

# FIAP GRADUAÇÃO

# ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

COMPUTATIONAL THINKING

PROF<sup>a</sup>. EVELYN CID

# CONTEÚDO PROGRAMÁTICO

## ☐ Métodos

### ➤ Procedimentos

### ➤ Funções

# I Modularidade

Em linhas gerais, problemas complexos exigem, para sua solução, algoritmos complexos, no entanto é possível dividir um problema grande em problemas menores, ou seja, usar o processo de modularidade. Cada parte menor ou módulo tem um algoritmo mais simples, o que facilita chegar a grande solução.

**Módulo é um bloco de programa que pode efetuar operações computacionais de entrada, processamento e saída.**

Podemos diminuir a complexidade dos problemas utilizando duas maneiras diferentes que veremos a seguir:

## **Procedimentos e Funções**

# I Procedimentos

Também chamamos de **sub-rotina**, é um conjunto de instruções que realiza uma determinada tarefa.

Um procedimento deve receber uma **identificação** (nome), pode possuir, variáveis, operações, chamar outros *procedimentos e funções*.

```
public static void Nome do Procedimento(){  
    <declaração de variáveis>  
    <conjunto de instruções do procedimento>  
}
```

# Exemplo de Procedimento

```
1 import java.util.Scanner;
2
3 public class Aula08_1 {
4
5     public static void main(String[] args) {
6
7         System.out.println("Verificando o número");
8         verificar(); ← Chamada do Procedimento
9     }
10
11     public static void verificar(){ ← Nome do Procedimento
12
13         Scanner entrada = new Scanner(System.in);
14         int numero=0; ← Variável do Procedimento
15
16         System.out.println("Digite um número:");
17         numero=entrada.nextInt();
18
19         if (numero %2==1){
20             System.out.println("Número Ímpar");
21         }else{
22             System.out.println("Número Par");
23         }
24         entrada.close();
25     }
26 }
27
28
29 }
```

