

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

1. Estuante

```
1 import java.io.Serializable;
2 public class Estudante implements Comparable<Estudante>, Serializable{
3     private int codigo;
4     private String nome;
5     private double test1;
6     private double test2;
7
8     public Estudante(int codigo, String nome, double test1, double test2) {
9         this.codigo = codigo;
10        this.nome = nome;
11        this.test1 = test1;
12        this.test2 = test2;
13    }
14    public int getCodigo() {
15        return codigo;
16    }
17    public void setCodigo(int codigo) {
18        this.codigo = codigo;
19    }
20    public String getNome() {
21        return nome;
22    }
23    public void setNome(String nome) {
24        this.nome = nome;
25    }
26    public double getTest1() {
27        return test1;
28    }
29    public void setTest1(double test1) {
30        this.test1 = test1;
31    }
32    public double getTest2() {
33        return test2;
34    }
35
36    public void setTest2(double test2) {
37        this.test2 = test2;
38    }
39    public double calculaMedia() {
40        return (test1 + test2)/2;
41    }
42    public int compareTo(Estudante estudante) {
43        if(this.codigo>estudante.codigo)return 1;
44        if(this.codigo<estudante.codigo)return -1;
45        return 0;
46    }
47    public String toString() {
48        return codigo + "-" + nome + "-" + test1 + "-" + test2 + "-" + calculaMedia();
49    }
50 }
```

2. ControllerEstudante

```
1 import java.io.FileInputStream;
2
3 public class ControllerEstudante {
4     static Scanner k=new Scanner(System.in);
5     public static ArrayList<Estudante> listaDeEstudantes(){
6         ArrayList<Estudante> estudantes=new ArrayList<Estudante>();
7
8         try{
9             FileInputStream abrir=new FileInputStream("estudantes.bin");
10            ObjectInputStream ler=new ObjectInputStream(abrir);
11
12            for(int i=0;i<100;i++){
13                estudantes.add((Estudante)ler.readObject());
14            }
15            ler.close();
16            abrir.close();
17        }
18        catch(Exception e){
19            System.out.println(e.getMessage());
20        }
21
22        return estudantes;
23    }
24 }
```

```

31 public static void adicionarEstudante() throws Exception{
32     ArrayList<Estudante> estudantes=ListaDeEstudantes();
33
34     FileOutputStream criar=new FileOutputStream("estudantes.bin");
35     ObjectOutputStream escrever=new ObjectOutputStream(criar);
36
37     int i=0;
38
39     while(i<estudantes.size()){
40         if(estudantes.get(i)!=null){
41             escrever.writeObject(estudantes.get(i));
42             i++;
43         }
44     }
45     System.out.print("Codigo:");int codigo=k.nextInt();
46     System.out.print("Nome:");String nome=k.next();
47     System.out.print("Teste1:");double teste1=k.nextDouble();
48     System.out.print("Teste2:");double teste2=k.nextDouble();
49
50     escrever.writeObject(new Estudante(codigo, nome, teste1, teste2));
51
52     escrever.close();
53     criar.close();
54     System.out.println("ESTUDANTE ADICIONADO COM SUCESSO:");
55
56 public static void listarEstudantes() throws Exception{
57     ArrayList<Estudante>estudantes=ListaDeEstudantes();
58
59
60     int i=0;
61     while(i<estudantes.size()){
62         if(estudantes.get(i)!=null){
63             System.out.println(estudantes.get(i).toString());
64             i++;
65         }
66     }
67
68 public static void atualizarEstudante(int codigo) throws Exception {
69
70     ArrayList<Estudante> estudantes=ListaDeEstudantes();
71     char opcao;
72     for (Estudante estudante : estudantes) {
73         if((estudante!=null) && (estudante.getCodigo() == codigo)) {
74
75             do{
76                 System.out.println("MENU DE ATUALIZACAO DO ESTUDANTE");
77                 System.out.println("1-Nome: ");
78                 System.out.println("2-Teste1: ");
79                 System.out.println("3-Teste2: ");
80                 System.out.println("0-Sair: ");
81                 System.out.println("Digite uma opcao: ");
82                 opcao=k.next().charAt(0);
83                 switch (opcao) {
84                     case '1':
85                     System.out.println("Nome:");
86                     String nome = k.next();
87                     estudante.setNome(nome);
88                     estudante.setTest1(estudante.getTest1());
89                     estudante.setTest2(estudante.getTest2());
90                     break;
91                     case '2':
92                     System.out.println("Teste 1:");
93                     double teste1= k.nextDouble();
94                     estudante.setTest1(teste1);
95                     estudante.setNome(estudante.getNome());
96                     estudante.setTest2(estudante.getTest2());
97                     break;

