



POO

Coleção de objetos

Prof. Alcides Calsavara

PUCPR

Conceitos



1. **Associação** entre classes
2. **Ligação** entre objetos
3. **Multiplicidade**
4. **Cardinalidade**
5. **Coleção de referências** (para objetos)
 - a. **vetor** de referências
 - b. **lista** de referências
 - c. classe **ArrayList**

Associação entre classes



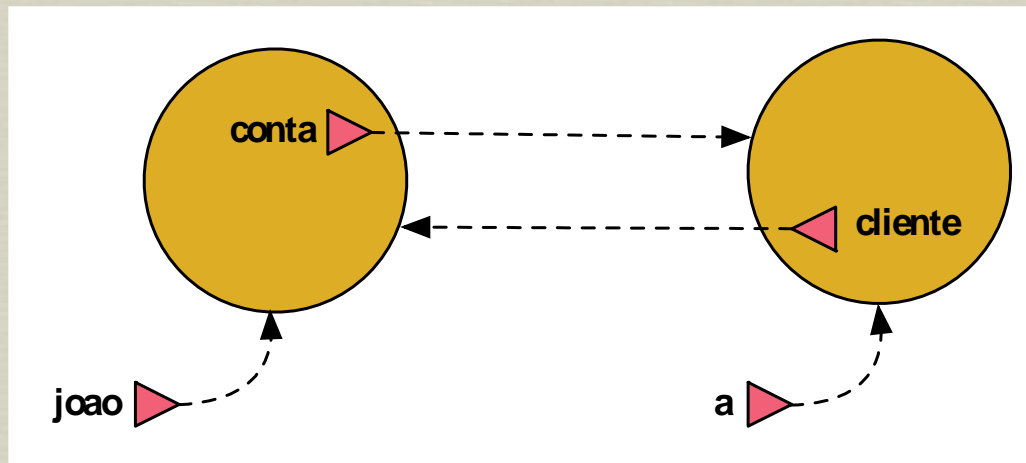
```
class Cliente {
    private ContaCorrente conta;
    . . .
    public void ligue(ContaCorrente c) {
        conta = c;
    }
}
```

Associação entre classes



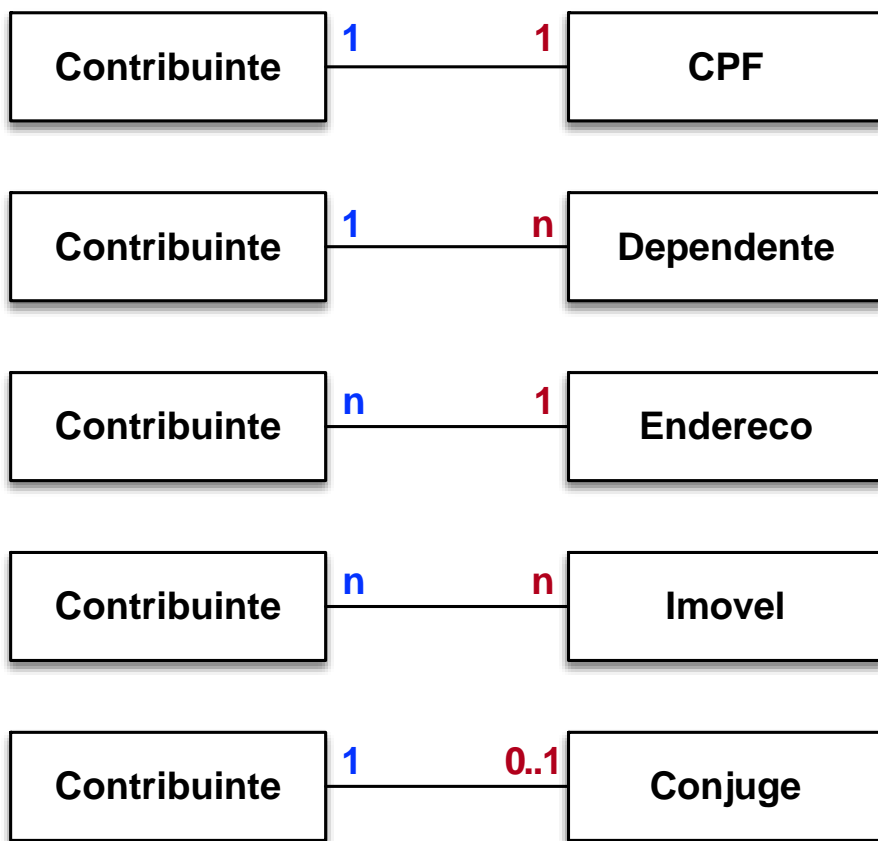
```
class ContaCorrente{
    private Cliente cliente;
    . . .
    public void ligue(Cliente c) {
        cliente = c;
    }
}
```

