# ESW 1 (103179): Pratica 04, by Cruz Ramon

< Back to Summary

| | |
|---|---|
| Assignment | ESW 1 (103179): Pratica 04 try #5 |
| Name | Cruz Ramon (Ramon.Cruz) |
| Partners | None |
| Submitted | 05/25/13 07:25PM, 1 day, 2 hrs, 34 mins early |
| Total Score | **5.0**/100.0 |

### Score Summary
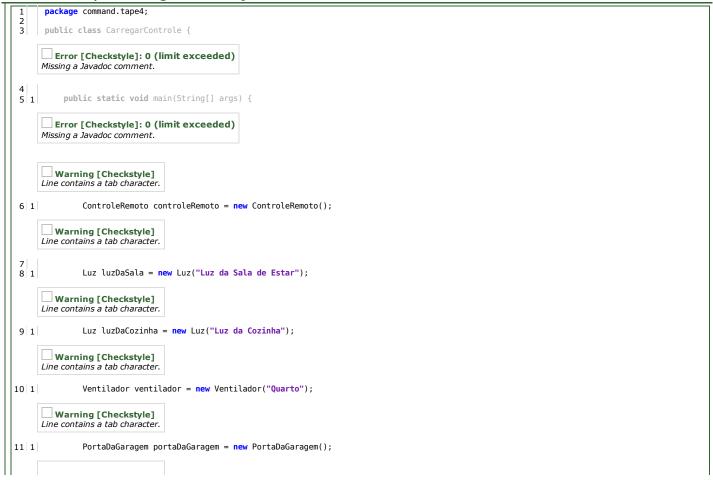
| | | |
|---|---|---|
| Design/Readability: | /95.0 | <Awaiting Staff> |
| Correctness/Testing: | 5.0/5.0 | |
| Final score: | **5.0/100.0** | |
| Position in class: | | |

| File | Remarks | Deductions | Methods Executed |
|---|---|---|---|
| command/tape4/CarregarControle.java | 49 | 0.0 | 100.0% |
| command/tape4/Command.java | 6 | 0.0 | |
| command/tape4/CommandVazio.java | 12 | 0.0 | 100.0% |
| command/tape4/ControleRemoto.java | 56 | 0.0 | 100.0% |
| command/tape4/ControleRemotoRamon.java | 91 | 0.0 | 100.0% |
| command/tape4/ControleRemotoRamonTest.java | 187 | 0.0 | 100.0% |
| command/tape4/ControleRemotoSimples.java | 15 | 0.0 | 100.0% |
| command/tape4/Luz.java | 28 | 0.0 | 100.0% |
| command/tape4/LuzDesligarCommand.java | 16 | 0.0 | 100.0% |
| command/tape4/LuzLigarCommand.java | 15 | 0.0 | 100.0% |
| command/tape4/PortaDaGaragem.java | 26 | 0.0 | 100.0% |
| command/tape4/PortaDaGaragemAbrirCommand.java | 17 | 0.0 | 100.0% |
| command/tape4/PortaDaGaragemFecharCommand.java | 15 | 0.0 | 100.0% |
| command/tape4/Som.java | 36 | 0.0 | 100.0% |
| command/tape4/SomComCDCommand.java | 21 | 0.0 | 100.0% |
| command/tape4/SomDesligarCommand.java | 22 | 0.0 | 100.0% |
| command/tape4/Ventilador.java | 74 | 0.0 | 100.0% |
| command/tape4/VentiladorDesligarCommand.java | 46 | 0.0 | 100.0% |
| command/tape4/VentiladorLigarCommand.java | 21 | 0.0 | 100.0% |

## ⊟command/tape4/CarregarControle.java

```
1   package command.tape4;
2
3   public class CarregarControle {
```

☐ **Error [Checkstyle]: 0 (limit exceeded)**
*Missing a Javadoc comment.*

```
4
5 1      public static void main(String[] args) {
```

☐ **Error [Checkstyle]: 0 (limit exceeded)**
*Missing a Javadoc comment.*

☐ **Warning [Checkstyle]**
*Line contains a tab character.*

```
6 1          ControleRemoto controleRemoto = new ControleRemoto();
```

☐ **Warning [Checkstyle]**
*Line contains a tab character.*

```
7
8 1          Luz luzDaSala = new Luz("Luz da Sala de Estar");
```

☐ **Warning [Checkstyle]**
*Line contains a tab character.*

```
9 1          Luz luzDaCozinha = new Luz("Luz da Cozinha");
```

☐ **Warning [Checkstyle]**
*Line contains a tab character.*

```
10 1          Ventilador ventilador = new Ventilador("Quarto");
```

☐ **Warning [Checkstyle]**
*Line contains a tab character.*

```
11 1          PortaDaGaragem portaDaGaragem = new PortaDaGaragem();
```

```
12 1          Som som = new Som();
```

```
13
14 1          LuzLigarCommand luzDaSalaLigarCommand = new LuzLigarCommand(luzDaSala);
```

```
15 1          LuzDesligarCommand luzDaSalaDesligarCommand = new LuzDesligarCommand(luzDaSala);
```

```
16
17 1          LuzLigarCommand luzDaCozinhaLigarCommand = new LuzLigarCommand(luzDaCozinha);
```

```
18 1          LuzDesligarCommand luzDaCozinhaDesligarCommand = new LuzDesligarCommand(luzDaCozinha);
```

```
19
20 1          VentiladorLigarCommand ventiladorLigarCommand = new VentiladorLigarCommand(ventilador);
```

```
21 1          VentiladorDesligarCommand ventiladorDesligarCommand = new VentiladorDesligarCommand(ventilador);
```

```
22
23 1          PortaDaGaragemAbrirCommand portaDaGaragemAbrirCommand = new PortaDaGaragemAbrirCommand(portaDaGaragem);
```

```
24 1          PortaDaGaragemFecharCommand portaDaGaragemFecharCommand = new PortaDaGaragemFecharCommand(portaDaGaragem);
```

```
25
26 1          SomComCDCommand somComCDCommand = new SomComCDCommand(som);
```

```
27 1            SomDesligarCommand somDesligarCommand = new SomDesligarCommand(som);
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
28
29 1            controleRemoto.setCommand(0, luzDaSalaLigarCommand,luzDaSalaDesligarCommand);
```

