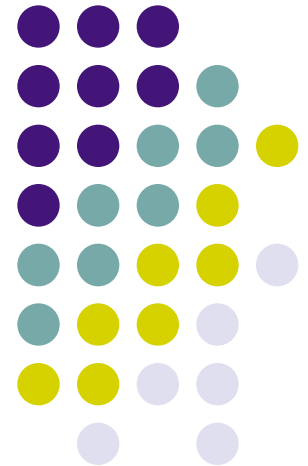


MATA 55

Programação Orientada a Objetos





Bibliografia Básica

- SANTOS, Rafael. **Introdução à Programação Orientada a Objetos Usando Java**. Ed. Campus, 1o Edição. 2003.
- HORSTMANN, Cays; CORNELL, Gary. **Core Java 2: Fundamentos**. Ed. Makron Books, 7o Edição, 2005.
- Barnes, David J; Kolling, Michael. **Programação Orientada a Objetos com Java**. Ed. Pearson-Prentice Hall.



Observações Importantes

- Chamadas
 - Serão realizadas no início e final de cada aula
- Segunda Chamada de Avaliação
 - Análise do Requerimento

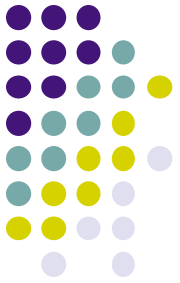


Avaliação

- 2 avaliações Teóricas
- 1 Trabalho + exercícios
- Pesos
 - Listas de Exercícios Peso 0,5
 - Primeira Avaliação Peso 1,0
 - Segunda Avaliação Peso 2,0
 - Trabalho Final Peso 1,5

Moodle

- mata5520152





Roteiro

- Paradigmas
 - Modelos
- Conceitos Básicos O.O
 - Objeto
 - Estado
 - Comportamento
 - Classe
 - Instância
 - Identidade



Paradigma

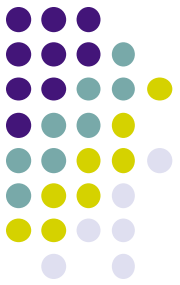
- Noção Geral
 - Modelo interpretativo ou conceitual de uma realidade
 - Ponto de Vista
 - Entendimento dessa realidade
 - Melhor forma de atuação
 - Conceitos Base
 - Exemplos
 - Políticos: Liberalismo, fascismo, comunismo, socialismo, etc
 - Econômicos: Monetarismo, Keynesianismo, etc
 - Programação: Imperativo, funcional, OO



Paradigma de Programação

- Programação
 - Processamento de informação
- Trata computacionalmente os problemas encontrados no mundo real
 - Fornece e determina a visão que o programador possui sobre a estruturação e execução do programa.
 - A forma com que o programador deve raciocinar e utilizar os recursos da linguagem.
 - Determina a forma com que o desenvolvedor do programa analisa os dados.

Alguns Paradigmas de Programação



- Paradigma imperativo
 - Conceitos: estado, atribuição, sequenciação
 - Linguagens: Basic, Pascal, C, Assembler.
- Paradigma funcional
 - Conceitos: função, aplicação, avaliação
 - Linguagens: Lisp, ML, OCaml, Haskell.
- Paradigma lógico
 - Conceitos: relação, dedução
 - Linguagens: Prolog.
- Paradigma orientado pelos objectos
 - Conceitos: objecto, mensagem
 - Linguagens: C++, Java, Eiffel.
- Paradigma concorrente
 - Conceitos: processo, comunicação (síncrona ou assíncrona)
 - Linguagens: Occam, Ada, Java.



