Introdução a programação Orientada a Objetos

Evolução das linguagens

- https://www.youtube.com/watch?v=VdTRiUe23os

Paradigmas de desenvolvimento

•**O que é um paradigma?**
Um exemplo, um modelo, um padrão;
Um conjunto de ideias, uma base filosófica.

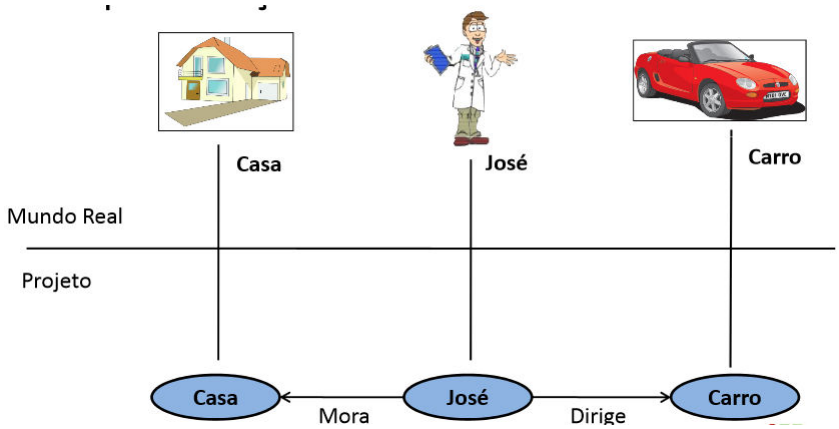
•Um paradigma de desenvolvimento agrupa métodos e técnicas que seguem um mesmo conjunto de princípios;

•Os dois mais conhecidos são:
Desenvolvimento Estruturado;
Orientação a Objetos (OO).

Paradigma Orientado a Objetos

- Sugere a **diminuição** da distância entre a **modelagem computacional** e o **mundo real**.
- Surgiu na tentativa de solucionar **problemas complexos** existentes através do desenvolvimento de softwares.
- Permite que **objetos** do **mundo real** sejam mapeados em Objetos no computador, pressupondo que o mundo é **composto por objetos**.
- Os **sistemas** são modelados como um **conjunto de objetos** que **interagem entre si**.

Representação



Programação orientada a objetos

Análise orientada a objetos

Consiste em definir quais objetos fazem parte de um sistema e a maneira como se comportam.

Programação orientada a objetos

Consiste em utilizar objetos computacionais para implementar as funcionalidades de um sistema.

Por que programar Orientado a Objetos?

- Aumento de Produtividade
- Ganho de Qualidade
- Ganho de Confiabilidade
- Conhecimento adquirido podendo ser compartilhado
- Custos

Orientação a Objetos (OO)

" Uma nova maneira de pensar os problemas utilizando conceitos do Mundo Real. O componente fundamental é o OBJETO que combina estrutura e comportamento em uma única entidade" [Raumbaugh]

" Um sistema orientado a objetos é uma coleção de objetos que interagem entre si" [Bertrand Meyer]

Conceitos da Orientação a Objetos

- Abstração
- Classe
- Objeto (Instância)
- Mensagem
- Encapsulamento
- Herança
- Polimorfismo
- ...

Abstração

"Ato de separar um ou mais elementos de uma totalidade complexa (coisa, representação, fato), os quais só mentalmente podem ser separados"

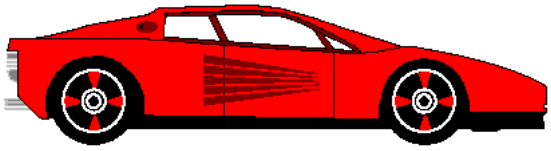
"Modelos mentais": visão simplificada do mundo construída por cada um em cada situação;

Abstrair consiste em ignorar aspectos irrelevante e concentrar nos principais.

Exemplo: Toda pessoa tem um atributo para cor dos olhos, ao modelarmos um sistema de folha de pagamento, esta informação é irrelevante, portanto não precisa ser incluída em nossa relação de atributos.

Abstração

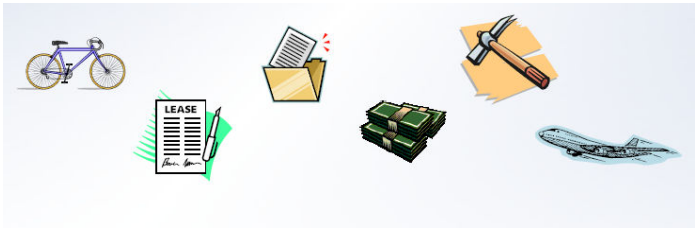
Motorista dirige um carro através dos pedais, alavanca de marchas e volante. Questões a respeito de motor estão escondidas para ele.



