
Modularização

Procedimentos e Funções

Procedimentos e Funções

- **Objetivo:**

Procedimentos, funções e parâmetros: recursos utilizados para tornar os algoritmos mais eficientes e possibilitar a reutilização de código;

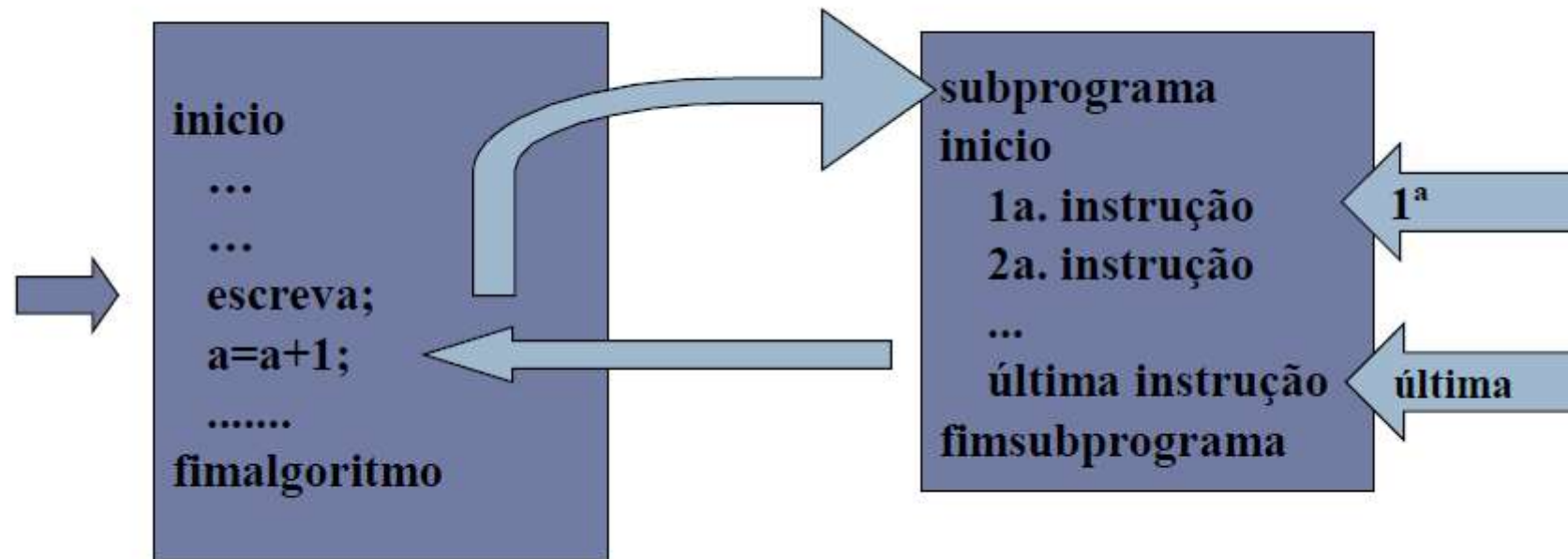
Uso de algumas rotinas em vários programas, inclusive com objetivos diferentes;

Escopo de variáveis;

Parâmetros;

Passagem de parâmetros.

Códigos para Códigos



Códigos para Códigos - Função (Function)

- A estrutura de uma função é muito parecida com um procedimento.
- Pode-se imaginar que uma função é um procedimento com características especiais **quanto ao retorno de valores.**
- A **função** é um subprograma que pode agir sobre os dados e **retornar um (OU VÁRIOS)** valor para o programa principal.

Códigos para Códigos - Função (Function)

- Uma característica que distingue uma função de um procedimento é que a função pode ser impressa, atribuída ou participar de cálculos como se fosse uma variável qualquer.
- Isso por que ela tem um valor retorno.