Paradigma Programação

- O “grau de sucesso” de um programador depende em parte:
 - Coleção de paradigmas que domina
 - Da habilidade em escolher o modelo conceitual (paradigma) mais indicado para analisar e resolver cada problema

Paradigma da Orientação a Objetos



- Partem de um ponto de vista distinto e intermediário
- Mundo real é composto de *objetos*
- Objeto é uma entidade que combina estrutura de dados e comportamento funcional
- Sistemas são modelados como um número de objetos que interagem entre si

Paradigma de Programação Orientada a Objetos



- Motivação
 - Perspectiva mais humana de observação da realidade
 - Diminuir o gap semântico
 - Trabalhar com noções intuitivas durante todo o ciclo de vida
 - Uso de uma representação básica consistente para a análise e projeto
 - Retardar, ao máximo, a introdução de conceitos de implementação

Paradigma de Programação Orientada a Objetos



- Motivação (cont)
 - Capacidade de enfrentar novos domínios de aplicação;
 - Melhoria da interação entre analistas e especialistas;
 - Aumento da consistência interna dos resultados da análise;
 - Possibilidade de ciclos de desenvolvimento variados;
 - Apoio à reutilização.



Modelos

- Arquitetos, Engenheiros e outros profissionais criam modelos antes de executarem um projeto
- Na programação O.O
 - **Modelos** podem ser usados para representar e processar dados usando programas de computadores



Modelos

- Representações simplificadas
 - Objetos, pessoas, itens, tarefas, processos, conceitos, idéias etc.
 - São usados comumente por pessoas no seu dia-a-dia, independente do uso de computadores.
 - O modelo representa certos **dados** ou informações.
 - Os dados contidos no modelo são somente relevantes à abstração do mundo real feita.



Modelos

- O quadro branco é um modelo do restaurante
 - Representando de forma simplificada as informações do restaurante necessárias para contabilizar os pedidos.

Restaurante Caseiro Hipotético					
Mesa 1		Mesa 2		Mesa 3	
Kg refeição		Kg refeição		Kg refeição	
Sobremesa		Sobremesa		Sobremesa	
Refrigerante		Refrigerante		Refrigerante	
Cerveja		Cerveja		Cerveja	



Modelos

- Um modelo normalmente contém **operações** ou procedimentos associados a ele, por exemplo:
 - Inclusão de um pedido para uma mesa
 - Modificação do status de um pedido
 - Encerramento dos pedidos de uma mesa
- **Operações**
 - São listas de comandos que processarão os dados contidos no modelo
- Também é possível a criação de modelos que possuem somente dados ou somente operações.
- Modelos podem conter submodelos e ser parte de outros modelos.

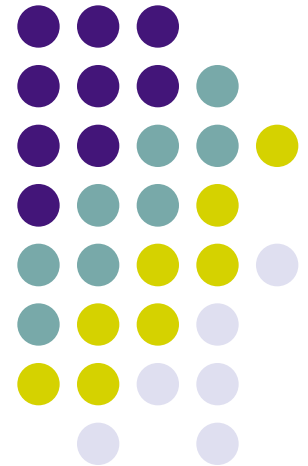


Modelos (Atividade)

- Em grupos de 4 pessoas, escolha um dos tópicos abaixo e defina um modelo que poderia ser aplicado para representá-lo de forma clara
 - Um controle remoto de televisão
 - A chamada efetuada em uma sala de aula
 - Um histórico escolar de um aluno
 - A alocação de laboratórios da faculdade
 - A ficha de um aluno na biblioteca

Orientação a Objeto

Conceitos Básicos





Objeto

- Unidade real ou abstrata
- Individualizada
- Representa um conceito da realidade humana
 - Pode ocupar
 - espaço físico (mundo físico)
 - lógico (memória)
- Entidade que incorpora uma abstração relevante para o conceito da aplicação

Objeto



- Significado semelhante aos objetos que conhecemos do mundo real.
- Objetos tem um conjunto de **características (estrutura), comportamentos** e relacionamentos entre si.
 - Descrito pela sua estrutura e comportamento
- Vejamos exemplos do mundo real: carro, casa, conta corrente, aluno, cachorro, etc.

Objeto



— Características e comportamentos do seguinte Carro:



Cor preta, vidros fumê, 2 portas, rebaixado, combustível gasolina, cd player, câmbio automático, esportivo etc.

Liga, desliga, abastece, trava portas, acelera, troca de marcha, pinta etc.