Objeto

OBJETO = DADOS + OPERAÇÕES

- Um objeto é qualquer indivíduo, lugar, evento, coisa, tela, relatório ou conceito que seja aplicável ao sistema.
- Podem ser coisas abstratas (ex.: uma reserva de passagem aérea) ou concretas (ex.: um documento).



OBJETO

- Um objeto é uma construção de software que encapsula **estado** e **comportamento**, através respectivamente de **propriedades** (atributos) e **operações** (métodos);

- Identidade: É o nome do objeto. Cada objeto é único.
- Exemplo: Aluno - João, José, Maria...

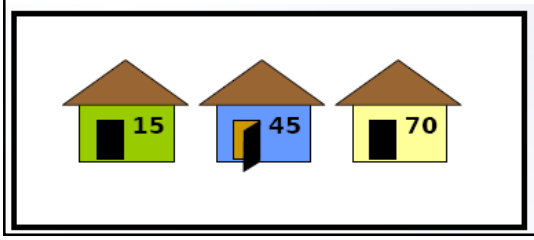
Como é um objeto?

Um objeto tem três características principais:

Estado (estrutura): conjunto de suas propriedades e seus valores correntes;

Comportamento: conjunto de serviços (operações e ações) que o objeto provê; O que você pode fazer com esse objeto.

Identidade: identificador único que diferencia cada objeto, mesmo que tenham o mesmo estado e comportamento.



CASA- Estado: altura, largura, cor, tamanho
Comportamento: arejada, clara, fria.
Identidade: número da casa

CARACTERÍSTICAS

Estado de um Objeto: composto por suas **propriedades** e seus respectivos **valores**;

Comportamento: a maneira como o objeto reage quando o seu estado é **alterado** ou quando uma mensagem é **recebida**.

•Exemplo:

- cachorros → estado: nome, cor, raça
  comportamento: latir, correr
- Bicicletas → estado: marcha atual, velocidade atual
  comportamento: trocar marcha, aplicar freios
- Lâmpada → estado: modelo, potência
  comportamento: acender, apagar

Exemplos

- Conta Bancária:
- Estado:agência, banco, saldo, movimentação, cpf, tipo da conta
- Comportamento: calcular_saldo, verificar_extrato, imprimir
- Identidade: número da conta

- Jogo:
- Estado: software, empresa, nome, versão, plataforma,
- Comportamento: Executar, pontuar, modificar
- Identidade: Cod_jogo

Vamos resolver:

- Computador:
- Cliente:
- Produto:
- Nota Fiscal:
- Vendedor:
- Livro:

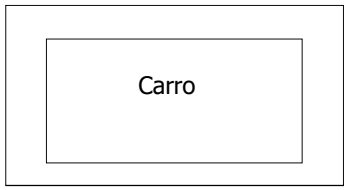
CLASSES

- A *unidade fundamental* de programação em orientação a objetos (POO) é a *classe*.
- Classes contém:
  - *Atributos*: determinam o *estado* do objeto;
  - *Métodos*: semelhantes a procedimentos em linguagens convencionais, são utilizados para *manipular os atributos*.

CLASSE

Uma Classe é representada por um retângulo que pode possuir até 3 divisões.
1º divisão: armazena o nome da classe;
2º divisão: os atributos da classe;
3º divisão: os métodos.

Geralmente uma classe possui atributos e métodos, porém é possível encontrar classes que contenham apenas uma dessas características ou mesmo nenhuma quando se tratar de coisa abstratas.



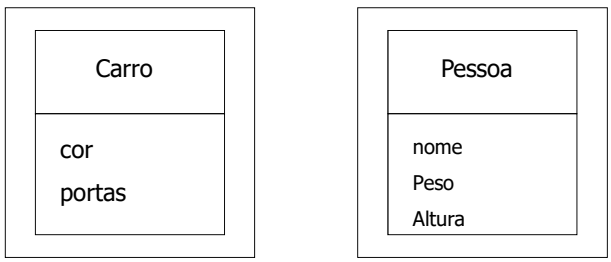
ATRIBUTOS

Representam um conjunto de informações, ou seja, elementos de dados que caracterizam um objeto.
Exemplos:

Classe Pessoa
* Atributos: Nome, Data de Nascimento, etc.

Classe Carro
* Atributos: Cor, Marca, Modelo, Ano, etc.

Os atributos são representados na 2º divisão.



