


 <small>DEPARTAMENTO DE ENGENHARIA DE COMPUTAÇÃO E SISTEMAS DIGITAIS</small> <b>PCS</b> <small>PCS 2302/2024 Laboratório de Fundamentos da Eng.de Computação</small>	<b>PCS-2302 / PCS-2024</b> <b>Lab. de Fundamentos de Eng. de Computação</b>
	<b>Aula 04</b>  <b>Introdução</b> <b>Paradigma de Objetos</b>  <b>Professores:</b> Marcos A. Simplicio Junior Paulo Sergio Muniz Silva

Aula 04:  
Introdução ao  
Paradigma de  
Objetos  
Autores:  
Anarosa A. F.  
Brandão  
Guilherme Gomes  
v.1.0 ago. 2013


1

   <small>DEPARTAMENTO DE ENGENHARIA DE COMPUTAÇÃO E SISTEMAS DIGITAIS</small> <b>PCS</b> <small>PCS 2302/2024 Laboratório de Fundamentos da Eng.de Computação</small>	<b>Roteiro</b>
	<ul style="list-style-type: none"><li>• Conceitos Básicos de OO<ul style="list-style-type: none"><li>– Classes e Objetos</li><li>– Atributos</li><li>– Comportamentos</li></ul></li><li>• Modelagem com UML</li><li>• Exemplos em Java</li></ul>

Aula 04:  
Introdução ao  
Paradigma de  
Objetos  
Autores:  
Anarosa A. F.  
Brandão  
Guilherme Gomes  
v.1.0 ago. 2013

2





DEPARTAMENTO DE ENGENHARIA DE COMPUTAÇÃO E SISTEMAS DIGITAIS

**PCS**

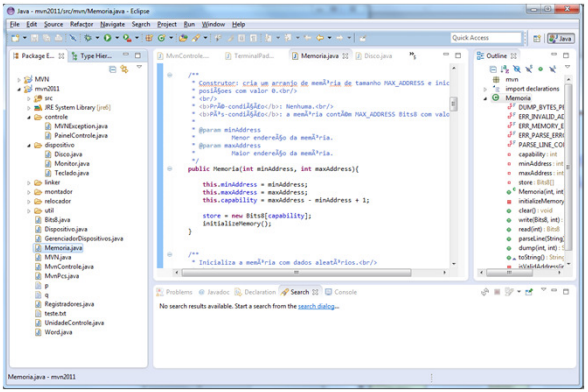
PCS 2302/2024  
Laboratório de Fundamentos da Eng.de Computação


Aula 04:  
Introdução ao Paradigma de Objetos

Autores:  
Anarosa A. F. Brandão  
Guilherme Gomes  
v.1.0 ago. 2013

## MVN como white box

- ... para entender como o simulador foi implementado, é importante entendermos alguns princípios de orientação a objetos.





DEPARTAMENTO DE ENGENHARIA DE COMPUTAÇÃO E SISTEMAS DIGITAIS

**PCS**



PCS 2302/2024  
Laboratório de Fundamentos da Eng.de Computação

Aula 04:  
Introdução ao Paradigma de Objetos


Autores:  
Anarosa A. F. Brandão  
Guilherme Gomes  
v.1.0 ago. 2013

## Diferentes paradigmas

- Principais paradigmas de desenvolvimento:
  - Estruturado
    - Modelo entrada – processamento – saída;
    - Dados separados das funções.
  - Orientado a objetos (OO)
    - O mundo é composto por objetos;
    - Objetos combinam dados e funções;
    - Conceitos do problema são modelados como objetos que são associados e interagem entre si.
- Outros
  - Funcional, Lógico, etc

DEPARTAMENTO DE ENGENHARIA DE  
COMPUTAÇÃO E SISTEMAS DIGITAIS



PCS 2302/2024  
Laboratório de  
Fundamentos da  
Eng.de Computação



Aula 04:  
Introdução ao  
Paradigma de  
Objetos

Autores:  
Anarosa A. F.  
Brandão  
Guilherme Gomes


v.1.0 ago. 2013

## Programação estruturada X OO

- Programação estruturada
  - Composição dos Programas
    - Um programa é composto por um conjunto de rotinas
    - A funcionalidade do programa é separada em rotinas
    - Os dados do programa são variáveis locais ou globais
  - Fluxo de Execução
    - O programa tem início em uma rotina principal
    - A rotina principal chama outras rotinas
    - Estas rotinas podem chamar outras rotinas, sucessivamente
    - Ao fim de uma rotina, o programa retorna para a chamadora

