PC1 - JAVA

Prof. Juarez Brandão

ETESP

ENCAPSULAMENTO

- É um conceito que propõe que os atributos de uma classe sejam protegidos e ocultados das demais classes do projeto. Possibilitando o controle de acesso dos mesmos através de métodos e garantindo que as regras definidas na própria classe sejam cumpridas.

- Ou seja, se for necessário armazenar uma informação dentro de um determinado atributo de uma instância de um objeto, deve-se chamar um método e passar a mesma por parâmetro. Proporcionando acesso somente as funções do objeto que compete ao “cliente”.

ENCAPSULAMENTO

- Cada atributo encapsulado, possui dois métodos públicos de acesso:

- **Get:** lê o conteúdo do atributo e retorna a informação

- **Set:** armazena a informação passada por parâmetro no atributo

```
void setNomeAtributo(String nome){
        this.nome = nome;
}

String getNomeAtributo(){
        return this.nome;
}
```

MODIFICADORES DE ACESSO

- Os modificadores de acesso permitem alterar a visibilidade dos elementos dentro de uma determinada classe. Existem quatro tipos, são:

• **public** – que permite que os elementos sejam acessados tanto internamente como externamente

• **private** – que permite que os elementos sejam acessados somente na classe que os possui

• **protected** – que permite que os elementos sejam acessados apenas por classes no mesmo pacote

• **default** – que permite que os elementos sejam acessados pelo métodos internos da mesma

```java
public class Acesso {
    private String usuario;
    private String senha;

    public String getUsuario() {
        return usuario;
    }
    public void setUsuario(String usuario) {
        this.usuario = usuario;
    }
    public String getSenha() {
        return senha;
    }
    public void setSenha(String senha) {
        this.senha = senha;
    }

    protected boolean validarSenha(){
        boolean ok = false;
        if (senha.equals("CursoJava")){
            ok = true;
        }

        return ok;
    }

}
```

