GAMA ACADEMY

# POO

## Iniciando o entendimento

# Classes

01

## Estrutura de uma classe básica

# O que é uma classe

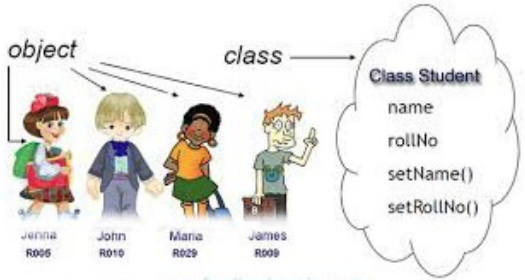
Classe é um tipo de dado

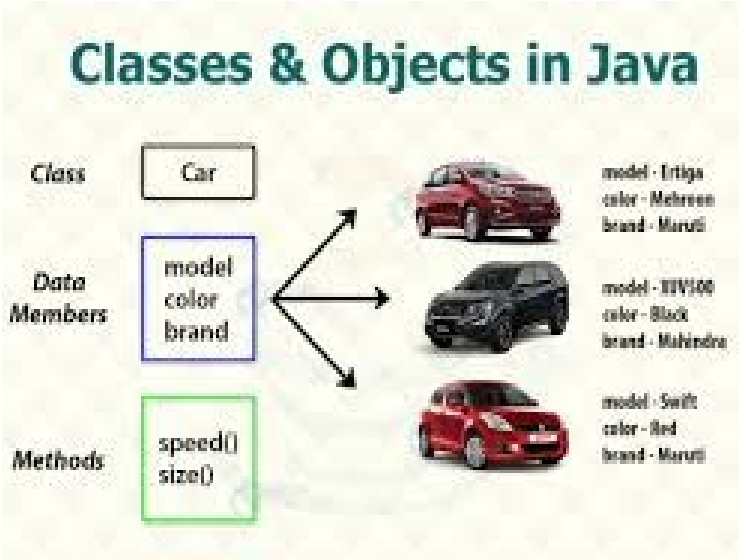
Toda Classe herda Object

A Classe serve para a definição de um Objeto

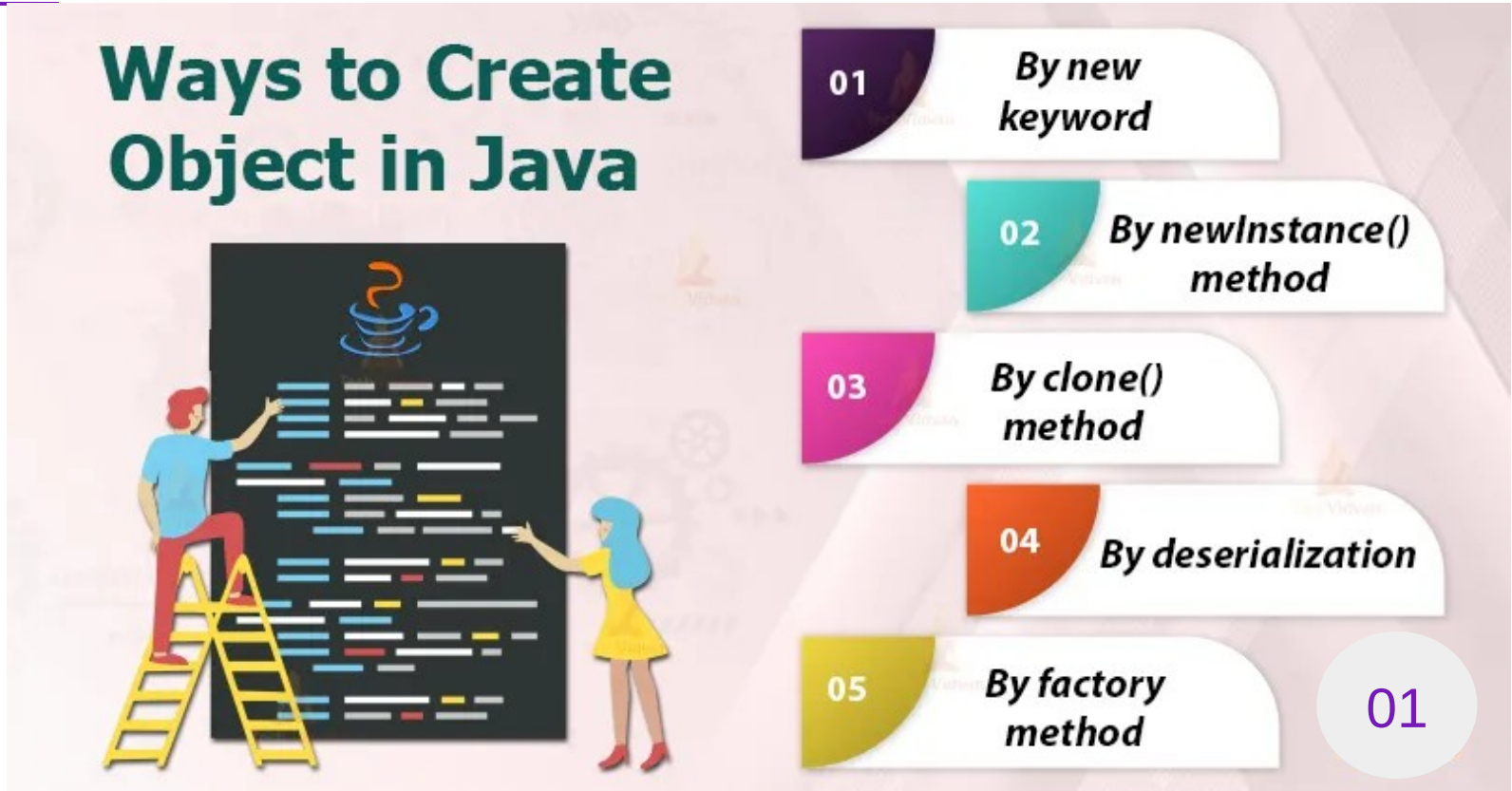
Uma classe deve ser coesa e ter baixo acoplamento