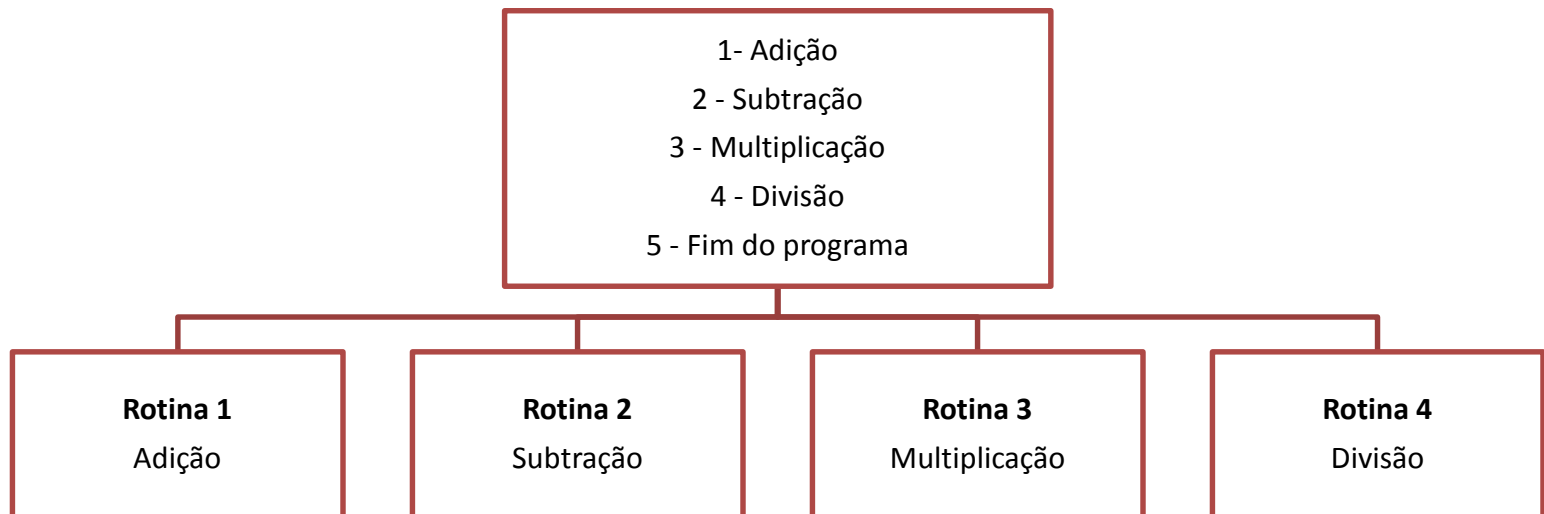
**Main**

**Procedimento Verificar**

**Instruções do Procedimento**

## Praticando

- 2) Desenvolva um programa para simulação de calculadora com um conjunto de cinco rotinas. Sendo uma principal e quatro secundárias. A rotina principal controla as quatro secundárias que pedem a leitura de dois valores, fazem a operação associada e apresentam o resultado obtido. A quinta opção não é uma rotina, apenas a opção que vai encerrar o laço de controle do menu:



# Praticando

## Programa Principal

1. Apresentar um menu de seleção com cinco opções:
  - Adição
  - Subtração
  - Multiplicação
  - Divisão
  - Fim de programa
2. Ao selecionar uma opção, a rotina correspondente deve ser executada.
3. Ao escolher o valor 5, o programa deve ser encerrado.

## Exemplo Rotina 1 – Adição

1. Ler dois valores, no caso variáveis A e B.
2. Efetuar a soma das variáveis A e B, colocando o resultado na variável R.
3. Apresentar o valor da variável R.
4. Retornar ao programa principal.



# Praticando

```

1 import java.util.Scanner;
2
3 public class Aula08_2 {
4     static Scanner entrada = new Scanner(System.in);
5
6     public static void main(String[] args) {
7
8         int opcao=0;
9
10        while (opcao!=5){
11            System.out.println("[1] - Adição");
12            System.out.println("[2] - Subtração");
13            System.out.println("[3] - Multiplicação");
14            System.out.println("[4] - Divisão");
15            System.out.println("[5] - Sair");
16
17            System.out.print("Escolha uma opção:");
18            opcao=entrada.nextInt();
19
20            switch (opcao){
21                case 1: adicao();
22                break;
23                case 2: subtracao();
24                break;
25                case 3: multiplicacao();
26                break;
27                case 4: divisao();
28                break;
29                case 5: System.exit(0);
30                break;
31                default:
32                    System.out.println("Opção inválida - Tente Novamente");
33            }
34        }
35    }
36 }
37

```

Main (principal)

Chamada dos  
Procedimentos

# Praticando

```
42 //Procedimento ADIÇÃO
43 public static void adicao(){
44     int a1=0,b1=0,r1=0;
45
46     System.out.println("Rotina Adição");
47     System.out.print("Digite o 1º valor:");
48     a1=entrada.nextInt();
49     System.out.print("Digite o 2º valor:");
50     b1=entrada.nextInt();
51     r1=a1+b1;
52     System.out.println("O resultado da operação:"+r1);
53 }
```

## Procedimento Adição

```
54
55 //Procedimento SUBTRAÇÃO
56 public static void subtracao(){
57     int a2=0,b2=0,r2=0;
58
59     System.out.println("Rotina Subtração");
60     System.out.print("Digite o 1º valor:");
61     a2=entrada.nextInt();
62     System.out.print("Digite o 2º valor:");
63     b2=entrada.nextInt();
64     r2=a2-b2;
65     System.out.println("O resultado da operação:"+r2);
66 }
```

## Procedimento Subtração

# I Escopo de Variável

**Globais:** São variáveis declaradas fora dos procedimentos e funções.

**Locais:** São as variáveis declaradas dentro do procedimento (sub-rotina), estas variáveis são utilizadas somente pelas instruções do próprio procedimento, ou procedimentos ou funções chamados a partir desta sub-rotina, não podendo ser utilizadas por outras partes do programa.