**Error [Checkstyle]: 0 (limit exceeded)**
*',' is not followed by whitespace.*

**Warning [Checkstyle]**
*Line contains a tab character.*

**Error [Checkstyle]: 0 (limit exceeded)**
*Line is longer than 80 characters.*

```
30 1            controleRemoto.setCommand(1, luzDaCozinhaLigarCommand,luzDaCozinhaDesligarCommand);
```

**Error [Checkstyle]: 0 (limit exceeded)**
*',' is not followed by whitespace.*

**Warning [Checkstyle]**
*Line contains a tab character.*

**Error [Checkstyle]: 0 (limit exceeded)**
*Line is longer than 80 characters.*

```
31 1            controleRemoto.setCommand(2, ventiladorLigarCommand,ventiladorDesligarCommand);
```

**Error [Checkstyle]: 0 (limit exceeded)**
*',' is not followed by whitespace.*

**Warning [Checkstyle]**
*Line contains a tab character.*

**Error [Checkstyle]: 0 (limit exceeded)**
*Line is longer than 80 characters.*

```
32 1            controleRemoto.setCommand(3, portaDaGaragemAbrirCommand,portaDaGaragemFecharCommand);
```

**Error [Checkstyle]: 0 (limit exceeded)**
*',' is not followed by whitespace.*

**Warning [Checkstyle]**
*Line contains a tab character.*

**Error [Checkstyle]: 0 (limit exceeded)**
*Line is longer than 80 characters.*

```
33 1            controleRemoto.setCommand(4, somComCDCommand, somDesligarCommand);
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
34
35 1            System.out.println(controleRemoto);
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
36
37 1            controleRemoto.botaoLigarPressionado(1);
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
38 1            controleRemoto.botaoDesligarPressionado(1);
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
39
40 1            controleRemoto.botaoLigarPressionado(2);
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
41 1            controleRemoto.botaoDesligarPressionado(2);
```

```
42
43 1            controleRemoto.botaoLigarPressionado(3);
```

```
44 1            controleRemoto.botaoDesligarPressionado(3);
```

```
45
46 1            controleRemoto.botaoLigarPressionado(4);
```

```
47 1            controleRemoto.botaoDesligarPressionado(4);
```

```
48
49        }
```

```
50    }
```

## command/tape4/Command.java

```
1    package command.tape4;
2
3    public interface Command {
```

```
4        public void execute();
```

```
5        public void undo();
```

```
6
```

```
7    }
```

## command/tape4/CommandVazio.java

```
1    package command.tape4;
2
3    public class CommandVazio implements Command {
```

```
4
5 2      @Override
```

```
6        public void execute() {
```

```
7
8            }
```

```
9  1      public void undo(){}
```

```
10
11       }
```

## command/tape4/ControleRemoto.java

```
1        package command.tape4;
2
3        public class ControleRemoto {
```

```
4            private Command[] commandsLigar;
```

```
5            private Command[] commandsDesligar;
```

```
6
```

```
7
```

```
8  3      public ControleRemoto()
```

```
9        {
```

```
10 3          commandsLigar = new Command[7];
```

```
11  3          commandsDesligar = new Command[7];
```

```
12  |
```

```
13  3          Command commandVazio = new CommandVazio();
```

```
14  3          for (int i = 0; i < 7; i++)
```

```
15  |          {
```

```
16  |
```

```
17 21              commandsLigar[i] = commandVazio;
```

```
18 21              commandsDesligar[i] = commandVazio;
```

```
19  |
```

```
20  |          }
```

```
21  |      }
```

```
22
23 10      public void setCommand(int posicao, Command commandLigar,Command commandDesligar)
```

```
24  |      {
```

```
25 10          commandsLigar[posicao] = commandLigar;
```

```
    |  |
```

```
26 10          commandsDesligar[posicao] = commandDesligar;
```

```
27   |      }
```

```
28
29  9      public void botaoLigarPressionado(int posicao)
```

```
30   |      {
```

```
31  9          commandsLigar[posicao].execute();
```

```
32   |      }
```

```
33
34  9      public void botaoDesligarPressionado(int posicao)
```

```
35   |      {
```

```
36  9          commandsDesligar[posicao].execute();
```

```
37   |      }
```

```
38  1      public void botaoDesfazerDesligar(int posicao)
```

```
39   |      {
```

```
40  1          commandsDesligar[posicao].undo();
```

```
41   |      }
```

```
42
43  3      public String toString(){
```

```
44  3        StringBuffer stringBuffer = new StringBuffer();
```

```
45  3        stringBuffer.append("\n ----- Controle Remoto ------ \n");
```

```
46  3        for (int i = 0; i < commandsLigar.length; i++) {
```

```
47 21            stringBuffer.append("[posicao " + i + "] "
```

```
48                    + commandsLigar[i].getClass().getName() + "\t"
```

```
49                    + commandsDesligar[i].getClass().getName() + "\n");
```

```
50
51        }
```

```
52  3        return stringBuffer.toString();
```

```
53     }
```

```
54
```

```
55  }
```

## command/tape4/ControleRemotoRamon.java

```
1  package command.tape4;
2
3  public class ControleRemotoRamon
```

```
4  {
5 1     public static void main(String args[])
```

```
6        {
```
**Warning [Checkstyle]**
*Line contains a tab character.*

```
7
```
**Warning [Checkstyle]**
*Line contains a tab character.*

```
8           // 1 Criar o controle Remoto da bagunca
```
**Warning [Checkstyle]**
*Line contains a tab character.*

```
9 1       ControleRemoto controleRamon=new ControleRemoto();
```
**Error [Checkstyle]: 0 (limit exceeded)**
*'=' is not followed by whitespace.*

