INSTITUTO FEDERAL
Santa Catarina

Ciência da Computação

Programação orientada a objetos

alexandre.perin@ifsc.edu.br

Lages (SC).

Associação entre objetos

- **Introdução**
  - No desenvolvimento de aplicações orientadas a objetos, certamente haverão diversos objetos envolvidos na solução de um problema
    - Várias classes de objetos terão que ser desenvolvidas para permitirem que objetos possam interagir entre si
    - Ex.:
      - Supermercado: produtos, clientes, vendas...
      - Aluno (faz perguntas), Professor (explica)
      - Jogador (conversam entre si), técnico (orienta), torcida...

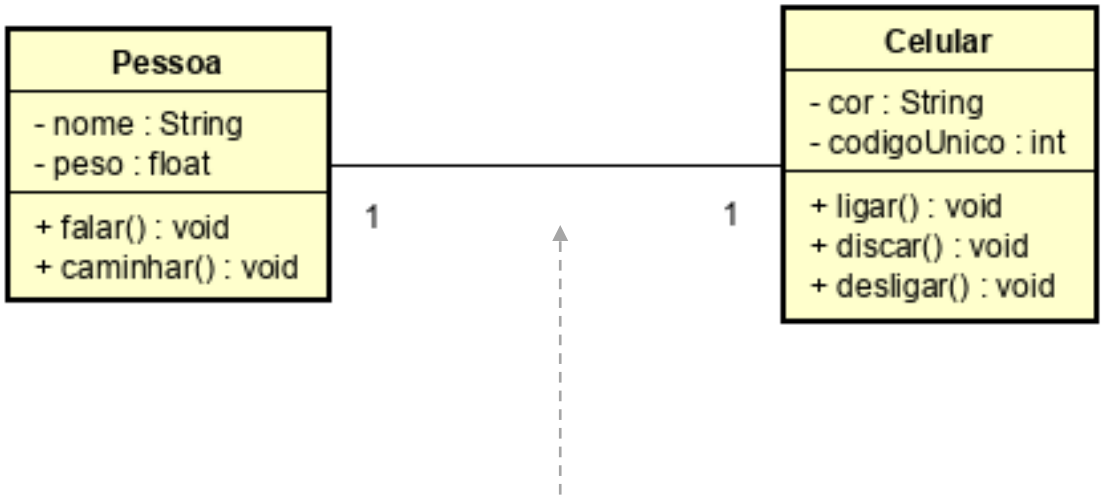
Associação entre objetos

- **Introdução**
  - Associação:
    - Simples
      - Com um (1) objeto
      - Com diversos (n) objetos
    - Agregação ou composição
    - Herança

} A serem estudadas mais à frente

Associação entre objetos

- **Associação simples com 1 objeto**
  - Representação gráfica



Associação entre objetos

- **Associação simples com 1 objeto com direção**

Associação entre objetos

- **Implementação em Java**

Pessoa.java

```
1  public class Pessoa {
2      private String nome;
3      private float peso;
4
5
6      private Celular celular;
7
8
9      public void falar() {}
10     public void caminhar() {}
11
12 }
```

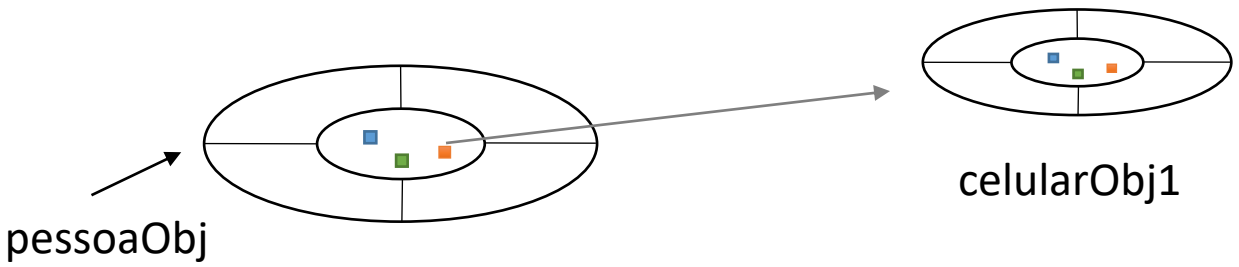
Celular.java

```
1  public class Celular {
2      private String cor;
3      private int codigoUnico;
4
5
6
7
8      public void ligar() {}
9      public void discar() {}
10     public void desligar() {}
11
12 }
```

Objetos da classe Pessoa terão acesso a objetos da classe Celular.