Ligação entre objetos



```
Cliente joao = new Cliente(...);  
ContaCorrente a = new ContaCorrente(...);  
joao.ligar(a);  
a.ligar(joao);
```

Multiplicidade



Um contribuinte possui **1** CPF.
Um CPF é vinculado a **1** contribuinte.

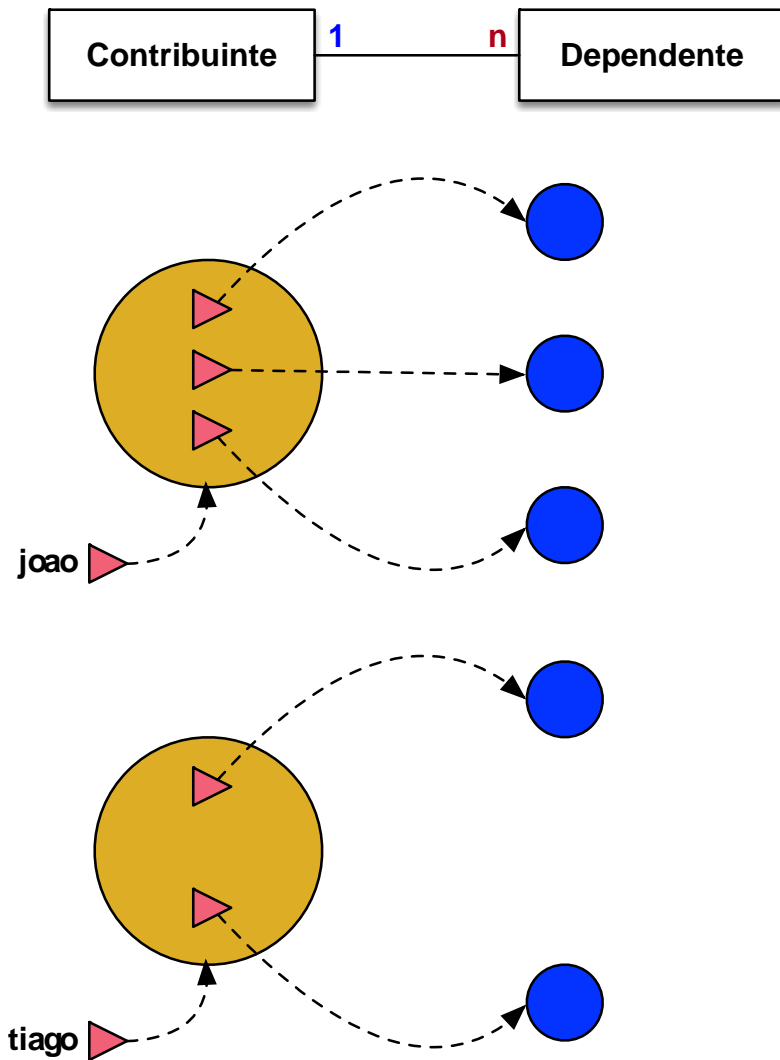
Um contribuinte possui **n** dependentes.
Um dependente é vinculado a **1** contribuinte.

Um contribuinte mora em **1** endereço.
Num endereço moram **n** contribuintes.

Um contribuinte possui **n** imóveis.
Um imóvel pertence a **n** contribuintes.

Um contribuinte **pode ter 1** cônjuge.
Um cônjuge é vinculado a **1** contribuinte.