DEPARTAMENTO DE ENGENHARIA DE  
COMPUTAÇÃO E SISTEMAS DIGITAIS



PCS 2302/2024  
Laboratório de  
Fundamentos da  
Eng.de Computação



Aula 04:  
Introdução ao  
Paradigma de  
Objetos

Autores:  
Anarosa A. F.  
Brandão  
Guilherme Gomes

v.1.0 ago. 2013

## Programação estruturada X OO

- Programação orientada a objetos
  - Composição dos Programas
    - A funcionalidade do programa é agrupada em objetos
    - Os dados do programa são agrupados em objetos
    - Os objetos agrupam dados e funções correlacionados
  - Fluxo de Execução
    - Similar ao estruturado
    - Os objetos colaboram entre si para a solução dos objetivos
    - A colaboração se realiza através de trocas de mensagens entre objetos

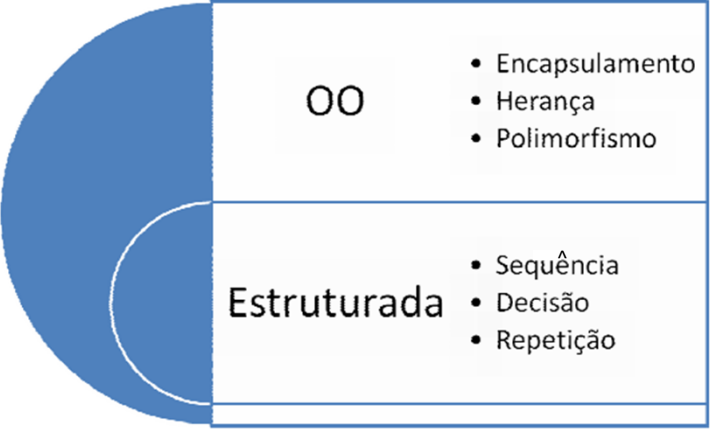



DEPARTAMENTO DE ENGENHARIA DE  
COMPUTAÇÃO E SISTEMAS DIGITAIS

**PCS**

PCS 2302/2024  
Laboratório de  
Fundamentos da  
Eng.de Computação

## Programação estruturada X OO



- Encapsulamento
- Herança
- Polimorfismo

OO



- Sequência
- Decisão
- Repetição

Estruturada

Aula 04:  
Introdução ao  
Paradigma de  
Objetos

Autores:  
Anarosa A. F.  
Brandão  
Guilherme Gomes

v.1.0 ago. 2013

DEPARTAMENTO DE ENGENHARIA DE  
COMPUTAÇÃO E SISTEMAS DIGITAIS

**PCS**

PCS 2302/2024  
Laboratório de  
Fundamentos da  
Eng.de Computação



## Objetos

- Definição
  - Um objeto é a representação computacional de um elemento ou processo do mundo real através de suas *características* e seu *comportamento*.
- Exemplos de objetos:
  - Memória, Registradores, UnidadeControle
  - Círculo, Quadrado, Triângulo
  - Cadeira, Poltrona, Sofá, Divã
  - Casa, Parede, Telhado, Porta, Janela
  - Universidade, Aluno, Professor, Curso, Disciplina

Aula 04:  
Introdução ao  
Paradigma de  
Objetos

Autores:  
Anarosa A. F.  
Brandão  
Guilherme Gomes

v.1.0 ago. 2013

DEPARTAMENTO DE ENGENHARIA DE  
COMPUTAÇÃO E SISTEMAS DIGITAIS

**PCS**

PCS 2302/2024  
Laboratório de  
Fundamentos da  
Eng.de Computação



Aula 04:  
Introdução ao  
Paradigma de  
Objetos

Autores:  
Anarosa A. F.  
Brandão  
Guilherme Gomes

v.1.0 ago. 2013

## Objetos: características

- Definição
  - Uma característica de um objeto é um elemento que descreve uma propriedade do objeto. É chamado *atributo* do objeto. O conjunto de atributos de um objeto define seu *estado*.
- Exemplos de atributos:
  - O objeto Memória tem como atributos, sua capacidade, o endereço inicial e o endereço final
  - O objeto Círculo tem como atributos, seu Raio e coordenadas de centro
  - O objeto Cadeira tem como atributos sua cor, seu modelo, seu preço
  - etc