**Error [Checkstyle]: 0 (limit exceeded)**
*'=' is not preceded with whitespace.*

**Warning [Checkstyle]**
*Line contains a tab character.*

```
10
```
**Warning [Checkstyle]**
*Line contains a tab character.*

```
11          // 2 criar Objetos do controle Remoto
```
**Warning [Checkstyle]**
*Line contains a tab character.*

```
12
```
**Warning [Checkstyle]**
*Line contains a tab character.*

```
13 1      Ventilador ventilador= new Ventilador("sala de estar");
```
**Error [Checkstyle]: 0 (limit exceeded)**
*'=' is not preceded with whitespace.*

**Warning [Checkstyle]**
*Line contains a tab character.*

```
14 1      Luz luzSalaEstar =new Luz("Sala de estar");
```
**Error [Checkstyle]: 0 (limit exceeded)**
*'=' is not followed by whitespace.*

**Warning [Checkstyle]**
*Line contains a tab character.*

```
15 1      Luz luzCozinha=new Luz("Cozinha");
```
**Error [Checkstyle]: 0 (limit exceeded)**
*'=' is not followed by whitespace.*

**Error [Checkstyle]: 0 (limit exceeded)**
*'=' is not preceded with whitespace.*

**Warning [Checkstyle]**
*Line contains a tab character.*

```
16 1      Som som =new Som();
```
**Error [Checkstyle]: 0 (limit exceeded)**
*'=' is not followed by whitespace.*

**Warning [Checkstyle]**
*Line contains a tab character.*

```
17
```

```
18        // 3 criar os camando dos objentos
```

```
19
```

```
20  1      VentiladorLigarCommand ventiladorLigarComando= new VentiladorLigarCommand(ventilador);
```

```
21  1      VentiladorDesligarCommand ventiladorDesligarComando=new VentiladorDesligarCommand(ventilador);
```

```
22
```

```
23  1      LuzLigarCommand luzSalaEstarLigar= new LuzLigarCommand(luzSalaEstar);
```

```
24  1      LuzDesligarCommand luzSalaEstarDesligar=new LuzDesligarCommand(luzSalaEstar);
```

```
25
```

```
26  1      LuzLigarCommand luzCozinhaLigar =new LuzLigarCommand(luzCozinha);
```

27 | 1 |         `LuzDesligarCommand luzCozinhaDesligar =new LuzDesligarCommand(luzCozinha);`

28 |

29 | 1 |         `SomComCDCommand somLigar =new SomComCDCommand(som);`

30 | 1 |         `SomDesligarCommand somDesligar=new SomDesligarCommand(som);`

31 |

32 |

33 |

34 | |     `//4 configure os comandos`

35 |

36 | 1 |         `controleRamon.setCommand(1, luzSalaEstarLigar, luzSalaEstarDesligar);`

37 | 1 |         `controleRamon.setCommand(2, luzCozinhaLigar, luzCozinhaDesligar);`

38 | 1 |         `controleRamon.setCommand(3, somLigar, somDesligar);// Som para incomodar o bairro inteiro`

```
39  1           controleRamon.setCommand(4, ventiladorLigarComando, ventiladorDesligarComando);// o trem vai pegar fogo.. liga e ventilador no maximo
```

```
40
```

```
41
```

```
42
```

```
43  1           System.out.println(controleRamon);
```

```
44
```

```
45           //5 use o controle remoto
```

```
46
```

```
47  1           System.out.println("\nLiga tudo!!! .. ta na hora de fazer o frevo ... passamos em FAS  =D");
```

```
48
```

```
49  1           controleRamon.botaoLigarPressionado(1);
```

```
50  1           controleRamon.botaoLigarPressionado(2);
```

```
51  1           controleRamon.botaoLigarPressionado(3);
```

```
52 1          controleRamon.botaoLigarPressionado(4);
```

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
53
```

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
54 1          System.out.println("\n\n===Ferrou, algum vizinho chamou a policia >:(\n" );
```

> **Warning [Checkstyle]**
> *Line contains a tab character.*

> **Error [Checkstyle]: 0 (limit exceeded)**
> *Line is longer than 80 characters.*

```
55
```

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
56 1          controleRamon.botaoDesligarPressionado(1);
```

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
57 1          controleRamon.botaoDesligarPressionado(2);
```

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
58 1          controleRamon.botaoDesligarPressionado(3);
```

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
59 1          controleRamon.botaoDesligarPressionado(4);
```

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
60
```

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
61
```

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
62
```

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
63
```

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
64      }
```

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
65   }
66
```

## command/tape4/ControleRemotoRamonTest.java

```
1   package command.tape4;
2
3   import static org.junit.Assert.*;
4   import junit.framework.Assert;
5
6   import org.junit.Test;
7
8   public class ControleRemotoRamonTest {
```

> **Error [Checkstyle]: 0 (limit exceeded)**
> *Missing a Javadoc comment.*

```
 9
10  1      @Test
```

**Error [Checkstyle]: 0 (limit exceeded)**
*Missing a Javadoc comment.*

**Warning [Checkstyle]**
*Line contains a tab character.*

```
11          public void testaRecivers()
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
12          {
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
13
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
14  1          Luz luz = new Luz("string");
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
15  1          PortaDaGaragem portaGaragem = new PortaDaGaragem();
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
16  1          Som som = new Som();
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
17  1          Ventilador ventilador= new Ventilador("string");
```

**Error [Checkstyle]: 0 (limit exceeded)**
*'=' is not preceded with whitespace.*

**Warning [Checkstyle]**
*Line contains a tab character.*

```
18
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
19          // instanciacao
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
20  1          Assert.assertNotNull(luz.getClass().getName());
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
21  1          Assert.assertNotNull(portaGaragem.getClass().getName());
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
22  1          Assert.assertNotNull(som.getClass().getName());
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
23  1          Assert.assertNotNull(ventilador.getClass().getName());
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
24
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
25  1          Assert.assertEquals("string", luz.getLocal());
```

26 | 1 |          Assert.assertEquals(**"string"**, ventilador.getLocal());

27 |  |

28 |  |        *//testa acoes*

29 |  |

30 | 1 |        luz.ligar();

31 | 1 |        luz.desligar();

32 | 1 |        portaGaragem.abrir();