Cardinalidade



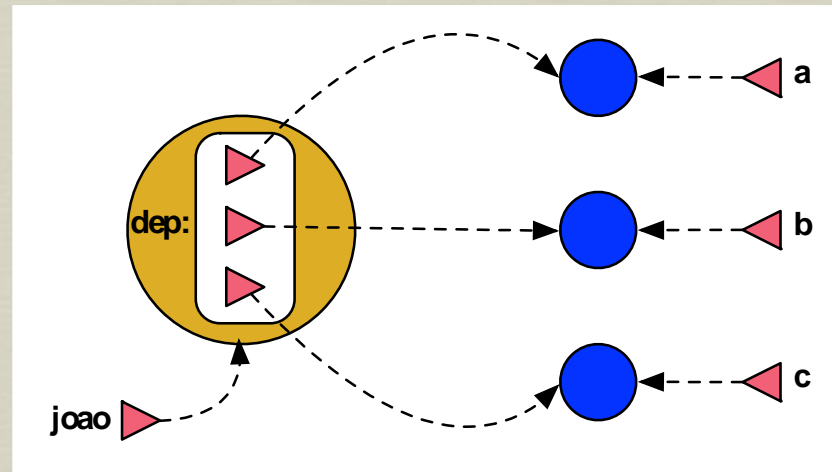
O contribuinte referenciado por “joao” tem **3** dependentes ligados a ele.

O conjunto de dependentes do contribuinte referenciado por “joao” tem cardinalidade **3**.

O contribuinte referenciado por “tiago” tem **2** dependentes ligados a ele.

O conjunto de dependentes do contribuinte referenciado por “tiago” tem cardinalidade **2**.

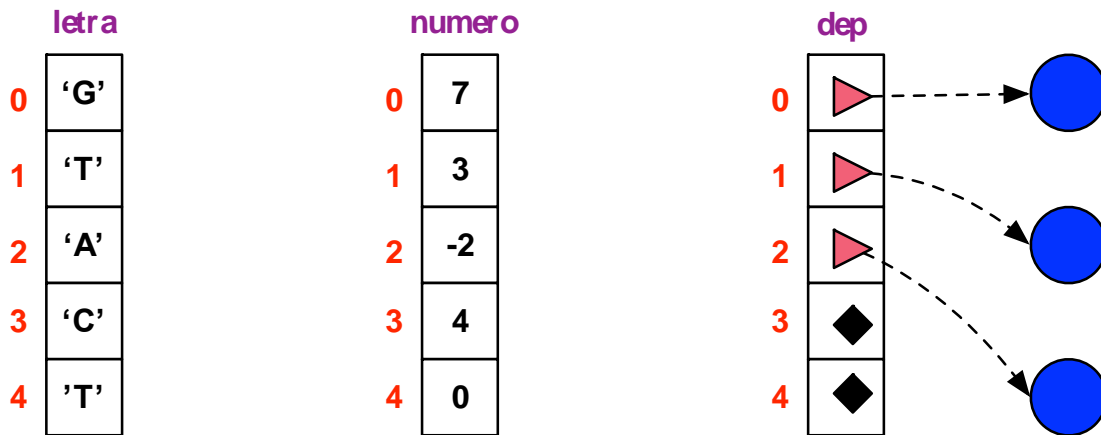
Coleção de referências



```
Contribuinte joao = new Contribuinte(...);  
Dependente a = new Dependente(...);  
Dependente b = new Dependente(...);  
Dependente c = new Dependente(...);  
joao.ligarDependente(a);  
joao.ligarDependente(b);  
joao.ligarDependente(c);
```


Vetor

Coleção de
tamanho fixo.



Um vetor de
tamanho **n** pode ser
indexado de **0** a **n-1**.

vetor \cong *array*

letra é um vetor de **5** caracteres

```
char[] letra = new char[5];
```

numero é um vetor de **5** valores inteiros

```
int[] numero = new int[5];
```

dep é um vetor de **5** referências para objetos da classe Dependente

```
Dependente[] dep = new Dependente[5];
```

Um vetor de referências para objetos da classe Dependente

```
Dependente[] dep = new Dependente[5]; // cria o vetor
```

```
dep[0] = new Dependente(...);
```

```
dep[1] = new Dependente(...);
```

```
dep[2] = new Dependente(...);
```

```
dep[0].imprimir(); // chama o método imprimir do dependente
```

```
dep[3].imprimir(); // erro de execução: NullPointerException
```

```
Dependente d = new Dependente(...);
```

```
dep[1] = d; // substitui a referência na posição 1 do vetor
```