DEPARTAMENTO DE ENGENHARIA DE  
COMPUTAÇÃO E SISTEMAS DIGITAIS

**PCS**

PCS 2302/2024  
Laboratório de  
Fundamentos da  
Eng.de Computação



Aula 04:  
Introdução ao  
Paradigma de  
Objetos

Autores:  
Anarosa A. F.  
Brandão  
Guilherme Gomes

v.1.0 ago. 2013

## Objetos: comportamento

- Definição
  - Um comportamento representa uma ação ou resposta de um objeto a uma ação do mundo real. Cada comportamento é descrito através de um *método* do objeto.
- Exemplos de métodos:
  - O objeto Memória tem métodos *ler* e *escrever*.
  - O objeto Círculo tem método *calcularArea*
  - O objeto Cadeira tem método *combinarCadeiraMesa*
  - etc

DEPARTAMENTO DE ENGENHARIA DE  
COMPUTAÇÃO E SISTEMAS DIGITAIS

**PCS**

PCS 2302/2024  
Laboratório de  
Fundamentos da  
Eng.de Computação

Aula 04:  
Introdução ao  
Paradigma de  
Objetos

Autores:  
Anarosa A. F.  
Brandão  
Guilherme Gomes

v.1.0 ago. 2013

## Mapeamento Mundo Real – Computacional visão OO

Objeto no Mundo Real

Características



Comportamento

→

Objeto Computacional

Atributos

Métodos

DEPARTAMENTO DE ENGENHARIA DE  
COMPUTAÇÃO E SISTEMAS DIGITAIS

**PCS**

PCS 2302/2024  
Laboratório de  
Fundamentos da  
Eng.de Computação

Aula 04:  
Introdução ao  
Paradigma de  
Objetos

Autores:  
Anarosa A. F.  
Brandão  
Guilherme Gomes

v.1.0 ago. 2013

## Orientação a objetos: fundamentos

- Encapsulamento
- Herança
- Polimorfismo

**USP**

DEPARTAMENTO DE ENGENHARIA DE  
COMPUTAÇÃO E SISTEMAS DIGITAIS

**PCS**

PCS 2302/2024  
Laboratório de  
Fundamentos da  
Eng.de Computação

Aula 04:  
Introdução ao  
Paradigma de  
Objetos

Autores:  
Anarosa A. F.  
Brandão  
Guilherme Gomes

v.1.0 ago. 2013

## Orientação a objetos: encapsulamento

**USP**

DEPARTAMENTO DE ENGENHARIA DE  
COMPUTAÇÃO E SISTEMAS DIGITAIS

**PCS**

PCS 2302/2024  
Laboratório de  
Fundamentos da  
Eng.de Computação

Aula 04:  
Introdução ao  
Paradigma de  
Objetos



Autores:  
Anarosa A. F.  
Brandão  
Guilherme Gomes

v.1.0 ago. 2013

## Orientação a objetos: encapsulamento

- Através do encapsulamento
  - Os métodos formam uma “cerca” em torno dos atributos
  - Os atributos *não podem* ser manipulados diretamente
  - Os atributos somente podem ser alterados ou consultados *através dos métodos do objeto*



DEPARTAMENTO DE ENGENHARIA DE  
COMPUTAÇÃO E SISTEMAS DIGITAIS

PCS 2302/2024  
Laboratório de  
Fundamentos da  
Eng.de Computação



Aula 04:  
Introdução ao  
Paradigma de  
Objetos

Autores:  
Anarosa A. F.  
Brandão  
Guilherme Gomes

v.1.0 ago. 2013

## Orientação a objetos: mensagens

- Um *cliente* de um objeto XPTO é um objeto que *usa* métodos definidos em XPTO.
- Um objeto usa um método de outro objeto através do envio (ou troca) de mensagens. O envio de uma mensagem possui:
  - Emissor
  - Receptor
  - Seletor de mensagens (nome do método chamado)
  - Parâmetros (opcionais)
- Uma mensagem pode retornar um valor

DEPARTAMENTO DE ENGENHARIA DE  
COMPUTAÇÃO E SISTEMAS DIGITAIS

PCS 2302/2024  
Laboratório de  
Fundamentos da  
Eng.de Computação

Aula 04:  
Introdução ao  
Paradigma de  
Objetos

Autores:  
Anarosa A. F.  
Brandão  
Guilherme Gomes

v.1.0 ago. 2013

## Orientação a objetos: mensagens



B é cliente de A e de D  
A e D são clientes de C



```



graph TD
    A[A: op1, op2] -- Message --> B[B: op3, op4, op5]
    B -- Message --> C[C: op6, op7, op8, op9]
    C -- Message --> D[D: op10, op11]
    D -- Return value --> C
    C -- Return value --> B
    B -- Return value --> A
            
```

Chamadas em B:  
A.op1();  
variavel=D.op10();

