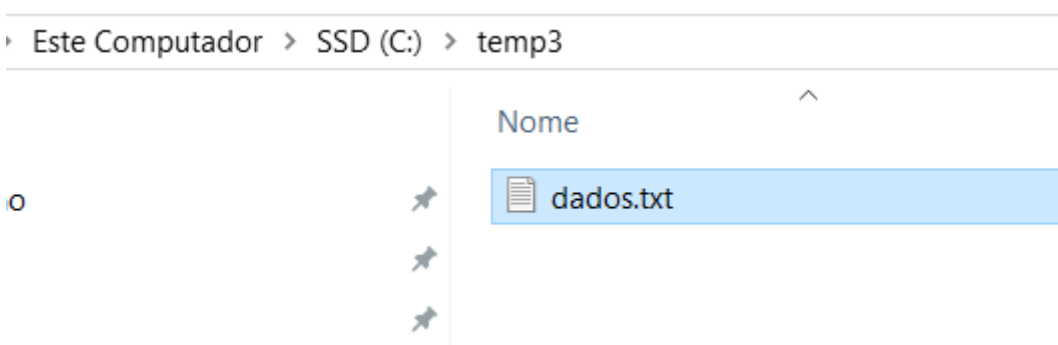
Clique na classe com o botão direito ➡ run as ➡ java application



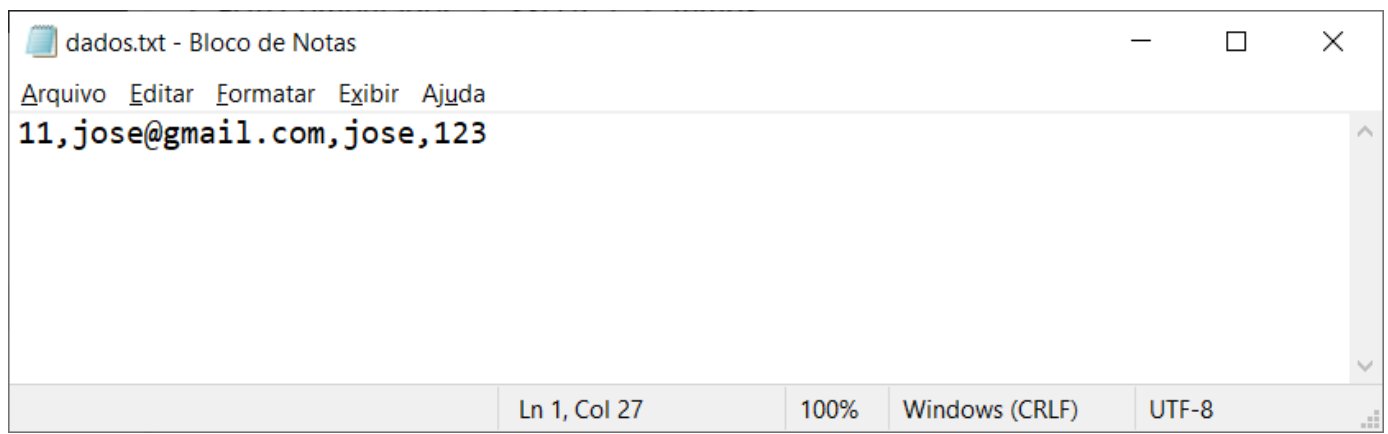
Resultado no console:



No explorer



Abrindo o arquivo



☕ Conta.java


```
1 package entity;
2
3 import java.text.SimpleDateFormat;
4 import java.util.Date;
5
6 public abstract class Conta {
7
8     // As RN estão nos metodos abstratos (adicionar os dados mesmo)
9     private String nome;
10    private Double saldo = 0.;
11    private Double deposito = 0.;
12    private Double retirada = 0.;
13    private Date dataHora;
14
15    // serao duas contas (poupanca) depositar (-10)...
16    // sobrescrita
17    // polimorfismo ...
18
19    public abstract void gerarDeposito(); // serão feitos na heranca
20    // serao duas contas (poupanca) depositar (-20)...
21
22    public abstract void gerarRetirada();
23
24    @Override
25    public String toString() {
26        return "Conta [nome=" + nome + ", saldo=" + saldo +
27            ", deposito=" + deposito + ", retirada=" + getRetirada() +
28            ", dataHora=" + new SimpleDateFormat("dd/MM/yyyy HH:mm:ss")
29            .format(dataHora) + "]\n";
30    }
31
32    public String getNome() {
33        return nome;
34    }
35
36    public void setNome(String nome) {
37        this.nome = nome;
38    }
39
40    public Double getSaldo() {
41        return saldo;
42    }
43
44    public void setSaldo(Double saldo) {
45        this.saldo = saldo;
46    }
47
48    public Double getDeposito() {
49        return deposito;
50    }
51
52    public void setDeposito(Double deposito) {
53        this.deposito = deposito;
54    }
```

```

55
56     public Date getDataHora() {
57         return dataHora;
58     }
59
60     public void setDataHora(Date dataHora) {
61         this.dataHora = dataHora;
62     }
63
64     public Double getRetirada() {
65         return retirada;
66     }
67
68     public void setRetirada(Double retirada) {
69         this.retirada = retirada;
70     }
71 }

