Programação para Web I
Revisão O.O.

Profa. Silvia Bertagnolli

Orientação a Objetos

Principais conceitos:

- abstração (classe, objeto, etc.)
- herança e composição de objetos (relacionamento entre objetos)
- polimorfismo
- encapsulamento

CLASSE E OBJETO

Qual a diferença entre classe e objeto?

nome	Fulano
telefone	33224455
endereco	Rua X, 20

nome	Juca
telefone	33224466
endereco	Rua Y, 30

Como declarar a classe Pessoa?


```
public class Pessoa{

        //Atributos



        //Construtores



        //Métodos

}
```

Como fica o código da classe Pessoa?


```
public class Pessoa{

        //Atributos
        private String nome;


        //Construtores

        //Métodos

}
```

Orientação a Objetos: Classe

```
public class Pessoa{
    //Atributos
    //Construtores
    public Pessoa(){}
    public Pessoa(String nome, String telefone, String endereco){
        this.nome = nome;
        this.telefone = telefone;
        this.endereco = endereco;
    }
    //Métodos
}
```

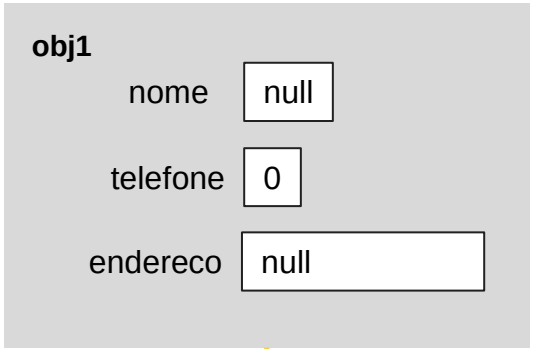
Orientação a Objetos: Classe


```
public class Pessoa{
    //Atributos
    //Construtores
    //Métodos
    public String toString(){
          return "Pessoa [nome=" + nome + "]";
    }
    public boolean equals(Object e){ … }
    public void setNome(String nome){
       this.nome = nome;
    }
    public String getNome(){return nome;}
    //outros métodos
}
```

Orientação a Objetos: Classe

```
public class Pessoa{
    private String nome;
    private String telefone;
    private String endereco;
    public Pessoa(){}
    public Pessoa(String nome, String telefone, String endereco){
        this.nome = nome;
        this.telefone = telefone;
        this.endereco – endereco;
    }
    public String toString(){return "Pessoa [nome=" + nome + "]";}
    public boolean equals(Object e){ … }
    public void setNome(String nome){ this.nome = nome; }
    public String getNome(){return nome;}
    //outros métodos
}
```

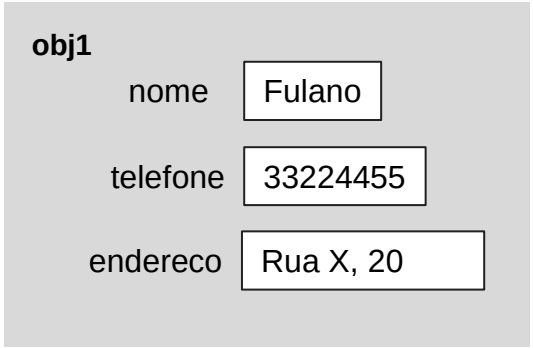
Qual a diferença entre classe e objeto?

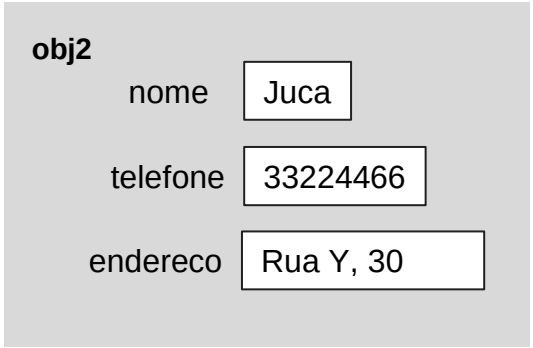



```
public class Teste{
  public static void main(String args[]){
    Pessoa obj1 = new Pessoa();
    System.out.println(obj1);
   }
}
```

Tipo	Valor inicial padrão do atributo
Inteiro (byte, short, int e long)	0
Ponto flutuante (float e double)	0.0
char	‘ ‘
boolean	false
Qualquer classe	null

Qual a diferença entre classe e objeto?