# Objeto é a instância de uma classe

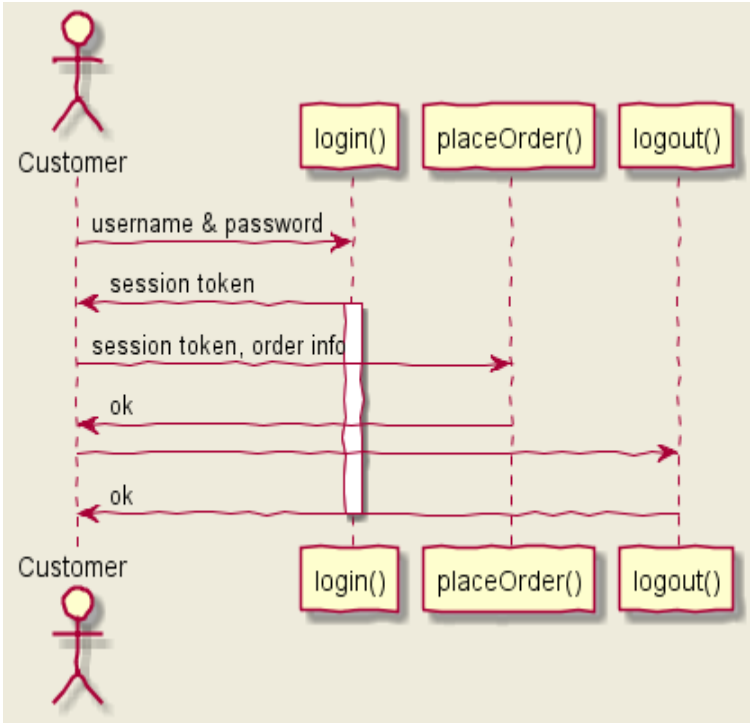


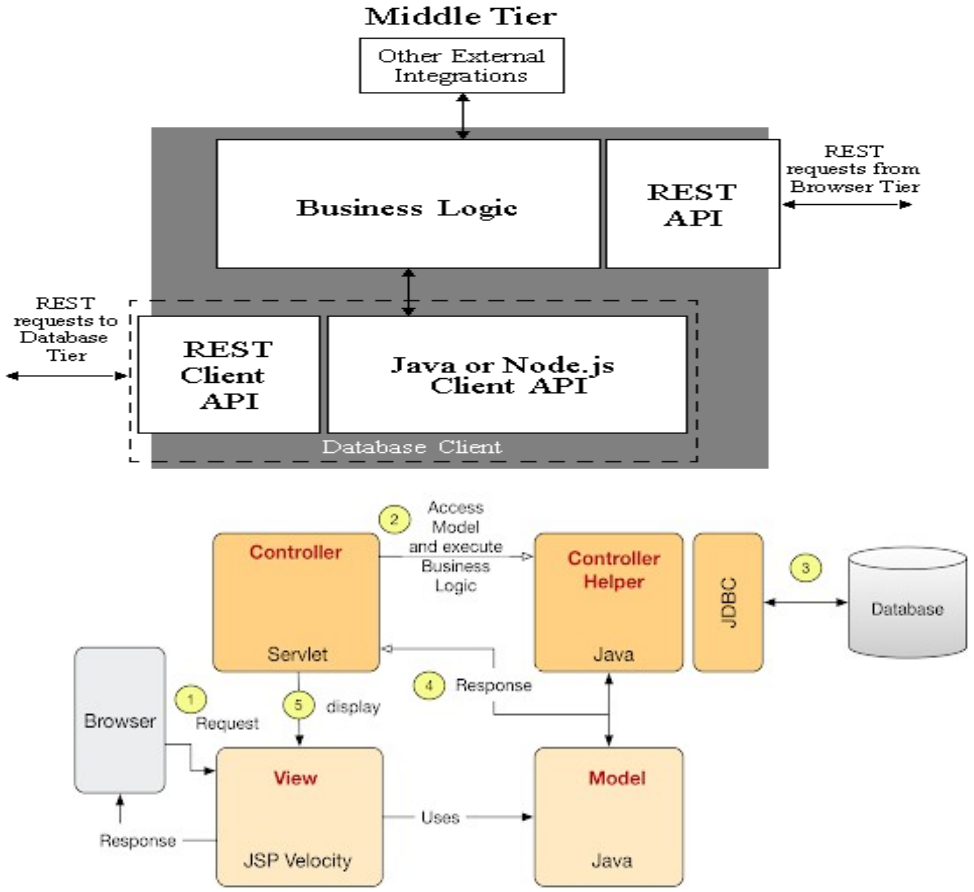



Como criar um Objeto

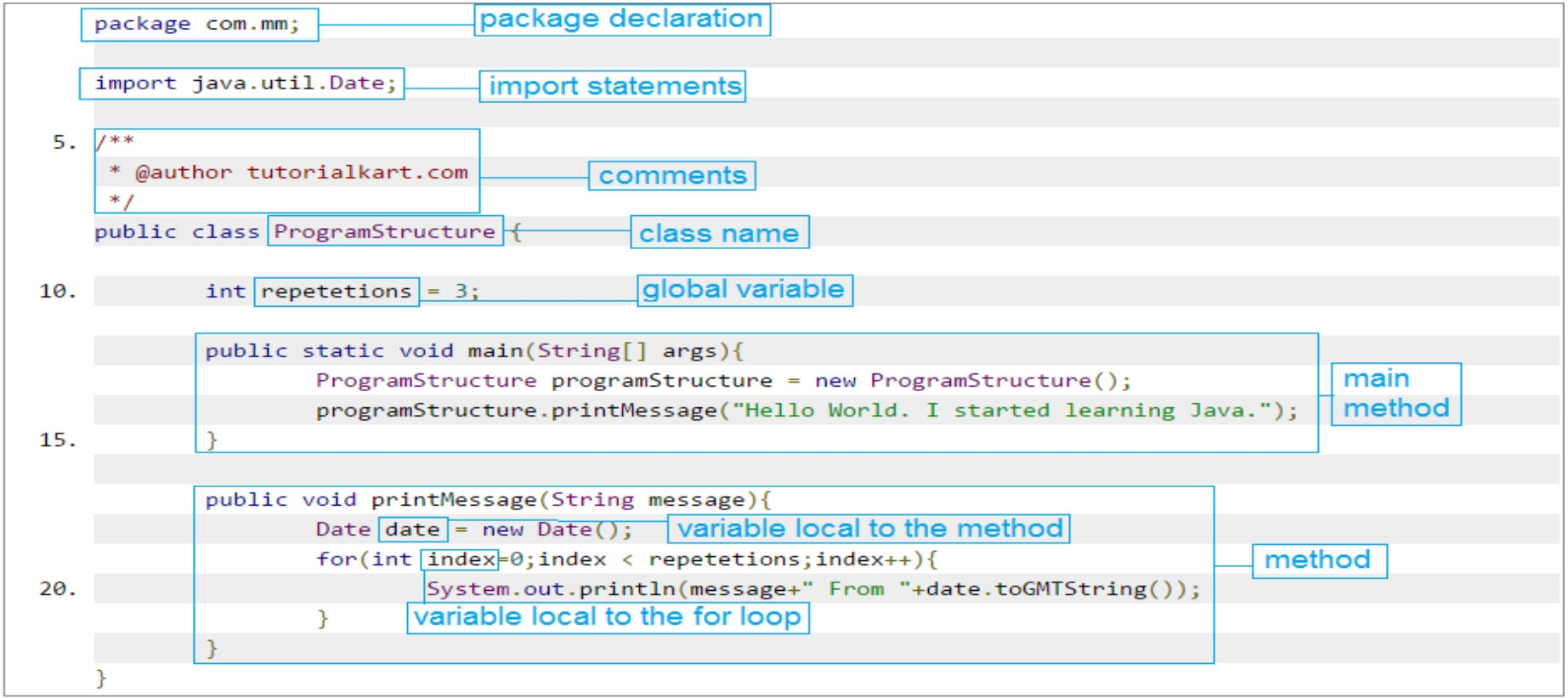