33 | 1 |        portaGaragem.fechar();

34 | 1 |        portaGaragem.ligarLuz();

35 | 1 |        portaGaragem.desligarLuz();

36 | 1 |        portaGaragem.parar();

37 | 1 |        som.definirCD();

38 | 1 |        som.definirVolume(12);

39 | 1 |        som.definirDVD();

40 | 1 |        som.definirRadio();

41 | 1 |        som.ligar();

42 | 1 |        som.desligar();

```
43
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
44
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
45  1          Assert.assertEquals(12, som.getVolume());
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
46  1          som.desligar();
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
47  1          ventilador.definirMaximo();
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
48  1          Assert.assertEquals(3,ventilador.getVelocidade());
```

**Error [Checkstyle]: 0 (limit exceeded)**
*',' is not followed by whitespace.*

**Warning [Checkstyle]**
*Line contains a tab character.*

```
49  1          Assert.assertEquals(-1,ventilador.getVelocidadeAnterior());
```

**Error [Checkstyle]: 0 (limit exceeded)**
*',' is not followed by whitespace.*

**Warning [Checkstyle]**
*Line contains a tab character.*

```
50
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
51  1          ventilador.definirMedio();
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
52  1          Assert.assertEquals(2,ventilador.getVelocidade());
```

**Error [Checkstyle]: 0 (limit exceeded)**
*',' is not followed by whitespace.*

**Warning [Checkstyle]**
*Line contains a tab character.*

```
53  1          Assert.assertEquals(3,ventilador.getVelocidadeAnterior());
```

**Error [Checkstyle]: 0 (limit exceeded)**
*',' is not followed by whitespace.*

**Warning [Checkstyle]**
*Line contains a tab character.*

```
54
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
55  1          ventilador.definirMinimo();
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
56  1          Assert.assertEquals(1,ventilador.getVelocidade());
```

**Error [Checkstyle]: 0 (limit exceeded)**
*',' is not followed by whitespace.*

```
57 1          Assert.assertEquals(2,ventilador.getVelocidadeAnterior());
```

```
58 |
```

```
59 1          ventilador.desligar();
```

```
60 1          Assert.assertEquals(0,ventilador.getVelocidade());
```

```
61 1          Assert.assertEquals(1,ventilador.getVelocidadeAnterior());
```

```
62 |
63 |      }
```

```
64 |
```

```
65 1     @Test
```

```
66 |      public void testaConcreteCommands()
```

```
67 |      {
```

```
68 1          Luz luz = new Luz("string");
```

```
69 1          PortaDaGaragem portaGaragem = new PortaDaGaragem();
```

```
70 1          Som som = new Som();
```

```
71  1          Ventilador ventilador= new Ventilador("string");
```

**Error [Checkstyle]: 0 (limit exceeded)**
*'=' is not preceded with whitespace.*

**Warning [Checkstyle]**
*Line contains a tab character.*

```
72  |
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
73  1          LuzDesligarCommand luzDesliga= new LuzDesligarCommand(luz);
```

**Error [Checkstyle]: 0 (limit exceeded)**
*'=' is not preceded with whitespace.*

**Warning [Checkstyle]**
*Line contains a tab character.*

```
74  1          LuzLigarCommand luzLiga = new LuzLigarCommand(luz);
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
75  |
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
76  1          PortaDaGaragemAbrirCommand portaAbre = new PortaDaGaragemAbrirCommand(portaGaragem);
```

**Warning [Checkstyle]**
*Line contains a tab character.*

**Error [Checkstyle]: 0 (limit exceeded)**
*Line is longer than 80 characters.*

```
77  1          PortaDaGaragemFecharCommand portaFecha =new PortaDaGaragemFecharCommand(portaGaragem);
```

**Error [Checkstyle]: 0 (limit exceeded)**
*'=' is not followed by whitespace.*

**Warning [Checkstyle]**
*Line contains a tab character.*

**Error [Checkstyle]: 0 (limit exceeded)**
*Line is longer than 80 characters.*

```
78  |
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
79  1          SomComCDCommand somLiga = new SomComCDCommand(som);
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
80  1          SomDesligarCommand somDesliga = new SomDesligarCommand(som);
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
81  |
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
82  1          VentiladorDesligarCommand ventiladorDesliga = new VentiladorDesligarCommand(ventilador);
```

**Warning [Checkstyle]**
*Line contains a tab character.*

**Error [Checkstyle]: 0 (limit exceeded)**
*Line is longer than 80 characters.*

```
83  1          VentiladorLigarCommand ventiladorLiga = new VentiladorLigarCommand(ventilador);
```