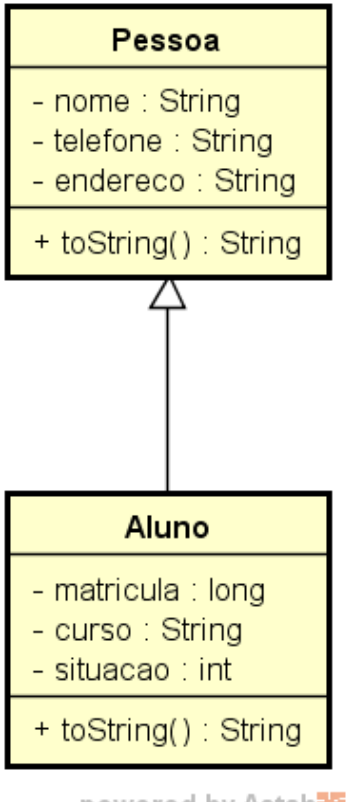

```
public class Teste{
  public static void main(String args[]){
    Pessoa obj1 = new Pessoa();
    obj1.setNome("Fulano");
    obj1.setNome("33224455");
    obj1.setNome("Rua X, 20");

    Pessoa obj2 = new Pessoa("Juca",
                "33224466","Rua Y, 30");

   }
}
```

Herança

Orientação a Objetos: Herança

- Superclasse e subclasse
- Relacionamento **é-um?**
- Na linguagem de programação Java **Object** é a superclasse das classes que não definem uma superclasse comum
- Para referenciar construtores e métodos da superclasse usamos a palavra reservada **super**

Como declarar atributos e construtores classe Aluno?



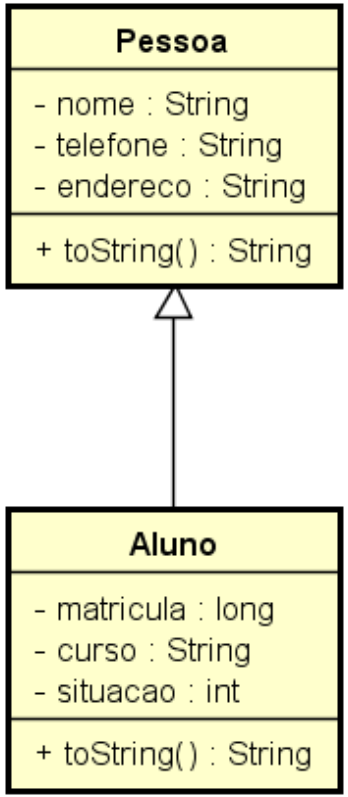

```
public class Aluno ___________________ {



}
```

Sobrescrita: anotação @override

- Esta anotação diz ao compilador que está sendo reescrito um método na classe
- Isso impede que alterações sejam realizadas na assinatura do método toString()
- Evita problemas com o uso da sobrecarga, exemplo:

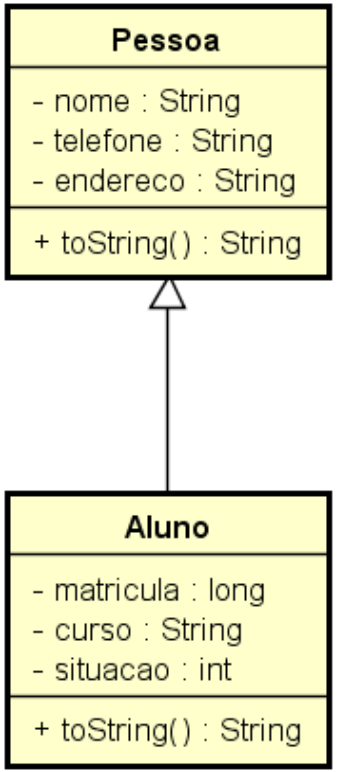
```
public double calculaSalario(double taxa){...}

public int calculaSalario(int taxa){...}
```

Como fica o código da classe Aluno?