— Características e comportamentos do seguinte Carro



Cor marrom, vidros simples, 2 portas, combustível álcool, câmbio manual, passeio etc.

Liga, desliga, abastece, trava portas, acelera, troca de marcha, pinta etc.

Objeto



— Características e comportamentos do seguinte monitor:

Contraste 50, brilho 45, status ligado, 17 polegadas etc.

Liga, desliga, controla contraste, controla brilho etc



— Características e comportamentos do seguinte monitor:

Contraste 68, brilho 20, status desligado, 19 polegadas, saída pra TV etc.

Liga, desliga, controla contraste, controla brilho etc

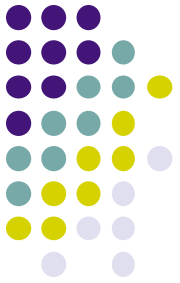




Objeto - Estado

- Estado
 - As características de um objeto em determinado momento chamamos de **estado**
 - Conjunto das propriedades de um objeto
 - Atributos e aos valores a eles associados

Objeto - Estado



- Por exemplo:
 - Se modificarmos o brilho do monitor, dizemos que este objeto mudou de estado
 - Se desligarmos o monitor ele também mudará de estado

Contraste 50, brilho 77, status ligado, 17 polegadas etc.

Liga, desliga, controla contraste, controla brilho etc



Objeto - Estado



— Como poderíamos modificar o estado do fusca?



Cor marrom, vidros simples, 2 portas,
combustível álcool, passeio etc.

Liga, desliga, abastece, trava portas,
acelera, troca de marcha, pinta etc.



Objeto - Comportamento

- Operações
 - Conjunto de operações do objeto
 - Recuperam ou manipulam a informação de estado de um objeto
 - Referem-se apenas às estruturas de dados do próprio objeto
 - Não devem acessar diretamente estruturas de outros objetos



Comportamento

- Exemplo
 - Uma lâmpada incandescente
 - Atenção para os nomes de dados e das operações.
 - A abrangência do modelo define os dados e operações. Ex.: consumo foi deixado de lado.

Lampada
estadoDaLampada
acende() apaga() mostraEstado()

Comportamento



```
Modelo Lampada // representa uma lâmpada em uso
Início do modelo
    dado estadoDaLampada; // indica se está ligada ou não
    operacao acende() // acende a lâmpada
    início
        estadoDaLampada = aceso;
    fim
    operacao apaga() // apaga a lâmpada
    início
        estadoDaLampada = apagado;
    fim
    operacao mostraEstado() // mostra o estado da lâmpada
    início
        se (estadoDaLampada == aceso)
            imprime "A lampada esta acesa"
        senao
            imprime "A lampada esta apagada";
    fim
Fim do modelo
```



Exercício

- Uma conta bancária simplificada
 - Aspectos práticos de contas reais (senhas, taxas) foram deixados de lado.
 - Defina o comportamento

ContaBancariaSimplificada
nomeDoCorrentista saldo contaÉEspecial
abreConta(nome,deposito,éEspecial) abreContaSimples(nome) deposita(valor) retira(valor) mostraDados()



Exercício - Resposta

Modelo ContaBancáriaSimplificada

Início do modelo

 dado nomeDoCorrentista,saldo,contaÉEspecial ; // dados da conta

// Inicializa simultaneamente todos os dados do modelo

 operacao abreConta(nome,deposito,"true") // argumentos para operação

 início

 nomeDoCorrentista = nome;

 saldo = deposito;

 contaÉEspecial = true;

 fim

// Inicializa simultaneamente todos os dados do modelo, usando o nome passado //
como argumento e os outros valores com valores default

 operacao abreContaSimples(nome) // argumentos para operação

 início

 nomeDoCorrentista = nome;

 saldo = 0.00;

 contaÉEspecial = false;

 fim

Exercício (continuação)



```
// Deposita um valor na conta
operacao deposita(valor) // argumentos para operação
início
    saldo = saldo + valor;
fim
// Retira um valor na conta
operacao retira(valor) // argumentos para operação
início
    se (contaÉEspecial == false) // a conta não é especial
    início
        se (valor <= saldo) // existe saldo suficiente
            saldo = saldo - valor;
        fim
    senão // conta especial, pode retirar a quantidade
        saldo = saldo - valor;
    fim
operacao mostraDados() // mostra os dados imprimindo seus valores
início
    imprime "o nome do correntista é " + nomeDoCorrentista;
    imprime "o saldo é " + saldo;
    se (contaÉEspecial) imprime "A conta é especial"
fim
Fim do modelo
```