84 |

85 1 `        Assert.assertNotNull(luzLiga.getClass().getName());`

86 1 `        Assert.assertNotNull(luzDesliga.getClass().getName());`

87 1 `        Assert.assertNotNull(portaAbre.getClass().getName());`

88 1 `        Assert.assertNotNull(portaFecha.getClass().getName());`

89 1 `        Assert.assertNotNull(somLiga.getClass().getName());`

90 1 `        Assert.assertNotNull(somDesliga.getClass().getName());`

91 1 `        Assert.assertNotNull(ventiladorDesliga.getClass().getName());`

92 1 `        Assert.assertNotNull(ventiladorLiga.getClass().getName());`

93 |

94 | `        //Testa execute()`

95 |

96 1 `        luzLiga.execute();`

97 1 `        luzLiga.undo();`

98 1 `        luzDesliga.execute();`

99 1 `        luzDesliga.undo();`

```
100

            Warning [Checkstyle]
            Line contains a tab character.
101  1          portaAbre.execute();

            Warning [Checkstyle]
            Line contains a tab character.
102  1          portaAbre.undo();

            Warning [Checkstyle]
            Line contains a tab character.
103  1          portaFecha.execute();

            Warning [Checkstyle]
            Line contains a tab character.
104  1          portaFecha.undo();

            Warning [Checkstyle]
            Line contains a tab character.
105

            Warning [Checkstyle]
            Line contains a tab character.
106  1          somLiga.execute();

            Warning [Checkstyle]
            Line contains a tab character.
107  1          somLiga.undo();

            Warning [Checkstyle]
            Line contains a tab character.
108  1          somDesliga.execute();

            Warning [Checkstyle]
            Line contains a tab character.
109  1          somDesliga.undo();

            Warning [Checkstyle]
            Line contains a tab character.
110

            Warning [Checkstyle]
            Line contains a tab character.
111  1          ventiladorLiga.execute();

            Warning [Checkstyle]
            Line contains a tab character.
112  1          Assert.assertEquals(3, ventilador.getVelocidade());

            Warning [Checkstyle]
            Line contains a tab character.
113  1          ventiladorLiga.undo();

            Warning [Checkstyle]
            Line contains a tab character.
114  1          Assert.assertEquals(0, ventilador.getVelocidade());

            Warning [Checkstyle]
            Line contains a tab character.
115

            Warning [Checkstyle]
            Line contains a tab character.
116  1          ventiladorLiga.execute();

            Warning [Checkstyle]
            Line contains a tab character.
```

```
117 1        ventiladorDesliga.execute();
```

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
118 1        Assert.assertEquals(3, ventilador.getVelocidadeAnterior());
```

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
119 1        Assert.assertEquals(0, ventilador.getVelocidade());
```

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
120 1        ventiladorDesliga.undo();
```

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
121 1        Assert.assertEquals(3, ventilador.getVelocidade());
```

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
122 |
```

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
123 |    }
```

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
124 |
```

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
125 1    @Test
```

> **Error [Checkstyle]: 0 (limit exceeded)**
> *Missing a Javadoc comment.*

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
126 |    public void testaInvokers()
```

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
127 |    {
```

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
128 |
```

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
129 1        ControleRemoto controleRemoto=new ControleRemoto();
```

> **Error [Checkstyle]: 0 (limit exceeded)**
> *'=' is not followed by whitespace.*

> **Error [Checkstyle]: 0 (limit exceeded)**
> *'=' is not preceded with whitespace.*

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
130 |
```

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
131 1        Assert.assertNotNull(controleRemoto.getClass().getName());
```

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
132
```

```
133  1          Ventilador ventilador= new Ventilador("string");
```

```
134
```

```
135  1          VentiladorDesligarCommand ventiladorDesliga = new VentiladorDesligarCommand(ventilador);
```

```
136  1          VentiladorLigarCommand ventiladorLiga = new VentiladorLigarCommand(ventilador);
```

```
137
```

```
138  1          controleRemoto.setCommand(1, ventiladorLiga, ventiladorDesliga);
```

```
139  1          controleRemoto.toString();
```

```
140
```

```
141  1          controleRemoto.botaoLigarPressionado(1);
```

```
142  1          Assert.assertEquals(3, ventilador.getVelocidade());
```

```
143
```

```
144  1          controleRemoto.botaoDesligarPressionado(1);
```

```
145  1          Assert.assertEquals(0, ventilador.getVelocidade());
```

```
146
```

```
147 1          controleRemoto.botaoDesfazerDesligar(1);
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
148 1          Assert.assertEquals(3, ventilador.getVelocidade());
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
149 |
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
150 |
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
151 |
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
152 |      }
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
153 1      @Test
```

**Error [Checkstyle]: 0 (limit exceeded)**
*Missing a Javadoc comment.*

**Warning [Checkstyle]**
*Line contains a tab character.*

```
154 |      public void testaOutrasClasses()
```

**Error [PMD]: 0 (limit exceeded)**
*A test case method should use one or more assert...() methods to check that the desired behavior is actually occurring.*

**Warning [Checkstyle]**
*Line contains a tab character.*

```
155 |      {
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
156 1          CarregarControle carregar = new CarregarControle();
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
157 1          carregar.main(null);
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
158 1          CommandVazio command = new CommandVazio();
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
159 1          command.execute();
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
160 1          command.undo();
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
161 |
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
162 1          ControleRemotoSimples controleSimples = new ControleRemotoSimples();
```

**Warning [Checkstyle]**

```
163 1            controleSimples.setCommand(command);
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
164 1            controleSimples.botaoPressionado();
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
165   |
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
166 1        ControleRemotoRamon controleRamon = new ControleRemotoRamon();
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
167 1            controleRamon.main(null);
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
168   |
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
169   |
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
170   |    }
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
171   |
172   |  }
```

## command/tape4/ControleRemotoSimples.java

```
1    package command.tape4;
2
3    public class ControleRemotoSimples {
```

**Error [Checkstyle]: 0 (limit exceeded)**
*Missing a Javadoc comment.*

```
4      Command posicao;
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
5
6 1      public ControleRemotoSimples() {}
```

**Error [PMD]: 0 (limit exceeded)**
*This constructor does not contain any code. Should it be doing something? Add a comment to the body explaining why it is empty.*

**Error [Checkstyle]: 0 (limit exceeded)**
*'}' is not preceded with whitespace.*

**Error [Checkstyle]: 0 (limit exceeded)**
*'{' is not followed by whitespace.*

**Error [Checkstyle]: 0 (limit exceeded)**
*Missing a Javadoc comment.*

**Warning [Checkstyle]**
*Line contains a tab character.*

```
7
8 1      public void setCommand(Command command) {
```

**Error [Checkstyle]: 0 (limit exceeded)**
*Missing a Javadoc comment.*

```
 9 1          posicao = command;
```

```
10        }
```

```
11
12 1      public void botaoPressionado() {
```

```
13 1          posicao.execute();
```

```
14        }
```

```
15    }
```

## command/tape4/Luz.java

```
 1    package command.tape4;
 2
 3    public class Luz {
```

```
 4        private String local;
```

```
 5
```

```
 6 6      public Luz(String local) {
```

```
 7 6          this.local = local;
```

```
 8        }
```

```
 9 6      public void ligar() {
```

```
10 6          System.out.println("Luz "+local+" ligada");
```

> **Error [Checkstyle]: 0 (limit exceeded)**
> *'+' is not preceded with whitespace.*

> **Error [Checkstyle]: 0 (limit exceeded)**
> *'+' is not followed by whitespace.*

> **Error [Checkstyle]: 0 (limit exceeded)**
> *'+' is not preceded with whitespace.*

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
11        }
```

