UNIVERSIDADE VEIGA DE ALMEIDA

JÔNATAS ELEOTÉRIO SILVA

TRABALHO PARA O DESENVOLVIMENTO DE APLICAÇÃO PARA CONTROLE DE COMPUTADORES DE UMA LOJA

Trabalho da Disciplina [AVA 1]

São João de Meriti - RJ
2021

JÔNATAS ELEOTÉRIO SILVA

TRABALHO PARA O DESENVOLVIMENTO DE APLICAÇÃO PARA CONTROLE DE COMPUTADORES DE UMA LOJA

Trabalho da Disciplina [AVA 1]

ALUNO: JÔNATAS ELEOTÉRIO SILVA
MATRÍCULA: 20191300220
MATÉRIA: PROGRMAÇÃO ORIENTADA A OBJETOS II
PROFESSOR: MARCIO AURELIO NOVAES ESTEVES
TURMA: IL10329

São João de Meriti - RJ
2021

ENTREGA DA AVALIAÇÃO - TRABALHO DA DISCIPLINA [AVA 1]

Trabalho para o desenvolvimento de Aplicação para controle de Computadores de uma loja.

Desenvolvimento de projeto com a aplicação dos conceitos de Particionamento, Agregação e Herança.

Um projeto de desenvolvimento de sistemas solicita a criação de uma aplicação para uma empresa de venda de computadores. A aplicação deverá gerenciar as quantidades e os preços de três diferentes tipos de computadores, sendo eles: Desktop, Notebook e Servidor. Cada objeto deve ser instanciado a partir do modelo do computador, por isso, este dado não faz parte dos atributos. Você deve analisar as classes e determinar os Particionamento e a Agregações necessárias, além de analisar se será necessário a aplicação de Herança. Após a análise das classes e determinação da Hierarquia para a criação das classes, você deverá codificar as classes e uma aplicação que utilize pelo menos três objetos de cada tipo (Desktop, Notebook e Servidor) com entrada de dados solicitadas ao usuário. Deverão ser realizados testes práticos e captura das telas para apresentação junto com a documentação a ser entregue.

Introtução

A aplicação foi desenvolvida com o foco na utilização dos conceitos de abstração, encapsulamento, agregação de classes e herança, onde foram criadas seis classes sendo elas ControleDeComputadores, Computador, Servidor, Monitor, Notebook e Desktop.

Todos os atributos são do tipo privado, sendo necessário a ultilização dos metodos getters e setters para acessá-los, também foram criados métodos para imprimir, entrada de dados, entrada de dados individuais, e metodos construtores.

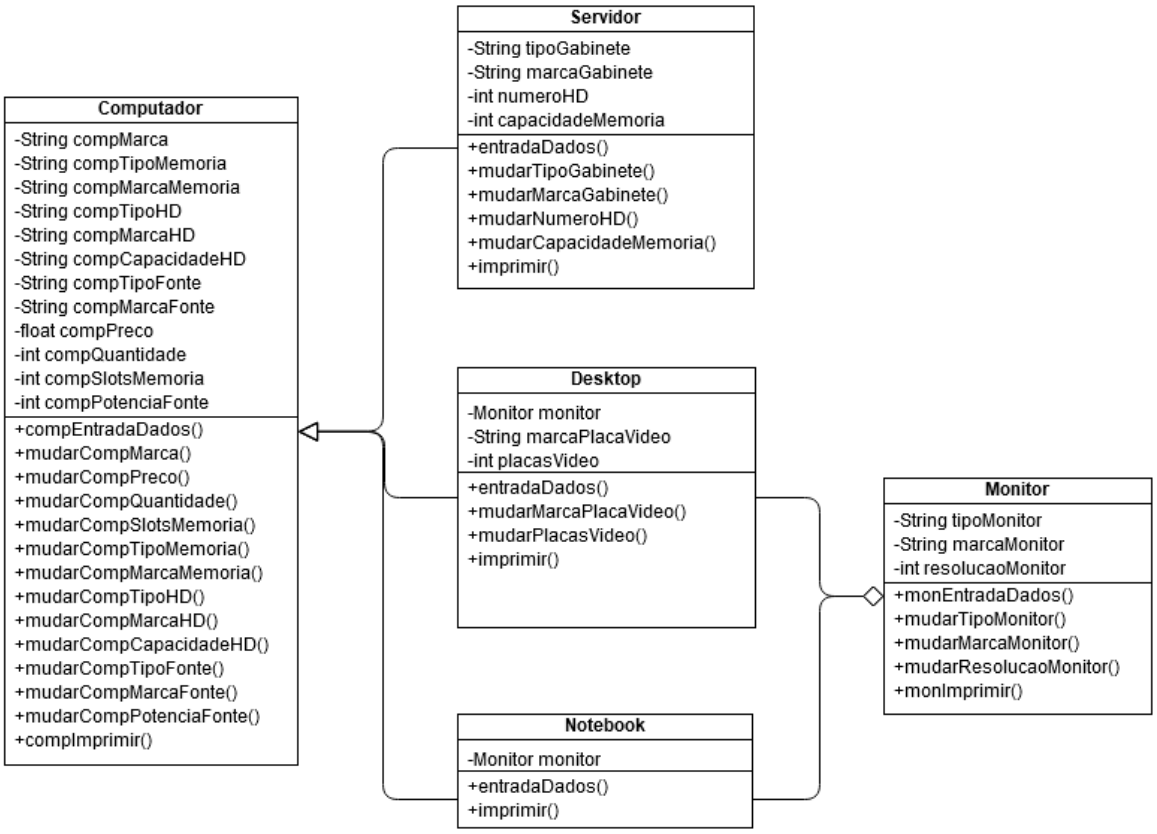
Observando os atributos sugeridos para as classes Servidor, Notebook e Desktop, pode-se notar similaridades entre a maioria deles, então foi criada a classe abstrata Computador para que fosse herdada.

Para utilizar o conceito de agregação de classes foi criada a classe Monitor para gerar o objeto monitor a ser agregados nas classes Notebook e Desktop.

Conteúdo

- **Hierrarquia das Classes**

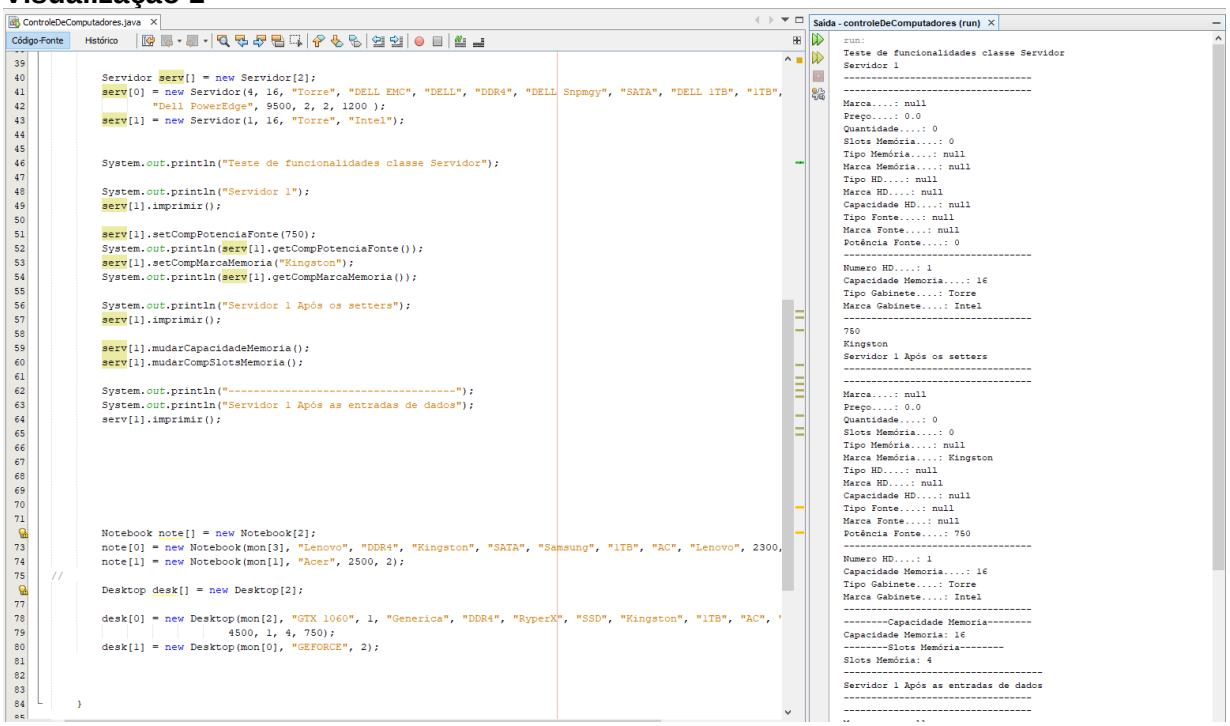
As classes Servidor, desktop e Notebook herdam da classe Computador e as classes Desktop e Notebook agregam a classe Monitor. Abaixo um pequeno diagrama mostrando as relações entre as classes.