```



ContaCorrente.java

```

1  package entity;
2
3  public class ContaCorrente extends Conta {
4
5      // Quando estou programando em ContaCorrente estou sobrescrevendo
6      // gerarDeposito da Classe abstrata
7      // por mudar o formato da atitude é um Polimorfismo ...
8      @Override
9      public void gerarDeposito() {
10         setSaldo(getSaldo() + getDeposito());
11     }
12
13     @Override
14     public void gerarRetirada() {
15         if (getRetirada() <= getSaldo()) {
16             setSaldo(getSaldo() - getRetirada());
17         } else {
18             throw new IllegalArgumentException
19                 ("Saldo insuficiente ...");
20         }
21     }
22 }

```



ContaPoupanca.java

```
1  package entity;
2
3  public class ContaPoupanca extends Conta {
4
5      @Override
6      public void gerarDeposito() {
7          setSaldo(getSaldo() + getDeposito() - 10.);
8      }
9
10     @Override
11     public void gerarRetirada() {
12         if (getRetirada() <= (getSaldo() + 20)) {
13             setSaldo(getSaldo() - getRetirada() - 20.);
14         } else {
15             throw new IllegalArgumentException
16                 ("Saldo insuficiente ...");
17         }
18     }
19 }
```




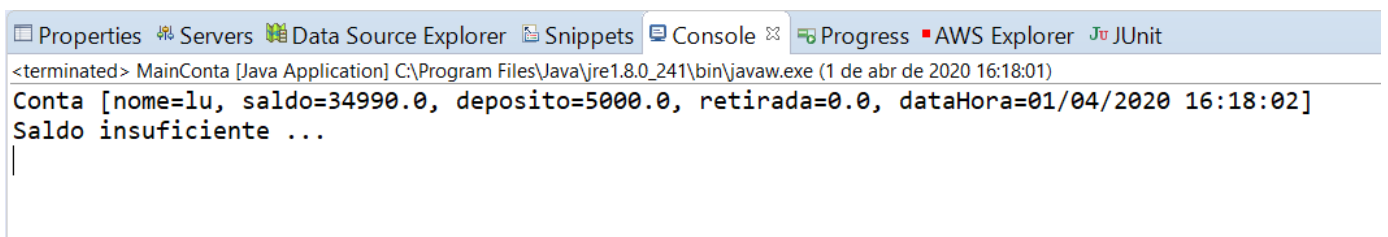
MainConta.java

```

1  package entity;
2
3  import java.util.Date;
4
5  public class MainConta {
6
7      public static void main(String[] args) {
8          // Nao estou dando new na Classe Abstrata
9          // estou dando new para criar o Vetor ....
10
11         Conta c[] = new Conta[2];
12         try {
13             c[0] = new ContaPoupanca();
14             // new () (cria espaço de mem)
15             c[0].setNome("lu");
16             c[0].setDataHora(new Date());
17             c[0].setSaldo(30000.);
18             c[0].setDeposito(5000.);
19             c[0].gerarDeposito();
20
21             System.out.println(c[0]);
22
23             c[1] = new ContaCorrente();
24             c[1].setNome("rafael");
25             c[1].setDataHora(new Date());
26             c[1].setSaldo(1000.);
27             c[1].setDeposito(0.);
28             c[1].setRetirada(2000.);
29
30             c[1].gerarRetirada();
31
32             System.out.println(c[1]);
33
34         } catch (Exception ex) {
35             System.out.println(ex.getMessage());
36         }
37     }
38 }

```

 Resultado no console:



```

<terminated> MainConta [Java Application] C:\Program Files\Java\jre1.8.0_241\bin\javaw.exe (1 de abr de 2020 16:18:01)
Conta [nome=lu, saldo=34990.0, deposito=5000.0, retirada=0.0, dataHora=01/04/2020 16:18:02]
Saldo insuficiente ...