Associação entre objetos

- **Associação simples com 1 objeto**
  - Representação na memória



Associação entre objetos

- **Codificação**

Passo 1 – Escolher uma IDE

Passo 2 – Criar a classe Pessoa

Passo 3 – Criar a classe Celular

Passo 4 – Na classe Pessoa inserir um atributo para poder acessar Celular

Passo 5 – Cria a classe executora

Passo 6 – Na classe executora, criar os objetos e utilizar toString() para mostrar o estado de cada um deles

Passo 7 – Compilar e testar

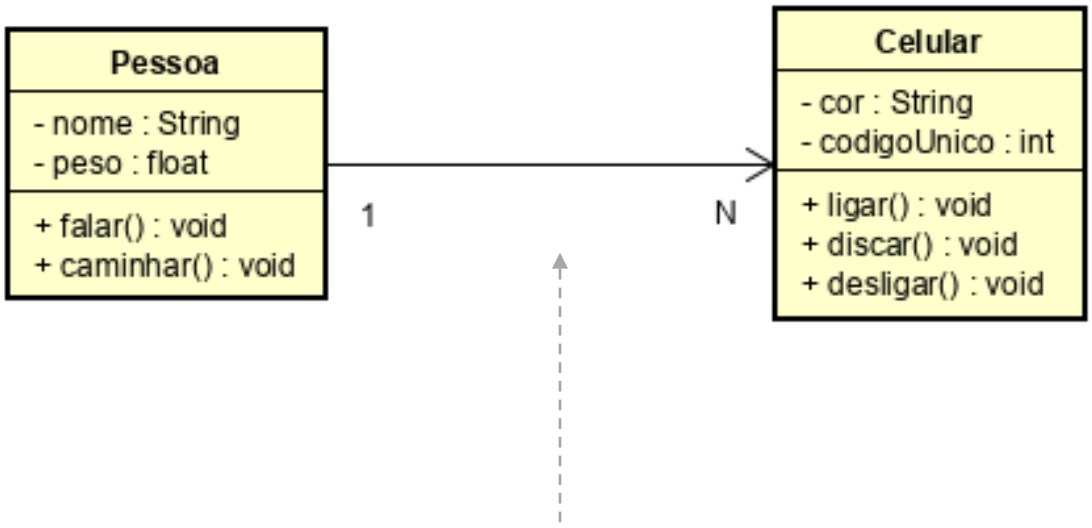
Exercícios

- Crie um projeto na IDE de sua preferência e codifique a seguinte associação (relação)

1. Criar as classes: Curso e Aluno
   - Construtor
   - Sets e Gets
   - toString
   - Associação: Curso acessa Aluno
2. Criar as classes: Livro e Editora
   - Construtor
   - Sets e Gets
   - toString
   - Associação: Livro acessa Editora

Associação entre objetos

- **Associação simples com n objetos**
  - Representação gráfica



Associação entre objetos

- **Há duas formas básicas para implementar:**
  - **Vetor**
    - Devido à estrutura ser estática, é necessário controlar o seu tamanho
    - Métodos específicos para manipulação devem ser implementados
  - **ArrayList**
    - Estrutura de dados disponível na linguagem Java
    - É um vetor dinâmico, que em tempo de execução aumenta ou diminui a sua capacidade de forma automática
    - Possui métodos que permitem uma mais fácil manipulação

Associação entre objetos

- **ArrayList**
  - Métodos principais:
    - boolean **add**(Object objeto): insere no final
    - void **add**(int index, Object objeto): insere na posição indicada
    - int **size**(): retorna o número de elementos da lista
    - Object **get**(int index): retorna um objeto
    - void **clear**(): remove todos os elementos
    - boolean **isEmpty**(): retorna true se a lista está vazia, false caso contrário
    - Object **remove**(int index): remove o i-ésimo elemento da lista
    - int **indexOf**(Object objeto): retorna a primeira posição da ocorrência de um objeto
    - Object **set**(int index, Object objeto): substitui o i-ésimo objeto pelo objeto enviado como parâmetro

Associação entre objetos

- **ArrayList**

  - Definição e criação:
    - `ArrayList<Nome da classe> listaObjetos = new ArrayList();`
    - Ex.:
      - `ArrayList<String> agendaNomes = new ArrayList();`

  - Utilização:
    - `agendaNomes.add("Romário");`
    - `agendaNomes.add("Falcão");`
    - `...`
    - `int nContatos = agendaNomes.size();`
    - `String nome = agendaNomes.get(1);`