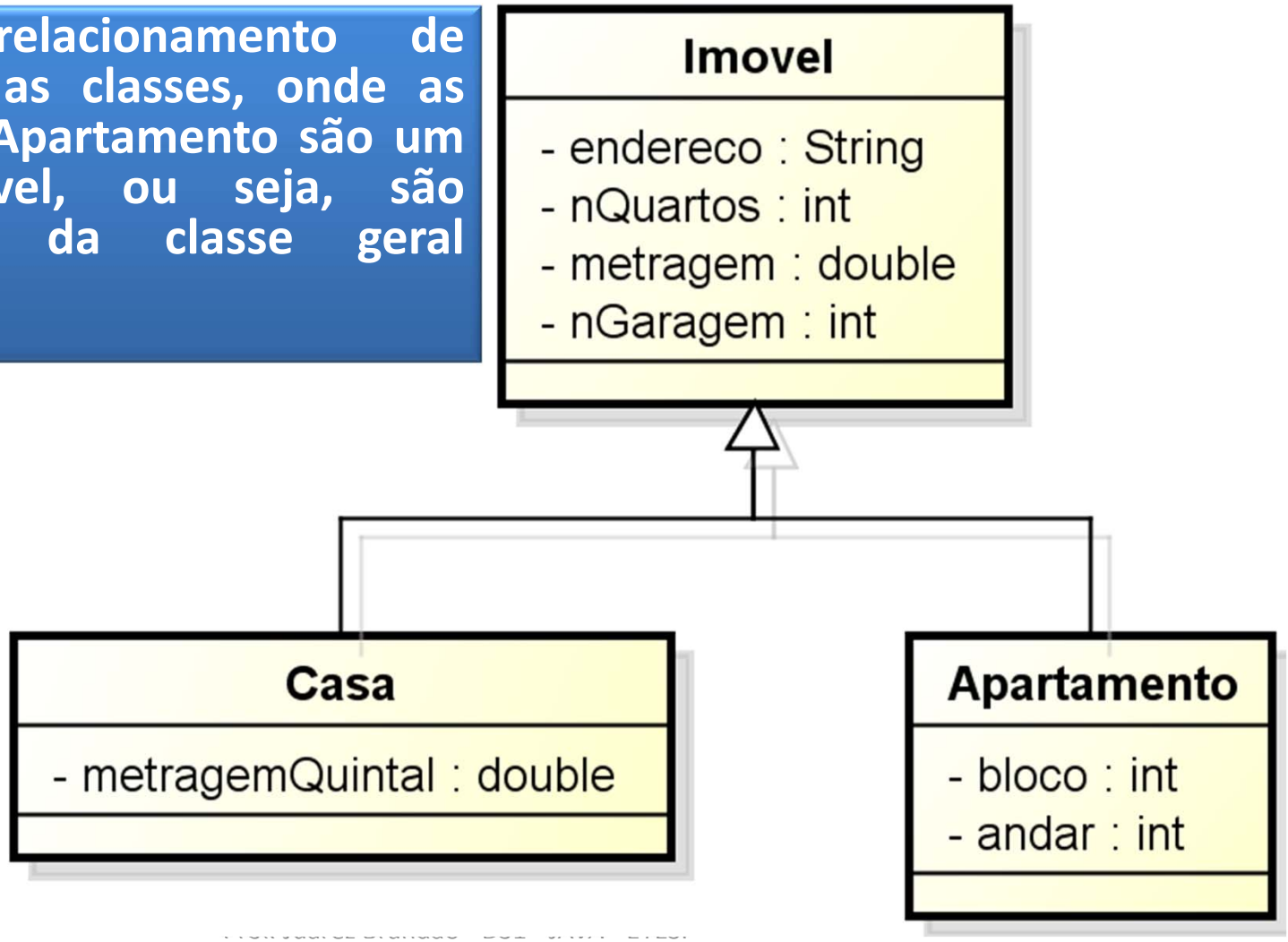
Acesso somente dentro da classe

Podem ser acessados por outras classes

Acesso somente para classes dentro do mesmo pacote

GENERALIZAÇÃO E ESPECIALIZAÇÃO (HERANÇA)

Criando o relacionamento de herança entre as classes, onde as classes Casa e Apartamento são um tipo de Imovel, ou seja, são especializações da classe geral Imovel



Superclasse

Atributos

Construtores

Gets e Sets

```java
public class Imovel {
    private String endereco;
    private int nQuartos;
    private double metragem;
    private int nGaragem;

    public Imovel() {
        this("",0,0.0,0);
    }

    public Imovel(String endereco,int nQuartos, double metragem, int nGaragem) {
        this.endereco = endereco;
        this.nQuartos = nQuartos;
        this.metragem = metragem;
        this.nGaragem = nGaragem;
    }

    public String getEndereco() {
        return endereco;
    }

    public void setEndereco(String endereco) {
        this.endereco = endereco;
    }

    public int getnQuartos() {
        return nQuartos;
    }
}
```

Superclasse

Gets e Sets

```
    public void setnQuartos(int nQuartos) {
        this.nQuartos = nQuartos;
    }

    public double getMetragem() {
        return metragem;
    }

    public void setMetragem(double metragem) {
        this.metragem = metragem;
    }

    public int getnGaragem() {
        return nGaragem;
    }

    public void setnGaragem(int nGaragem) {
        this.nGaragem = nGaragem;
    }
}
```

Subclasse - Apartamento

Veja a linha da declaração da classe, utilizamos o comando extends para demonstrar a Herança, onde a classe Apartamento herda a classe Imovel

Já no construtor vazio, passamos os parâmetros para os atributos herdados além dos atributos já existentes na classe.

•No construtor com passagem de parâmetro, passamos os parâmetros necessários para os atributos da superclasse, e dentro do construtor chamamos o método super(), que chama o construtor por parâmetro da superclasse.