```

```

98         case '3':
99             System.out.println("Teste 2:");
100             double teste2= k.nextDouble();
101             estudante.setTest2(teste2);
102             estudante.setNome(estudante.getNome());
103             estudante.setTest1(estudante.getTest1());
104             break;
105         case '0':
106             System.out.println("Saiu do menu de atualizacao");
107             break;
108         default:
109             System.out.println("Opcao invalida:");
110             break;
111     }
112     }while(opcao!='0');
113 }
114 }
115 FileOutputStream criar=new FileOutputStream("estudantes.bin");
116 ObjectOutputStream escrever=new ObjectOutputStream(criar);
117
118 int i=0;
119 while (i<estudantes.size()) {
120     if(estudantes.get(i)!=null){
121         escrever.writeObject(estudantes.get(i));
122         i++;
123     }
124 }
125 escrever.close();
126 criar.close();
127 System.out.println("ESTUDANTE ATUALIZADO COM SUCESSO:");
128 }

```

```

129
130 public static void removerEstudante(int codigo) throws Exception {
131
132     ArrayList<Estudante> estudantes=ListaDeEstudantes();
133
134     for (int i=0;i<estudantes.size();i++) {
135         if((estudantes.get(i)!=null) && (estudantes.get(i).getCodigo() == codigo)){
136             estudantes.remove(estudantes.get(i));
137         }
138     }
139     FileOutputStream criar=new FileOutputStream("estudantes.bin");
140     ObjectOutputStream escrever=new ObjectOutputStream(criar);
141
142     int i=0;
143     while (i<estudantes.size()) {
144         if(estudantes.get(i)!=null){
145             escrever.writeObject(estudantes.get(i));
146             i++;
147         }
148     }
149     escrever.close();
150     criar.close();
151     System.out.println("ESTUDANTE REMOVIDO COM SUCESSO:");
152 }

```

```

153 public static void ordenarEstudantes () throws IOException{
154     ArrayList<Estudante> estudantes=ListaDeEstudantes();
155     Collections.sort(estudantes);
156     FileOutputStream criar=new FileOutputStream("estudantes.bin");
157     ObjectOutputStream escrever=new ObjectOutputStream(criar);
158
159     int i=0;
160     while (i<estudantes.size()) {
161         if(estudantes.get(i)!=null){
162             escrever.writeObject(estudantes.get(i));
163             i++;
164         }
165     }
166     escrever.close();
167     criar.close();
168     System.out.println("ESTUDANTES ORDENADOS COM SUCESSO:");
169
170 }
171 }

```

3. ViewEstudante

```
1 import java.util.Scanner;
2 public class ViewEstudante {
3     static Scanner k=new Scanner(System.in);
4     public static void main(String[] args) throws Exception {
5         int opcao;
6         do{
7             System.out.println("+++MENU PRINCIPAL+++");
8             System.out.println("1-Adicionar Estudante: ");
9             System.out.println("2-Listar Estudante: ");
10            System.out.println("3-Atualizar Estudante: ");
11            System.out.println("4-Remover Estudante: ");
12            System.out.println("5-Ordenar Estudantes: ");
13            System.out.println("0-Sair: ");
14            System.out.print("Digite uma opcao: ");
15            opcao=k.nextInt();
16
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.println("Digite um codigo que existe na lista: ");
26                    int codigoAtualizacao=k.nextInt();
27                    ControllerEstudante.atualizarEstudante(codigoAtualizacao);
28                    break;
29                case 4:
30                    System.out.println("Digite um codigo que existe na lista: ");
31                    int codigoRemocao=k.nextInt();
32                    ControllerEstudante.removerEstudante(codigoRemocao);
33                    break;
34
35                case 5:
36                    ControllerEstudante.ordenarEstudantes();
37                    break;
38                case 0:
39                    System.out.println("SAIU DO MENU PRINCIPAL");
40                    break;
41                default:
42                    System.out.println("OPCAO INVALIDA");
43                    break;
44            }
45        }while(opcao!=0);
46    }
47 }
```