# Escopo de Variável

```

1 import java.util.Scanner;
2
3 public class Aula08_2 {
4
5     static Scanner entrada = new Scanner(System.in);
6     static int opcao=0;
7
8     public static void main(String[] args) {
9
10         while (opcao!=5){
11             System.out.println("[1] - Adição");
12             System.out.println("[2] - Subtração");
13             System.out.println("[3] - Multiplicação");
14             System.out.println("[4] - Divisão");
15             System.out.println("[5] - Sair");
16
17             System.out.print("Escolha uma opcao=");
18             opcao=entrada.nextInt();
19
20             switch (opcao){
21                 case 1: adicao();
22                 break;
23                 case 2: subtracao();
24                 break;
25                 case 3: multiplicacao();
26                 break;

```

→ Variáveis de escopo Global

```

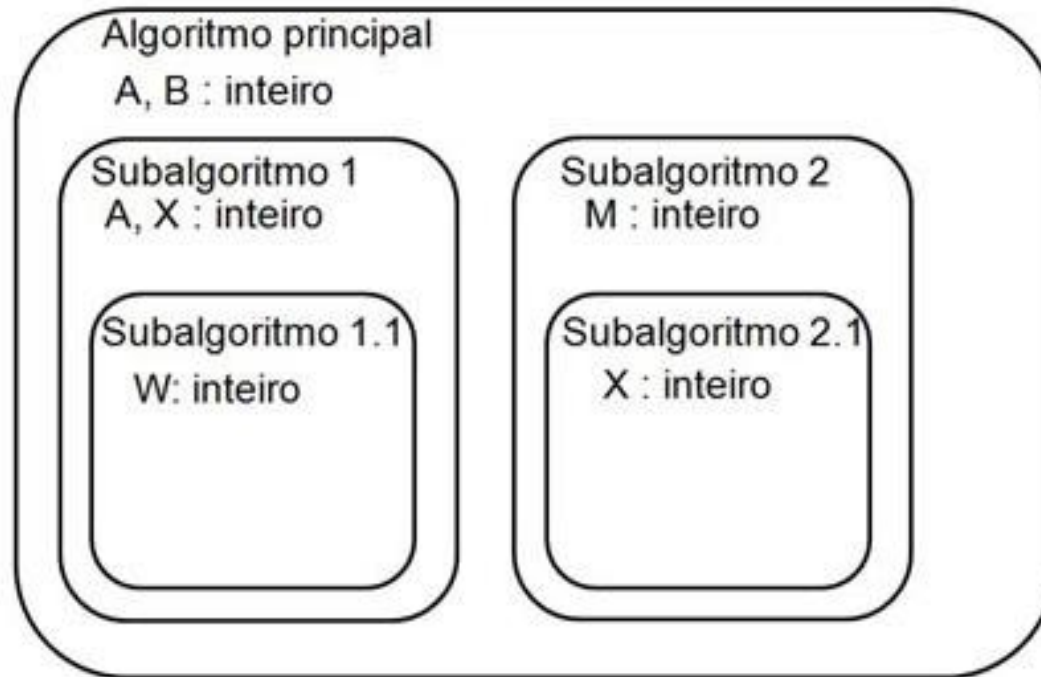
42 //Procedimento ADIÇÃO
43 public static void adicao(){
44     int a1=0,b1=0,r1=0;
45
46     System.out.println("Rotina Adição");
47     System.out.print("Digite o 1º valor:");
48     a1=entrada.nextInt();
49     System.out.print("Digite o 2º valor:");
50     b1=entrada.nextInt();
51     r1=a1+b1;
52     System.out.println("O resultado da operação:"+r1);
53 }
54

```

→ Variáveis de escopo Local

# Escopo de Variável

Veja, na Figura 1, como podemos ilustrar a hierarquia da visibilidade das variáveis.



**Figura 1** - Hierarquia no escopo de variáveis. Fonte: Adaptado de Manzano (2005, p 186).

## Escopo de Variável

**Uma variável local pode ter o mesmo nome de uma variável global. Porém, uma vez declaradas em contextos diferentes, elas são distintas.**

Além de melhorar o desempenho do algoritmo, essa divisão entre **variáveis locais e globais** serve para definir os **parâmetros** de uma sub-rotina. Ou seja, elas estabelecem a comunicação entre a sub-rotina e o algoritmo principal, que o chamou.