Fonte: Pressman, 2005

  <p>DEPARTAMENTO DE ENGENHARIA DE COMPUTAÇÃO E SISTEMAS DIGITAIS</p> <p><b>PCS</b></p> <p>PCS 2302/2024 Laboratório de Fundamentos da Eng.de Computação</p> <p>Aula 04: Introdução ao Paradigma de Objetos</p> <p>Autores: Anarosa A. F. Brandão Guilherme Gomes</p> <p>v.1.0 ago. 2013</p>	<h2>Orientação a objetos: benefícios</h2>
	<ul style="list-style-type: none"> <li>• Benefícios do encapsulamento             <ul style="list-style-type: none"> <li>– Clientes de um objeto podem fazer uso de seus métodos sem conhecer os detalhes de sua implementação (basta conhecer o comportamento esperado, receptor, nome do método chamado e parâmetros)</li> <li>– A implementação de um objeto pode ser alterada sem o conhecimento de seus clientes, desde que mantida a interface visível (receptor, nome do método chamado e parâmetros).</li> <li>– É possível “esconder” de verdade, por questões de segurança, recursos que não devem ser acessados</li> </ul> </li> </ul>

  <p>DEPARTAMENTO DE ENGENHARIA DE COMPUTAÇÃO E SISTEMAS DIGITAIS</p> <p><b>PCS</b></p> <p>PCS 2302/2024 Laboratório de Fundamentos da Eng.de Computação</p> <p>Aula 04: Introdução ao Paradigma de Objetos</p> <p>Autores: Anarosa A. F. Brandão Guilherme Gomes</p> <p>v.1.0 ago. 2013</p>	<h2>Orientação a objetos: classes</h2>
	<ul style="list-style-type: none"> <li>• A adoção do paradigma de objetos pressupõe o uso de Classes.</li> <li>• Classes descrevem as características e o comportamento de um conjunto de objetos através de seus atributos (parte estática) e métodos (parte dinâmica), respectivamente.             <ul style="list-style-type: none"> <li>– Atributos são descritos por variáveis, que podem ser de tipos primitivos ou definidos por outros objetos</li> </ul> </li> </ul>

PCS 2302/2024  
Laboratório de  
Fundamentos da  
Eng.de Computação

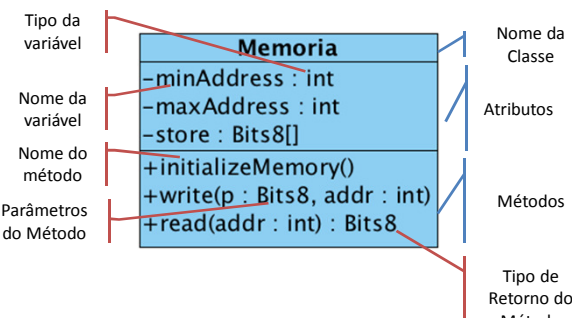
Aula 04:  
Introdução ao  
Paradigma de  
Objetos



Autores:  
Anarosa A. F.  
Brandão  
Guilherme Gomes

v.1.0 ago. 2013

## Orientação a objetos: classes

- A classe **Memoria**, do simulador MVN possui:
  - 2 atributos descritos por números inteiros (int) e um atributo descrito por um vetor de objetos Bits8.
  - 3 métodos: um sem parâmetros de entrada nem tipo de retorno; um apenas com parâmetros de entrada e um com parâmetro de entrada e saída.



PCS 2302/2024  
Laboratório de  
Fundamentos da  
Eng.de Computação



Aula 04:  
Introdução ao  
Paradigma de  
Objetos

Autores:  
Anarosa A. F.  
Brandão  
Guilherme Gomes

v.1.0 ago. 2013

## Orientação a objetos: classes

- Toda classe é responsável pela criação de seus objetos. Isto é feito através de um (ou mais) método(s), chamado de *construtor(es)*.
  - Métodos construtores têm o nome da classe que vai gerar os objetos
  - Existem também os métodos *destrutores*, responsáveis pela destruição do objeto após seu uso.

PCS 2302/2024  
Laboratório de  
Fundamentos da  
Eng.de Computação

Aula 04:  
Introdução ao  
Paradigma de  
Objetos

Autores:  
Anarosa A. F.  
Brandão  
Guilherme Gomes



v.1.0 ago. 2013

## Orientação a objetos: classes em Java

```

public class Memoria {
    public final static int MIN_ADDRESS = 0x0000;
    public final static int MAX_ADDRESS = 0xFFFF;
    public final static int CAPACITY = MAX_ADDRESS - MIN_ADDRESS + 1;
    private Bits8[] store;
    /**
     * Construtor: cria um arranjo de memória de tamanho MAX_ADDRESS e inicia as
     * posições com valor 0.<br>
     * <b>Pós-condição </b>: a memória contém MAX_ADDRESS Bits8 com valor
     * zerado.
     */
    public Memoria() {
        store = new Bits8[CAPACITY];
        for (int i = 0; i < CAPACITY; i++) {
            store[i] = new Bits8(0);
        }
    }

    //CONTINUA NO PRÓXIMO SLIDE.....
  