Objeto (Atividade)



- Em grupos, selecionar uma das opções abaixo e definir características e comportamentos relacionados ao item.
 - Casa;
 - Controle remoto de televisão;
 - Jogador de futebol;
 - Celular;
 - Livro de biblioteca.
- Listar um conjunto de objetos com as características definidas.

Classe

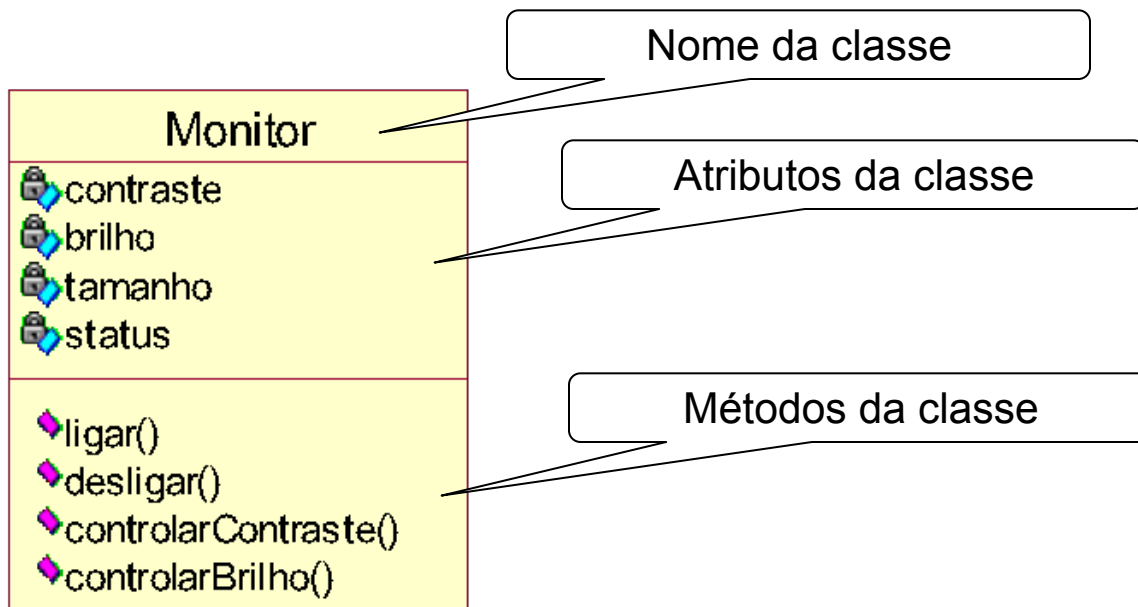


- É uma descrição de um conjunto de **objetos** que compartilham as mesmas:
 - Características,
 - Comportamentos,
 - Relacionamentos e
 - Semântica.

Classe (Modelo)



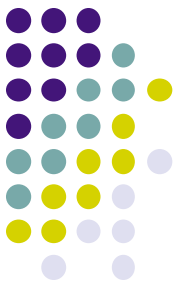
- As características chamaremos de **atributos**.
- Os comportamentos chamaremos de **métodos**.
- **Modelo em UML**