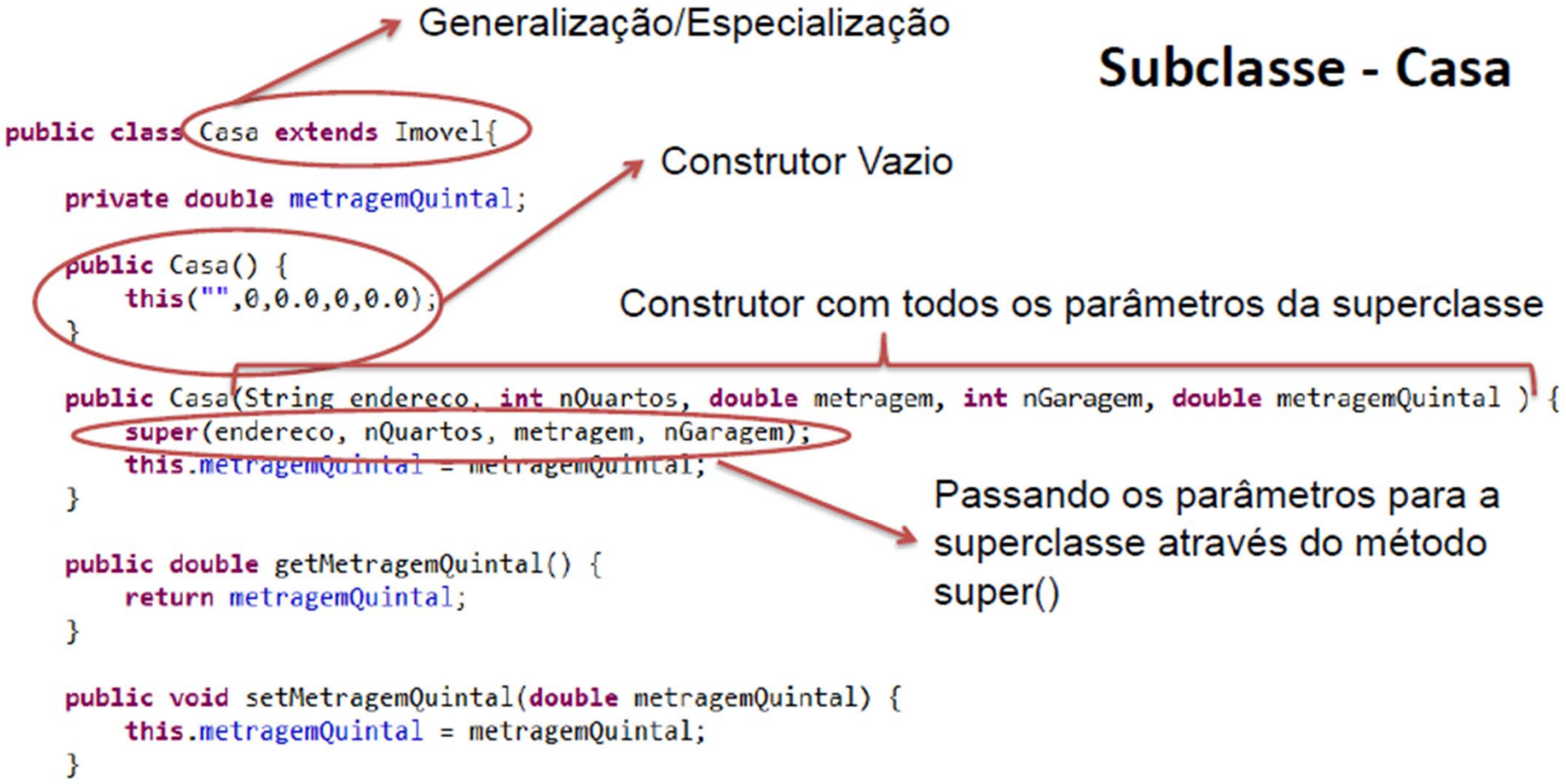
```
public class Apartamento extends Imovel {
    private int bloco;
    private int andar;

    public Apartamento() {
        this("",0,0.0,0,0,0);
    }

    public Apartamento(String endereco, int nQuartos, double metragem, int nGaragem,int bloco, int andar){
        super(endereco, nQuartos, metragem, nGaragem);
        this.bloco = bloco;
        this.andar = andar;
    }
```

Subclasse - Apartamento

```
public int getBloco() {
    return bloco;
}

public void setBloco(int bloco) {
    this.bloco = bloco;
}

public int getAndar() {
    return andar;
}

public void setAndar(int andar) {
    this.andar = andar;
}
```

Os métodos gets/sets somente para os atributos da classe, pois os outros já foram herdados ao utilizar o comando extends

Subclasse - Casa




```
public class Casa extends Imovel{

    private double metragemQuintal;

    public Casa() {
        this("",0,0.0,0,0.0);
    }

    public Casa(String endereco, int nQuartos, double metragem, int nGaragem, double metragemQuintal ) {
        super(endereco, nQuartos, metragem, nGaragem);
        this.metragemQuintal = metragemQuintal;
    }

    public double getMetragemQuintal() {
        return metragemQuintal;
    }

    public void setMetragemQuintal(double metragemQuintal) {
        this.metragemQuintal = metragemQuintal;
    }
```