



TECNOLOGIA E INOVAÇÃO  
EM PROL DA INDÚSTRIA



Curso Técnico em  
Informática

# Desenvolvimento de Sistemas I – 100h

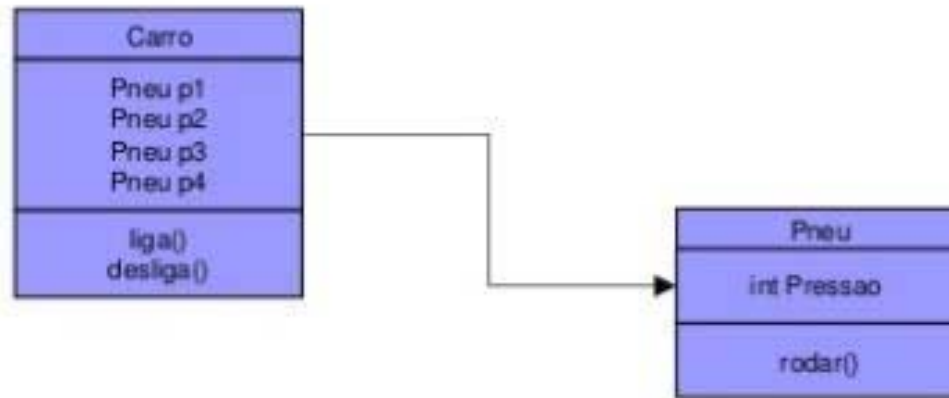
Prof<sup>a</sup>: Francisleide Almeida

Baseados nos slides de  
Antonio Rodrigues Carvalho Neto

# Associação

# Associação

- Ocorre quando uma classe possui atributos do tipo de outra classe.



- Nota : Neste caso estamos dizendo que carro possui pneu (4 pneus)



# Associação

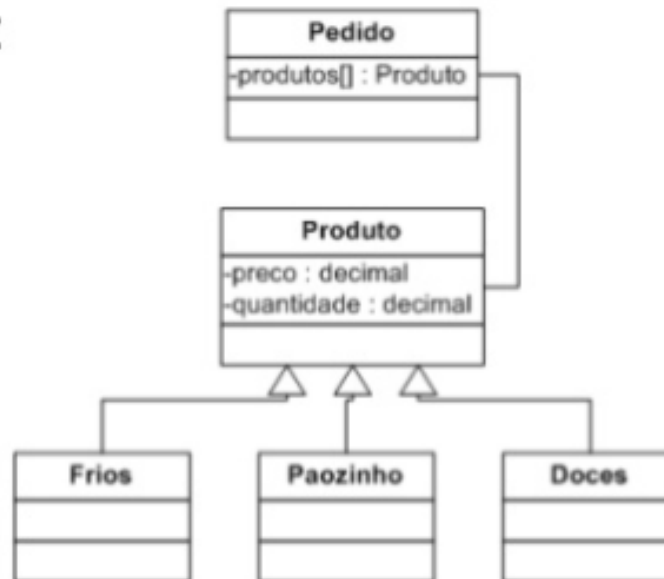
- A associação pode ser representada em Java da seguinte forma:

```
public class Pneu {  
    int Pressao;  
  
    void roda() {  
        System.out.println("Pneu em movimento");  
    }  
}
```

```
public class Carro {  
    Pneu p1;  
    Pneu p2;  
    Pneu p3;  
    Pneu p4;  
  
    void liga() {  
        System.out.println("Carro ligado");  
    }  
  
    void desliga() {  
        System.out.println("Carro desligado");  
    }  
}
```

# Exercício

- Crie as classes em java conforme o diagrama abaixo:



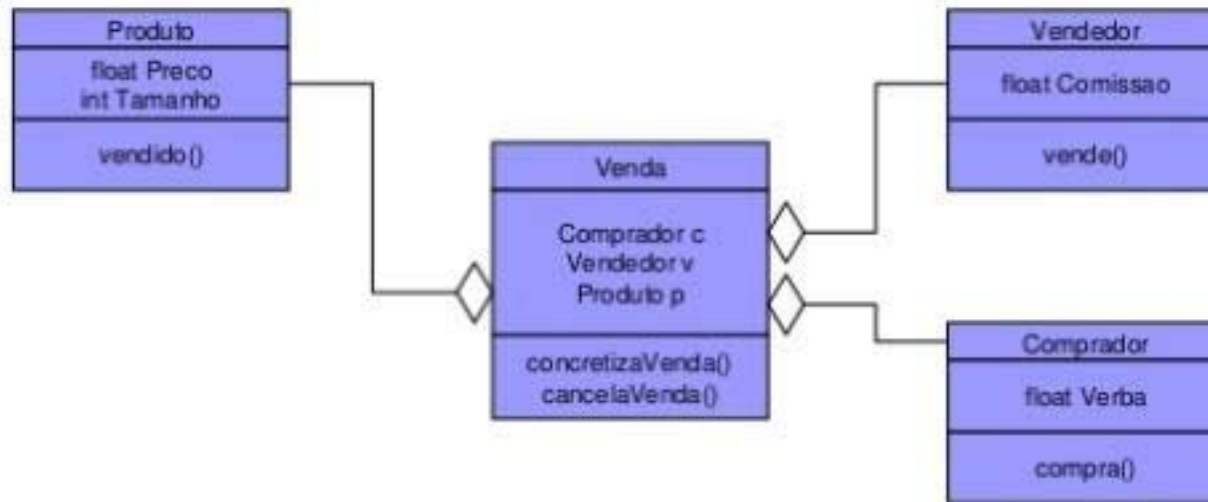
# Agregação



# Agregação

- Ocorre quando uma classe usa outras classes em suas operações. As classes utilizadas participam da classe principal, mas a classe principal não contém estas classes utilizadas como sendo partes suas.

# Agregação



- Neste caso **Venda** é o objeto definido como sendo o todo. E este objeto somente pode existir caso os demais objetos que o compõem também existam.

# Agregação

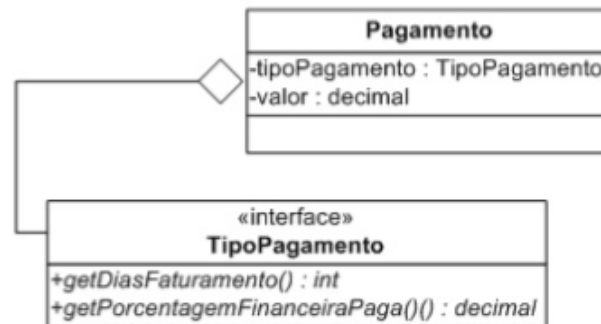
- A agregação pode ser representada da seguinte forma:

```
public class Vendedor {  
    float Comissao;  
  
    void vende() {  
        System.out.println("Vendido");  
    }  
}  
  
public class Comprador {  
    float Verba;  
  
    void compra() {  
        System.out.println("Comprado");  
    }  
}  
  
public class Produto {  
    float Preço;  
    int Tamanho  
  
    void vendido() {  
        System.out.println("Vendido");  
    }  
}
```

```
public class Venda {  
    Comprador c;  
    Vendedor v;  
    Produto p;  
  
    void concretizaVenda() {  
        System.out.println("Venda efetuada");  
        c.Verba -= p.Preço;  
        v.Comissao += p.Preço * 0.1f;  
        p.vendido();  
    }  
  
    void cancelaVenda() {  
        System.out.println("Venda cancelada");  
    }  
}
```

# Agregação

- Crie as classes no java, conforme o digrama abaixo:

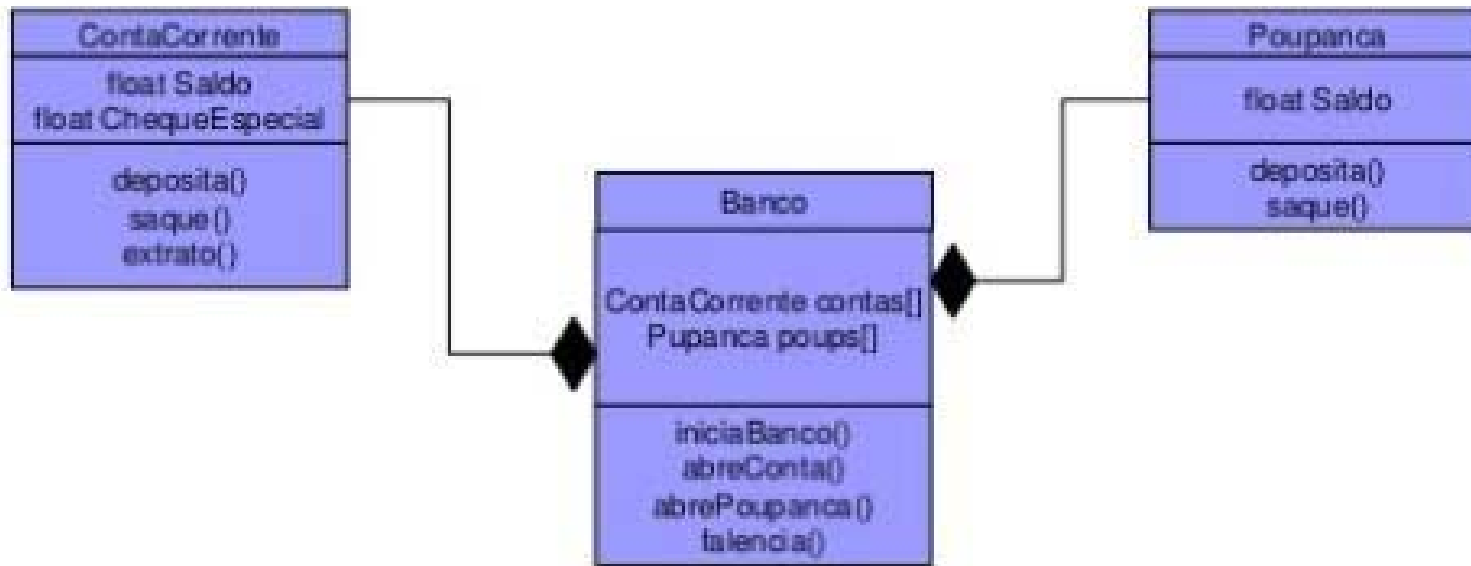


# Composição

# Composição

- Semelhante a agregação, a composição também é um conjunto onde há uma classe representando o todo e classes satélites funcionando como partes. Sua principal diferença ocorre que quando o objeto todo deixar de existir os seus objetos partes deverão deixar de existir também.

# Composição



- No caso desta composição uma vez que o Objeto banco for destruído todas os objetos Poupanca e ContaCorrente deverão ser destruídos também.



- A composição pode ser representada da seguinte forma:

```
public class Poupanca {
    float Saldo;

    void saque() {
        Saldo -= 10.0f;
        System.out.println("Novo Saldo →" + Saldo);
    }
    void deposito() {
        Saldo += 10.0f;
        System.out.println("Novo Saldo →" + Saldo);
    }
}

public class ContaCorrente {
    float Saldo;

    void saque() {
        Saldo -= 100.0f;
        System.out.println("Novo Saldo →" + Saldo);
    }
    void saque() {
        Saldo -= 100.0f;
        System.out.println("Novo Saldo →" + Saldo);
    }
}
```

```
public class Banco {
    Poupanca[] pops;
    ContaCorrente[] cc;
    int numConta, numPoupanca;
    void iniciaBanco() {
        pops = new Poupanca[100];
        cc = new ContaCorrente[100];
        numConta = 1;
        numPoupanca = 1;
    }
    void abreConta() {
        cc[ numConta ] = new ContaCorrente();
        numConta++;
    }
    void abrePoupanca() {
        pops[ numConta ] = new Poupanca();
        numPoupanca++;
    }
    void falencia() {
        for (int i = 0; i < 100; i++) {
            pops[ i ] = null;
            cc[ i ] = null;
        }
    }
}
```