```

PCS 2302/2024  
Laboratório de  
Fundamentos da  
Eng.de Computação

Aula 04:  
Introdução ao  
Paradigma de  
Objetos

Autores:  
Anarosa A. F.  
Brandão  
Guilherme Gomes

v.1.0 ago. 2013

## Orientação a objetos: classes

```



/**
 * Escreve um byte em um endereço da memória. <br>
 * <b>Pré-condição </b>: o Bits8 a ser escrito não pode ser nulo. <br>
 * <b>Pós-condição </b>: a posição de memória <i>addr</i> contém o valor
 * <i>p</i>.
 * @param p
 *         O Bits8 a ser escrito.
 * @param addr
 *         O endereço o qual o Bits8 será escrito (entre MIN_ADDRESS e
 *         MAX_ADDRESS).
 */
  
```

```

public void write(Bits8 p, int addr) throws MVNException {
    // TO DO Código para verificar se o endereço é válido.
    // TO DO armazena p no endereço indicado
    return;
}

/**
 * Retorna um Byte8 armazenado em um endereço da memória.
 * @param addr
 *         O endereço de memória do Bits8 a ser retornado.
 */
public Bits8 read(int addr) throws MVNException {
    if (!validAddress(addr)) {
        throw new MVNException("Endereco de memoria invalido [" + addr + "]");
    }
    return store[addr - MIN_ADDRESS];
}
  
```

Abstração procedural  
– gera JavaDoc

DEPARTAMENTO DE ENGENHARIA DE  
COMPUTAÇÃO E SISTEMAS DIGITAIS

PCS 2302/2024  
Laboratório de  
Fundamentos da  
Eng.de Computação



Aula 04:  
Introdução ao  
Paradigma de  
Objetos

Autores:  
Anarosa A. F.  
Brandão  
Guilherme Gomes

v.1.0 ago. 2013

## Orientação a objetos: herança

- Classes são organizadas em estruturas hierárquicas
  - Uma classe pode herdar características e comportamento de outras classes
  - A classe que forneceu os elementos herdados é chamada de **superclasse**
  - A classe herdeira é chamada de **subclasse**
  - A subclasse herda todos os métodos e atributos de suas superclasses
  - A subclasse pode definir novos atributos e métodos específicos

DEPARTAMENTO DE ENGENHARIA DE  
COMPUTAÇÃO E SISTEMAS DIGITAIS

PCS 2302/2024  
Laboratório de  
Fundamentos da  
Eng.de Computação

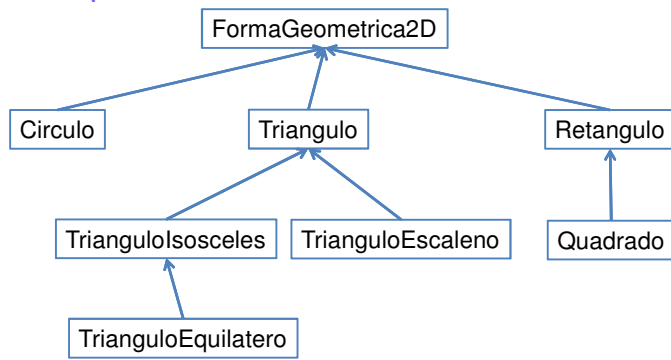
Aula 04:  
Introdução ao  
Paradigma de  
Objetos

Autores:  
Anarosa A. F.  
Brandão  
Guilherme Gomes

v.1.0 ago. 2013


## Orientação a objetos: herança

- Classes são organizadas em estruturas hierárquicas



```

graph BT
    FormaGeometrica2D --> Circulo
    FormaGeometrica2D --> Triangulo
    FormaGeometrica2D --> Retangulo
    Triangulo --> TrianguloIsosceles
    Triangulo --> TrianguloEscaleno
    TrianguloEquilatero --> TrianguloIsosceles
    Quadrado --> Retangulo
            
```