```


 MainConta.java

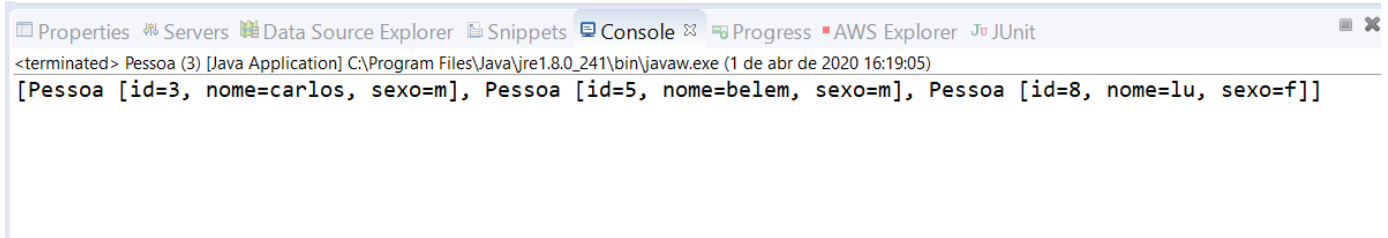
```
1  package entity;
2
3  import java.util.ArrayList;
4  import java.util.Arrays;
5  import java.util.Collections;
6  import java.util.List;
7
8  //ordenar (compareTo)
9  public class Pessoa implements Comparable<Pessoa> {
10
11      private Integer id;
12      private String nome;
13      private String sexo;
14
15      public Pessoa() {
16
17      }
18
19      public Pessoa(Integer id, String nome, String sexo) {
20          this.id = id;
21          this.nome = nome;
22          this.sexo = sexo;
23      }
24
25      @Override
26      public String toString() {
27          return "Pessoa [id=" + id + ", nome=" + nome + ", sexo="
28              + sexo + "]";
29      }
30
31      public Integer getId() {
32          return id;
33      }
34
35      public void setId(Integer id) {
36          this.id = id;
37      }
38
39      public String getNome() {
40          return nome;
41      }
42
43      public void setNome(String nome) {
44          this.nome = nome;
45      }
46
47      public String getSexo() {
48          return sexo;
49      }
50
51      public void setSexo(String sexo) {
52          this.sexo = sexo;
53      }
54
```

```

55         // Criterio de Ordenacao aqui default pelo Id...
56         @Override
57         public int compareTo(Pessoa p) {
58             // ordenando id com getId() ... Ascendente
59             return this.id.compareTo(p.getId());
60         }
61
62         public static void main(String[] args) {
63
64             // exemplo ....
65             List<Pessoa> lista = new ArrayList<Pessoa>();
66
67             Pessoa p1 = new Pessoa(5, "belem", "m");
68             Pessoa p2 = new Pessoa(3, "carlos", "m");
69             Pessoa p3 = new Pessoa(8, "lu", "f");
70
71             lista.addAll(Arrays.asList(p1, p2, p3));
72             // lista.add(p1); lista.add(p2); lista.add(p3);
73
74             Collections.sort(lista);
75
76             System.out.println(lista);
77         }
78     }

```

 Resultado no console:



```

<terminated> Pessoa (3) [Java Application] C:\Program Files\Java\jre1.8.0_241\bin\javaw.exe (1 de abr de 2020 16:19:05)
[Pessoa [id=3, nome=carlos, sexo=m], Pessoa [id=5, nome=belem, sexo=m], Pessoa [id=8, nome=lu, sexo=f]]

```



ManagerPessoa.java

```

1  package manager;
2
3  import java.util.ArrayList;
4  import java.util.Collections;
5  import java.util.Comparator;
6  import java.util.List;
7  import java.util.stream.Collectors;
8
9  import entity.Pessoa;
10
11 public class ManagerPessoa {
12
13     public static List<Pessoa> pessoas = new ArrayList<Pessoa>();
14
15     static {
16         pessoas.add(new Pessoa(10, "lu", "f"));
17         pessoas.add(new Pessoa(5, "rafa", "m"));
18         pessoas.add(new Pessoa(8, "carlos", "m"));
19     }
20
21     public void ordenarNome() {
22         Comparator<Pessoa> comp = (a, b) -> a.getNome()
23             .compareTo(b.getNome());
24         Collections.sort(pessoas, comp);
25     }
26
27     public void atualizarNome() {
28         String pessoa = pessoas.stream().filter(res -> res.getId()
29             .equals(10)).map(rs -> "" + rs.getId() + "," + "luciana" +
30             "," + rs.getSexo()).collect(Collectors.toList()).get(0);
31
32         pessoas.set(1, new Pessoa(10, "luciana", "f"));
33
34         System.out.println("\n" + pessoa);
35         System.out.println("\n" + pessoas);
36     }
37
38     public static void main(String[] args) {
39         ManagerPessoa mb = new ManagerPessoa();
40         mb.ordenarNome();
41         System.out.println(mb.pessoas);
42
43         mb.atualizarNome();
44     }
45 }

```

 Resultado no console:

<terminated> ManagerPessoa [Java Application] C:\Program Files\Java\jre1.8.0_241\bin\javaw.exe (1 de abr de 2020 16:19:38)

[Pessoa [id=8, nome=carlos, sexo=m], Pessoa [id=10, nome=lu, sexo=f], Pessoa [id=5, nome=rafa, sexo=m]]

10,luciana,f

[Pessoa [id=8, nome=carlos, sexo=m], Pessoa [id=10, nome=luciana, sexo=f], Pessoa [id=5, nome=rafa, sexo=m]]

— Coti Informática <https://www.cotiinformatica.com.br> (<https://www.cotiinformatica.com.br>)