Introdução à Programação

Funções (Métodos)

emerson@paduan.pro.br

Modularização

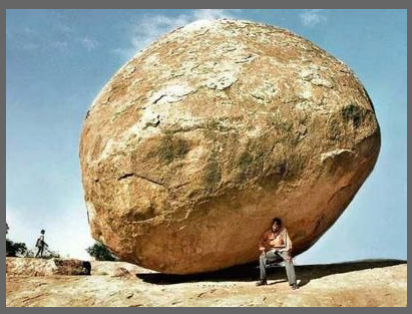
Dividir um problema em partes, denominadas módulos

- Cada módulo resolve uma tarefa específica
- O gerenciamento é feito pelo módulo principal que ”chama” ou “aciona” os outros módulos

emerson@paduan.pro.br

Modularização

Por que modularizar?



Modularização

Na programação estruturada são chamados de Procedimentos ou Funções.

Procedimentos não retornam valor ao final

Funções retornam um valor ao final da execução

Na programação OO são chamados **Métodos**

Estrutura

```
static void nomeMetodo( <parametros> )
{
    <instruções>
}
```


Estrutura

void: não retorna nada ao chamador do procedimento!

nomeMetodo: é um identificador. Obedece às regras de definição de nomes de variáveis.

parâmetros: (*opcional*) Lista de argumentos que serão passados para o método, separados por ” , ” . Deve-se especificar o tipo de cada parâmetro

Exemplo

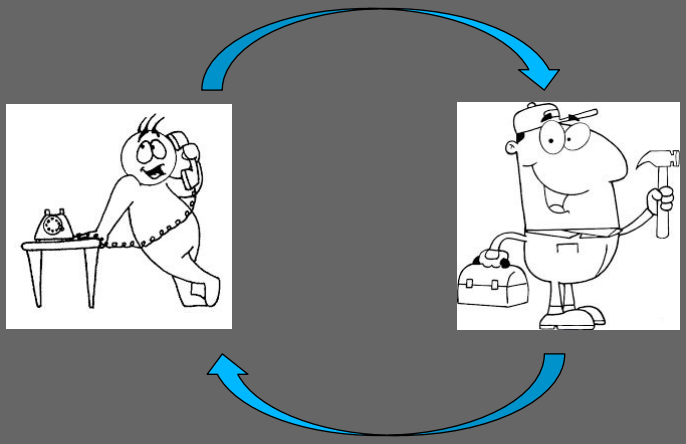
```java
public class teste1 {
    public static void main(String args[]) {
        exibirMenu();
    }

    static void exibirMenu(){
        System.out.println("*****************************");
        System.out.println("Programação OO - Prof. Emerson");
        System.out.println("*****************************");
    }
}
```


Chamadas

Um método pode chamar outro método . Isso é bastante comum para que os métodos tenham uma única tarefa e a manutenção e clareza do código seja o melhor possível.

Chamada de método



Chamadas

```java
public class teste1 {
    public static void main(String args[]) {
        exibir1();
    }

    static void exibir1(){
        System.out.println("No exibir 1");
        exibir2();
    }

    static void exibir2(){
        System.out.println("No exibir 2");
    }

}
```

Parâmetros

Parâmetros podem ser entendidos como informações que são enviadas para o método para que ele cumpra sua tarefa.

Por exemplo, considere que o método precisa saber qual o nome deve ser exibido, para que assim, o método fique genérico e sirva para qualquer nome.

Parâmetros

```
public class teste1 {
    public static void main(String args[]) {
        exibir("Emerson");
    }

    static void exibir(String nome){
        System.out.println("******************************");
        System.out.println("Programação OO - " + nome);
        System.out.println("******************************");
    }
}
```

Exercício

Escreva um método que receba uma mensagem e um caractere como parâmetros, e exiba na tela a mensagem com uma linha do tamanho da mensagem em abaixo desta. A linha deve ser formada com o caractere passado como parâmetro. *Dica: para saber o tamanho da String: use o método length()*

Exemplo:

Parâmetros: Emerson, -

Emerson
- - - - - - -

Exercício

Escreva um método que receba como parâmetro um número inteiro e exiba na tela se o valor é positivo ou negativo.

Variação do exercício: o método retorna uma informação para o main, ao invés de exibir na tela o resultado.

Retorno de métodos

```java
public static void main(String[] args) {
    int valorRetornado;

    valorRetornado = somar( 10, 50);

    System.out.println("A soma é: " + valorRetornado);

}
```

```java
static int somar(int n1, int n2){
    int resultado;
    resultado = n1 + n2;
    return resultado;
}
```


Escopo de variáveis

O escopo diz respeito ao local ou bloco onde a variável é válida.

Uma variável é válida, ou existe, somente dentro do método onde foi criada. Assim, se uma variável é criada dentro de um método, ela somente pode ser usada dentro deste método.

Exercício

Criar o método **encontrarMaior**() que retorne o maior valor entre três números inteiros recebidos como parâmetro.

Os valores devem ser digitados pelo usuário no main.

emerson@paduan.pro.br