As **variáveis globais** do algoritmo servem como dados de entrada para a sub-rotina e as **variáveis locais** da sub-rotina armazenam os dados recebidos para, com eles, efetuar os cálculos necessários.

# Passagem de Parâmetros

Sabemos que deve haver uma **comunicação** entre a **sub-rotina** e o **algoritmo que o chama**, através das variáveis globais do algoritmo e as variáveis locais do sub-rotina. A essa comunicação, damos o nome de **passagem de parâmetros**.

**Parâmetro: Espécie de variável utilizada para que seja possível passar um valor para um determinado procedimento. O procedimento utilizará este valor na sequência de suas instruções.**

Os **tipos** de parâmetros que utilizamos nesse processo são dois, veja-os a seguir.

- **Parâmetros formais** – são aquelas variáveis locais que declaramos entre parêntesis, nos cabeçalhos dos sub-rotinas. São utilizados para realizar os cálculos dentro da sub-rotina.
- **Parâmetros reais** – são os valores que substituem os parâmetros formais no momento da chamada de uma sub-rotina.

# Declaração de um procedimento com passagem de parâmetros

## Criando o procedimento com parâmetro:

```
public static void Nome do Procedimento(tipos de parâmetros){  
    <declaração de variáveis>  
    <conjunto de instruções do procedimento>  
}
```

## Chamada do procedimento com parâmetro:

```
Nome do Procedimento(parâmetros)
```



# Declaração de um procedimento com passagem de parâmetros

```
13 public static void main(String[] args) {  
14  
15     Scanner entrada = new Scanner (System.in);  
16  
17     int num1 = 0;  
18  
19     int num2 = 0;  
20  
21     System.out.println("Informe o primeiro número:");  
22  
23     num1 = entrada.nextInt();  
24  
25     System.out.println("Informe o segundo número: ");  
26  
27     num2 = entrada.nextInt();  
28  
29  
30     somaNumeros(num1,num2);  
31  
32     entrada.close();  
33  
34 }
```

## Procedimento somaNumeros

```
35  
36  
37 public static void somaNumeros(int num1, int num2){  
38  
39     int soma = 0;  
40  
41     soma = num1 + num2;  
42  
43     System.out.println("Soma dos números: " + soma);  
44  
45 }  
46  
47 }
```

# Tipos de Parâmetros

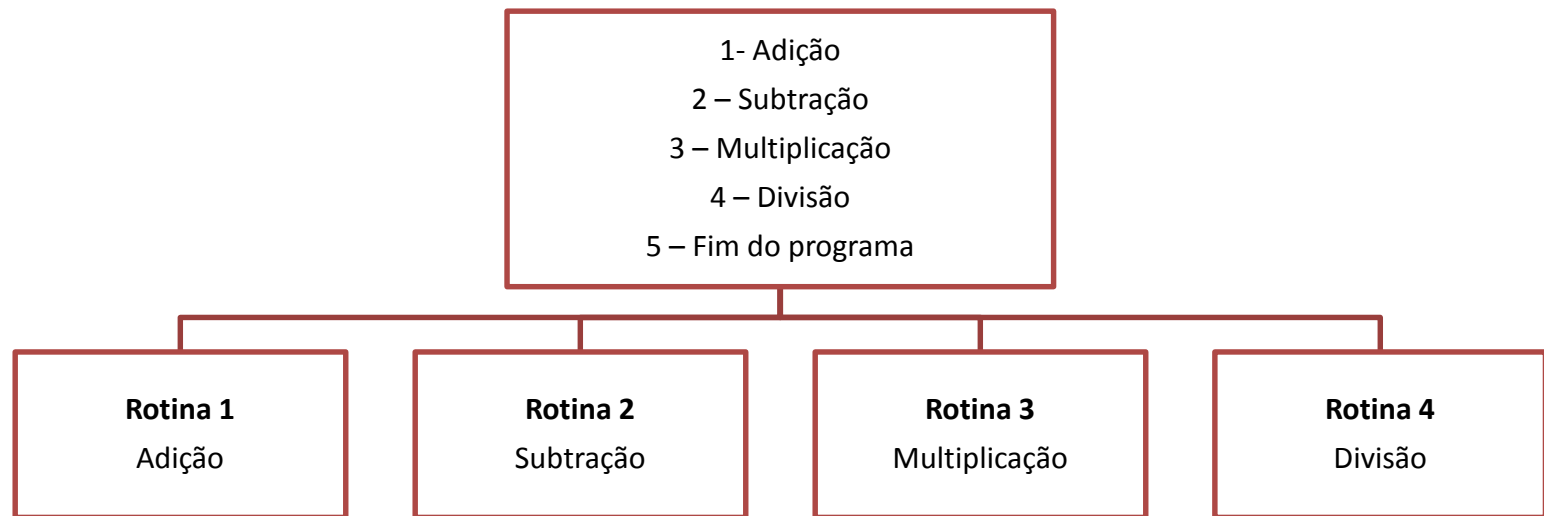
```
13 public static void main(String[] args) {  
14  
15     Scanner entrada = new Scanner (System.in);  
16  
17     int num1 = 0;  
18  
19     int num2 = 0;  
20  
21     System.out.println("Informe o primeiro número:");  
22  
23     num1 = entrada.nextInt();  
24  
25     System.out.println("Informe o segundo número: ");  
26  
27     num2 = entrada.nextInt();  
28  
29     // Chamada do procedimento (Método)  
30     somaNumeros(num1,num2);  
31  
32     entrada.close();  
33  
34 }
```