# Papel de um Objeto Java no mundo dos sistemas

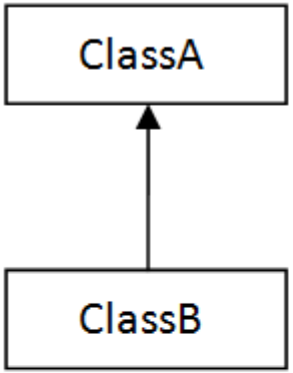


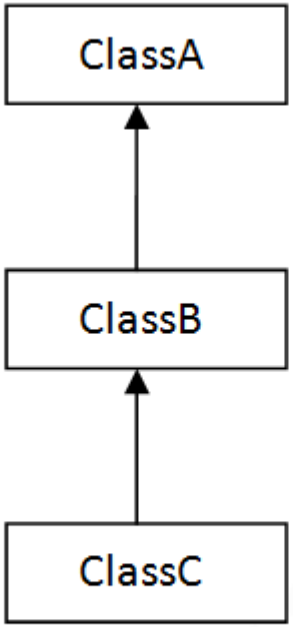
# Estrutura de uma Classe



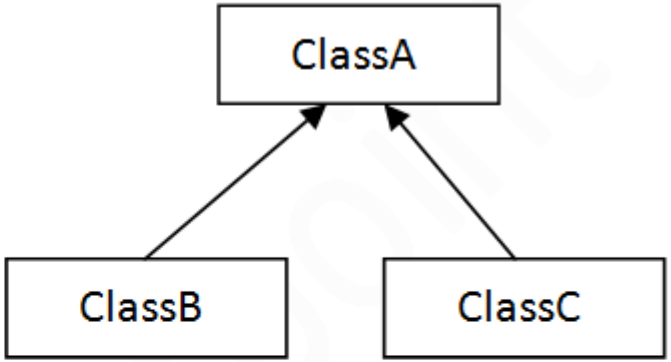
# Herança (extends)