PCS 2302/2024  
Laboratório de  
Fundamentos da  
Eng.de Computação

Aula 04:  
Introdução ao  
Paradigma de  
Objetos

Autores:  
Anarosa A. F.  
Brandão  
Guilherme Gomes


v.1.0 ago. 2013

## Orientação a objetos: herança em Java

```

package mvn;
import java.util.*;
/**
 * Representação de um Byte para a MVN.
 * @author PSMuniz
 * @version 1.0 - PCS/EPUSP
 * @viz.diagram Bits8.tpx
 */
public class Bits8 extends BitSet {
/**
 * Constrói um Bits8 com o determinado valor inteiro.
 */
public Bits8(int x) {
    super(8); // pattern of 8 bits, using constructor for BitSet
    for (int pos = 0; pos < 8; pos++) {
        super.clear(pos);
    } // 'clear' is a method in class BitSet
    int quot = x;
    int rem = 0;
    int position = 7;
    while (quot > 0 && position >= 0) {
        rem = quot % 2;
        if (rem == 1)
            super.set(position); // 'set' is a method in class BitSet
        quot = quot / 2;
        position--;
    }
}

```



PCS 2302/2024  
Laboratório de  
Fundamentos da  
Eng.de Computação



Aula 04:  
Introdução ao  
Paradigma de  
Objetos

Autores:  
Anarosa A. F.  
Brandão  
Guilherme Gomes

v.1.0 ago. 2013

## Orientação a objetos: herança em Java

<http://docs.oracle.com/javase/7/docs/api/java/util/BitSet.html>

PCS 2302/2024  
Laboratório de  
Fundamentos da  
Eng.de Computação



Aula 04:  
Introdução ao  
Paradigma de  
Objetos

Autores:  
Anarosa A. F.  
Brandão  
Guilherme Gomes

v.1.0 ago. 2013

## Orientação a objetos: polimorfismo

- Classes organizadas hierarquicamente podem redefinir características ou comportamentos herdados
- O mecanismo de programação que processam objetos que compartilham a mesma superclasse como se todos fossem objetos da superclasse é chamado de **polimorfismo**

PCS 2302/2024  
Laboratório de  
Fundamentos da  
Eng.de Computação

Aula 04:  
Introdução ao  
Paradigma de  
Objetos

Autores:  
Anarosa A. F.  
Brandão  
Guilherme Gomes

v.1.0 ago. 2013

## Orientação a objetos: polimorfismo

```



public class GerenciadorDispositivos {

    /** Escreve um Bits8 em um dispositivo. (...) */
    public void escreverDispositivo(int deviceType, int logicalUnit,
                                    Bits8 outData) throws MVNException {
        Dispositivo dispositivo = getDevice(deviceType, logicalUnit);
        dispositivo.escrever(outData);
    }

    /** Lê um Bits8 (byte da MVN) de um dispositivo. (...) */
    public Bits8 lerDispositivo(int deviceType, int logicalUnit)
        throws MVNException {
        Dispositivo dispositivo = getDevice(deviceType, logicalUnit);
        return dispositivo.escrever(outData);
    }
}

```

Usam métodos de objetos da classe "Dispositivo", independentemente do tipo específico de instância (Disco, Monitor, Teclado, etc)

PCS 2302/2024  
Laboratório de  
Fundamentos da  
Eng.de Computação

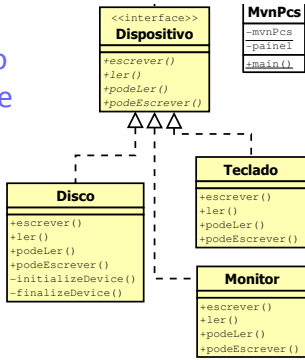
Aula 04:  
Introdução ao  
Paradigma de  
Objetos

Autores:  
Anarosa A. F.  
Brandão  
Guilherme Gomes

v.1.0 ago. 2013



## Polimorfismo e interfaces

- Classes em Java só possam herdar (via **extends**) de uma classe, mas podem implementar quantas **interfaces** forem necessárias
- **Interface** (não confundir com interfaces visuais – GUIs):  
“Contratos” de código, de modo que uma classe que adere a este contrato (via palavra reservada **“implements”**) garante que os serviços do contrato serão implementados por ela.



```

classDiagram
    class Dispositivo {
        <<interface>>
        +escrever()
        +ler()
        +podeLer()
        +podeEscrever()
    }
    class MvnPcs {
        -mvnPcs
        -panel
        +main()
    }
    class Disco {
        +escrever()
        +ler()
        +podeLer()
        +podeEscrever()
        -initializeDevice()
        -finalizeDevice()
    }
    class Teclado {
        +escrever()
        +ler()
        +podeLer()
        +podeEscrever()
    }
    class Monitor {
        +escrever()
        +ler()
        +podeLer()
        +podeEscrever()
    }
    Dispositivo <|-- MvnPcs
    Dispositivo <|.. Disco
    Dispositivo <|.. Teclado
    Dispositivo <|.. Monitor
    
```

