

## MVC: ARRAYLIS, FICHEIRO DE TEXTOS, E ORDENACAO COM COLLECTIONS.SORT

### 1. Estuante

```
1 public class Estudante implements Comparable<Estudante> {
2     private int codigo;
3     private String nome;
4     private double test1;
5     private double test2;
6
7     public Estudante(int codigo, String nome, double test1, double test2) {
8         this.codigo = codigo;
9         this.nome = nome;
10        this.test1 = test1;
11        this.test2 = test2;
12    }
13    public int getCodigo() {return codigo;}
14
15    public void setCodigo(int codigo) {this.codigo = codigo;}
16
17    public String getNome() {return nome;}
18
19    public void setNome(String nome) {this.nome = nome;}
20
21    public double getTest1() {return test1;}
22
23    public void setTest1(double test1) {this.test1 = test1;}
24
25    public double getTest2() {return test2;}
26
27    public void setTest2(double test2) {this.test2 = test2;}
28
29    public double calculaMedia() {return (test1 + test2)/2;}
30
31    public String toString() {
32        return codigo + "-" + nome + "-" + test1 + "-" + test2 + "-" + calculaMedia();
33    }
34    public int compareTo(Estudante estudante) {
35        if(this.codigo>estudante.codigo) return 1;
36        if(this.codigo<estudante.codigo) return -1;
37        return 0;
38    }
39 }
```

### 2. ControllerEstudante

```
1 import java.io.BufferedReader;
2
3
4
5
6
7
8
9
10
11 public class ControllerEstudante {
12     static Scanner k = new Scanner(System.in);
13     public static void adicionarEstudante() throws IOException {
14
15         FileWriter criar = new FileWriter("notas.txt", true);
16         BufferedWriter adicionar = new BufferedWriter(criar);
17         System.out.print("Codigo: ");
18         int codigo = k.nextInt();
19         System.out.print("Nome: ");
20         String nome = k.next();
21         System.out.print("Teste 1: ");
22         double test1 = k.nextDouble();
23         System.out.print("Teste 2: ");
24         double test2 = k.nextDouble();
25         Estudante estudante = new Estudante(codigo, nome, test1, test2);
26         adicionar.write(estudante.toString());
27         adicionar.newLine();
28
29         adicionar.close();
30         criar.close();
31         System.out.println("ADICIONADO COM SUCESSO:");
32     }
33 }
```

```

34 public static ArrayList<Estudante> listaDeEstudantes() throws IOException {
35     ArrayList<Estudante> estudantes = new ArrayList<Estudante>();
36     FileReader abrir = new FileReader("notas.txt");
37     BufferedReader ler = new BufferedReader(abrir);
38     String linha = ler.readLine();
39
40     String[] elementosDaLinha;
41
42     while(linha != null && !linha.isEmpty()) {
43         elementosDaLinha = linha.split("-");
44         int codigo = Integer.parseInt(elementosDaLinha[0]);
45         String nome = elementosDaLinha[1];
46         double test1 = Double.parseDouble(elementosDaLinha[2]);
47         double test2 = Double.parseDouble(elementosDaLinha[3]);
48         estudantes.add(new Estudante(codigo, nome, test1, test2));
49         linha=ler.readLine();
50     }
51
52     ler.close();
53     abrir.close();
54     return estudantes;
55 }
56
57 public static void listarEstudantes()throws IOException {
58     ArrayList<Estudante> estudantes=listaDeEstudantes();
59     for (int i = 0; i < estudantes.size(); i++) {
60         System.out.println(estudantes.get(i).toString());
61     }
62 }

```

```

63 static void atualizarEstudante(int codigo) throws IOException {
64     ArrayList<Estudante> estudantes=listaDeEstudantes();
65     char opcao;
66     for (Estudante estudante : estudantes) {
67         if(estudante.getCodigo() == codigo) {
68             do{
69                 System.out.println("MENU DE ATUALIZACAO DO ESTUDANTE");
70                 System.out.println("1-Nome:");
71                 System.out.println("2-Teste1:");
72                 System.out.println("3-Teste2:");
73                 System.out.println("0-Sair:");
74                 opcao=k.next().charAt(0);
75                 switch (opcao) {
76                     case '1':
77                     System.out.println("Nome:");
78                     String nome = k.next();
79                     estudante.setNome(nome);
80                     estudante.setTest1(estudante.getTest1());
81                     estudante.setTest2(estudante.getTest2());
82                     break;
83                     case '2':
84                     System.out.println("Teste 1:");
85                     double teste1= k.nextDouble();
86                     estudante.setTest1(teste1);
87                     estudante.setNome(estudante.getNome());
88                     estudante.setTest2(estudante.getTest2());
89                     break;