- **Captura das Telas de teste**

Teste de funcionalidade da classe Servidor:

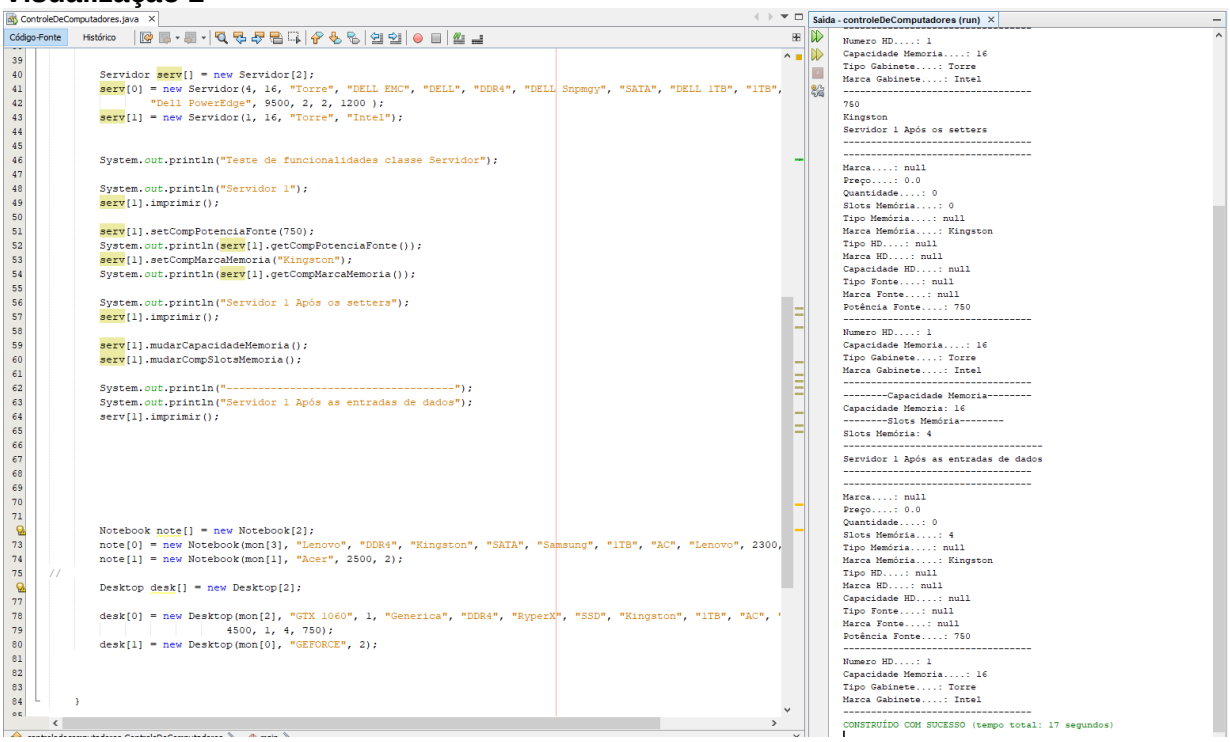
Visualização 1



```
Servidor serv[] = new Servidor[2];
serv[0] = new Servidor(4, 16, "Torre", "DELL EMC", "DELL", "DDR4", "DELL Snpmgy", "SATA", "DELL 1TB", "1TB",
        "Dell PowerEdge", 9500, 2, 2, 1200 );
serv[1] = new Servidor(1, 16, "Torre", "Intel");


System.out.println("Teste de funcionalidades classe Servidor");

System.out.println("Servidor 1");
serv[1].imprimir();

serv[1].setCompPotenciaFonte(750);
System.out.println(serv[1].getCompPotenciaFonte());
serv[1].setCompMarcaMemoria("Kingston");
System.out.println(serv[1].getCompMarcaMemoria());

System.out.println("Servidor 1 Após os setters");
serv[1].imprimir();

serv[1].mudarCapacidadeMemoria();
serv[1].mudarCompSlotsMemoria();

System.out.println("-------------------------------------");
System.out.println("Servidor 1 Após as entradas de dados");
serv[1].imprimir();




Notebook note[] = new Notebook[2];
note[0] = new Notebook(mon[3], "Lenovo", "DDR4", "Kingston", "SATA", "Samsung", "1TB", "AC", "Lenovo", 2300,
note[1] = new Notebook(mon[1], "Acer", 2500, 2);
//
Desktop desk[] = new Desktop[2];

desk[0] = new Desktop(mon[2], "GTX 1060", 1, "Generica", "DDR4", "RyperX", "SSD", "Kingston", "1TB", "AC",
                4500, 1, 4, 750);
desk[1] = new Desktop(mon[0], "GEFORCE", 2);


}
```

```
run:
Teste de funcionalidades classe Servidor
Servidor 1
-----------------------------------
-----------------------------------
Marca....: null
Preço....: 0.0
Quantidade....: 0
Slots Memória....: 0
Tipo Memória....: null
Marca Memória....: null
Tipo HD....: null
Marca HD....: null
Capacidade HD....: null
Tipo Fonte....: null
Marca Fonte....: null
Potência Fonte....: 0
-----------------------------------
Numero HD....: 1
Capacidade Memoria....: 16
Tipo Gabinete....: Torre
Marca Gabinete....: Intel
-----------------------------------
750
Kingston
Servidor 1 Após os setters
-----------------------------------
-----------------------------------
Marca....: null
Preço....: 0.0
Quantidade....: 0
Slots Memória....: 0
Tipo Memória....: null
Marca Memória....: Kingston
Tipo HD....: null
Marca HD....: null
Capacidade HD....: null
Tipo Fonte....: null
Marca Fonte....: null
Potência Fonte....: 750
-----------------------------------
Numero HD....: 1
Capacidade Memoria....: 16
Tipo Gabinete....: Torre
Marca Gabinete....: Intel
-----------------------------------
--------Capacidade Memoria--------
Capacidade Memoria: 16
--------Slots Memória--------
Slots Memória: 4
-----------------------------------
Servidor 1 Após as entradas de dados
-----------------------------------
-----------------------------------
```