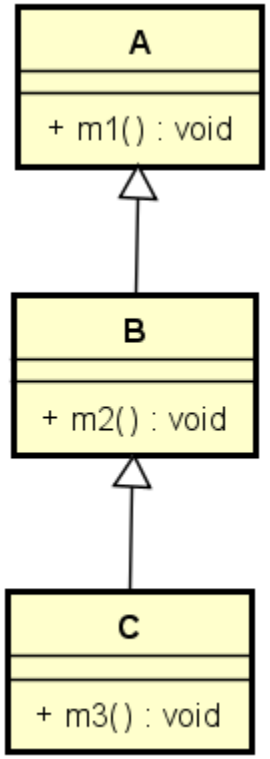

```
public class Aluno ___________________ {



    @Override
    public String toString(){
        return __________;
    }

}
```

Ligação Dinâmica: chamada de métodos




```
Aluno aluno = new Aluno();

aluno.toString();

aluno.imprime();

aluno.hashCode();

aluno.charAt(3);
```

Ligação Dinâmica: chamada de métodos




```
A a = new A();
a.m1();
B b = new B();
b.m1();
b.m2();
A b2 = new B();
b2.m1();
C c = new C();
c.m1();
c.m2();
c.m3();
```

TRANSIÇÃO

Polimorfismo

Orientação a Objetos: anotação @override

- Esta anotação diz ao compilador que está sendo reescrito um método na classe
- Isso impede que alterações sejam realizadas na assinatura do método toString(), por exemplo
- Evita problemas com o uso da sobrecarga, exemplo:

```
public double calculaSalario(double taxa){...}

public int calculaSalario(int taxa){...}
```

Instanceof x Cast

instanceof é um operador e determina qual a classe de um dado objeto

```
Pessoa p[] = new Pessoa[3];
p[0] = new Pessoa();
p[1] = new Aluno();
if(p[0] instanceof Pessoa)
    System.out.println(“Objeto é uma pessoa”);
```

Instanceof x Cast

cast “força” que um objeto seja convertido para outro

```
Pessoa p[] = new Pessoa[3];
p[0] = new Pessoa();
p[1] = new Aluno();
if(p[1] instanceof Aluno){
   Aluno a = (Aluno)p[1];
   System.out.println(a.getCpf());

//ou:
if(p[1] instanceof Aluno){
System.out.println(((Aluno)p[1]).getCpf());
```

Relembrando...

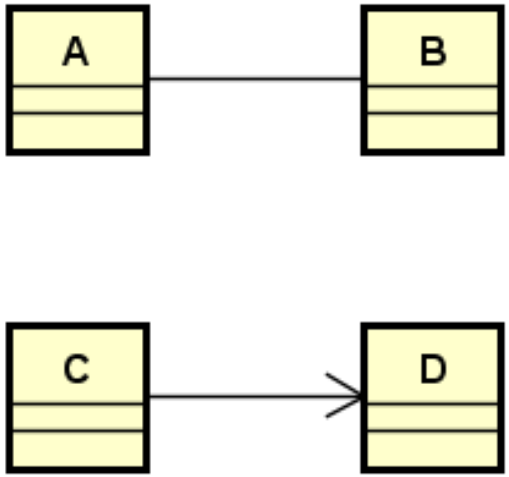
1. O que é uma variável polimórfica?
2. Dê um exemplo de polimorfismo de sobrecarga
3. Dê um exemplo de polimorfismo de sobrescrita