Um objeto monitor e seu estado:
Meu Monitor

contraste = 50
brilho = 80
tamanho = 15 polegadas
status = ligado

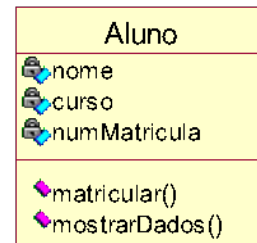




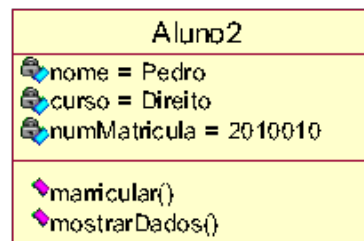
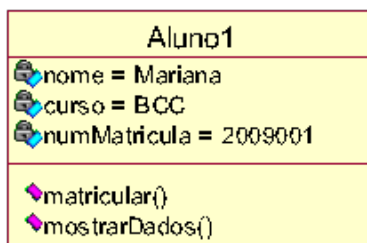
Classe

- **Objetos** são **instâncias** de classes:
 - “Materialização” do que é representado nas classes.
 - um membro descrito por uma classe
 - objetos que se comportam da maneira especificada pela classe
 - Possuem **estado** e comportamento.

Classe



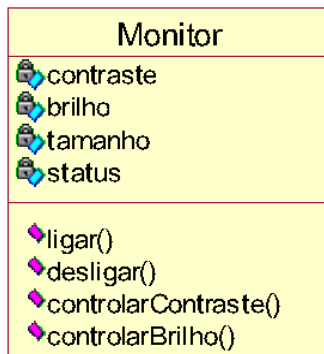
Instâncias



Classe



- Quais seriam os atributos e comportamento de um avião?
- Quais seriam os atributos e comportamento de um elevador?
 - Escrever através da representação gráfica
 - Exemplificar possíveis estados
 - Exemplo:



Um objeto monitor e seu estado:

Meu Monitor

contraste = 50

brilho = 80

tamanho = 15 polegadas

status = ligado

Classe



Avião
<ul style="list-style-type: none">capacidadePassageirosmodelocomprimentolarguraanoultimaManutencaopesoMaxDecolagem
<ul style="list-style-type: none">decolar()pousar()embarcarPassageiros()desembarcarPassageiros()abastecer()realizarManutencao()

Um objeto Avião e seu estado:

capacidadePassageiros = 200

modelo = Airbus A320

comprimento = 37,57 metros

largura = 11,76 metros

ano = 1999











ultimaManutencao = 12/01/2008

pesoMaxDecolagem = 77.000 kg



Classe



Elevador
 andarCorrente
 statusOperacional
 qtdAndares
 limitePeso
 abrirPorta()
 fecharPorta()
 subir()
 descer()
 desligar()
 ligar()

Um objeto Elevador e seu estado:

andarCorrente = 2o andar

statusOperacional = ligado/funcionando

qtdAndares = 2

limitePeso = 500 kg





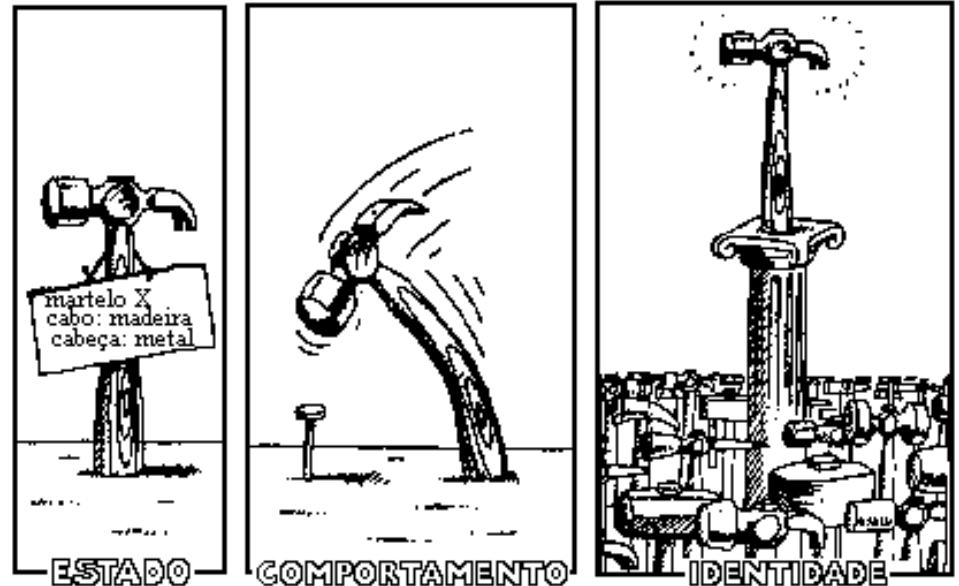
Identidade

- Cada objeto tem uma identidade própria
- Objetos têm existência própria
 - são distintos mesmo se seu estado e comportamento forem iguais
- Cada objeto dispõe de um identificador único pelo qual pode ser referenciado inequivocamente