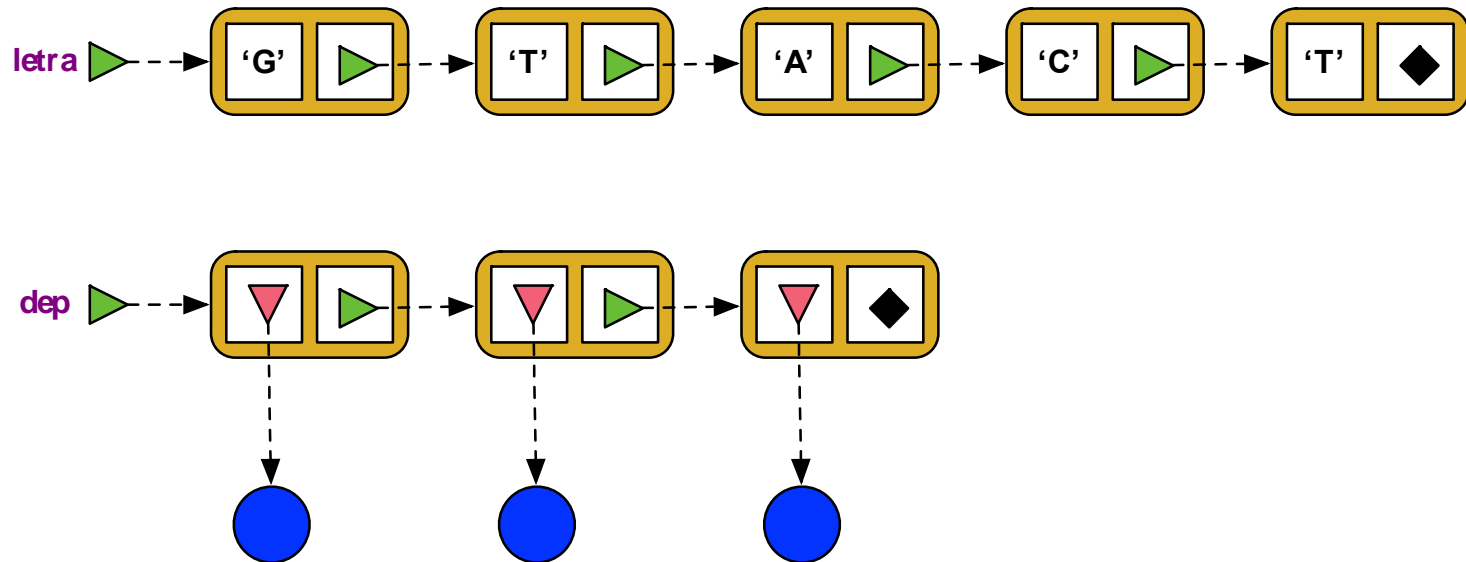
```
d = new Dependente(...);
```

```
dep[3] = d;
```

```
dep[3].imprimir(); // chama o método imprimir do dependente
```

Lista

Coleção de
tamanho variável.



letra é uma referência para o primeiro elemento de uma lista composta por 5 elementos. Cada elemento contém um caracter e uma referência para o próximo elemento.

dep é uma referência para o primeiro elemento de uma lista composta por 3 elementos. Cada elemento contém uma referência para um objeto da classe Dependente e uma referência para o próximo elemento.

