LÓGICA DE PROGRAMAÇÃO

“Procedimentos são uma classe de algoritmos que efetuam um ou mais cálculos determinados. Ao contrário das funções, eles não produzem resultado de retorno, possuindo sua aplicação muito mais focada na modularização do algoritmo principal.

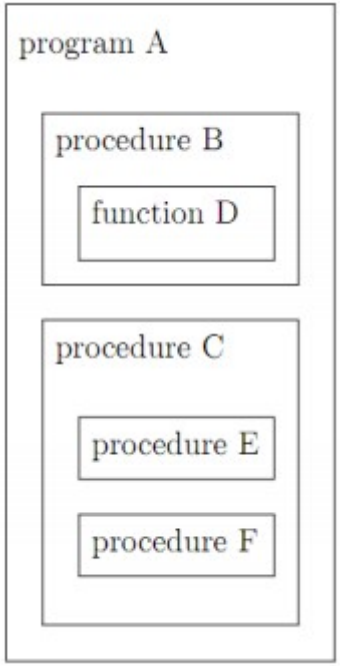
PROCEDIMENTOS

- Assim como funções buscam promover o reaproveitamento de código;
- Se diferem levemente das funções por conta de não serem obrigados a retornarem um valor para o código que o invocou;
- Se bem modelados os procedimentos, assim como as funções, é possível tornar o código muito mais limpo e simples de entender;
- As regras de posicionamento dentro do algoritmo principal continuam sendo as mesas que utilizadas nas funções;

PROCEDIMENTOS

- Assim como já foi visto em funções, os parâmetros são um novo conceito que se mantém dentro da mesma ideia;
- O que se mantém são as variáveis de escopo, onde agora existem as variáveis globais, que são declaradas junto com os procedimentos no início do algoritmo, e as variáveis locais, que são declaradas apenas no escopo da função e/ou procedimentos desenvolvidos;

PROCEDIMENTOS



FUNÇÕES EM PSEUDOCÓDIGO

```
Algoritmo "<nome do algoritmo>"
var
   <declaração de variáveis globais>
<definição da função>
inicio
   < lista de comandos>
fimalgoritmo
```

```
procedimento <identificador> ([var]<parâmetros>)

var

<declaração de variáveis locais>

inicio

<lista de comandos>
fimprocedimento
```

PRODIMENTOS EM PSEUDOCÓDIGO

```
1.    ALGORITMO "Procedimento"
2.
3.    var
4.    A,B,C,D,CONT,AUX:Inteiro
5.
6.    Procedimento TROCA(var x, y: inteiro)
7.    var
8.    Aux : inteiro
9.    INICIO
10.   Aux <- x
11.   x <- y
12.   y <- Aux
13.   FIMProcedimento
14.
15.   INICIO
16.   LEIA (A,B,C,D)
17.   Enquanto NAO((A<=B) e (B<=C) e (C<=D)) faca
18.      se (D<C) entao
19.       TROCA(D,C)
20.      FIMSE
21.      SE C<B ENTAO
22.         TROCA(C,B)
23.      FIMSE
24.      SE B<A ENTAO
25.         TROCA(A,B)
26.      FIMSE
27.   FIMENQUANTO
28.   ESCREVA (A," ",B," ",C," ",D)
29.   FIMALGORITMO
```

PROCEDIMENTOS EM JAVA

- Procedimentos em Java são bem similares as funções em pseudocódigo;
- Nesse caso o que muda são as posições dos elementos constituintes;
- A ordem de declaração no algoritmo principal nesse momento, será mantida do mesmo jeito que em pseudocódigo;
- Isso se deve para manter o conteúdo programático alinhado com a nova proposta abordada na unidade curricular;

PROCEDIMENTOS EM JAVA

```java
import javax.swing.JOptionPane;

public class Procedimentos {

    static int[] numeros;

    static int numeroDigitado;

    static int i;

    static void iniciarVetor(int qtdeDeElementos) {
        numeros = new int[qtdeDeElementos];
    }

    static void adicionar(int numeroConvertido, int indice) {
        numeros[indice] = numeroConvertido;
    }

    public static void main(String[] args) {

        iniciarVetor(10);

        for (i = 0; i < numeros.length; i = i + 1) {

            numeroDigitado = Integer.parseInt(JOptionPane
                    .showInputDialog("Digite um número"));

            adicionar(numeroDigitado, i);

        }

    }

}
```

FUNÇÕES: PSEUDOCÓDIGO X JAVA

```
1.   ALGORITMO "Procedimento"
2.
3.   var
4.   A,B,C,D,CONT,AUX:Inteiro
5.
6.   Procedimento TROCA(var x, y: inteiro)
7.   var
8.   Aux : inteiro
9.   INICIO
10.  Aux <- x
11.  x <- y
12.  y <- Aux
13.  FIMProcedimento
14.
15.  INICIO
16.  LEIA (A,B,C,D)
17.  Enquanto NAO((A<=B) e (B<=C) e (C<=D)) faca
18.     se (D<C) entao
19.      TROCA(D,C)
20.     FIMSE
21.     SE C<B ENTAO
22.        TROCA(C,B)
23.     FIMSE
24.     SE B<A ENTAO
25.        TROCA(A,B)
26.     FIMSE
27.  FIMENQUANTO
28.  ESCREVA (A," ",B," ",C," ",D)
29.  FIMALGORITMO
```

```java
import javax.swing.JOptionPane;

public class Procedimentos {

    static int[] numeros;

    static int numeroDigitado;

    static int i;

    static void iniciarVetor(int qtdeDeElementos) {
        numeros = new int[qtdeDeElementos];
    }

    static void adicionar(int numeroConvertido, int indice) {
        numeros[indice] = numeroConvertido;
    }

    public static void main(String[] args) {

        iniciarVetor(10);

        for (i = 0; i < numeros.length; i = i + 1) {

            numeroDigitado = Integer.parseInt(JOptionPane
                    .showInputDialog("Digite um número"));

            adicionar(numeroDigitado, i);

        }

    }

}
```

MÃO NA MASSA

Caso 1

Construa um algoritmo que realize uma das quatro operações matemáticas sobre dois números decimais positivos. O usuário deve fazer uso de um menu para selecionar qual operação deseja realizar, e após informar os números, o resultado deve ser apresentado. O algoritmo deve ser modularizado em 4 procedimentos além do algoritmo principal.

Caso 2

Crie um algoritmo que permita o armazenamento de 5 nomes em um vetor.

Após a armazenagem, converta todos os nomes para maiúsculo e mostre ao final. O algoritmo deve ser modularizado em 2 procedimentos além do algoritmo principal.

MÃO NA MASSA

Caso 3

Construa um algoritmo que apresente a tabuada (até 10) de um número inteiro positivo. O algoritmo deve ser modularizado para que a tabuada seja apresentada por um procedimento além do algoritmo principal.

Caso 4

Desenvolva um algoritmo que realize a leitura de salário e nome de 50 funcionários. Os nomes e salários devem ser armazenados nos devidos vetores respeitando o tipo de cada dado. Ao final um terceiro vetor de salários reajustados deve ser apresentado ao usuário. O algoritmo deve ser modularizado para que o vetor resultante seja apresentado em um procedimento especifico para