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
12 6     public void desligar() {
```

> **Error [Checkstyle]: 0 (limit exceeded)**
> *Missing a Javadoc comment.*

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
13 6         System.out.println("Luz "+local+" desligada");
```

> **Error [Checkstyle]: 0 (limit exceeded)**
> *'+' is not followed by whitespace.*

> **Error [Checkstyle]: 0 (limit exceeded)**
> *'+' is not preceded with whitespace.*

> **Error [Checkstyle]: 0 (limit exceeded)**
> *'+' is not followed by whitespace.*

> **Error [Checkstyle]: 0 (limit exceeded)**
> *'+' is not preceded with whitespace.*

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
14        }
```

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
15 1     public String getLocal(){
```

> **Error [Checkstyle]: 0 (limit exceeded)**
> *'{' is not preceded with whitespace.*

> **Error [Checkstyle]: 0 (limit exceeded)**
> *Missing a Javadoc comment.*

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
16 1         return local;
```

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
17        }
```

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
18
19    }
```

## command/tape4/LuzDesligarCommand.java

```
1    package command.tape4;
2
3    public class LuzDesligarCommand implements Command {
```

```
4      Luz luz;
```

```
5
6 5      public LuzDesligarCommand(Luz luz) {
```

```
7 5          this.luz = luz;
```

```
8      }
```

```
9
10 4      public void execute() {
```

```
11 4          luz.desligar();
```

```
12      }
```

```
13
```

```
14 1      public void undo()
```

```
15      {
```

```
16 1          luz.ligar();
```

```
17      }
```

```
18   }
```

## command/tape4/LuzLigarCommand.java

```
1    package command.tape4;
2
3    public class LuzLigarCommand implements Command {
```

```
           Missing a Javadoc comment.

  4 │       Luz luz;

        ☐ Warning [Checkstyle]
          Line contains a tab character.

  5
  6 5       public LuzLigarCommand(Luz luz) {

        ☐ Error [Checkstyle]: 0 (limit exceeded)
          Missing a Javadoc comment.


        ☐ Warning [Checkstyle]
          Line contains a tab character.

  7 5           this.luz = luz;

        ☐ Warning [Checkstyle]
          Line contains a tab character.

  8 │       }

        ☐ Warning [Checkstyle]
          Line contains a tab character.

  9
 10 4       public void execute() {

        ☐ Error [Checkstyle]: 0 (limit exceeded)
          Missing a Javadoc comment.


        ☐ Warning [Checkstyle]
          Line contains a tab character.

 11 4           luz.ligar();

        ☐ Warning [Checkstyle]
          Line contains a tab character.

 12 │       }

        ☐ Warning [Checkstyle]
          Line contains a tab character.

 13 1       public void undo()

        ☐ Error [Checkstyle]: 0 (limit exceeded)
          Missing a Javadoc comment.


        ☐ Warning [Checkstyle]
          Line contains a tab character.

 14 │       {

        ☐ Warning [Checkstyle]
          Line contains a tab character.

 15 1           luz.desligar();

        ☐ Warning [Checkstyle]
          Line contains a tab character.

 16 │       }

        ☐ Warning [Checkstyle]
          Line contains a tab character.

 17 │   }
```

## command/tape4/PortaDaGaragem.java

```
  1 │   package command.tape4;
  2
  3 │   public class PortaDaGaragem {

        ☐ Error [Checkstyle]: 0 (limit exceeded)
          Missing a Javadoc comment.

  4
  5 3       public PortaDaGaragem() {

        ☐ Error [PMD]: 0 (limit exceeded)
```

*This constructor does not contain any code. Should it be doing something? Add a comment to the body explaining why it is empty.*

☐ **Error [Checkstyle]: 0 (limit exceeded)**
*Missing a Javadoc comment.*

☐ **Warning [Checkstyle]**
*Line contains a tab character.*

6 |     }

☐ **Warning [Checkstyle]**
*Line contains a tab character.*

7 |

☐ **Warning [Checkstyle]**
*Line contains a tab character.*

8 4 |    public void abrir() {

☐ **Error [Checkstyle]: 0 (limit exceeded)**
*Missing a Javadoc comment.*

☐ **Warning [Checkstyle]**
*Line contains a tab character.*

9 4 |        System.out.println("Porta da garagem aberta");

☐ **Warning [Checkstyle]**
*Line contains a tab character.*

10 |     }

☐ **Warning [Checkstyle]**
*Line contains a tab character.*

11
12 4 |    public void fechar() {

☐ **Error [Checkstyle]: 0 (limit exceeded)**
*Missing a Javadoc comment.*

☐ **Warning [Checkstyle]**
*Line contains a tab character.*

13 4 |        System.out.println("Porta da garagem fechada");

☐ **Warning [Checkstyle]**
*Line contains a tab character.*

14 |     }

☐ **Warning [Checkstyle]**
*Line contains a tab character.*

15
16 1 |    public void parar() {

☐ **Error [Checkstyle]: 0 (limit exceeded)**
*Missing a Javadoc comment.*

☐ **Warning [Checkstyle]**
*Line contains a tab character.*

17 1 |        System.out.println("Porta da garagem parada");

☐ **Warning [Checkstyle]**
*Line contains a tab character.*

18 |     }

☐ **Warning [Checkstyle]**
*Line contains a tab character.*

19
20 1 |    public void ligarLuz() {

☐ **Error [Checkstyle]: 0 (limit exceeded)**
*Missing a Javadoc comment.*

```
21 1          System.out.println("Luz da garagem ligada");
```

```
22    |      }
```

```
23
24 1      public void desligarLuz() {
```

```
25 1          System.out.println("Luz da garagem desligada");
```

```
26    |      }
```

```
27
28    |  }
```

## command/tape4/PortaDaGaragemAbrirCommand.java

```
1     package command.tape4;
2
3     public class PortaDaGaragemAbrirCommand implements Command {
```