Visualização 2

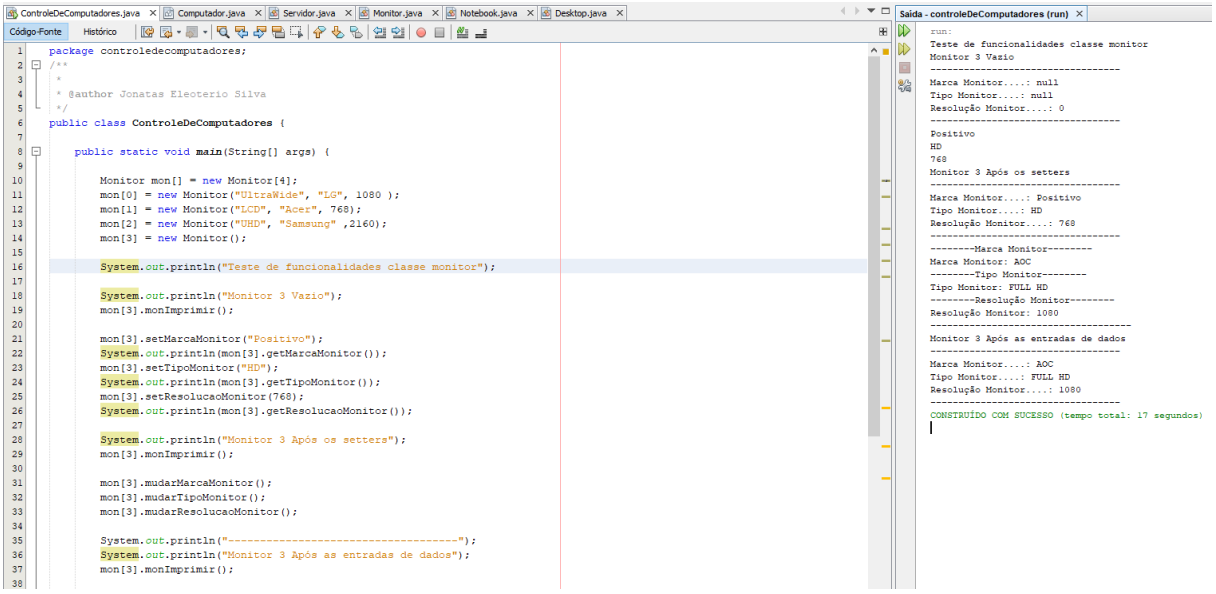


```
Servidor serv[] = new Servidor[2];
serv[0] = new Servidor(4, 16, "Torre", "DELL EMC", "DELL", "DDR4", "DELL Snpmgy", "SATA", "DELL 1TB", "1TB",
        "Dell PowerEdge", 9500, 2, 2, 1200 );
serv[1] = new Servidor(1, 16, "Torre", "Intel");


System.out.println("Teste de funcionalidades classe Servidor");

System.out.println("Servidor 1");
serv[1].imprimir();

serv[1].setCompPotenciaFonte(750);
System.out.println(serv[1].getCompPotenciaFonte());
serv[1].setCompMarcaMemoria("Kingston");
System.out.println(serv[1].getCompMarcaMemoria());

System.out.println("Servidor 1 Após os setters");
serv[1].imprimir();

serv[1].mudarCapacidadeMemoria();
serv[1].mudarCompSlotsMemoria();

System.out.println("-------------------------------------");
System.out.println("Servidor 1 Após as entradas de dados");
serv[1].imprimir();




Notebook note[] = new Notebook[2];
note[0] = new Notebook(mon[3], "Lenovo", "DDR4", "Kingston", "SATA", "Samsung", "1TB", "AC", "Lenovo", 2300,
note[1] = new Notebook(mon[1], "Acer", 2500, 2);
//
Desktop desk[] = new Desktop[2];

desk[0] = new Desktop(mon[2], "GTX 1060", 1, "Generica", "DDR4", "RyperX", "SSD", "Kingston", "1TB", "AC",
                4500, 1, 4, 750);
desk[1] = new Desktop(mon[0], "GEFORCE", 2);


}
```

```
Numero HD....: 1
Capacidade Memoria....: 16
Tipo Gabinete....: Torre
Marca Gabinete....: Intel
-----------------------------------
750
Kingston
Servidor 1 Após os setters
-----------------------------------
-----------------------------------
Marca....: null
Preço....: 0.0
Quantidade....: 0
Slots Memória....: 0
Tipo Memória....: null
Marca Memória....: Kingston
Tipo HD....: null
Marca HD....: null
Capacidade HD....: null
Tipo Fonte....: null
Marca Fonte....: null
Potência Fonte....: 750
-----------------------------------
Numero HD....: 1
Capacidade Memoria....: 16
Tipo Gabinete....: Torre
Marca Gabinete....: Intel
-----------------------------------
--------Capacidade Memoria--------
Capacidade Memoria: 16
--------Slots Memória--------
Slots Memória: 4
-----------------------------------
Servidor 1 Após as entradas de dados
-----------------------------------
-----------------------------------
Marca....: null
Preço....: 0.0
Quantidade....: 0
Slots Memória....: 4
Tipo Memória....: null
Marca Memória....: Kingston
Tipo HD....: null
Marca HD....: null
Capacidade HD....: null
Tipo Fonte....: null
Marca Fonte....: null
Potência Fonte....: 750
-----------------------------------
Numero HD....: 1
Capacidade Memoria....: 16
Tipo Gabinete....: Torre
Marca Gabinete....: Intel
-----------------------------------
CONSTRUÍDO COM SUCESSO (tempo total: 17 segundos)
```

Teste de funcionalidade da classe Monitor:

Visualização

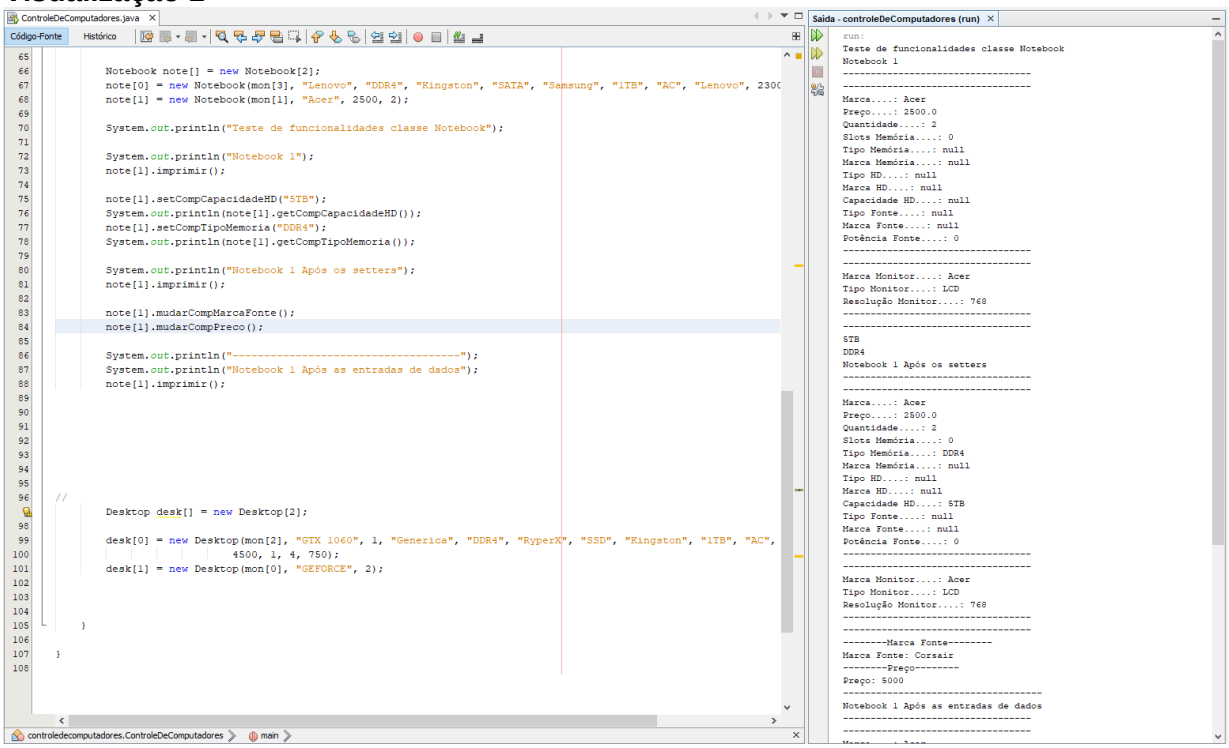


```
package controledecomputadores;
/**
 *
 * @author Jonatas Eleoterio Silva
 */
public class ControleDeComputadores {

    public static void main(String[] args) {

        Monitor mon[] = new Monitor[4];
        mon[0] = new Monitor("UltraWide", "LG", 1080 );
        mon[1] = new Monitor("LCD", "Acer", 768);
        mon[2] = new Monitor("UHD", "Samsung" ,2160);
        mon[3] = new Monitor();

        System.out.println("Teste de funcionalidades classe monitor");

        System.out.println("Monitor 3 Vazio");
        mon[3].monImprimir();

        mon[3].setMarcaMonitor("Positivo");
        System.out.println(mon[3].getMarcaMonitor());
        mon[3].setTipoMonitor("HD");
        System.out.println(mon[3].getTipoMonitor());
        mon[3].setResolucaoMonitor(768);
        System.out.println(mon[3].getResolucaoMonitor());

        System.out.println("Monitor 3 Após os setters");
        mon[3].monImprimir();

        mon[3].mudarMarcaMonitor();
        mon[3].mudarTipoMonitor();
        mon[3].mudarResolucaoMonitor();

        System.out.println("-------------------------------------");
        System.out.println("Monitor 3 Após as entradas de dados");
        mon[3].monImprimir();
```

```
run:
Teste de funcionalidades classe monitor
Monitor 3 Vazio
-------------------------------------
Marca Monitor....: null
Tipo Monitor....: null
Resolução Monitor....: 0
-------------------------------------
Positivo
HD
768
Monitor 3 Após os setters
-------------------------------------
Marca Monitor....: Positivo
Tipo Monitor....: HD
Resolução Monitor....: 768
-------------------------------------
--------Marca Monitor--------
Marca Monitor: AOC
--------Tipo Monitor--------
Tipo Monitor: FULL HD
--------Resolução Monitor--------
Resolução Monitor: 1080
-------------------------------------
Monitor 3 Após as entradas de dados
-------------------------------------
Marca Monitor....: AOC
Tipo Monitor....: FULL HD
Resolução Monitor....: 1080
-------------------------------------
CONSTRUÍDO COM SUCESSO (tempo total: 17 segundos)
```

Teste de funcionalidade da classe Notebook:

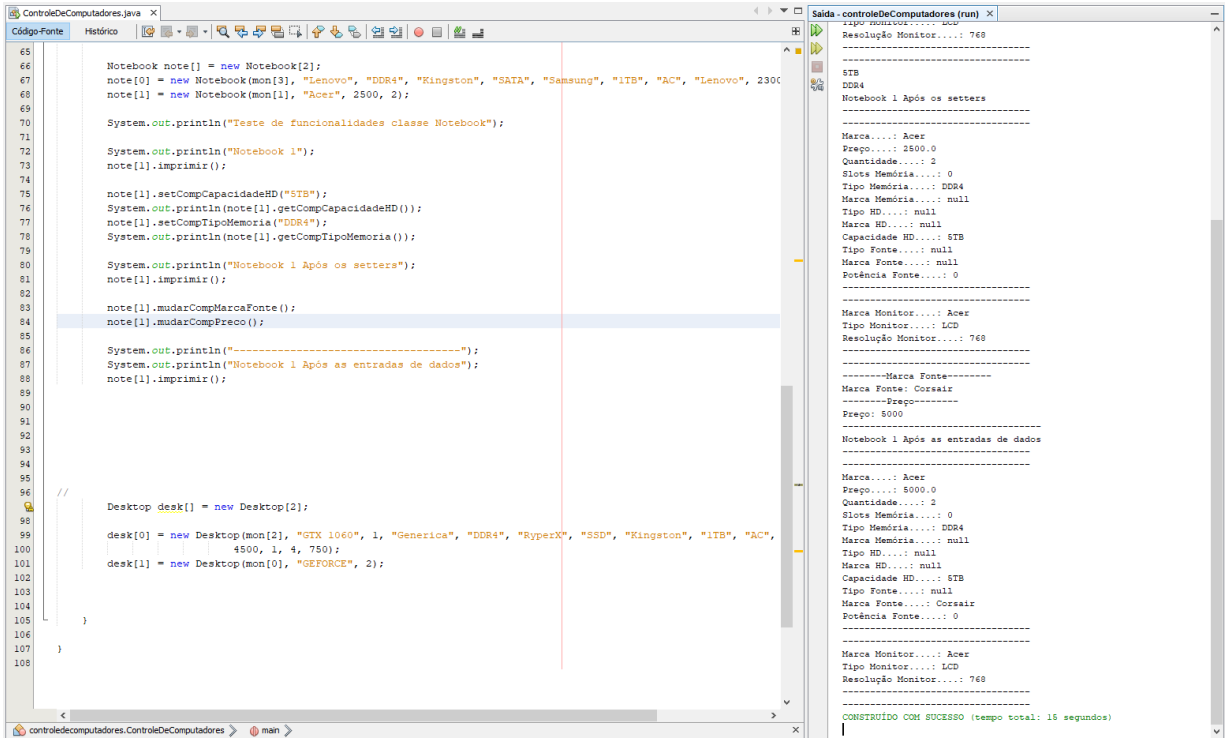
Visualização 1



```
        Notebook note[] = new Notebook[2];
        note[0] = new Notebook(mon[3], "Lenovo", "DDR4", "Kingston", "SATA", "Samsung", "1TB", "AC", "Lenovo", 2300
        note[1] = new Notebook(mon[1], "Acer", 2500, 2);

        System.out.println("Teste de funcionalidades classe Notebook");

        System.out.println("Notebook 1");
        note[1].imprimir();

        note[1].setCompCapacidadeHD("5TB");
        System.out.println(note[1].getCompCapacidadeHD());
        note[1].setCompTipoMemoria("DDR4");
        System.out.println(note[1].getCompTipoMemoria());

        System.out.println("Notebook 1 Após os setters");
        note[1].imprimir();

        note[1].mudarCompMarcaFonte();
        note[1].mudarCompPreco();

        System.out.println("-------------------------------------");
        System.out.println("Notebook 1 Após as entradas de dados");
        note[1].imprimir();

//
        Desktop desk[] = new Desktop[2];

        desk[0] = new Desktop(mon[2], "GTX 1060", 1, "Generica", "DDR4", "RyperX", "SSD", "Kingston", "1TB", "AC",
                              4500, 1, 4, 750);
        desk[1] = new Desktop(mon[0], "GEFORCE", 2);
    }
}
```

```
run:
Teste de funcionalidades classe Notebook
Notebook 1
-------------------------------------
-------------------------------------
Marca....: Acer
Preço....: 2500.0
Quantidade....: 2
Slots Memória....: 0
Tipo Memória....: null
Marca Memória....: null
Tipo HD....: null
Marca HD....: null
Capacidade HD....: null
Tipo Fonte....: null
Marca Fonte....: null
Potência Fonte....: 0
-------------------------------------
-------------------------------------
Marca Monitor....: Acer
Tipo Monitor....: LCD
Resolução Monitor....: 768
-------------------------------------
-------------------------------------
5TB
DDR4
Notebook 1 Após os setters
-------------------------------------
-------------------------------------
Marca....: Acer
Preço....: 2500.0
Quantidade....: 2
Slots Memória....: 0
Tipo Memória....: DDR4
Marca Memória....: null
Tipo HD....: null
Marca HD....: null
Capacidade HD....: 5TB
Tipo Fonte....: null
Marca Fonte....: null
Potência Fonte....: 0
-------------------------------------
-------------------------------------
Marca Monitor....: Acer
Tipo Monitor....: LCD
Resolução Monitor....: 768
-------------------------------------
-------------------------------------
--------Marca Fonte--------
Marca Fonte: Corsair
--------Preço--------
Preço: 5000
-------------------------------------
Notebook 1 Após as entradas de dados
-------------------------------------
-------------------------------------
```

Visualização 2

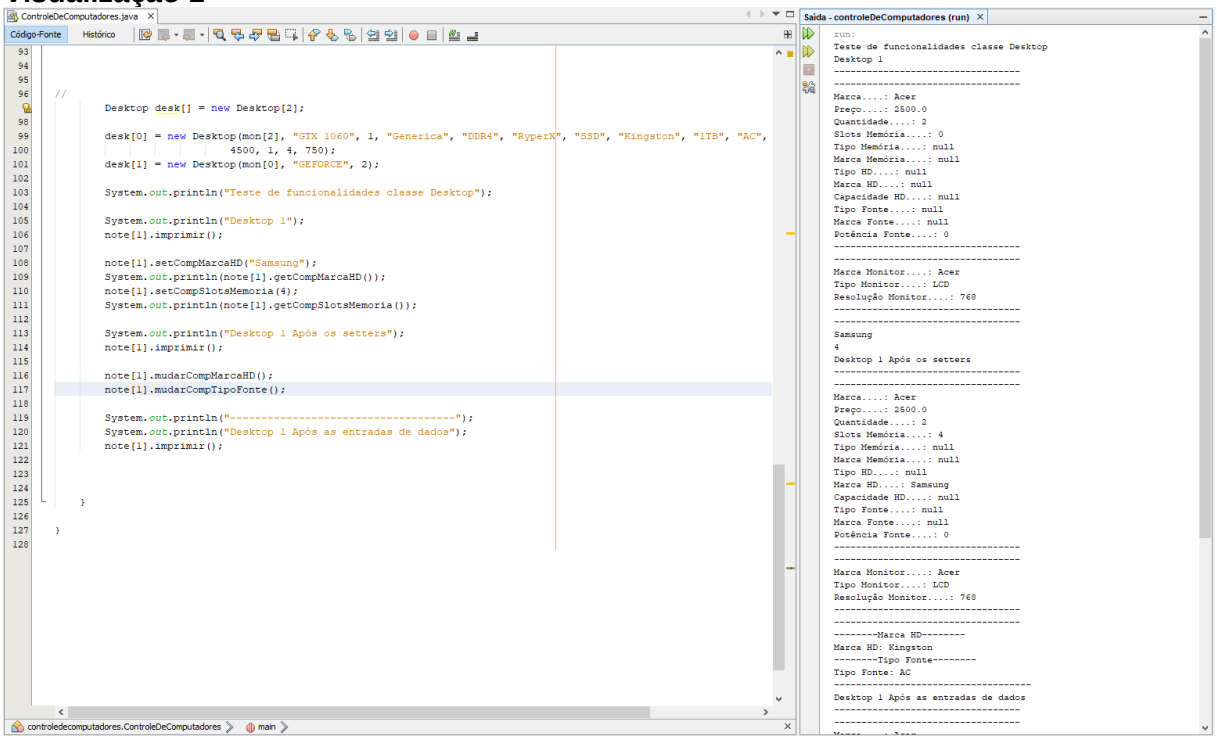


```
        Notebook note[] = new Notebook[2];
        note[0] = new Notebook(mon[3], "Lenovo", "DDR4", "Kingston", "SATA", "Samsung", "1TB", "AC", "Lenovo", 2300
        note[1] = new Notebook(mon[1], "Acer", 2500, 2);

        System.out.println("Teste de funcionalidades classe Notebook");

        System.out.println("Notebook 1");
        note[1].imprimir();

        note[1].setCompCapacidadeHD("5TB");
        System.out.println(note[1].getCompCapacidadeHD());
        note[1].setCompTipoMemoria("DDR4");
        System.out.println(note[1].getCompTipoMemoria());

        System.out.println("Notebook 1 Após os setters");
        note[1].imprimir();

        note[1].mudarCompMarcaFonte();
        note[1].mudarCompPreco();

        System.out.println("-------------------------------------");
        System.out.println("Notebook 1 Após as entradas de dados");
        note[1].imprimir();

//
        Desktop desk[] = new Desktop[2];

        desk[0] = new Desktop(mon[2], "GTX 1060", 1, "Generica", "DDR4", "RyperX", "SSD", "Kingston", "1TB", "AC",
                              4500, 1, 4, 750);
        desk[1] = new Desktop(mon[0], "GEFORCE", 2);


    }
}
```

```
Resolução Monitor....: 768
-----------------------------------
-----------------------------------
5TB
DDR4
Notebook 1 Após os setters
-----------------------------------
-----------------------------------
Marca....: Acer
Preço....: 2500.0
Quantidade....: 2
Slots Memória....: 0
Tipo Memória....: DDR4
Marca Memória....: null
Tipo HD....: null
Marca HD....: null
Capacidade HD....: 5TB
Tipo Fonte....: null
Marca Fonte....: null
Potência Fonte....: 0
-----------------------------------
-----------------------------------
Marca Monitor....: Acer
Tipo Monitor....: LCD
Resolução Monitor....: 768
-----------------------------------
-----------------------------------
--------Marca Fonte--------
Marca Fonte: Corsair
--------Preço--------
Preço: 5000
-----------------------------------
Notebook 1 Após as entradas de dados
-----------------------------------
-----------------------------------
Marca....: Acer
Preço....: 5000.0
Quantidade....: 2
Slots Memória....: 0
Tipo Memória....: DDR4
Marca Memória....: null
Tipo HD....: null
Marca HD....: null
Capacidade HD....: 5TB
Tipo Fonte....: null
Marca Fonte....: Corsair
Potência Fonte....: 0
-----------------------------------
-----------------------------------
Marca Monitor....: Acer
Tipo Monitor....: LCD
Resolução Monitor....: 768
-----------------------------------
-----------------------------------
CONSTRUÍDO COM SUCESSO (tempo total: 15 segundos)
```