PCS 2302/2024  
Laboratório de  
Fundamentos da  
Eng.de Computação

Aula 04:  
Introdução ao  
Paradigma de  
Objetos

Autores:  
Anarosa A. F.  
Brandão  
Guilherme Gomes

v.1.0 ago. 2013



## Polimorfismo e interfaces: Exemplos em Java

```


package mvn;
import mvn.controle.MVNException;
public interface Dispositivo{
    public static final String ERR_WRITEONLYDEVICE= "Dispositivo
        \">%s\%\" disponível somente para escrita.";
    public static final String ERR_READONLYDEVICE= "Dispositivo
        \">%s\%\" disponível somente para leitura.";

    public void escrever(Bits8 in) throws MVNException;
    public Bits8 ler() throws MVNException;
    public boolean podeLer();
    public boolean podeEscrever();
    public void reset() throws MVNException;
    public Bits8 skip(Bits8 val) throws MVNException;
    public Bits8 position() throws MVNException;
    public Bits8 size() throws MVNException;
}
    
```



DEPARTAMENTO DE ENGENHARIA DE  
COMPUTAÇÃO E SISTEMAS DIGITAIS



PCS 2302/2024  
Laboratório de  
Fundamentos da  
Eng.de Computação

Aula 04:  
Introdução ao  
Paradigma de  
Objetos

Autores:  
Anarosa A. F.  
Brandão  
Guilherme Gomes

v.1.0 ago. 2013



## Polimorfismo e interfaces: Exemplos em Java

```


package mvn.dispositivo;

public class Disco implements Dispositivo {

    public Disco(String arquivo, char modoOperacao) throws MVNException {
        switch (modoOperacao) {
            case MODO_LEITURA:
                this.modoOperacao = LEITURA;
                break;
            case MODO_ESCRITA:
                this.modoOperacao = ESCRITA;
                break;
            case MODO_LEITURAESCRITA:
                this.modoOperacao = LEITURAESCRITA;
                break;
            default:
                this.modoOperacao = INVALIDO; }
        this.arquivo = new File(arquivo);
        outFile = null;
        inFile = null;
        initializeDevice();
    }
  
```

DEPARTAMENTO DE ENGENHARIA DE  
COMPUTAÇÃO E SISTEMAS DIGITAIS



PCS 2302/2024  
Laboratório de  
Fundamentos da  
Eng.de Computação

Aula 04:  
Introdução ao  
Paradigma de  
Objetos

Autores:  
Anarosa A. F.  
Brandão  
Guilherme Gomes


v.1.0 ago. 2013

## Polimorfismo e interfaces: Exemplos em Java

```

package mvn.dispositivo;

public class Disco implements Dispositivo {
    ...
    public void escrever(Bits8 in) throws MVNException{
        //código de escrever} ;
    public Bits8 ler() throws MVNException{
        //código de ler} ;
    public boolean podeLer(){
        return modoOperacao == LEITURA || modoOperacao == LEITURAESCRITA;}
    ...
  
```



PCS 2302/2024  
Laboratório de  
Fundamentos da  
Eng.de Computação

Aula 04:  
Introdução ao  
Paradigma de  
Objetos

Autores:  
Anarosa A. F.  
Brandão  
Guilherme Gomes

v.1.0 ago. 2013

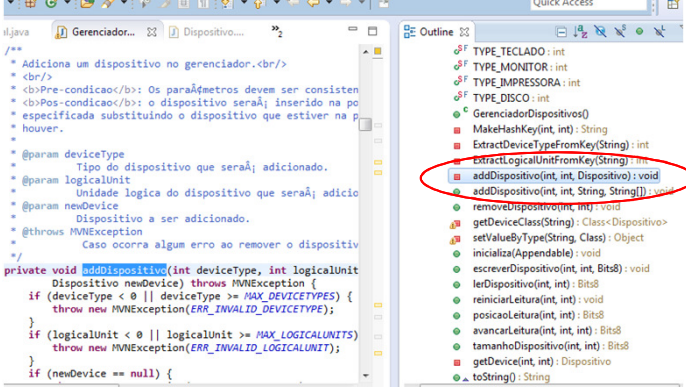
## Orientação a objetos: modelagem


### Diagrama de Classes

```

classDiagram
    class GerenciadorDispositivos {
        +addDispositivo(deviceType : int, logicalUnit : int, deviceClass : String, params : String [])
        +removeDispositivo(deviceType : int, logicalUnit : int)
    }
    class Dispositivo {
        +escrever(in : Bits8)
        +ler() : Bits8
    }
    GerenciadorDispositivos "0..*" -- "1" Dispositivo
    
```