Associação entre objetos

- **ArrayList**: navegação

a) Com o índice

```
int n = agendaNomes.size();
for (i=0; i<n; i++) {
     System.out.println("Posição" + i + agendaNomes.get(i));
}
```

b) Utilizando for-each

```
i = 0;
for (String objeto: agendaNomes) {
    System.out.println("Posição " + i + objeto);
    i++;
}
```

c) Usando iterator

```
i = 0;
Iterator<String> iterator = agendaNomes.iterator();
while (iterator.hasNext()) {
      System.out.println("Posição " + i + iterator.next());
      i++;
}
```

Associação entre objetos

- **Implementação em Java**

Pessoa.java

```
1  public class Pessoa {
2      private String nome;
3      private float peso;
4
5      private ArrayList<Celular>  listaCelular;
6
7
8
9      public void falar() {}
10     public void caminhar() {}
11
12 }
```

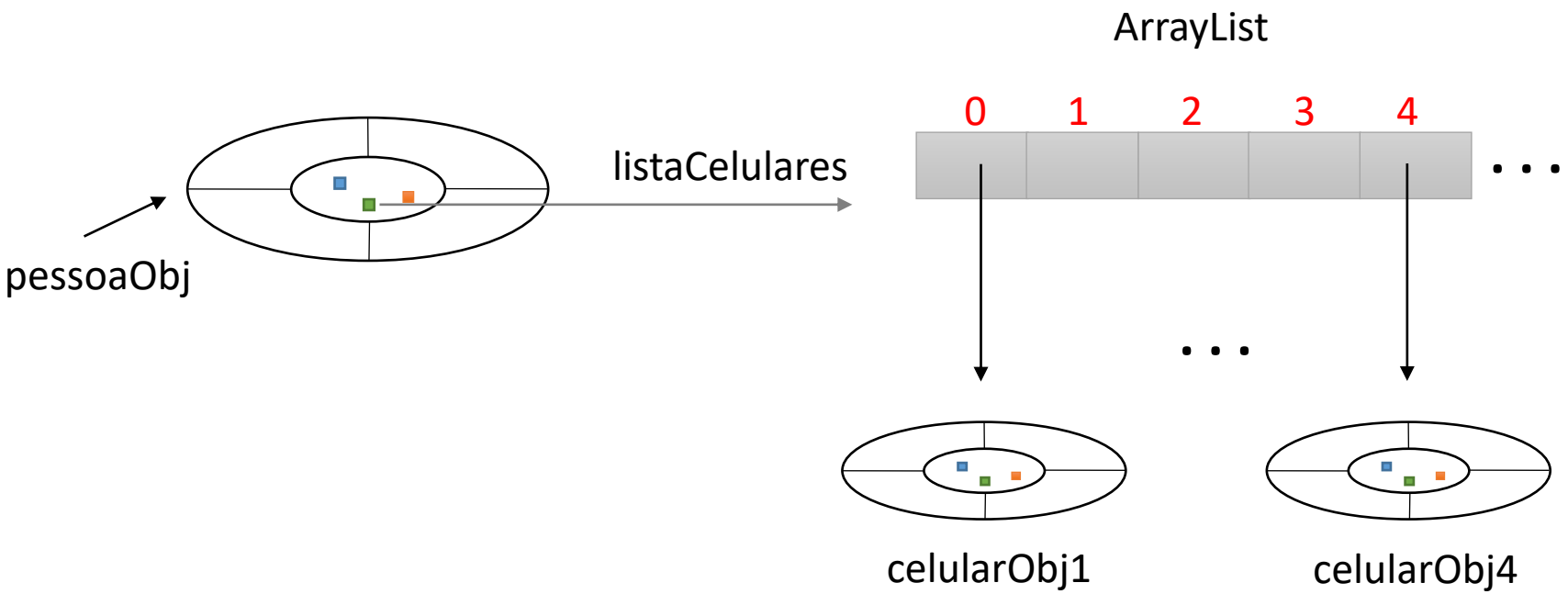
Celular.java

```
1  public class Celular {
2      private String cor;
3      private int codigoUnico;
4
5
6
7
8      public void ligar() {}
9      public void discar() {}
10     public void desligar() {}
11
12 }
```

Objetos da classe Pessoa terão acesso a objetos da classe Celular.

Associação entre objetos

- **Representação na memória**



Exemplos

- Crie um projeto na IDE de sua preferência e codifique a seguinte associação (relação)

1. Criar as classes: Professor e Disciplina
   - Construtor
   - Sets e Gets
   - toString
   - Associação: Professor acessa um conjunto de Disciplinas
2. Criar as classes: Time e Jogador
   - Construtor
   - Sets e Gets
   - toString
   - Associação: Time acessa um conjunto de Jogadores