1) Single

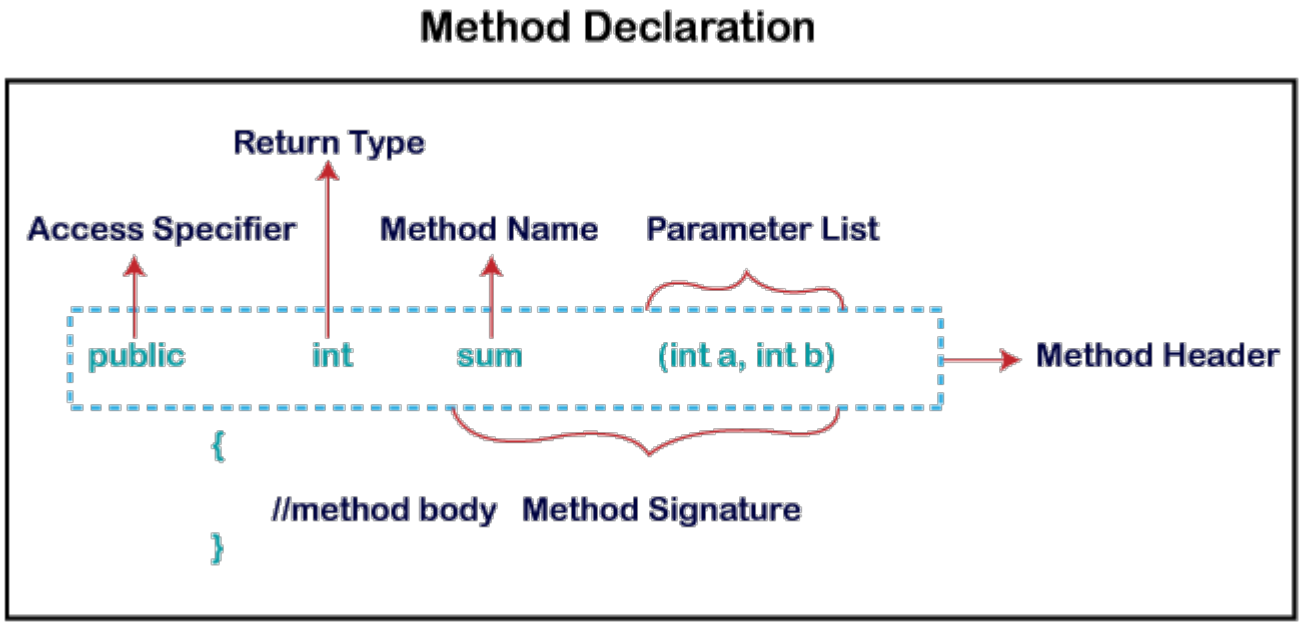



2) Multilevel

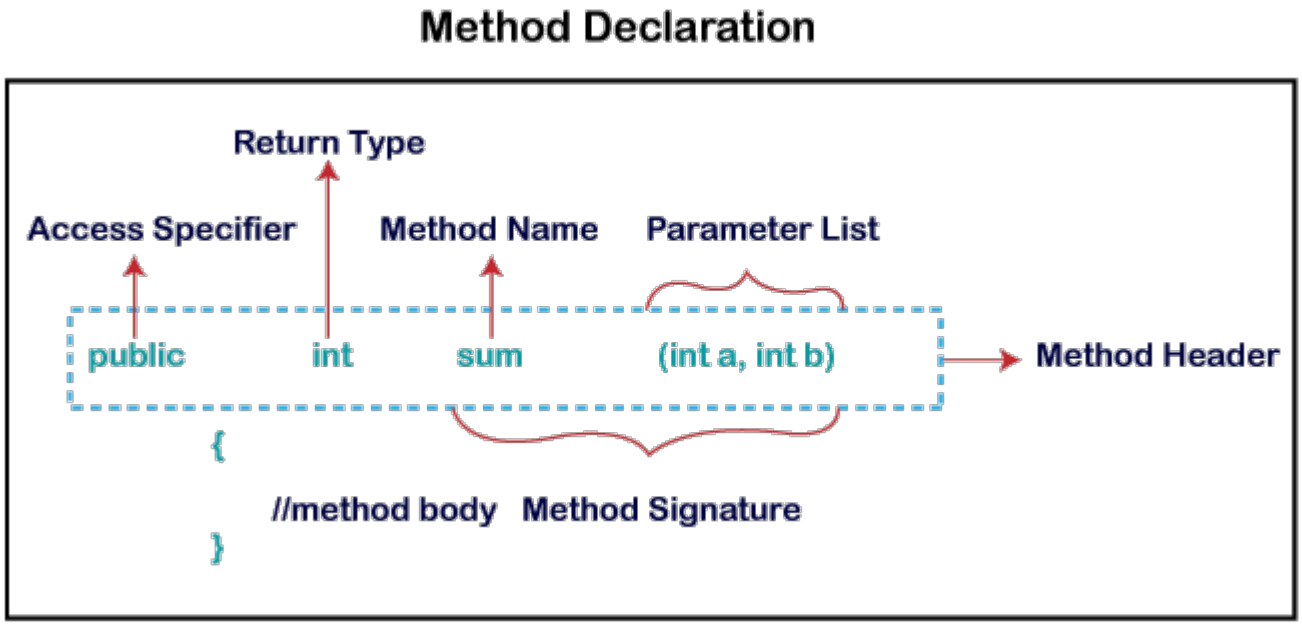



3) Hierarchical

# Métodos


Method Declaration
Return Type
Access Specifier
Method Name
Parameter List
public
int
sum
(int a, int b)
Method Header
{
//method body
Method Signature
}

# Métodos


Method Declaration
Return Type
Access Specifier
Method Name
Parameter List
public
int
sum
(int a, int b)
Method Header
{
//method body
Method Signature
}

# Sobrescrita e Sobrecarga dos Métodos

Overriding

```
class Dog{