```
4    |      PortaDaGaragem portaDaGaragem;
```

```
5
6 2      public PortaDaGaragemAbrirCommand(PortaDaGaragem portaDaGaragem) {
```

```
7 2          this.portaDaGaragem = portaDaGaragem;
```

```
8    |      }
```

```
9
10 2      public void execute()
```

```
11   |      {
```

```
12 2          portaDaGaragem.abrir();
```

```
13 |        }
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
14 |
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
15 1|       public void undo()
```

**Error [Checkstyle]: 0 (limit exceeded)**
*Missing a Javadoc comment.*

**Warning [Checkstyle]**
*Line contains a tab character.*

```
16 |       {
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
17 1|           portaDaGaragem.fechar();
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
18 |       }
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
19 | }
```

## command/tape4/PortaDaGaragemFecharCommand.java

```
1 | package command.tape4;
2 |
3 | public class PortaDaGaragemFecharCommand implements Command {
```

**Error [Checkstyle]: 0 (limit exceeded)**
*Missing a Javadoc comment.*

```
4 |       PortaDaGaragem portaDaGaragem;
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
5 |
6 2|       public PortaDaGaragemFecharCommand(PortaDaGaragem portaDaGaragem) {
```

**Error [Checkstyle]: 0 (limit exceeded)**
*Missing a Javadoc comment.*

**Warning [Checkstyle]**
*Line contains a tab character.*

```
7 2|           this.portaDaGaragem = portaDaGaragem;
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
8 |       }
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
9 |
10 2|       public void execute() {
```

**Error [Checkstyle]: 0 (limit exceeded)**
*Missing a Javadoc comment.*

**Warning [Checkstyle]**
*Line contains a tab character.*

```
11 2|           portaDaGaragem.fechar();
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
12 |       }
```

```
13 1      public void undo()
```

```
14 |      {
```

```
15 1          portaDaGaragem.abrir();
```

```
16 |      }
```

```
17 |  }
```

## command/tape4/Som.java

```
1      package command.tape4;
2
3      public class Som {
```

```
4          private int volume;
```

```
5 5      public void ligar() {
```

```
6 5          System.out.println("Som ligado");
```

```
7 |      }
```

```
8 |
```

```
9 6      public void desligar() {
```

```
10 6          System.out.println("Som desligado");
```

```
11 |      }
```

```
12 |
```

```java
13 5        public void definirCD() {
```

**Error [Checkstyle]: 0 (limit exceeded)**
*Missing a Javadoc comment.*

**Warning [Checkstyle]**
*Line contains a tab character.*

```java
14 5            System.out.println("Lendo CD");
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```java
15      }
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```java
16
17 1        public void definirDVD() {
```

**Error [Checkstyle]: 0 (limit exceeded)**
*Missing a Javadoc comment.*

**Warning [Checkstyle]**
*Line contains a tab character.*

```java
18 1            System.out.println("Lendo DVD");
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```java
19      }
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```java
20
21 1        public void definirRadio() {
```

**Error [Checkstyle]: 0 (limit exceeded)**
*Missing a Javadoc comment.*

**Warning [Checkstyle]**
*Line contains a tab character.*

```java
22 1            System.out.println("Buscando faixa");
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```java
23      }
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```java
24
25 5        public void definirVolume(int volume) {
```

**Error [Checkstyle]: 0 (limit exceeded)**
*'volume' hides a field.*

**Error [Checkstyle]: 0 (limit exceeded)**
*Missing a Javadoc comment.*

**Warning [Checkstyle]**
*Line contains a tab character.*

```java
26 5            this.volume= volume;
```

**Error [Checkstyle]: 0 (limit exceeded)**
*'=' is not preceded with whitespace.*

**Warning [Checkstyle]**
*Line contains a tab character.*

```java
27 5            System.out.println("Volume "+ volume);
```

**Error [Checkstyle]: 0 (limit exceeded)**
*'+' is not preceded with whitespace.*

28 | `        }`

29 1 | `    public int getVolume()`

30 | `    {`

31 1 | `        return volume;`

32 | `    }`

33 |
34 | `}`

## command/tape4/SomComCDCommand.java

1 | `package command.tape4;`
2 |
3 | `public class SomComCDCommand implements Command {`

4 |
5 | `    private Som som;`

6 |

7 3 | `    public SomComCDCommand(Som som){`

8 3 | `        this.som = som;`

9 | `    }`

10 |

11 3 | `    @Override`

```
12    public void execute() {
```

```
13 3        som.ligar();
```

```
14 3        som.definirCD();
```

```
15 3        som.definirVolume(200);
```

```
16      }
```

```
17 1    public void undo()
```

```
18      {
```

```
19 1        som.desligar();
```

```
20      }
```

```
21
22    }
23
```

## command/tape4/SomDesligarCommand.java

```
1    package command.tape4;
2
3    public class SomDesligarCommand implements Command {
```

```
4
5        private Som som;
```

```
6
```

```
7 3    public SomDesligarCommand(Som som){
```

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
8 3         this.som = som;
```

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
9     }
```

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
10
```

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
11 3     @Override
```

> **Error [Checkstyle]: 0 (limit exceeded)**
> *Missing a Javadoc comment.*

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
12     public void execute() {
```

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
13 3         som.desligar();
```

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
14     }
```

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
15
```

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
16 1     public void undo()
```

> **Error [Checkstyle]: 0 (limit exceeded)**
> *Missing a Javadoc comment.*

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
17     {
```

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
18 1         som.ligar();
```

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
19 1         som.definirCD();
```

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
20 1         som.definirVolume(15);
```

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
21     }
```

> **Warning [Checkstyle]**
> *Line contains a tab character.*

```
22
23   }
24
```

## command/tape4/Ventilador.java

```
1    package command.tape4;
2
3    public class Ventilador {
```

**Error [Checkstyle]: 0 (limit exceeded)**
*Missing a Javadoc comment.*

```
4        public static final int MAXIMO = 3;
```

**Error [Checkstyle]: 0 (limit exceeded)**
*Missing a Javadoc comment.*

**Warning [Checkstyle]**
*Line contains a tab character.*