MÉTODOS

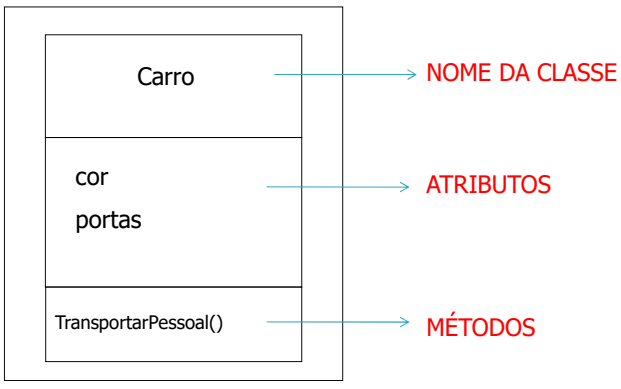
Classes costumam possuir métodos, que representam as atividades que um objeto de uma classe pode executar.

Ex: Classe Carro
Métodos: andar(), ligar(), desligar().

Um método pode receber ou não parâmetros (valores que são utilizados durante a execução do método) e pode retornar valores. Esses valores podem representar o resultado da operação executada ou simplesmente indicar se o processo foi concluído com sucesso ou não. Assim, um método representa um conjunto de instruções que são executadas quando o método é chamado.
Um objeto da classe Carro, por exemplo, pode executar a atividade de transportar pessoas.

MÉTODOS

Os métodos são armazenados na 3º divisão de uma classe.



Exemplo de uma classe com Métodos

Classe

- A classe é a implementação de tipo abstrato de dados (TAD) no paradigma orientado a objetos.
- A classe define as propriedades (atributos) e os comportamentos (métodos).

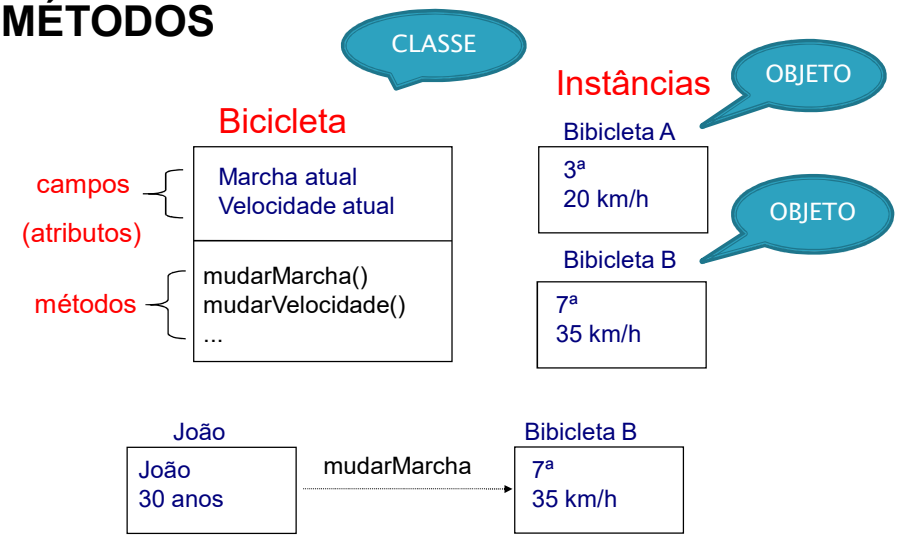
CLASSE
Atributos
Métodos

Documento
Autor DataDeChegada
Imprimir Editar

Classe

Classe	Objetos
Funcionário	Funcionária Ana Cristina
	Funcionário Gustavo
Empresa	Empresa Casa de Festas ABC
	Empresa Software Ltda
Computador	Notebook
	Tablet

MÉTODOS



Visibilidade:

- Visibilidade:
  + público: visível em qualquer classe de qualquer pacote
  # protegido: visível para classes do mesmo pacote
  – privado: visível somente para classe

Exemplo:

+ nome : String
– VerNota(): String

EXERCÍCIOS

Identificar as Classes, Objetos, Atributos e os métodos existentes nas classes:
PROFESSOR
ALUNO
DISCIPLINA
ANIMAL

Criar 3 atributos e métodos de cada classe.

Classes e Objetos

- Como definir uma classe e seus atributos

```
public class Cliente
{
    private int clienteId;
    private string nome;
    private decimal limiteCredito;
    private int quantidadeProdutos;
}
```

```
Cliente novoCliente = new Cliente();
```

Os pilares da OO

- Os pilares da OO são mecanismos fundamentais que garantem a filosofia de Orientação a Objetos. São eles:
  - Encapsulamento;
  - Herança;
  - Polimorfismo.

Encapsulamento

- É a técnica utilizada para esconder uma ideia, ou seja, não expor detalhes internos para o usuário, tornando partes do sistema mais independentes possível.

Encapsulamento

- Como um exemplo mais técnico podemos descrever o que acontece em um sistema de vendas, aonde temos cadastros de funcionários, usuários, gerentes, clientes, produtos entre outros.
- Se por acaso acontecer um problema na parte do usuário é somente nesse setor que será realizada a manutenção não afetando os demais.