Objeto, Estado, Identidade e Comportamento



- Um objeto possui estado, exibe algum comportamento bem definido e possui identidade própria



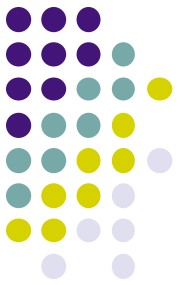
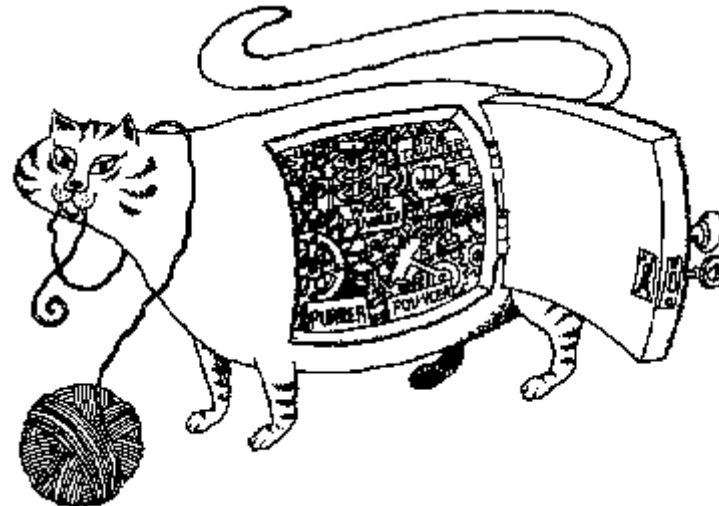


Objeto

- Um objeto é uma entidade independente que:
 - armazena dados,
 - encapsula serviços e
 - é modelado para executar funções do sistema.

Encapsulamento

- Interação entre objetos sem conhecimento do funcionamento interno





Encapsulamento

- Combinação de dados e comportamentos em uma classe
 - “escondendo” do usuário do objeto os detalhes de implementação.
- Um objeto em um programa “encapsula”
 - estado e o comportamento, de modo que podemos tratar o objeto como uma coisa só.
- Possibilitam a criação de programas com menos erros e mais clareza.

Encapsulamento (exemplos)



- Impressora:
 - Não sabemos como a impressora faz para imprimir as páginas internamente. Uma série de operações são realizadas, mas apenas solicitamos a impressão e esperamos pelo resultado.
 - Não precisamos abrir a impressora e medir o nível de tinta do cartucho. Apenas solicitamos a informação do status do nível de tinta.
- Celular:
 - Não sabemos o que o aparelho celular faz para se comunicar com a operadora e realizar as chamadas, apenas solicitamos a ligação através do número desejado. Os sub-passos estão encapsulados.
- Conta Bancária
 - Não sabemos os passos operacionais que o banco realiza para registrar um saque. Apenas solicitamos e recebemos o dinheiro.



Encapsulamento

- Previne manipulações incorretas de um objeto
- Permite que a implementação de um objeto possa ser modificada sem afetar as aplicações que usam este objeto



Exercício

Em um sistema para locadora de filmes on-line precisamos representar os filmes da locadora respondendo perguntas, tais como:

- Qual o ator principal do filme?
- Quem dirigiu o filme?
- A quantos anos o filme foi lançado?
- Quantas copias existem disponíveis para o filme?
- É um filme nacional?

Como seria representado um cliente da locadora?

Como um cliente poderia realizar a locação/devolução de um filme?