Teste de funcionalidade da classe Desktop:

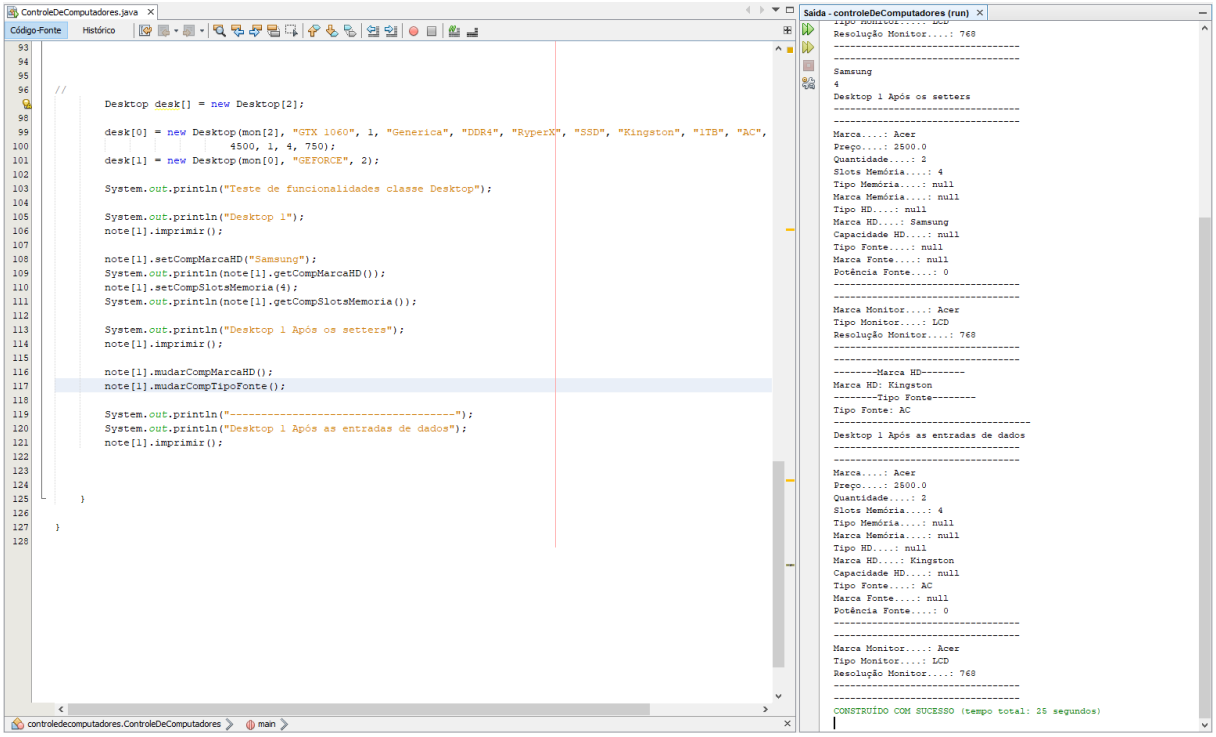
Visualização 1



```
//
        Desktop desk[] = new Desktop[2];

        desk[0] = new Desktop(mon[2], "GTX 1060", 1, "Generica", "DDR4", "RyperX", "SSD", "Kingston", "1TB", "AC",
                              4500, 1, 4, 750);
        desk[1] = new Desktop(mon[0], "GEFORCE", 2);

        System.out.println("Teste de funcionalidades classe Desktop");

        System.out.println("Desktop 1");
        note[1].imprimir();

        note[1].setCompMarcaHD("Samsung");
        System.out.println(note[1].getCompMarcaHD());
        note[1].setCompSlotsMemoria(4);
        System.out.println(note[1].getCompSlotsMemoria());

        System.out.println("Desktop 1 Após os setters");
        note[1].imprimir();

        note[1].mudarCompMarcaHD();
        note[1].mudarCompTipoFonte();

        System.out.println("-------------------------------------");
        System.out.println("Desktop 1 Após as entradas de dados");
        note[1].imprimir();


    }
}
```

```
run:
Teste de funcionalidades classe Desktop
Desktop 1
-----------------------------------
-----------------------------------
Marca....: Acer
Preço....: 2500.0
Quantidade....: 2
Slots Memória....: 0
Tipo Memória....: null
Marca Memória....: null
Tipo HD....: null
Marca HD....: null
Capacidade HD....: null
Tipo Fonte....: null
Marca Fonte....: null
Potência Fonte....: 0
-----------------------------------
-----------------------------------
Marca Monitor....: Acer
Tipo Monitor....: LCD
Resolução Monitor....: 768
-----------------------------------
-----------------------------------
Samsung
4
Desktop 1 Após os setters
-----------------------------------
-----------------------------------
Marca....: Acer
Preço....: 2500.0
Quantidade....: 2
Slots Memória....: 4
Tipo Memória....: null
Marca Memória....: null
Tipo HD....: null
Marca HD....: Samsung
Capacidade HD....: null
Tipo Fonte....: null
Marca Fonte....: null
Potência Fonte....: 0
-----------------------------------
-----------------------------------
Marca Monitor....: Acer
Tipo Monitor....: LCD
Resolução Monitor....: 768
-----------------------------------
-----------------------------------
--------Marca HD--------
Marca HD: Kingston
--------Tipo Fonte--------
Tipo Fonte: AC
-----------------------------------
Desktop 1 Após as entradas de dados
-----------------------------------
-----------------------------------
```

Visualização 2



```
//
        Desktop desk[] = new Desktop[2];

        desk[0] = new Desktop(mon[2], "GTX 1060", 1, "Generica", "DDR4", "RyperX", "SSD", "Kingston", "1TB", "AC",
                              4500, 1, 4, 750);
        desk[1] = new Desktop(mon[0], "GEFORCE", 2);

        System.out.println("Teste de funcionalidades classe Desktop");

        System.out.println("Desktop 1");
        note[1].imprimir();

        note[1].setCompMarcaHD("Samsung");
        System.out.println(note[1].getCompMarcaHD());
        note[1].setCompSlotsMemoria(4);
        System.out.println(note[1].getCompSlotsMemoria());

        System.out.println("Desktop 1 Após os setters");
        note[1].imprimir();

        note[1].mudarCompMarcaHD();
        note[1].mudarCompTipoFonte();

        System.out.println("-----------------------------------");
        System.out.println("Desktop 1 Após as entradas de dados");
        note[1].imprimir();


    }
}
```

```
Resolução Monitor....: 768
-----------------------------------
-----------------------------------
Samsung
4
Desktop 1 Após os setters
-----------------------------------
-----------------------------------
Marca....: Acer
Preço....: 2500.0
Quantidade....: 2
Slots Memória....: 4
Tipo Memória....: null
Marca Memória....: null
Tipo HD....: null
Marca HD....: Samsung
Capacidade HD....: null
Tipo Fonte....: null
Marca Fonte....: null
Potência Fonte....: 0
-----------------------------------
-----------------------------------
Marca Monitor....: Acer
Tipo Monitor....: LCD
Resolução Monitor....: 768
-----------------------------------
-----------------------------------
--------Marca HD--------
Marca HD: Kingston
--------Tipo Fonte--------
Tipo Fonte: AC
-----------------------------------
Desktop 1 Após as entradas de dados
-----------------------------------
-----------------------------------
Marca....: Acer
Preço....: 2500.0
Quantidade....: 2
Slots Memória....: 4
Tipo Memória....: null
Marca Memória....: null
Tipo HD....: null
Marca HD....: Kingston
Capacidade HD....: null
Tipo Fonte....: AC
Marca Fonte....: null
Potência Fonte....: 0
-----------------------------------
-----------------------------------
Marca Monitor....: Acer
Tipo Monitor....: LCD
Resolução Monitor....: 768
-----------------------------------
-----------------------------------
CONSTRUÍDO COM SUCESSO (tempo total: 25 segundos)
```

