# Aula 02 – Conceitos de orientação a objetos

Baseado nos slides oficiais do livro Java – Como programar – Deitel e Deitel – 10ª edição

# Introdução a tecnologia de objetos

- Procura mudar a forma de pensar nos programas
- Concentrar esforços em definir estruturas e funções que as manipulam, bem como elas se comunicam
- SmallTalk é a primeira linguagem que segue o paradigma
- Seu objetivo era modelar o mundo através da linguagem.

# O que são objetos?

- Conceito central baseia-se em **objetos**
  - Componente de software que representa uma **entidade do sistema**, seja ela física, conceitual ou de software
  - Cada objeto tem **características** (atributos) e **comportamento** (operações) próprios
  - Oferecem uma **abstração** da entidade com limites e significados para o domínio da aplicação
- São exemplos de objetos: data, automóvel, pessoa, janela do sistema, time de futebol, ...

# Exemplo de objetos

- Quais as características e comportamentos inerentes a um carro?
  - Para o dono de uma loja de automóveis?
  - Para o DETRAN?
  - Para um software de controle autônomo?
- Quais as características e comportamentos inerentes a uma pessoa?
  - Para um jogo de video game?
  - Para fichas de uma delegacia de polícia?
  - Para uma academia de musculação?
  -

# Princípios norteadores da POO

- Tipos abstratos de dados (classes)
- Encapsulamento
- Herança
- Polimorfismo

# Classes

- Objetos semelhantes tem normalmente um conjunto de características e comportamentos iguais
  - Todas as pessoas tem um nome, idade, cor do cabelo, …
  - Todo carro pode acelerar, frear, mudar de direção…
- Chamamos de classe uma representação de um conjunto de objetos que compartilham características semelhantes
- Uma **classe** é um **tipo abstrato de dados**
  - Um objeto está para uma classe assim como uma variável está para um tipo de dado
  - Cada classe é identificada por um nome

# Instanciação

- Pensando nas classes como moldes, pode-se criar quantas instâncias dela se quiser
- Por isso, costumamos dizer que **um objeto é uma instância de uma classe**
- Maior clareza de código, uma vez que os dados e as funções que os manipulam estão juntos

# Reuso

- Uma das grandes vantagens da orientação a objetos
- Vários programas diferentes podem usar as mesmas definições de classes, evitando tempo e esforço
- Software construído como um produto:
  - Componentes de vários fabricantes
  - Nenhum conhece como foi implementado, mas todos devem saber como utilizá-lo
  - Em caso de troca de uma parte, o todo não é afetado.

# Mensagens

- Cada objeto é independente e não pode ter seu estado interno alterado por qualquer outro

- Para que isso ocorra os objetos trocam mensagens através de uma interface conhecida por ambos
  - Ex: um carro  acelera pressionando-se o pedal do acelerador

- **Um programa orientado a objetos é uma coleção de objetos que trocam mensagens entre si**

- As mensagens são codificadas como chamadas de métodos

# Atributos de instância

- Apesar de possuir os mesmos atributos, o estado de cada objeto pode ser diferente um do outro
- Cada objeto conhece seu próprio estado, mas não conhece o estado dos outros
- Os atributos são codificados como variáveis de instância

# Encapsulamento

- Ocorre quando uma classe ( e seus objetos ) ocultam dos demais os detalhes da sua implementação

- O encapsulamento consiste em controlar e **restringir o acesso** aos atributos e comportamentos do objeto a apenas aqueles informados na sua interface pública
  - O motorista não tem acesso direto ao motor, apenas através dos pedais do carro

# Herança

- Quando todos os objetos tem características comuns, mas alguns tem uma especialização nas características
  - Todas as pessoas da Universidade tem características semelhantes, mas estudantes e professores tem atribuições específicas
- Na **herança**, pode-se criar classes genéricas (**superclasse**) com os atributos comuns e classes mais especializadas (**subclasses**). As subclasses não precisam repetir o código das superclasses, apenas declarar que herdam das mesmas.

# Polimorfismo

- Apesar de ter comportamentos semelhantes, a forma como está implementado pode ser diferente
  - Ex: Todo carro tem os métodos de ligar, acelerar e frear, mas a forma como cada um faz isso é diferente

- **Polimorfismo** é a propriedade de várias formas de implementar um mesmo comportamento para um objeto.

# Análise Orientada a Objetos

- Capacidade de criar modelos, que podem ou não virar código, para descrever um sistema que usa o paradigma de orientação a objetos

- UML – Unified Modelling Language: linguagem de modelagem que usa símbolos para descrição dos componentes de um modelo de sistema e seus relacionamentos
  - Ex: Diagrama de classe, Diagrama de objetos, Diagrama de sequência, Diagrama de atividades.

# Exemplo

- Modelar classes e objetos envolvendo o seguinte problema:
  - Um jogo utiliza personagens baseados em lendas medievais. Cada personagem tem um nome, uma classe, pontos de vida, 6 atributos, 2 itens de combate e 1 mochila p/ carregar um número variável de itens de cura. A classe de personagem pode ser guerreiro, mago, ladrão ou sacerdote. Os itens de combate tem um nome, valor de ataque, valor de defesa e 6 valores representando bônus para os atributos. Cada mochila possui uma capacidade máxima de carga e um conjunto de itens de cura. Cada item de cura tem um nome, um peso e o número de pontos de vida que cura.