Computação Eletrônica

Subprograma

Função

- O que este programa faz?

```
Program funcao;
        var n,x: real;
        begin
            write('Entre n: ');
            readln(n);
            x := cos(n);
            write(x);
 end.
```

Função

- Uma função sempre retorna um valor

```
Program funcao;
        var n,x: real;
        begin
            write('Entre n: ');
            readln(n);
            x := cos(n);
            write(x);
 end.
```

Ou seja, é sempre importante atribuir o resultado da função a alguma variável.

Função

```
Program funcao1;

  function dobro(x: integer):integer;
      begin
        x := 2 * x;
        dobro := x;
      end;

  var n,d: integer;
      begin
        write('Entre n: ');
        readln(n);
        d := 6;
        write(n);
      end.
}
```

Memória

n	d	x
3	6	6

Monitor / Teclado

Entre n: 3 <ENTER>

2)

3) O comp
"dobro := x
nome da fu
ponto de c

amada.
bro" é o
orne ao

4) O valor retornado é 6 posto no lugar da chamada. As variáveis da função são retiradas da memória (no caso, a variável x).

Função

```
Program funcao2;

        function dobro(x: integer):integer;
           var dois: integer;
           begin
               dois := 2;
               x := dois * x;
               dobro := x;
           end;

        var n: integer;
        begin
            write('Entre n: ');
            readln(n);
            writeln(dobro(n));
            write(n); readln;
        end.
```

Função

```
Program funcao3;

    function elevado(x: integer; n: integer):
   integer;
        var i, resultado: integer;
        begin
            resultado := 1;
            i := 1;
            while (i <= n) do
               begin
                  resultado := x * resultado;
                  i := i+1;
               end;
            elevado := resultado;
        end;

var x, n: integer;
begin
    write('Entre x e n: ');
    readln(x,n);
    writeln(elevado(x,n));
    readln;
end.
```

Função

```
Program funcao5;
   function soma(x,y,z,w:integer):integer;
      var k: integer;
      begin
          k := x + y + z + w;
          soma := k;
      end;

   var s: integer;
   begin
      s := soma(1,2,3,4);
      writeln(s);
      s := soma(4,3,2,1);
      writeln(s);
      s := soma(1,1,1,1);
      writeln(s);
      readln;
   end.
```

Função

```
Program funcao6;
   function quadrado(x:integer):integer;
      begin
          quadrado := x * x;
      end;

   var n,q: integer;
   begin
      write('Entre n: ');
      readln(n);
      q := quadrado(n);
      write('Quadrado: ',q);
   end.
```

Procedimento

```
Program procedimento1;

        procedure hello();
           begin
               writeln('Hello world');
           end;
        begin
            hello();
            hello();
        end.
```

Função: retorna um valor.
Procedimento: não retorna valor.

Procedimento

```
Program procedimento2;

        procedure dobro(x: integer);
           begin
               x := 2 * x;
           end;

        var n,d: integer;
        begin
            write('Entre n: ');
            readln(n);
            d := dobro(n);
            write(d);
        end.
```

ERRO!!
`dobro` é um procedimento.
Não retorna nenhum valor.
Não pode ser atribuído à `d`.
A variável `x` é *local* ao procedimento.
Ou seja, `x` é retirada da memória ao fim do procedimento.

Então, como recuperar o valor de `x`?

Procedimento

```
Program procedimento3;

   procedure dobro(var x: integer);
      begin
         x := 2 * x;
      end;

   var n: integer;
      begin
         write('Entre n: ');
         readln(n);
         dobro(n);
         write(n);
     end.
```

Note o parâmetro **`var`** `x: integer`

Com isto, `x` ocupa a mesma posição de memória de `n`.

Esta passagem de parâmetro é chamada de passagem **por referência**.

Procedimento

```
Program procedimento3;

  procedure dobro(var x: integer);
     begin
       x := 2 * x;
     end;

  var n: integer;
     begin
       write('Entre n: ');
       readln(n);
       dobro(3);
       write(n);
  end.
```

Memória

n x 6	

Monitor / Teclado

Entre x: 3 <ENTER>
6

Procedimento

```
Program procedimento4;

   procedure leVetor(var x: array of integer);
     var i: integer;
     begin
        i := 0;
        while (i < 3) do
           begin
              write('Entre o numero: ');
              readln(x[i]);
              i := i + 1;
           end;
      end;

var  y: array[1..3] of integer;
     i: integer;
begin
   leVetor(y);
   i := 1;
   while (i <= 3) do
       begin
          writeln('Numero ',i,':
                  ',y[i]);
          i := i+1;
       end;
end.
```

Obs. Dentro de um procedimento os vetores são indexados de 0 a (tamanho-1)!