Arquivo Computador.java

```
package controledecomputadores;

import java.util.Scanner;

/**

*

* @author Jonatas Eleoterio Silva

*/

public abstract class Computador {

    // Atributos

    private String compMarca, compTipoMemoria, compMarcaMemoria, compTipoHD, compMarcaHD, compCapacidadeHD, compTipoFonte, compMarcaFonte;

    private float compPreco;

    private int compQuantidade, compSlotsMemoria, compPotenciaFonte;


    // Métodos Públicos

    // Entrada de Dados

    public void compEntradaDados() {
```

```
Scanner sc = new Scanner(System.in);
System.out.println("--------Computador--------");
System.out.print("Marca: ");
setCompMarca(sc.nextLine());

System.out.print("Marca Memória: ");
setCompMarcaMemoria(sc.nextLine());

System.out.print("Tipo Memória: ");
setCompTipoMemoria(sc.nextLine());

System.out.print("Tipo HD: ");
setCompTipoHD(sc.nextLine());

System.out.print("Marca HD: ");
setCompMarcaHD(sc.nextLine());

System.out.print("Capacidade HD: ");
setCompCapacidadeHD(sc.nextLine());

System.out.print("Tipo Fonte: ");
setCompTipoFonte(sc.nextLine());

System.out.print("Marca Fonte: ");
setCompMarcaFonte(sc.nextLine());

System.out.print("Preço: ");
setCompPreco(sc.nextFloat());

System.out.print("Potência Fonte: ");
setCompPotenciaFonte(sc.nextInt());
```

```
        System.out.print("Quantidade: ");
        setCompQuantidade(sc.nextInt());

        System.out.print("Slots Memória: ");
        setCompSlotsMemoria(sc.nextInt());
    }

    public void mudarCompMarca() {
        Scanner sc = new Scanner(System.in);
        System.out.println("--------Marca--------");
        System.out.print("Marca: ");
        setCompMarca(sc.nextLine());
    }

    public void mudarCompPreco() {
        Scanner sc = new Scanner(System.in);
        System.out.println("--------Preço--------");
        System.out.print("Preço: ");
        setCompPreco(sc.nextFloat());
    }

    public void mudarCompQuantidade() {
        Scanner sc = new Scanner(System.in);
        System.out.println("--------Quantidade--------");
        System.out.print("Quantidade: ");
        setCompQuantidade(sc.nextInt());
    }

    public void mudarCompSlotsMemoria() {
        Scanner sc = new Scanner(System.in);
        System.out.println("--------Slots Memória--------");
        System.out.print("Slots Memória: ");
```

```
    setCompSlotsMemoria(sc.nextInt());
}

public void mudarCompTipoMemoria() {
    Scanner sc = new Scanner(System.in);
    System.out.println("--------Tipo Memória--------");
    System.out.print("Tipo Memória: ");
    setCompTipoMemoria(sc.nextLine());
}

public void mudarCompMarcaMemoria() {
    Scanner sc = new Scanner(System.in);
    System.out.println("--------Marca Memória--------");
    System.out.print("Marca Memória: ");
    setCompMarcaMemoria(sc.nextLine());
}

public void mudarCompTipoHD() {
    Scanner sc = new Scanner(System.in);
    System.out.println("--------Tipo HD--------");
    System.out.print("Tipo HD: ");
    setCompTipoHD(sc.nextLine());
}

public void mudarCompMarcaHD() {
    Scanner sc = new Scanner(System.in);
    System.out.println("--------Marca HD--------");
    System.out.print("Marca HD: ");
    setCompMarcaHD(sc.nextLine());
}

public void mudarCompCapacidadeHD() {
```

```
    Scanner sc = new Scanner(System.in);
    System.out.println("--------Capacidade HD--------");
    System.out.print("Capacidade HD: ");
    setCompCapacidadeHD(sc.nextLine());
}

public void mudarCompTipoFonte() {
    Scanner sc = new Scanner(System.in);
    System.out.println("--------Tipo Fonte--------");
    System.out.print("Tipo Fonte: ");
    setCompTipoFonte(sc.nextLine());
}

public void mudarCompMarcaFonte() {
    Scanner sc = new Scanner(System.in);
    System.out.println("--------Marca Fonte--------");
    System.out.print("Marca Fonte: ");
    setCompMarcaFonte(sc.nextLine());
}

public void mudarCompPotenciaFonte() {
    Scanner sc = new Scanner(System.in);
    System.out.println("--------Potência Fonte--------");
    System.out.print("Potência Fonte: ");
    setCompPotenciaFonte(sc.nextInt());
}

// Imprimir
public void compImprimir() {
    System.out.println("---------------------------------");
    System.out.println("Marca....: " + getCompMarca());
    System.out.println("Preço....: " + getCompPreco());
```

```
        System.out.println("Quantidade....: " + getCompQuantidade());
        System.out.println("Slots Memória....: " + getCompSlotsMemoria());
        System.out.println("Tipo Memória....: " + getCompTipoMemoria());
        System.out.println("Marca Memória....: " + getCompMarcaMemoria());
        System.out.println("Tipo HD....: " + getCompTipoHD());
        System.out.println("Marca HD....: " + getCompMarcaHD());
        System.out.println("Capacidade HD....: " + getCompCapacidadeHD());
        System.out.println("Tipo Fonte....: " + getCompTipoFonte());
        System.out.println("Marca Fonte....: " + getCompMarcaFonte());
        System.out.println("Potência Fonte....: " + getCompPotenciaFonte());
        System.out.println("---------------------------------");
    }

    // Métodos Construtores

    public Computador(){ }

    public Computador(String compMarca, String compTipoMemoria, String
compMarcaMemoria, String compTipoHD, String compMarcaHD, String
compCapacidadeHD, String compTipoFonte, String compMarcaFonte, float compPreco, int
compQuantidade, int compSlotsMemoria, int compPotenciaFonte) {
        this.compMarca = compMarca;
        this.compTipoMemoria = compTipoMemoria;
        this.compMarcaMemoria = compMarcaMemoria;
        this.compTipoHD = compTipoHD;
        this.compMarcaHD = compMarcaHD;
        this.compCapacidadeHD = compCapacidadeHD;
        this.compTipoFonte = compTipoFonte;
        this.compMarcaFonte = compMarcaFonte;
        this.compPreco = compPreco;
        this.compQuantidade = compQuantidade;
        this.compSlotsMemoria = compSlotsMemoria;
        this.compPotenciaFonte = compPotenciaFonte;
```

```
    }

    public Computador(String compMarca, float compPreco, int compQuantidade) {
        this.compMarca = compMarca;
        this.compPreco = compPreco;
        this.compQuantidade = compQuantidade;
    }

    public Computador(String compMarca, String compMarcaMemoria, String compMarcaHD,
String compMarcaFonte) {
        this.compMarca = compMarca;
        this.compMarcaMemoria = compMarcaMemoria;
        this.compMarcaHD = compMarcaHD;
        this.compMarcaFonte = compMarcaFonte;
    }

    public Computador(String compTipoMemoria, String compTipoHD, String compTipoFonte)
{
        this.compTipoMemoria = compTipoMemoria;
        this.compTipoHD = compTipoHD;
        this.compTipoFonte = compTipoFonte;
    }

    // Setters e Getters
    public String getCompMarca() {
        return compMarca;
    }

    public void setCompMarca(String compMarca) {
        this.compMarca = compMarca;
    }

    public String getCompTipoMemoria() {
```

```
    return compTipoMemoria;
}

public void setCompTipoMemoria(String compTipoMemoria) {
    this.compTipoMemoria = compTipoMemoria;
}

public String getCompMarcaMemoria() {
    return compMarcaMemoria;
}

public void setCompMarcaMemoria(String compMarcaMemoria) {
    this.compMarcaMemoria = compMarcaMemoria;
}

public String getCompTipoHD() {
    return compTipoHD;
}

public void setCompTipoHD(String compTipoHD) {
    this.compTipoHD = compTipoHD;
}

public String getCompMarcaHD() {
    return compMarcaHD;
}

public void setCompMarcaHD(String compMarcaHD) {
    this.compMarcaHD = compMarcaHD;
}

public String getCompCapacidadeHD() {
```

```
    return compCapacidadeHD;
}

public void setCompCapacidadeHD(String compCapacidadeHD) {
    this.compCapacidadeHD = compCapacidadeHD;
}

public String getCompTipoFonte() {
    return compTipoFonte;
}

public void setCompTipoFonte(String compTipoFonte) {
    this.compTipoFonte = compTipoFonte;
}

public String getCompMarcaFonte() {
    return compMarcaFonte;
}

public void setCompMarcaFonte(String compMarcaFonte) {
    this.compMarcaFonte = compMarcaFonte;
}

public float getCompPreco() {
    return compPreco;
}

public void setCompPreco(float compPreco) {
    this.compPreco = compPreco;
}

public int getCompQuantidade() {
```

```
        return compQuantidade;
    }

    public void setCompQuantidade(int compQuantidade) {
        this.compQuantidade = compQuantidade;
    }

    public int getCompSlotsMemoria() {
        return compSlotsMemoria;
    }

    public void setCompSlotsMemoria(int compSlotsMemoria) {
        this.compSlotsMemoria = compSlotsMemoria;
    }

    public int getCompPotenciaFonte() {
        return compPotenciaFonte;
    }

    public void setCompPotenciaFonte(int compPotenciaFonte) {
        this.compPotenciaFonte = compPotenciaFonte;
    }

}
```