    public void bark(){
        System.out.println("woof ");
    }
}
class Hound extends Dog{
    public void sniff(){
        System.out.println("sniff ");
    }

    public void bark(){
        System.out.println("bowl");
    }
}
```

Same Method Name,
Same parameter

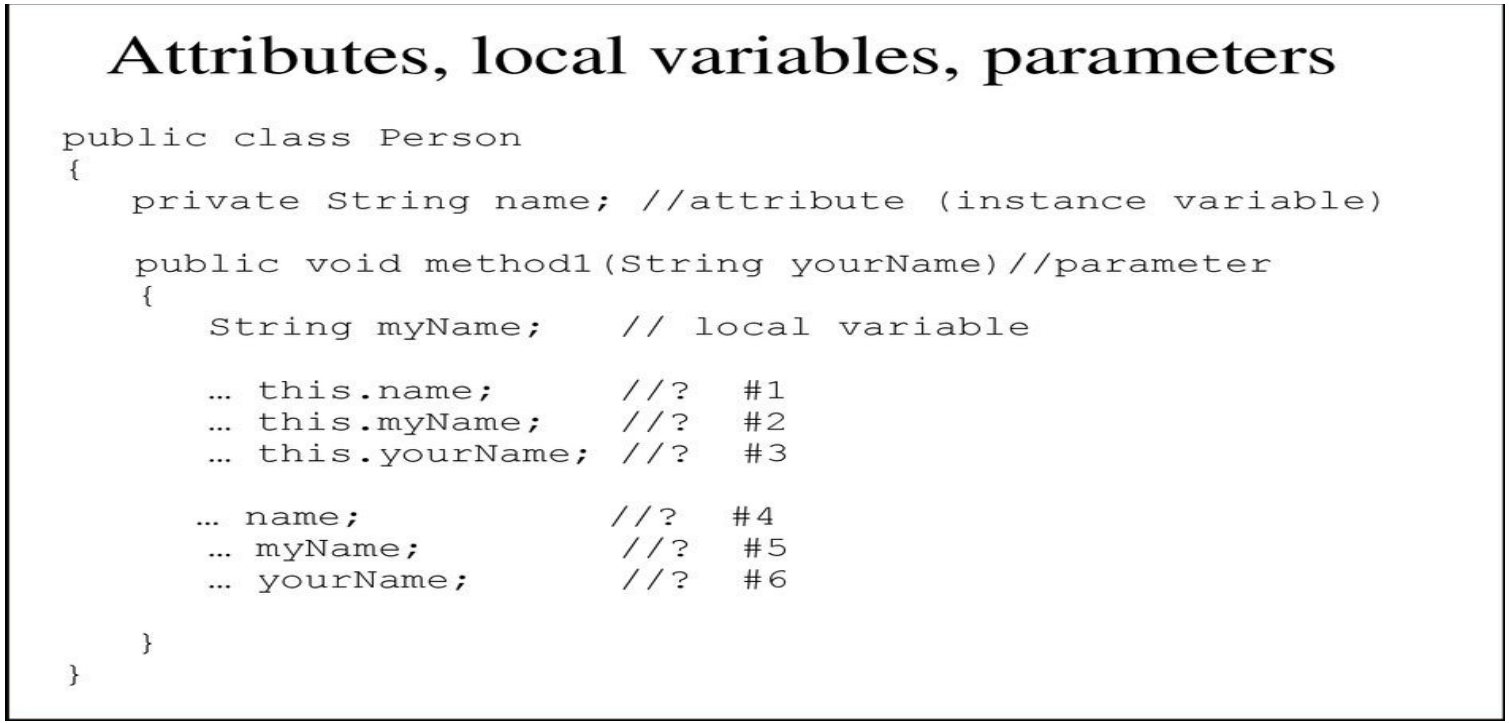
Overloading

```
class Dog{
    public void bark(){
        System.out.println("woof ");
    }

    //overloading method
    public void bark(int num){
        for(int i=0; i<num; i++)
            System.out.println("woof ");
    }
}
```