Classe ArrayList

- ☞ Uma instância de ArrayList é uma lista de referências para objetos.
- ☞ A classe é parametrizada com o nome da classe de objetos referenciados pelos elementos da lista.
- ☞ Exemplo de criação de uma lista denominada **dep** de referências para objetos da classe **Dependente**:

```
ArrayList<Dependente> dep; // declaração da variável dep  
  
dep = new ArrayList<Dependente>(); // criação da lista
```

Exemplo: Um ArrayList de referências para objetos da classe String

```
import java.util.ArrayList;

public class TesteLista {
    public static void main(String[] args)
    {
        ArrayList<String> cidades = new ArrayList<String>();

        System.out.println(cidades.size());

        cidades.add("Curitiba");
        cidades.add("Maringa");
        cidades.add("Londrina");

        System.out.println(cidades.size());

        for (String x : cidades)
            System.out.println(x);
    }
}
```

Exemplo: Um ArrayList de referências para objetos da classe String

```
import java.util.ArrayList;

public class TesteLista {
    public static void main(String[] args)
    {
        ArrayList<String> cidades = new ArrayList<String>();

        System.out.println(cidades.size());

        cidades.add("Curitiba");
        cidades.add("Maringa");
        cidades.add("Londrina");

        System.out.println(cidades.size());

        for (String x : cidades)
            System.out.println(x);
    }
}
```

Cria um ArrayList de referências para objetos da classe String.

Exemplo: Um ArrayList de referências para objetos da classe String

```
import java.util.ArrayList;

public class TesteLista {
    public static void main(String[] args)
    {
        ArrayList<String> cidades = new ArrayList<String>();

        System.out.println(cidades.size());

        cidades.add("Curitiba");
        cidades.add("Maringa");
        cidades.add("Londrina");

        System.out.println(cidades.size());

        for (String x : cidades)
            System.out.println(x);
    }
}
```

Adiciona referências para objetos da classe String ao ArrayList cidades.

Exemplo: Um ArrayList de referências para objetos da classe String

```
import java.util.ArrayList;

public class TesteLista {
    public static void main(String[] args)
    {
        ArrayList<String> cidades = new ArrayList<String>();

        System.out.println(cidades.size());

        cidades.add("Curitiba");
        cidades.add("Maringa");
        cidades.add("Londrina");

        System.out.println(cidades.size());

        for (String x : cidades)
            System.out.println(x);
    }
}
```

Retorna o número de referências contidas no ArrayList cidades.

Exemplo: Um ArrayList de referências para objetos da classe String

```
import java.util.ArrayList;

public class TesteLista {
    public static void main(String[] args)
    {
        ArrayList<String> cidades = new ArrayList<String>();

        System.out.println(cidades.size());

        cidades.add("Curitiba");
        cidades.add("Maringa");
        cidades.add("Londrina");

        System.out.println(cidades.size());

        for (String x : cidades)
            System.out.println(x);
    }
}
```

Para cada String x no ArrayList cidades faça o seguinte: ...

Métodos de ArrayList

1. `add(Object obj)` // insere um objeto no fim da lista
2. `add(int index, Object obj)` // insere um objeto na posição especificada
3. `remove(Object obj)` // remove da lista o objeto especificado
4. `remove(int index)` // remove da lista o objeto na posição especificada
5. `set(int index, Object obj)` // atualiza o objeto na posição especificada
6. `int indexOf(Object obj)` // retorna a posição do objeto especificado
7. `Object get(int index)` // retorna o objeto na posição especificada
8. `int size()` // retorna o tamanho da lista
9. `boolean contains(Object obj)` // verifica se o objeto especificado está na lista
10. `clear()` // remove todos os objetos da lista

Exemplo: Classes Dependente, Contribuinte e ReceitaFederal

```
class Dependente {  
    private String nome;  
  
    public Dependente(String nome) {  
        this.nome = nome;  
    }  
  
    public void imprimir() {  
        System.out.println("Dependente: " + nome);  
    }  
}
```

```
import java.util.ArrayList;

public class Contribuinte {
    private String nome;

    private ArrayList<Dependente> dep;

    public Contribuinte(String nome) {
        this.nome = nome;
        dep = new ArrayList<Dependente>();
    }

    public void ligarDependente(Dependente d) {
        dep.add(d);
    }

    public void imprimir() {
        System.out.println("Contribuinte: " + nome);
        imprimirDependentes();
    }

    private void imprimirDependentes() {
        for (Dependente d : dep) { // d é um iterador
            d.imprimir();
        }
    }

    public int numeroDependentes() {
        return dep.size();
    }
}
```



```
import java.util.ArrayList;
```

```
public class Contribuinte {  
    private String nome;
```