Arquivo Servidor.java

```
package controledecomputadores;
import java.util.Scanner;
/**
*
```

```
* @author Jonatas Eleoterio Silva
*/
public class Servidor extends Computador {

    // Atributos
    private int numeroHD, capacidadeMemoria;
    private String tipoGabinete, marcaGabinete;

    // Métodos Públicos
    // Entrada de Dados
    public void entradaDados() {
        Scanner sc = new Scanner(System.in);
        System.out.println("--------Servidor--------");
        this.compEntradaDados();
        System.out.print("Tipo Gabinete: ");
        setTipoGabinete(sc.nextLine());

        System.out.print("Tipo HD: ");
        setMarcaGabinete(sc.nextLine());

        System.out.print("Numero HD: ");
        setNumeroHD(sc.nextInt());

        System.out.print("Capacidade Memoria: ");
        setCapacidadeMemoria(sc.nextInt());

    }

    public void mudarTipoGabinete() {
        Scanner sc = new Scanner(System.in);
        System.out.println("--------Tipo Gabinete--------");
        System.out.print("Tipo Gabinete: ");
```

```
    setTipoGabinete(sc.nextLine());
}

public void mudarMarcaGabinete() {
    Scanner sc = new Scanner(System.in);
    System.out.println("--------Marca Gabinete--------");
    System.out.print("Marca Gabinete: ");
    setMarcaGabinete(sc.nextLine());
}

public void mudarNumeroHD() {
    Scanner sc = new Scanner(System.in);
    System.out.println("--------Numero HD--------");
    System.out.print("Numero HD: ");
    setNumeroHD(sc.nextInt());
}

public void mudarCapacidadeMemoria() {
    Scanner sc = new Scanner(System.in);
    System.out.println("--------Capacidade Memoria--------");
    System.out.print("Capacidade Memoria: ");
    setCapacidadeMemoria(sc.nextInt());
}


// Imprimir
public void imprimir() {

    System.out.println("---------------------------------");
    this.compImprimir();
    System.out.println("Numero HD....: " + getNumeroHD());
    System.out.println("Capacidade Memoria....: " + getCapacidadeMemoria());
```

```
        System.out.println("Tipo Gabinete....: " + getTipoGabinete());
        System.out.println("Marca Gabinete....: " + getMarcaGabinete());
        System.out.println("----------------------------------");
    }
    // Métodos Construtores

    public Servidor() {

    }

    public Servidor(int numeroHD, int capacidadeMemoria, String tipoGabinete, String
marcaGabinete, String compMarca, String compTipoMemoria, String compMarcaMemoria,
String compTipoHD, String compMarcaHD, String compCapacidadeHD, String
compTipoFonte, String compMarcaFonte, float compPreco, int compQuantidade, int
compSlotsMemoria, int compPotenciaFonte) {
        super(compMarca, compTipoMemoria, compMarcaMemoria, compTipoHD,
compMarcaHD, compCapacidadeHD, compTipoFonte, compMarcaFonte, compPreco,
compQuantidade, compSlotsMemoria, compPotenciaFonte);
        this.numeroHD = numeroHD;
        this.capacidadeMemoria = capacidadeMemoria;
        this.tipoGabinete = tipoGabinete;
        this.marcaGabinete = marcaGabinete;
    }

    public Servidor(int numeroHD, int capacidadeMemoria, String tipoGabinete, String
marcaGabinete) {
        this.numeroHD = numeroHD;
        this.capacidadeMemoria = capacidadeMemoria;
        this.tipoGabinete = tipoGabinete;
        this.marcaGabinete = marcaGabinete;
    }

    // Getters e Setters
    public int getNumeroHD() {
```

```
        return numeroHD;
    }

    public void setNumeroHD(int numeroHD) {
        this.numeroHD = numeroHD;
    }

    public int getCapacidadeMemoria() {
        return capacidadeMemoria;
    }

    public void setCapacidadeMemoria(int capacidadeMemoria) {
        this.capacidadeMemoria = capacidadeMemoria;
    }

    public String getTipoGabinete() {
        return tipoGabinete;
    }

    public void setTipoGabinete(String tipoGabinete) {
        this.tipoGabinete = tipoGabinete;
    }

    public String getMarcaGabinete() {
        return marcaGabinete;
    }

    public void setMarcaGabinete(String marcaGabinete) {
        this.marcaGabinete = marcaGabinete;
    }

}
```

Arquivo Monitor.java

```
package controledecomputadores;

import java.util.Scanner;

/**
*
* @author Jonatas Eleoterio Silva
*/
public class Monitor {
    // Atributos
    private String tipoMonitor, marcaMonitor;
    private int resolucaoMonitor;

    // Métodos Públicos
    // Entrada de Dados
    public void monEntradaDados() {
        Scanner sc = new Scanner(System.in);
        System.out.println("--------Monitor--------");
        System.out.print("Tipo Monitor: ");
        setTipoMonitor(sc.nextLine());

        System.out.print("Marca Monitor: ");
        setMarcaMonitor(sc.nextLine());

        System.out.print("Resolução Monitor: ");
        setResolucaoMonitor(sc.nextInt());
    }

    public void mudarTipoMonitor() {
        Scanner sc = new Scanner(System.in);
```

```
    System.out.println("--------Tipo Monitor--------");
    System.out.print("Tipo Monitor: ");
    setTipoMonitor(sc.nextLine());
}

public void mudarMarcaMonitor() {
    Scanner sc = new Scanner(System.in);
    System.out.println("--------Marca Monitor--------");
    System.out.print("Marca Monitor: ");
    setMarcaMonitor(sc.nextLine());
}

public void mudarResolucaoMonitor() {
    Scanner sc = new Scanner(System.in);
    System.out.println("--------Resolução Monitor--------");
    System.out.print("Resolução Monitor: ");
    setResolucaoMonitor(sc.nextInt());
}

// Imprimir
public void monImprimir() {
    System.out.println("-----------------------------------");
    System.out.println("Marca Monitor....: " + getMarcaMonitor());
    System.out.println("Tipo Monitor....: " + getTipoMonitor());
    System.out.println("Resolução Monitor....: " + getResolucaoMonitor());
    System.out.println("-----------------------------------");
}

// Métodos Construtores
public Monitor() {
}
```

```
public Monitor(String marcaMonitor) {
    this.marcaMonitor = marcaMonitor;
}

public Monitor(String tipoMonitor, String marcaMonitor, int resolucaoMonitor) {
    this.tipoMonitor = tipoMonitor;
    this.marcaMonitor = marcaMonitor;
    this.resolucaoMonitor = resolucaoMonitor;
}

// Getters e Setters
public String getTipoMonitor() {
    return tipoMonitor;
}

public void setTipoMonitor(String tipoMonitor) {
    this.tipoMonitor = tipoMonitor;
}

public String getMarcaMonitor() {
    return marcaMonitor;
}

public void setMarcaMonitor(String marcaMonitor) {
    this.marcaMonitor = marcaMonitor;
}

public int getResolucaoMonitor() {
    return resolucaoMonitor;
}

public void setResolucaoMonitor(int resolucaoMonitor) {
```

```
        this.resolucaoMonitor = resolucaoMonitor;
    }



}
```