Composição

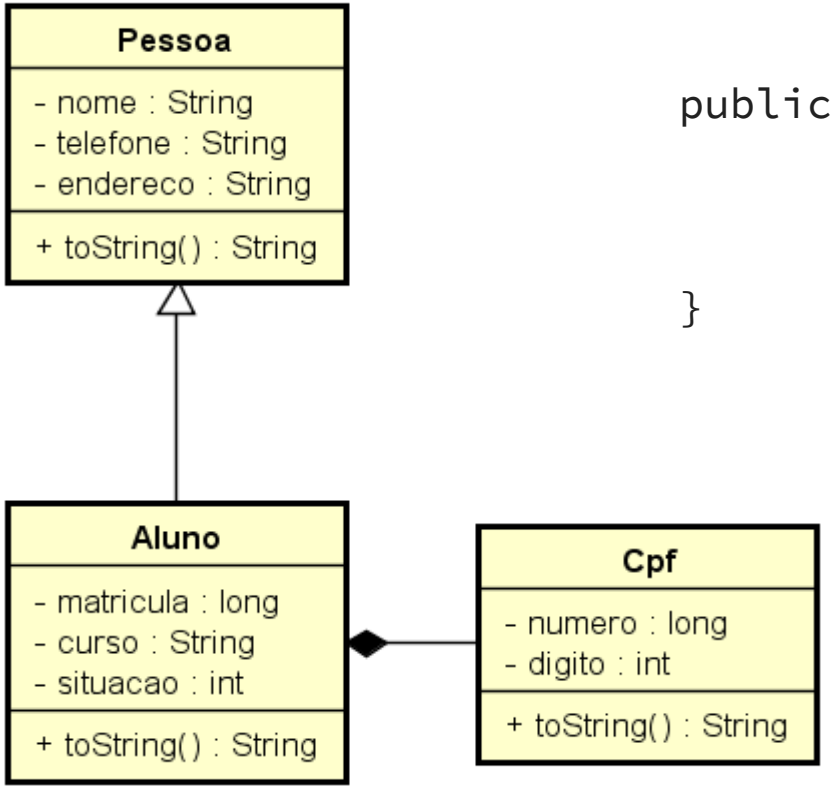
Orientação a Objetos: composição

- Relacionamento **tem-um?**
- Composição e Agregação (todo-parte)
- Associação (origem-destino)
- Multiplicidade deve ser analisada
    - 1..1
    - 1..*
    - 2..3

Como implementar os relacionamentos abaixo?

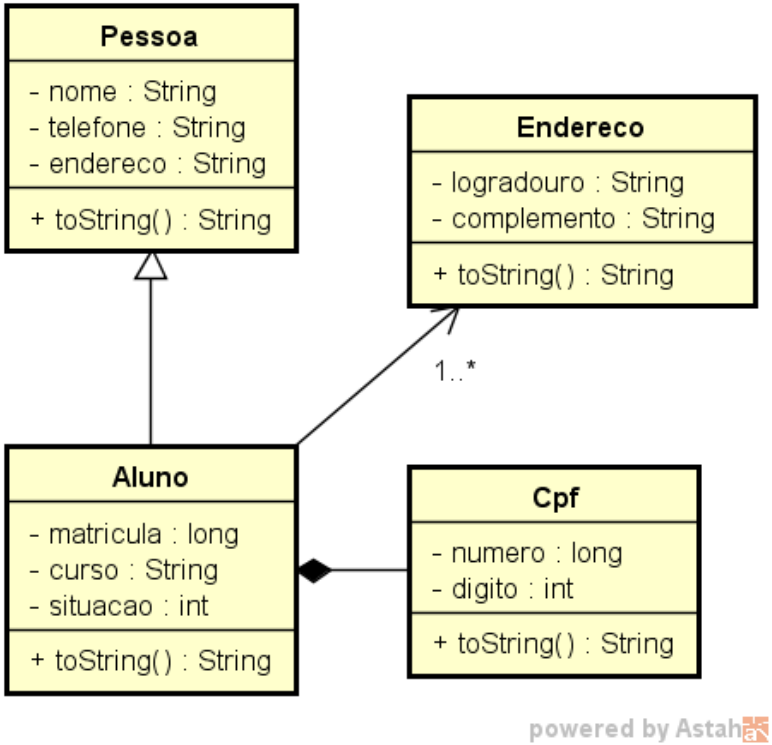


Como fica o código da classe Aluno?




```
public class Aluno extends Pessoa {

        private Cpf cpf;

}
```

Como fica o código da classe Aluno?