Declaração

```
    private ArrayList<Dependente> dep;
```

```
    public Contribuinte(String nome) {  
        this.nome = nome;
```

Criação

```
        dep = new ArrayList<Dependente>();  
    }
```

```
    public void ligarDependente(Dependente d) {
```

```
        dep.add(d);
```

Inserção

```
    }  
    public void imprimir() {
```

```
        System.out.println("Contribuinte: " + nome);  
        imprimirDependentes();  
    }
```

```
    private void imprimirDependentes() {
```

```
        for (Dependente d : dep) { // d é um iterador  
            d.imprimir();
```

Iteração

```
        }
```

```
    }  
    public int numeroDependentes() {
```

```
        return dep.size();  
    }
```

```
}
```

```
public class ReceitaFederal {  
    public static void main(String[] args) {  
        Contribuinte julia = new Contribuinte( "Julia");  
        Dependente jorge = new Dependente( "Jorge");  
        Dependente sandra = new Dependente( "Sandra");  
        julia.ligarDependente(jorge);  
        julia.ligarDependente(sandra);  
        julia.imprimir();  
        System.out.println("Numero de dependentes: " +  
            julia.numeroDependentes( ) + '\n');  
  
        Contribuinte leonardo = new Contribuinte("Leonardo");  
        Dependente marta = new Dependente("Marta");  
        leonardo.ligarDependente(marta);  
        Dependente diego = new Dependente("Diego");  
        leonardo.ligarDependente(diego);  
        Dependente claudia = new Dependente("Claudia");  
        leonardo.ligarDependente(claudia);  
        leonardo.imprimir();  
        System.out.println("Numero de dependentes: " +  
            leonardo.numeroDependentes( ) + '\n');  
    }  
}
```

Exemplo: Classes Bovino, Fazendeiro e Agro

```
public class Bovino {  
  
    private int codigo;  
    private double peso;  
  
    public Bovino(int codigo, double peso)  
    {  
        this.codigo = codigo;  
        this.peso = peso;  
    }  
    public void atualizar_peso( double peso )  
    {  
        this.peso = peso;  
    }  
  
    public int codigo() { return codigo; }  
  
    public boolean pronto() { return (peso >= 500); }  
  
    public double peso() { return peso; }  
}
```

```
import java.util.ArrayList;

public class Fazendeiro {

    private ArrayList<Bovino> gado;

    public Fazendeiro()
    {
        gado = new ArrayList<Bovino>( );
    }

    public void incluir(Bovino bovino)
    {
        gado.add(bovino);
    }
}
```

[continua]

[continuação]

```
public boolean atualizar_peso( int codigo_busca,
                             double novo_peso ) {
    int posicao = 0;
    boolean encontrado = false;

    while (posicao < gado.size() && !encontrado) {
        if (gado.get(posicao).codigo() == codigo_busca)
            encontrado = true;
        else
            posicao++;
    }

    if (encontrado)
        gado.get(posicao).atualizar_peso(novo_peso);

    return encontrado;
}
```

[continua]

[continuação]

```
public double peso_disponivel()
{
    double peso_pronto = 0;
    for (Bovino b: gado)
    {
        if ( b.pronto() )
            peso_pronto = peso_pronto + b.peso();
    }
    return peso_pronto;
}
```



```
public class Agro {  
    public static void main(String args[]) {  
  
        Fazendeiro garcia = new Fazendeiro();  
  
        garcia.incluir( new Bovino(86, 120) );  
        garcia.incluir( new Bovino(25, 500) );  
        garcia.incluir( new Bovino(38, 180) );  
        garcia.incluir( new Bovino(90, 700) );  
  
        System.out.println( garcia.peso_disponivel() );  
  
        garcia.atualizar_peso( 38, 400 );  
        garcia.atualizar_peso( 90, 800 );  
        garcia.atualizar_peso( 25, 600 );  
        garcia.atualizar_peso( 86, 300 );  
  
        System.out.println( garcia.peso_disponivel() );  
    }  
}
```

Java Collections



<https://www.codejava.net/java-core/collections/overview-of-java-collections-framework-api-uml-diagram>

<https://docs.oracle.com/javase/8/docs/technotes/guides/collections/reference.html>