Arquivo Notebook.java

```
package controledecomputadores;

import java.util.Scanner;

/**
*
* @author Jonatas Eleoterio Silva
*/
public class Notebook extends Computador {
    // Atributos
    private Monitor monitor;

    // Métodos Públicos
    // Entrada de Dados
    public void entradaDados() {
        Scanner sc = new Scanner(System.in);
        System.out.println("--------Notebook--------");
        this.compEntradaDados();
    }

    // Imprimir
    public void imprimir() {
        System.out.println("---------------------------------");
        this.compImprimir();
```

```
        this.monitor.monImprimir();
        System.out.println("---------------------------------");
    }
    // Métodos Construtores

    public Notebook(Monitor monitor) {
        this.monitor = monitor;
    }

    public Notebook(Monitor monitor, String compMarca, float compPreco, int
compQuantidade) {
        super(compMarca, compPreco, compQuantidade);
        this.monitor = monitor;
    }

    public Notebook(Monitor monitor, String compMarca, String compTipoMemoria, String
compMarcaMemoria, String compTipoHD, String compMarcaHD, String
compCapacidadeHD, String compTipoFonte, String compMarcaFonte, float compPreco, int
compQuantidade, int compSlotsMemoria, int compPotenciaFonte) {
        super(compMarca, compTipoMemoria, compMarcaMemoria, compTipoHD,
compMarcaHD, compCapacidadeHD, compTipoFonte, compMarcaFonte, compPreco,
compQuantidade, compSlotsMemoria, compPotenciaFonte);
        this.monitor = monitor;
    }

    // Getters e Setters
    public Monitor getMonitor() {
        return monitor;
    }

    public void setMonitor(Monitor monitor) {
        this.monitor = monitor;
    }
```

```
}
```

Arquivo Desktop.java

```
package controledecomputadores;

import java.util.Scanner;

/**
*
* @author Jonatas Eleoterio Silva
*/
public class Desktop extends Computador {
    private Monitor monitor;
    private String marcaPlacaVideo;
    private int placasVideo;


    // Métodos Públicos
    // Entrada de Dados
    public void entradaDados() {
        Scanner sc = new Scanner(System.in);
        System.out.println("--------Desktop--------");
        this.compEntradaDados();

        System.out.print("Marca PlacaVideo: ");
        setMarcaPlacaVideo(sc.nextLine());

        System.out.print("Placas Video: ");
        setPlacasVideo(sc.nextInt());
    }
```

```
public void mudarMarcaPlacaVideo() {
    Scanner sc = new Scanner(System.in);
    System.out.println("--------Marca PlacaVideo--------");
    System.out.print("Marca PlacaVideo: ");
    setMarcaPlacaVideo(sc.nextLine());
}

public void mudarPlacasVideo() {
    Scanner sc = new Scanner(System.in);
    System.out.println("--------Placas Video--------");
    System.out.print("Placas Video: ");
    setPlacasVideo(sc.nextInt());
}

// Imprimir
public void imprimir() {
    System.out.println("---------------------------------");
    this.compImprimir();
    System.out.println("Marca PlacaVideo....: " + getMarcaPlacaVideo());
    System.out.println("placas Video....: " + getPlacasVideo());
    this.monitor.monImprimir();
    System.out.println("---------------------------------");
}

// Métodos Construtores
public Desktop(Monitor monitor) {
    this.monitor = monitor;
}

public Desktop(Monitor monitor, String marcaPlacaVideo, int placasVideo) {
    this.monitor = monitor;
```

```
        this.marcaPlacaVideo = marcaPlacaVideo;
        this.placasVideo = placasVideo;
    }

    public Desktop(Monitor monitor, String marcaPlacaVideo, int placasVideo, String compMarca, String compTipoMemoria, String compMarcaMemoria, String compTipoHD, String compMarcaHD, String compCapacidadeHD, String compTipoFonte, String compMarcaFonte, float compPreco, int compQuantidade, int compSlotsMemoria, int compPotenciaFonte) {
        super(compMarca, compTipoMemoria, compMarcaMemoria, compTipoHD, compMarcaHD, compCapacidadeHD, compTipoFonte, compMarcaFonte, compPreco, compQuantidade, compSlotsMemoria, compPotenciaFonte);
        this.monitor = monitor;
        this.marcaPlacaVideo = marcaPlacaVideo;
        this.placasVideo = placasVideo;
    }



    // Getters e Setters
    public Monitor getMonitor() {
        return monitor;
    }

    public void setMonitor(Monitor monitor) {
        this.monitor = monitor;
    }

    public String getMarcaPlacaVideo() {
        return marcaPlacaVideo;
    }

    public void setMarcaPlacaVideo(String marcaPlacaVideo) {
        this.marcaPlacaVideo = marcaPlacaVideo;
```

```
    }

    public int getPlacasVideo() {
        return placasVideo;
    }

    public void setPlacasVideo(int placasVideo) {
        this.placasVideo = placasVideo;
    }


}
```

Arquivo ControleDeComputadores.java

```
package controledecomputadores;
/**
*
* @author Jonatas Eleoterio Silva
*/
public class ControleDeComputadores {

    public static void main(String[] args) {

        Monitor mon[] = new Monitor[4];
        mon[0] = new Monitor("UltraWide", "LG", 1080 );
        mon[1] = new Monitor("LCD", "Acer", 768);
        mon[2] = new Monitor("UHD", "Samsung" ,2160);
        mon[3] = new Monitor();

        Servidor serv[] = new Servidor[2];
```

```
        serv[0] = new Servidor(4, 16, "Torre", "DELL EMC", "DELL", "DDR4", "DELL Snpmgy",
"SATA", "DELL 1TB", "1TB", "AC",
                "Dell PowerEdge", 9500, 2, 2, 1200 );
        serv[1] = new Servidor(1, 16, "Torre", "Intel");


        Notebook note[] = new Notebook[2];
        note[0] = new Notebook(mon[3], "Lenovo", "DDR4", "Kingston", "SATA", "Samsung",
"1TB", "AC", "Lenovo", 2300, 1, 2, 400);
        note[1] = new Notebook(mon[1], "Acer", 2500, 2);


        Desktop desk[] = new Desktop[2];


        desk[0] = new Desktop(mon[2], "GTX 1060", 1, "Generica", "DDR4", "RyperX", "SSD",
"Kingston", "1TB", "AC", "Corser",
                        4500, 1, 4, 750);
        desk[1] = new Desktop(mon[0], "GEFORCE", 2);


        System.out.println("---------Controle de Computadores---------");


        System.out.println("Servidor 1");
        serv[1].imprimir();


        serv[1].setCompPotenciaFonte(750);
        System.out.println(serv[1].getCompPotenciaFonte());
        serv[1].setCompMarcaMemoria("Kingston");
        System.out.println(serv[1].getCompMarcaMemoria());


        System.out.println("Servidor 1 Após os setters");
        serv[1].imprimir();


        serv[1].mudarCapacidadeMemoria();
        serv[1].mudarCompSlotsMemoria();
```

```
System.out.println("-----------------------------------");
System.out.println("Servidor 1 Após as entradas de dados");
serv[1].imprimir();

System.out.println("Notebook 1");  ;
note[1].imprimir();

note[1].setCompCapacidadeHD("5TB");
System.out.println(note[1].getCompCapacidadeHD());
note[1].setCompTipoMemoria("DDR4");
System.out.println(note[1].getCompTipoMemoria());

System.out.println("Notebook 1 Após os setters");
note[1].imprimir();

note[1].mudarCompMarcaFonte();
note[1].mudarCompPreco();

System.out.println("-----------------------------------");
System.out.println("Notebook 1 Após as entradas de dados");
note[1].imprimir();

System.out.println("Desktop 1");
note[1].imprimir();

note[1].setCompMarcaHD("Samsung");
System.out.println(note[1].getCompMarcaHD());
note[1].setCompSlotsMemoria(4);
System.out.println(note[1].getCompSlotsMemoria());

System.out.println("Desktop 1 Após os setters");
note[1].imprimir();
```

```
        note[1].mudarCompMarcaHD();
        note[1].mudarCompTipoFonte();

        System.out.println("-----------------------------------");
        System.out.println("Desktop 1 Após as entradas de dados");
        note[1].imprimir();

    }

}
```

Referências

DEITEL, P. Java: como programar. 10. ed. São Paulo: Pearson Education do Brasil, 2017.

MANZANO, J. A. N. G.; COSTA JUNIOR, R. A. Java 7: programação de computadores: guia prático de introdução, orientação e desenvolvimento. São Paulo: Érica, 2011.