**Parâmetros REAIS**

**Parâmetros FORMAIS**

```
35  
36 // Método chamado somaNumeros  
37 public static void somaNumeros(int num1, int num2){  
38  
39     int soma = 0;  
40  
41     soma = num1 + num2;  
42  
43     System.out.println("Soma dos números: " + soma);  
44 }  
45  
46 }  
47
```

3) Faça as correções no exercício da calculadora e utilize parâmetros:



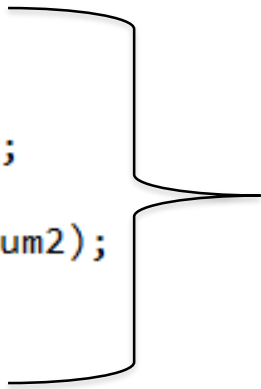
## Praticando

```
1 import java.util.Scanner;
2
3 public class Aula08_4{
4
5     static Scanner entrada = new Scanner(System.in);
6
7     public static void main(String[] args) {
8         int opcao=0;
9         double num1=0, num2=0;
10
11         while (opcao!=5){
12             System.out.println("[1] - Adição");
13             System.out.println("[2] - Subtração");
14             System.out.println("[3] - Multiplicação");
15             System.out.println("[4] - Divisão");
16             System.out.println("[5] - Sair");
17
18             System.out.print("Escolha uma opção:");
19             opcao=entrada.nextInt();
20
21             if(opcao>0 && opcao<5){
22                 System.out.print("Digite o 1º valor:");
23                 num1=entrada.nextDouble();
24                 System.out.print("Digite o 2º valor:");
25                 num2=entrada.nextDouble();
26             }
```

Main (principal)

## Praticando

```
27     switch (opcao){
28     case 1: adicao(num1,num2);
29     break;
30     case 2: subtracao(num1,num2);
31     break;
32     case 3: multiplicacao(num1,num2);
33     break;
34     case 4: divisao(num1,num2);
35     break;
36     case 5: System.exit(0);
37     break;
38     default:
39         System.out.println("Opção inválida - Tente Novamente");
40     }
41 }
42
43
44 }
```



Chamadas dos  
Procedimentos com  
parâmetro

## Praticando

```
public static void adicao(double a1, double b1){  
    double r1=0;  
  
    System.out.println("Rotina Adição");  
    r1=a1+b1;  
    System.out.println("O resultado da operação:"+r1);  
}
```

## I Praticando

- 4) Faça um programa que receba duas notas e o nome da disciplina, em um primeiro procedimento passe as duas notas como parâmetro, faça o cálculo da média, imprima e chame um segundo procedimento que receba a média e faça a verificação de aprovado para médias maiores iguais a 6 ou reprovado para menores que 6.