Encapsulamento

- Em um processo de encapsulamento os atributos das classes são do tipo **private**.
- Para acessar esses tipos de modificadores, é necessário criar métodos **setters** e **getters**.
- Por entendimento os métodos setters servem para alterar a informação de uma propriedade de um objeto.
- E os métodos getters para retornar o valor dessa propriedade.

Encapsulamento

Veja um exemplo de encapsulamento, e é realizado o processo de geração dos métodos setters e getters.

Métodos getters	Métodos setters

public String getNome() {
return nome;
}
 	
public double getSalario() {
return salario;
}
	public void setNome(String nome) {
	this.nome = nome;
	}
	
	public void setSalario(double salario) {
	this.salario = salario;
	}

Encapsulamento da classe Funcionário.

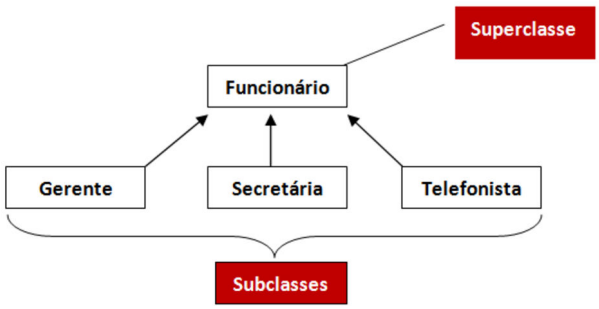
```
public class Funcionario {
private double salario;
private String nome;

public String getNome() {
      return nome;
}
 public void setNome(String nome) {
 this.nome = nome;
}
public void setSalario(double salario) {
 this.salario = salario;
}
 public double getSalario() {
 return salario;
 }
 }
```

Herança

- Na Programação Orientada a Objetos o significado de herança tem o mesmo significado para o mundo real.
- Assim como um filho pode herdar alguma característica do pai, na Orientação a Objetos é permitido que uma classe herde atributos e métodos da outra, tendo apenas uma restrição para a herança.
- Os modificadores de acessos das classes, métodos e atributos só podem estar com visibilidade **public** e **protected** para que sejam herdados.

Herança

Uma das grandes vantagens de usar o recurso da herança é na reutilização do código. Esse reaproveitamento pode ser acionado quando se identifica que o atributo ou método de uma classe será igual para as outras. Para efetuar uma herança de uma classe é utilizada a palavra reservada chamada **extends**.



Herança

Hierarquia Pessoa-Aluno

```
class Pessoa {
    // info pessoal
    private String nome;
    private int idade;
    private String morada;
    private String tlm;

    public Pessoa(String n; int id; String mor; String tel) {
        nome= n; idade= id; morada= mor; tlm= tel;
    }
    // métodos
}
class Aluno extends Pessoa {
    private double  media;

    public Aluno(String n; int id; String mor; String tel; double m) {
        super(n, id, mor, tel); // invoca o construtor da superclasse
        media= m;
    }
    // métodos
}
```

Polimorfismo

- Significa muitas formas, na orientação a objetos você pode enviar uma mesma mensagem para diferentes objetos e fazê-los responder da maneira correta.
- Você pode enviar a mensagem **mover** para cada objeto semelhante a um veiculo e cada um vai se comportar de maneira diferente para atender a sua solicitação.
- Quando uma mesma mensagem pode ser processada de diferentes formas temos um exemplo de polimorfismo.

- Uma definição mais formal diria:

- *"Polimorfismo é o princípio pelo qual duas ou mais classes derivadas de uma mesma superclasse podem invocar métodos que têm a mesma identificação (assinatura), mas comportamentos distintos, especializados para cada classe derivada, usando para tanto uma referência a um objeto do tipo da superclasse"*

Polimorfismo

Usando polimorfismo podemos :
1-Invocar métodos da classe derivada através da classe base em tempo de execução;
2-Permitir que classes forneçam diferentes implementações de métodos que são chamados com o mesmo nome;

Existem dois tipos básicos de polimorfismo:
Polimorfismo em tempo de compilação (Overloading/Sobrecarga);
O polimorfismo em tempo de compilação utiliza a sobrecarga de métodos e operadores sendo também chamado de ligação precoce*(early binding)*. A utilização da sobrecarga de métodos realiza a tarefa com distintos parâmetros de entrada.

Polimorfismo em tempo de execução (Overriding/Sobrescrita);
O polimorfismo em tempo de execução pode ser feito usando herança e métodos virtuais. Quando sobrescrevemos(*override*) os métodos virtuais estamos alterando o comportamento dos métodos para a classe derivada. Isto também é conhecido como ligação tardia.