Códigos para Códigos - Função (Function)

Sintaxe:

funcao <identificador> ([var]<parâmetros>) <tipo de retorno>

var <declare as variáveis>

inicio

<comandos>

retorne <variável de retorno>

fimfuncao

Códigos para Códigos - Função (Function)

Algoritmo "Soma"

var

a, b, result: inteiro

funcao ImprimeSoma(x:inteiro; y:inteiro):inteiro

Var

soma: inteiro

inicio

soma <- x + y

retorne soma

fimfuncao

{Programa Principal}

inicio

Escreva("Entre com o primeiro valor: ")

Leia(a)

Escreva("Entre com o segundo valor: ")

Leia(b)

result <- **ImprimeSoma(a,b)**

Escreva("A soma dos valores é igual a: ", result)

fimalgoritmo

Códigos para Códigos - Função (Function)

Quando declaramos as variáveis, nós podemos fazê-las;

- **Local** - declaradas dentro de um procedimento ou função - só têm validade dentro do escopo ao qual foram declaradas.
- **Global** - são as variáveis do programa principal. São acessíveis a todos os subprogramas e ao programa principal.

Obs: Parâmetros são variáveis locais.

Códigos para Códigos - Função (Function)

Algoritmo "Soma"

var

a, b, result: inteiro

funcao ImprimeSoma(x:inteiro; y:inteiro):inteiro

Var

soma: inteiro

inicio

soma <- x + y

retorne soma

fimfuncao

{Programa Principal}

inicio

Escreva("Entre com o primeiro valor: ")

Leia(a)

Escreva("Entre com o segundo valor: ")

Leia(b)

result <- ImprimeSoma(a,b)

Escreva("A soma dos valores é igual a: ", result)

fimalgoritmo

Variável Global (funciona em
TODO código principal)



Variável Local (funciona dentro
da estrutura, no caso a função)

Códigos para Códigos - Procedimentos (procedure)

É um subprograma que **não retorna nenhum valor**

Deve ser definido após a declaração das variáveis.

É ativado ao ser chamado no programa principal.

Pode ou não ter parâmetros.

Exemplo – Cálculo de Imposto

```
1  import java.util.Scanner;
2  import java.io.*;
3  import java.text.DecimalFormat;
4  public class Aula5F_1 {
5      public static void main(String[] args) {
6          double salario, imposto;
7          Scanner ler = new
8              Scanner(System.in);
9          DecimalFormat deci = new DecimalFormat("0.00");
10
11          System.out.println("Digite seu salario: ");
12          salario = ler.nextDouble();
13          imposto=CalculaImposto(salario);
14          System.out.println("Você pagará em impostos: R$"+ deci.format(imposto));
15      }
16      public static double CalculaImposto(Double x){
17          double desconto;
18          desconto=x*0.14;
19          return desconto;
20      }
21  }
22  }
23  }
```

Vamos praticar

Fazer um programa para obter três notas e calcular e exibir a sua média, usando uma função para o cálculo da média.

Elabore um procedimento e uma função que receba um número como entrada e retorne verdade caso este número for par, e falso caso contrário.

Escrever um algoritmo que leia uma quantidade desconhecida de números e conte quantos deles estão nos seguintes intervalos: $[0,25]$, $[26,50]$, $[51,75]$ e $[76,100]$. A entrada de dados deve terminar quando for lido um número negativo.

Vetores

Diferença entre índice e elemento

- Elemento: conteúdo
- Índice: posição do elemento no vetor

1	2	3	4	5	← Índice
10	23	90	17	98	← Elemento

- Ou seja:
 - o elemento de índice 1 é o 10;
 - o elemento de índice 3 é o 90, e assim por diante

Acessando e atribuindo valores a um vetor

- Acessando um elemento de um vetor:

$x \leftarrow \text{vet}[2]$

$y \leftarrow \text{vet}[4]$

- Atribuindo valores a um vetor:

$\text{vet}[2] \leftarrow 90$

$\text{vet}[3] \leftarrow x$

Vetores

- Ler um vetor de 5 posições, e em seguida, imprimir esse vetor:

ALGORITMO

```
DECLARE vetor[5], i NUMÉRICO
PARA i ← 1 ATÉ 5 FAÇA
INÍCIO
    ESCREVA("Digite o elemento ", i)
    LEIA(vetor[i])
```

FIM

```
PARA i ← 1 ATÉ 5 FAÇA
    ESCREVA(vetor[i])
```

FIM_ALGORITMO

Vetores

- Ler um vetor de 5 posições, e em seguida, imprimir esse vetor em ordem inversa:

ALGORITMO

DECLARE vetor[5], i NUMÉRICO

PARA i \leftarrow 1 ATÉ 5 FAÇA

INÍCIO

 ESCREVA("Digite o elemento ", i)

 LEIA(vetor[i])

FIM

PARA i \leftarrow 5 ATÉ 1 PASSO -1 FAÇA

 ESCREVA(vetor[i])

FIM_ALGORITMO

Vetores em Java

- Declaração

- `int c[];`

- Vetor em Java é um objeto, então deve ser instanciado

- `c = new int[10];`

- Declarando e criando

- `int c[] = new int [10];`

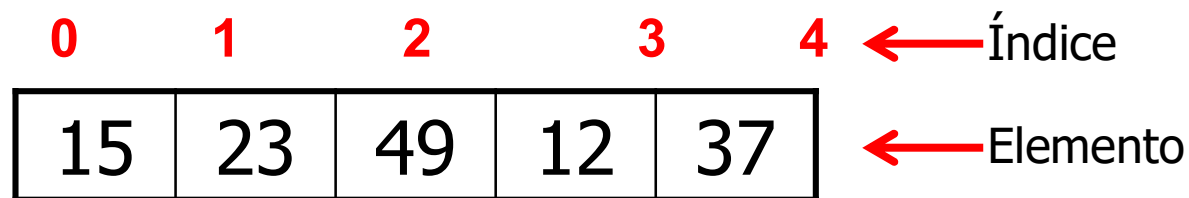
Vetores em Java

- Outros exemplos:

- `String` nomes[] = `new String` [100] ;
- `double` notas[] = `new double` [150] ;

- Iniciando vetores com valores

- `int` dados[] = {15,23,49,12,37};



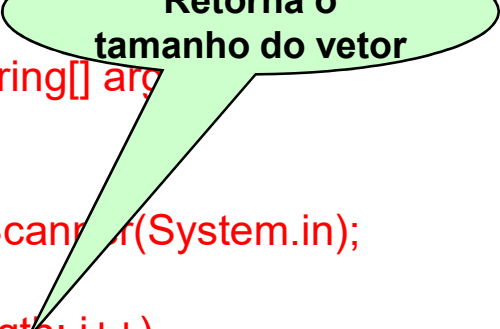
- `String` meses[] = { "Janeiro", "Fevereiro",
"Dezembro" }; 0 1 2

Janeiro	Fevereiro	Dezembro
---------	-----------	----------

Vetores em Java

Lendo e imprimindo um vetor de tamanho 5:

```
public class Vetor {  
    public static void main(String[] args)  
    {  
        int vet[] = new int[5];  
  
        Scanner input = new Scanner(System.in);  
  
        for (int i = 0; i < vet.length; i++)  
        {  
            System.out.format("Digite o elemento %d do vetor: ", i);  
            vet[i] = input.nextInt();  
        }  
  
        System.out.println("Imprimindo o vetor...");  
        for (int i = 0; i < vet.length; i++)  
        {  
            System.out.println(vet[i]);  
        }  
    }  
}
```



Retorna o tamanho do vetor

Vetores

- Ler um vetor de 5 posições, e em seguida, a soma de seus elementos:

```
public class Vetor {  
    public static void main(String[] args) {  
        int vet[] = new int[5];  
  
        int soma = 0;  
  
        Scanner input = new Scanner(System.in);  
  
        for (int i = 0; i < vet.length; i++)  
        {  
            System.out.format("Digite o elemento %d do vetor: ", i);  
            vet[i] = input.nextInt();  
            soma += vet[i];  
        }  
  
        System.out.format("\nSoma dos elementos do vetor: %d\n", soma);  
    }  
}
```

Exemplo

Ler um vetor de inteiros de 5 posições e imprimir este vetor em ordem inversa

```
public class Vetor {  
    public static void main(String[] args) {  
        int vet[] = new int[5];  
  
        Scanner input = new Scanner(System.in);  
  
        for (int i = 0; i < vet.length; i++)  
        {  
            System.out.format("Digite o elemento %d do vetor: ", i);  
            vet[i] = input.nextInt();  
        }  
  
        System.out.println("Imprimindo o vetor em ordem inversa...");  
        for (int i = vet.length-1; i >= 0; i--)  
            System.out.format("%d \t", vet[i]);  
    }  
}
```