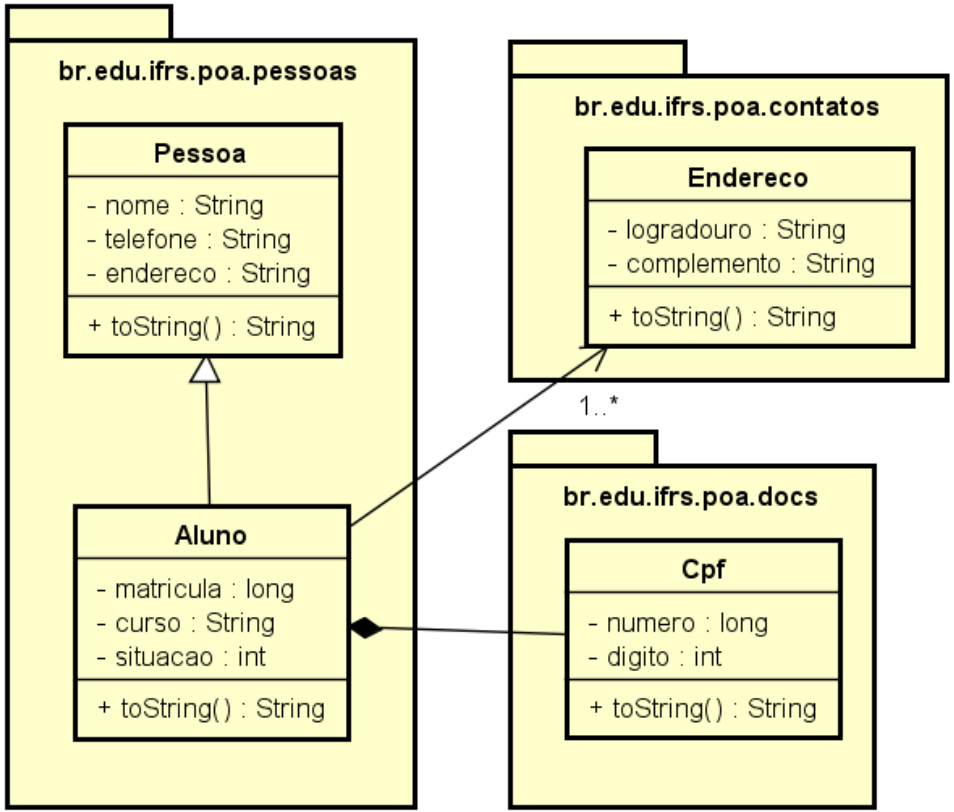



```
import java.util.*;

public class Aluno extends Pessoa {

    private Endereco enderecos[];

    //OU:

    private List<Endereco> enderecos;



}
```

Como fica Aluno com a composição de múltiplos?

```
Import java.util.*;

public class Aluno extends Pessoa {

        private Endereco enderecos[];

        //OU

        private List<Endereco> enderecos;



}
```

Como fica o código da classe usando Pacotes?