Same Method Name,
Different Parameter

Atributos de Classe, Variável local e parâmetros



# Classes, Métodos e atributos estáticos

Classe estática

Método estático

Atributos/variáveis estáticas

Bloco estáticos

# Classes estáticas

InnerClassDemo.java

```java
public class InnerClassDemo {
    public static void main(String[] args) {
        // Accessing the Static Inner class
        InnerClassDemo.StaticInnerClass innerClass = new StaticInnerClass();
        System.out.println("StaticInnerClass value  : " + innerClass.getValue());
        // Accesing the Normal Inner class
        InnerClassDemo.NormalInnerCalss normalInnerCalss = new InnerClassDemo().new NormalInnerCalss();
        System.out.println("normalInnerCalss Value : "+ normalInnerCalss.getValue());
    }
    static class StaticInnerClass {
        int a = 10;
        public int getValue() {
            return a;
        }
    }
    class NormalInnerCalss {
        int instVar = 20;
        public int getValue() {
            return instVar;
        }
    }
}
```

# Métodos estáticos

Os métodos estáticos podem ser acessados diretamente usando o nome da classe.

Os métodos estáticos não podem ser substituídos.

Os métodos não estáticos podem acessar métodos estáticos apenas usando o nome da classe.

Os métodos estáticos também podem acessar os métodos não estáticos usando a instância da classe.

Os métodos estáticos e não estáticos não são acessados diretamente.

Um método estático não pode se referir a “this” ou “super” em qualquer lugar.

## Atributos/Variáveis estáticas

Variáveis estáticas são declaradas com a palavra-chave static.

Variáveis estáticas também são chamadas de variáveis de classe.

As variáveis de classe pertencem a toda a classe e não a uma instância específica da classe.

Uma única variável estática pode ser compartilhada por todas as instâncias de uma classe.

Não podemos acessar as variáveis estáticas dos métodos normais.

Variáveis de classe são alocadas na memória apenas uma vez no momento do carregamento da classe, e que pode ser comumente acessado por todas as instâncias da classe.

Variáveis estáticas são alocadas na memória do pool estático.

Como a memória para as variáveis da classe é alocada no momento do carregamento da própria classe, podemos acessar as variáveis estáticas diretamente com o próprio nome da classe.