**Aggregação**  
(Relacionamento “tem-um”)





PCS 2302/2024  
Laboratório de  
Fundamentos da  
Eng.de Computação

Aula 04:  
Introdução ao  
Paradigma de  
Objetos

Autores:  
Anarosa A. F.  
Brandão  
Guilherme Gomes

v.1.0 ago. 2013

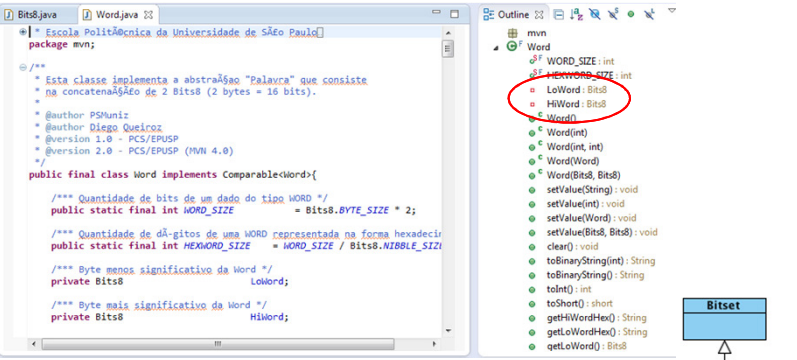
## Orientação a objetos: modelagem



### Diagrama de Classes

```

classDiagram
    class Word {
        +loWord : Bits8
        +hiWord : Bits8
        +clear()
        +setValue(loWord : Bits8, hiWord : Bits8)
    }
    class Bits8 {
        +BYTE_SIZE : int
        +NIBBLE_SIZE : int
        +setValue(value : byte)
        +shiftLeft()
        +shiftRight()
    }
    Word "2" -- "1" Bits8
    
```

**Composição**  
(Relacionamento “Todo – Parte”)



DEPARTAMENTO DE ENGENHARIA DE  
COMPUTAÇÃO E SISTEMAS DIGITAIS

**PCS**

PCS 2302/2024  
Laboratório de  
Fundamentos da  
Eng.de Computação

Aula 04:  
Introdução ao  
Paradigma de  
Objetos



Autores:  
Anarosa A. F.  
Brandão  
Guilherme Gomes

v.1.0 ago. 2013

## Orientação a objetos: modelagem

- Por que modelar o software antes de construir?
  - Porque se desenha uma casa no papel antes de construí-la em concreto?
    - Para garantir que ela não vai desabar!
    - Para garantir que as portas e janelas serão do tamanho correto para encaixar nas paredes
  - Software que é desenhado antes de ser construído possui uma chance maior de funcionar corretamente...
- O que é o Design Orientado a Objetos?
  - OO Design é uma das metodologias de design mais populares para software
  - No Design OO, você começa analisando as entidades do mundo real que existem no ambiente e adiciona características e comportamentos a essas entidades

37

DEPARTAMENTO DE ENGENHARIA DE  
COMPUTAÇÃO E SISTEMAS DIGITAIS

**PCS**

PCS 2302/2024  
Laboratório de  
Fundamentos da  
Eng.de Computação

Aula 04:  
Introdução ao  
Paradigma de  
Objetos



Autores:  
Anarosa A. F.  
Brandão  
Guilherme Gomes

v.1.0 ago. 2013

## Orientação a objetos: modelagem

- Passo a Passo:
  - Identifique as entidades do mundo real em Classes e Objetos
  - Estabeleça os relacionamentos entre estas classes:
    - Uma Palavra é **Composta** por 2 Bytes
    - O Gerenciador de Dispositivos **Agrega** Dispositivos
    - Uma MVNException **Herda de** (é uma) Exception
  - Analise as ações que um objeto pode pedir que outro objeto execute
    - Crie métodos para essas ações

38



DEPARTAMENTO DE ENGENHARIA DE  
COMPUTAÇÃO E SISTEMAS DIGITAIS

# Créditos

- Parte dos slides foi cedida pela profa. Viviane Torres da Silva, da UFF

PCS 2302/2024  
Laboratório de  
Fundamentos da  
Eng.de Computação

Aula 04:

Introdução ao  
Paradigma de  
Objetos

Autores:

Anarosa A. F.  
Brandão  
Guilherme Gomes

v.1.0 ago. 2013

39