**Métodos**

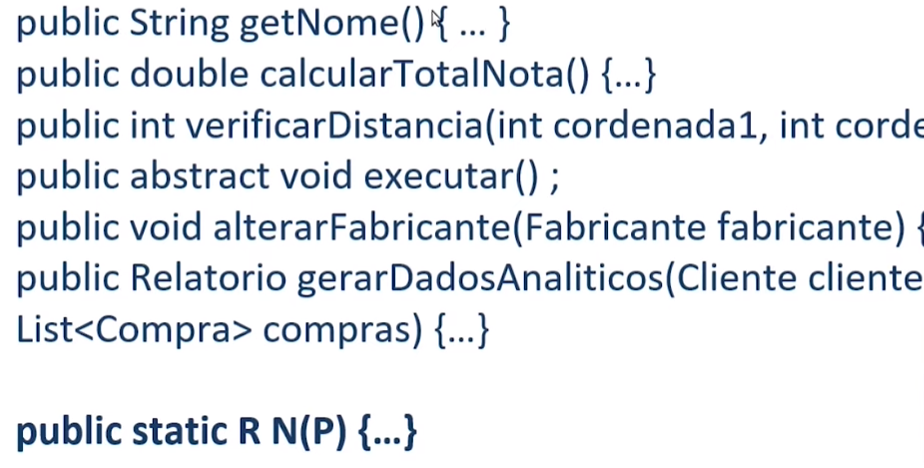
**Criação**

“é uma porção de código (sub-rotina) que é disponibilizada por uma classe. Este é executado quando é feita uma requisição a ele. São responsáveis por definir e realizar um determinado comportamento”

**Padrão de definição:**

**<?visibilidade?> <?tipo?> <?modificador?> retorno nome (<?parametros?>) <?exceções?> corpo**

* **V**: “public”, “protected” e “private”
* **M**: “static” e “final”
* **T**: concreto ou abstrato
* **N**: nome que é fornecido ao método
* **R:** tipo de dado ou “void”
* **P:** parâmetros que pode receber
* **E:** exceções que pode lançar
* **C:** código que possui ou vazio



**public static TipoDeDado NomeFornecidoAoMetodo (parâmetros) {...}**

**Utilização**

Passa-se uma mensagem através de uma classe ou objeto

nome\_da\_classe.nome\_do\_metodo(); ou nome\_da\_classe.nome\_do\_metodo (...);

nome\_do\_objeto.nome\_do\_metodo(); ou nome\_do\_objeto.nome\_do\_metodo (...);

* **classe** – primeira letra maiúscula
* **objeto** – primeira letra minúscula

**Particularidades**

**Assinatura:** é a forma de identificar unicamente o método

**Ass = nome + parâmetros**

**Método:**

public double calcularTotalVenda (double precoItem1, double precoItem2, double precoItem3) { ... }

**Assinatura:**

calcularTotalVenda (double precoItem1, double precoItem2, double precoItem3)

**Construtor e Destrutor:** são métodos especiais usados na Orientação a Objetos

**Mensagem:** esta pode ser direcionada a um objeto ou a uma classe

**Passagem de parâmetros:**

* por valor (cópia)
* por referência (endereço)
* *int* i = 10;
* public void fazerAlgo (*int* i) {
* i = i + 10;
* System.out.println ("Valor de i dentro: " + i);
* }
* System.out.println ("Valor de i fora: " + i);

**Boas práticas:**

* nomes devem ser descritivos, mas curtos
* notação camelo: verificarSaldo (); executarTransferencia (...); existeDebito ();
* deve possuir entre 80 e 120 linhas
* evite lista de parâmetros longas
* visibilidades adequadas

**Sobrecarga**

“É a capacidade de definir métodos para diferentes contextos, mas preservando seu nome”

**Criação:**

**Alterar a assinatura do método**

**Ass = nome + parâmetros**

converterParaInteiro (float f);

converterParaInteiro (doubled);

**Retornos**

* é uma instrução de interrupção
* simbologia: return

**Funcionamento:**

O método executa seu retorno quando:

* completa todas suas instruções internas
* chega a uma declaração explicita de retorno
* lança uma exceção

**Considerações:**

* o tipo de retorno do método é definido na sua criação e pode ser um tipo primitivo ou objeto
* o tipo de dado do return deve ser compatível com o do método
* se o método for sem retorno (void), pode ou não ter um “return” para encerrar sua execução

Exemplos:

public *String* getMensagem(){

    return "Olá";

}

public *double* getJuros(){

    return 2.36;

}

public *void* setIdade(){

    return 10;

}

public *void* executar (){

    ...

    return;

    ....

}