# Blocos estáticos

Temos diferentes tipos de blocos em Java, como bloco de inicialização, bloco sincronizado e bloco estático.
Cada bloco tem sua própria importância.
Aqui, um bloco estático é um bloco de instruções, que é definido usando a palavra-chave static.

```
static{
    //Code
}
```

# Objetos e referência

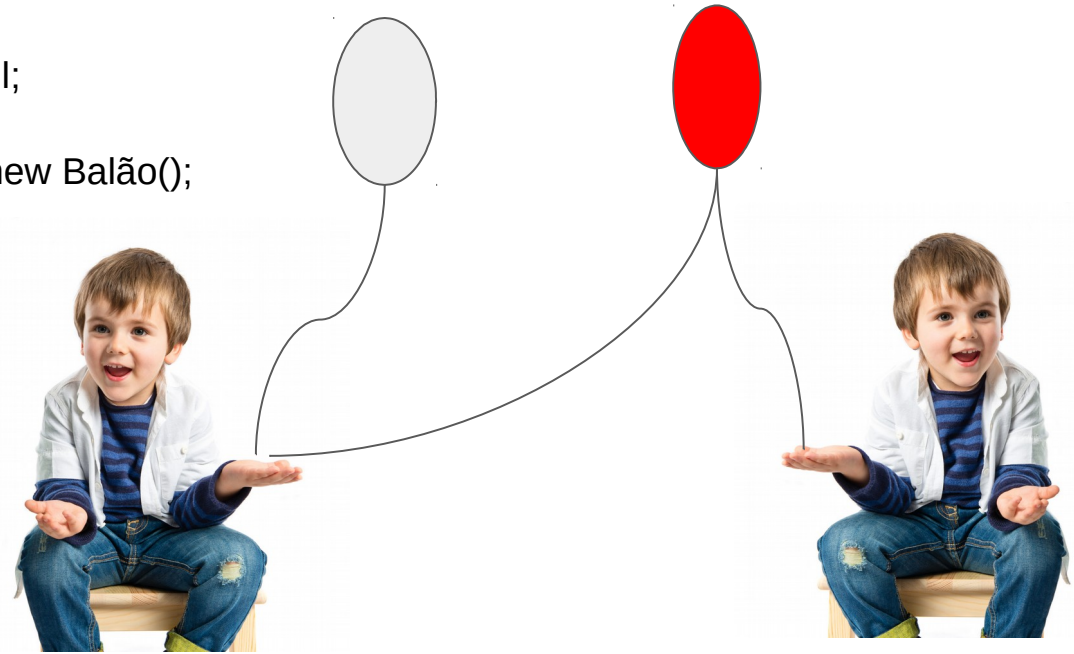
Uma referência  é utilizada para armazenar o endereço de um objeto alocado na memória.
Pela referência é manipulado o estado do objeto.
Pode ser um atributo, variável local ou argumentos de métodos.

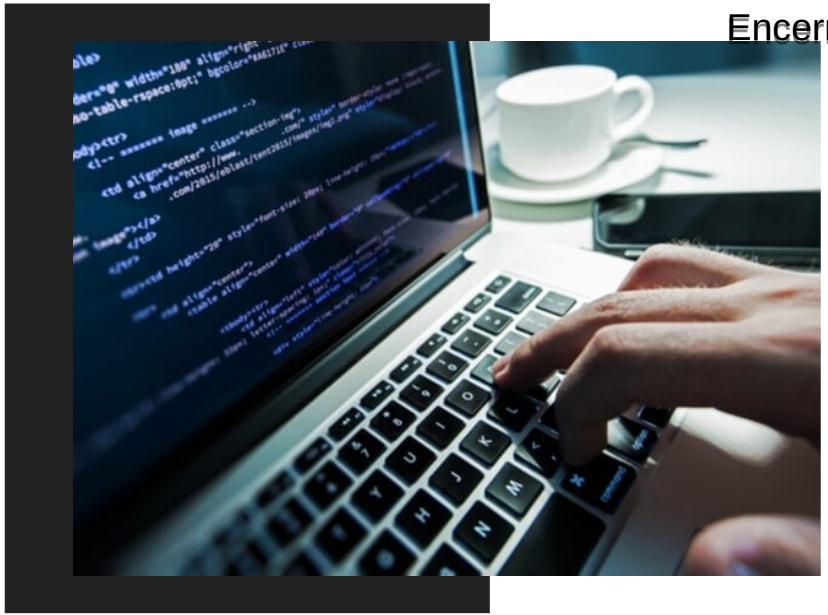
Balão criança = null;

Balão criança = new Balão();

criança = new Balão();

Balão criança2 = criança;





Encerramos por hoje !!!