```

```

90         case '3':
91             System.out.println("Teste 2:");
92             double teste2= k.nextDouble();
93             estudante.setTest2(teste2);
94             estudante.setNome(estudante.getNome());
95             estudante.setTest1(estudante.getTest1());
96             break;
97         case '0':
98             System.out.println("Saiu do menu de atualizacao");
99             break;
100         default:
101             System.out.println("Opcao invalida:");
102             break;
103     }
104 }
105 }while(opcao!='0');
106 }
107 }
108     FileWriter criar = new FileWriter("notas.txt");
109     BufferedWriter adicionar = new BufferedWriter(criar);
110     for (int i=0;i<estudantes.size();i++) {
111         adicionar.write(estudantes.get(i).toString());
112         adicionar.newLine();
113     }
114     adicionar.close();
115     criar.close();
116     System.out.println("ACTUALIZADO COM SUCESSO:");
117 }

119 public static void removerEstudante(int codigo) throws IOException {
120     ArrayList<Estudante> estudantes=ListaDeEstudantes();
121     for (int i=0;i<estudantes.size();i++) {
122         if(estudantes.get(i).getCodigo() == codigo) {
123             estudantes.remove(estudantes.get(i));
124         }
125     }
126
127     FileWriter criar = new FileWriter("notas.txt");
128     BufferedWriter adicionar = new BufferedWriter(criar);
129     for (int i=0;i<estudantes.size();i++) {
130         adicionar.write(estudantes.get(i).toString());
131         adicionar.newLine();
132     }
133     adicionar.close();
134     criar.close();
135     System.out.println("REMOVIDO COM SUCESSO:");
136 }

137 public static void ordenarEstudantes() throws IOException{
138     ArrayList<Estudante> estudantes=ListaDeEstudantes();
139     Collections.sort(estudantes);
140     FileWriter criar = new FileWriter("notas.txt");
141     BufferedWriter adicionar = new BufferedWriter(criar);
142     for (int i=0;i<estudantes.size();i++) {
143         adicionar.write(estudantes.get(i).toString());
144         adicionar.newLine();
145     }
146     adicionar.close();
147     criar.close();
148     System.out.println("ESTUDANSTES ORDENADOS COM SUCESSO:");
149 }
150 }
151 }
152 }

```

### 3. ViewEstudante

```
1*import java.io.IOException;
4 public class ViewEstudante {
5     public static void main(String[] args) throws IOException {
6         Scanner k = new Scanner(System.in);
7         int opcao;
8         do {
9             System.out.println("MENU DE PRINCIPAL DO SISTEMA");
10            System.out.println("1 - Adicionar Estudante: ");
11            System.out.println("2 - Listar Estudantes: ");
12            System.out.println("3 - Atualizar Estudante: ");
13            System.out.println("4 - Remover Estudante: ");
14            System.out.println("5 - Listar Estudantes Ordenados: ");
15            System.out.println("0 - Sair: ");
16            System.out.print("Opcao: "); opcao = k.nextInt();
17            switch (opcao) {
18                case 1:
19                    ControllerEstudante.adicionarEstudante();
20                    break;
21                case 2:
22                    ControllerEstudante.listarEstudantes();
23                    break;
24                case 3:
25                    System.out.print("Digite um codigo de estudante que existe na lista: "); int codigoAtualizacao = k.next
26                    ControllerEstudante.atualizarEstudante(codigoAtualizacao);
27                    break;
28                case 4:
29                    System.out.print("Digite um codigo de estudante que existe na lista: ");
30                    int codigoRemocao = k.nextInt();
31                    ControllerEstudante.removerEstudante(codigoRemocao);
32                    break;
33
34                case 5:
35                    ControllerEstudante.ordenarEstudantes();
36                    break;
37                case 0:
38                    System.out.println("Saiu do menu principal");
39                    break;
40                default:
41                    System.out.println("Opcao invalida");
42                    break;
43            } while(opcao != 0);
44        }
45    }
```