# Praticando

Nota.java ✕

```

1
2 import java.util.Scanner;
3  /**
4   * Métodos em Java - Procedimento
5   */
6 public class Nota {
7
8     public static void main(String[] args) {
9
10         Scanner entrada = new Scanner(System.in);
11
12         int Nota1, Nota2=0;
13         String Discp;
14
15
16         System.out.println("Digite o nome da disciplina:");
17         Discp = entrada.next();
18         System.out.println("Digite a primeira nota:");
19         Nota1 = entrada.nextInt();
20         System.out.println("Digite a segunda nota:");
21         Nota2 = entrada.nextInt();
22
23         //Chamada do procedimento
24         CalculoMedia(Nota1,Nota2);
25
26         entrada.close();
27
28     }

```

```

30     //Procedimento Cálculo Média
31     public static void CalculoMedia(int N1,int N2){
32         int M=0;
33
34         M=(N1 + N2)/2;
35
36         System.out.println("A média é:" + M);
37         Verificar(M);
38     }
39
40     //Procedimento Verificar Situação
41     public static void Verificar(int Media){
42         if(Media>=6){
43             System.out.println("Aprovado");
44         }else{
45             System.out.println("Reprovado");
46         }
47     }
48
49
50
51 }
52

```



As funções, em algoritmos, seguem o mesmo princípio das funções matemáticas que aprendemos na escola. Quando dizemos que  $f(x)=x+1$ , temos duas variáveis ( $x$  e  $f(x)$ ) e, com base no valor de uma delas, encontramos o valor da outra.

**Uma função é um sub-rotina que é chamado dentro do algoritmo através da citação de seu nome (identificador) e deve retornar um único valor.**

**Como a função retorna um valor é necessário associar um tipo de dado correspondente ao valor que é retornado.**

! A sintaxe da criação de uma função é a seguinte:

### Função

```
public static tipo da variável de retorno Nome da função(Parâmetro){  
    <declaração de variáveis>  
    <conjunto de instruções da função>  
    return variável;  
}
```

### Chamada da função (Retorno):

**Variável** = Nome da Função (**parâmetros**)

# Exemplo de Função - Java

```
1 import java.util.Scanner;
2
3 public class Soma {
4
5     /**
6      * Métodos em Java - Função
7      */
8
9     public static void main(String[] args) {
10
11         Scanner entrada = new Scanner(System.in);
12
13         int a,b,soma=0;
14
15         System.out.println("Digite o 1º número:");
16         a = entrada.nextInt();
17         System.out.println("Digite o 2º número:");
18         b = entrada.nextInt();
19         soma=fsoma(a,b);
20
21         System.out.println("A soma dos números:"+ soma);
22
23         entrada.close();
24     }
25
```

```
26 //Função Soma
27 public static int fsoma(int a1,int b1){
28     int resultado=0;
29
30     resultado=a1+b1;
31     return resultado;
32 }
33
34
```

Função

Chamada da Função

## Dicas:

- Os **parâmetros da função** são separados por vírgula (como na declaração de variáveis), se forem do mesmo tipo. Se forem de tipos diferentes, devemos separá-los por ponto-e-vírgula (;).
- O **tipo de retorno da função**, bem como todas as suas variáveis locais, pode ser qualquer um entre os tipos de dados básicos que conhecemos: inteiro, real, lógico, literal ou caractere.
- O **comando *return*** é utilizado para informar o valor de retorno da função ao algoritmo, no momento em que a função é chamada.
- A **função é chamada** sempre em uma expressão, em que o valor armazenado na variável de retorno da função é atribuído a uma variável do algoritmo.
- Sempre que chamar a função é imprescindível que os parâmetros passados devem, obrigatoriamente, estar entre parêntesis, **na mesma ordem**, ser do mesmo tipo e em igual quantidade aos parâmetros da função (variáveis locais *a1* e *b1*).

## **Referências Bibliográficas**

**Manzano, Oliveira** – Algoritmos – Ed. Érica – 3º Edição

**Forbellone, Frederico** – Lógica de Programação, Ed. Person, 2008

**Puga, Rissetti** – Lógica de programação e estrutura de dados com aplicações em Java - Ed. Person