```
5        public static final int MEDIO = 2;
```

**Error [Checkstyle]: 0 (limit exceeded)**
*Missing a Javadoc comment.*

**Warning [Checkstyle]**
*Line contains a tab character.*

```
6        public static final int MINIMO = 1;
```

**Error [Checkstyle]: 0 (limit exceeded)**
*Missing a Javadoc comment.*

**Warning [Checkstyle]**
*Line contains a tab character.*

```
7        public static final int DESLIGADO = 0;
```

**Error [Checkstyle]: 0 (limit exceeded)**
*Missing a Javadoc comment.*

**Warning [Checkstyle]**
*Line contains a tab character.*

```
8
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
9        private int velocidade;
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
10       private int velocidadeAnterior;
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
11       private String local;
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
12
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
13  5    public Ventilador(String local)
```

**Error [Checkstyle]: 0 (limit exceeded)**
*Missing a Javadoc comment.*

**Warning [Checkstyle]**
*Line contains a tab character.*

```
14       {
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
15  5        this.local = local;
```

```
16  5          velocidade=-1;
```

```
17  |      }
```

```
18  |
```

```
19  8      public void definirMaximo()
```

```
20  |      {
```

```
21  8          velocidadeAnterior=velocidade;
```

```
22  8          velocidade = MAXIMO;
```

```
23  8          System.out.println("Ventilador na velocidade maxima");
```

```
24  |
```

```
25  |      }
```

```
26
27  1      public void definirMinimo()
```

```
28  |      {
```

*Line contains a tab character.*

```
29  1           velocidadeAnterior=velocidade;
```

**Error [Checkstyle]: 0 (limit exceeded)**
*'=' is not followed by whitespace.*

**Error [Checkstyle]: 0 (limit exceeded)**
*'=' is not preceded with whitespace.*

**Warning [Checkstyle]**
*Line contains a tab character.*

```
30  1           velocidade = MINIMO;
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
31  1           System.out.println("Ventilador na velocidade minima");
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
32          }
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
33
34  1       public void definirMedio()
```

**Error [Checkstyle]: 0 (limit exceeded)**
*Missing a Javadoc comment.*

**Warning [Checkstyle]**
*Line contains a tab character.*

```
35          {
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
36  1           velocidadeAnterior=velocidade;
```

**Error [Checkstyle]: 0 (limit exceeded)**
*'=' is not followed by whitespace.*

**Error [Checkstyle]: 0 (limit exceeded)**
*'=' is not preceded with whitespace.*

**Warning [Checkstyle]**
*Line contains a tab character.*

```
37  1           velocidade = MEDIO;
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
38  1           System.out.println("Ventilador na velocidade media");
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
39          }
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
40
41  6       public void desligar()
```

**Error [Checkstyle]: 0 (limit exceeded)**
*Missing a Javadoc comment.*

**Warning [Checkstyle]**
*Line contains a tab character.*

```
42          {
```

43 | 6 |             velocidadeAnterior=velocidade;

44 | 6 |          velocidade = DESLIGADO;

45 | 6 |          System.out.println("Ventilador desligado");

46 | |       }

47 |
48 | 11 |       public int getVelocidade() {

49 | 11 |          return velocidade;

50 | |       }

51 | 9 |       public int getVelocidadeAnterior()

52 | |       {

53 | 9 |          return velocidadeAnterior;

54 | |       }

55 | 1 |       public String getLocal()

56 | |       {

```
57  1          return local;
```

```
58          }
```

```
59
60      }
```

## command/tape4/VentiladorDesligarCommand.java

```
1      package command.tape4;
2
3      public class VentiladorDesligarCommand implements Command {
```

```
4
5          private Ventilador ventilador;
```

```
6
7  4      public VentiladorDesligarCommand(Ventilador ventilador) {
```

```
8  4          this.ventilador = ventilador;
```

```
9          }
```

```
10
11  4      @Override
```

```
12      public void execute() {
```

```
13  4          ventilador.desligar();
```

```
14
15          }
```

```
16  2      public void undo()
```

```
17          {
```

```
18  0            if(ventilador.getVelocidadeAnterior()<0) ventilador.definirMaximo();
```

```
19            else
```

```
20            {
```

```
21  2                switch(ventilador.getVelocidadeAnterior())
```

```
22                {
```

```
23  0                case 1:ventilador.definirMinimo();System.out.println("Ventilador religado na velocidade Minima");break;
```

```
24  0                case 2:ventilador.definirMedio();System.out.println("Ventilador religado na velocidade Media");break;
```

25 2            `case 3:ventilador.definirMaximo();System.out.println("Ventilador religado na Maxima");break;`

26

27              `}`

28

29          `}`

30      `}`

31
32    `}`

## command/tape4/VentiladorLigarCommand.java

1    `package command.tape4;`
2
3    `public class VentiladorLigarCommand implements Command {`

4

5        `private Ventilador ventilador;`

6

```
 7 4        public VentiladorLigarCommand(Ventilador ventilador){
```

**Error [Checkstyle]: 0 (limit exceeded)**
*'{' is not preceded with whitespace.*

**Error [Checkstyle]: 0 (limit exceeded)**
*Missing a Javadoc comment.*

**Warning [Checkstyle]**
*Line contains a tab character.*

```
 8 4            this.ventilador = ventilador;
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
 9 |        }
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
10 |
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
11 5        @Override
```

**Error [Checkstyle]: 0 (limit exceeded)**
*Missing a Javadoc comment.*

**Warning [Checkstyle]**
*Line contains a tab character.*

```
12 |        public void execute()
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
13 |        {
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
14 5            ventilador.definirMaximo();
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
15 |        }
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
16 1        public void undo()
```

**Error [Checkstyle]: 0 (limit exceeded)**
*Missing a Javadoc comment.*

**Warning [Checkstyle]**
*Line contains a tab character.*

```
17 |        {
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
18 1            ventilador.desligar();
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
19 |        }
```

**Warning [Checkstyle]**
*Line contains a tab character.*

```
20 |
21 |    }
```

< Back to Summary