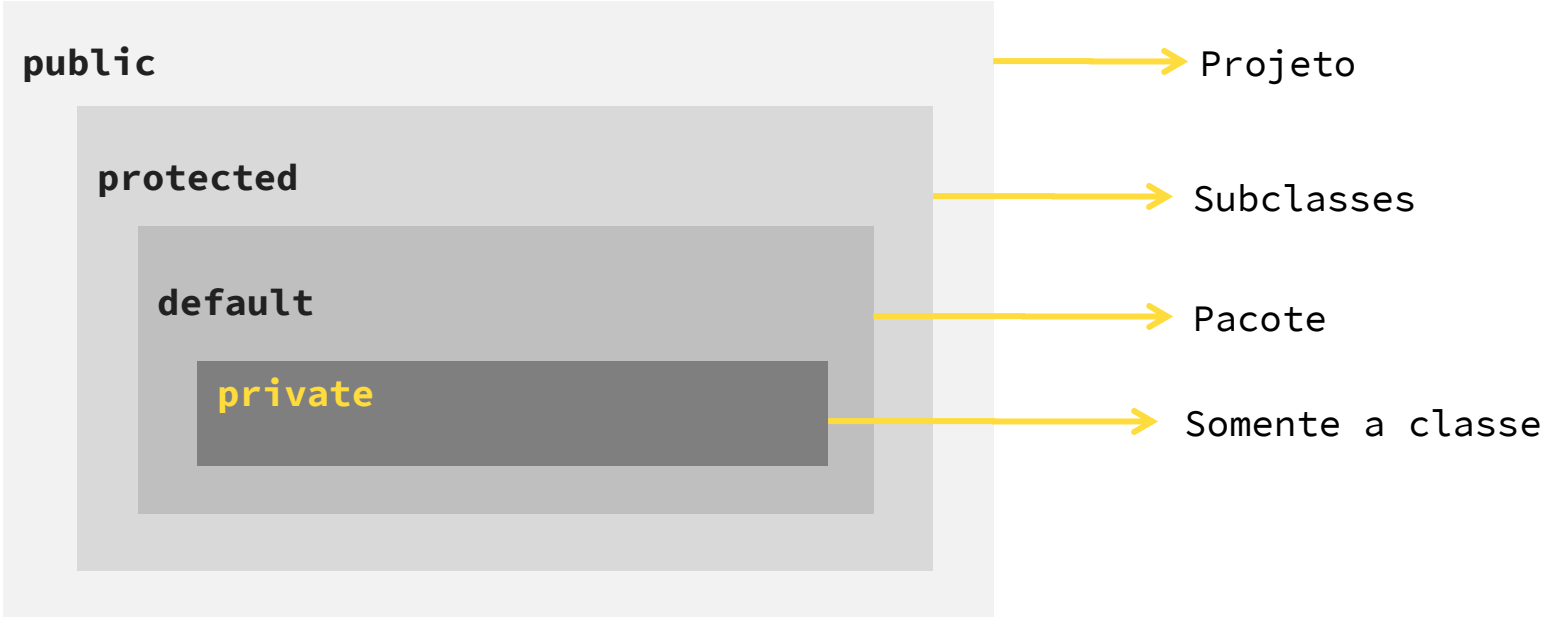


Como fica o código da classe Aluno agora?

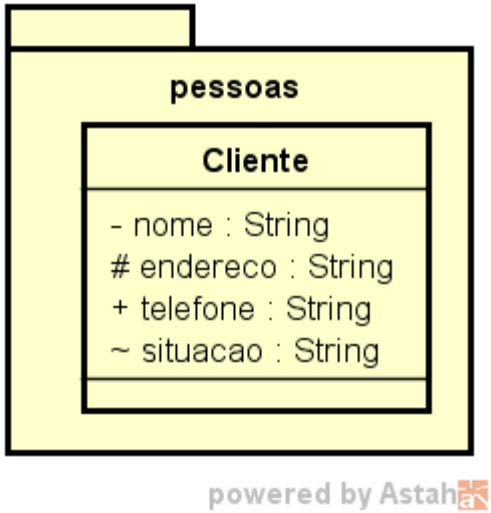
```
package br.edu.ifrs.poa.pessoas;

import br.edu.ifrs.poa.contatos.*;

import br.edu.ifrs.poa.contatos.Endereco;



_____________________________________

_____________________________________

public class Aluno ___________________ {
```

Encapsulamento

Modificadores e sua Visibilidade



Uma Classe na UML e uma classe na linguagem Java




```
package pessoas;

public class Cliente{

        private String nome;

        __________ String endereco;

        __________ String telefone;

        __________ String situacao;

...

}
```

Recomendações

Modificador/Elemento	Classe	Método	Atributo
public	✓	✓	✓
private	✗	✓	✓
protected	✗	✓	✓

Resumo

Visibilidade	public	protected	private	default
Da mesma classe	✓	✓	✓	✓
De qualquer classe no mesmo pacote	✓	✓		✓
De qualquer classe que não seja uma subclasse externa ao pacote	✓			
De uma subclasse do mesmo pacote	✓	✓		✓
De uma subclasse externa ao pacote	✓	✓		

Exercícios

1. Crie as classes da figura ao lado

2. Agora, monte o menu abaixo usando vetores

1 - Cadastrar Aluno
2 - Pesquisar Aluno pelo nome
3 - Pesquisar Aluno pelo CPF
4 - Listar todos os alunos
5 - Sair

Obs.: Para resolver o item 3 do menu você deverá utilizar **instanceof** e/ou **cast** de objetos