



USP - ICMC - SSC SSC 0300 - 2o. Semestre 2013

Disciplina de Linguagem de Programação e Aplicações [Eng. Elétrica / Automação]

Prof. Dr. Fernando Santos Osório / PAE: Rafael Klaser (LRM / ICMC)

LRM - Laboratório de Robótica Móvel do ICMC / CROB-SC

Email: fosorio@icmc.usp.br ou fosorio@gmail.com

Página Pessoal: <http://www.icmc.usp.br/~fosorio/>

Material on-line:

Wiki ICMC - <http://wiki.icmc.usp.br/index.php>

Wiki SSC0300 - [http://wiki.icmc.usp.br/index.php/SSC-300-2013\(fosorio\)](http://wiki.icmc.usp.br/index.php/SSC-300-2013(fosorio))

Agenda:

- **Modularização de Programas:**
 - Funções e Procedimentos da Linguagem C: Sub-rotinas
 - Programas Modulares: Blocos {} e Sub-rotinas()
- **Criando uma Sub-Rotina:**
 - Declaração (como se faz nos includes)
 - Definição dos comandos (detalhamento da sub-rotina)
- **Passagem de Parâmetros em uma Sub-Rotina:**
 - Passagem de Parâmetros de Entrada e Saída [return]
 - Parâmetros de entrada: Por Valor (by value)
 - Parâmetros de entrada: Por Referência (by reference)

Informações Complementares a Atualizadas:

Consulte REGULARMENTE o material disponível na WIKI

[http://wiki.icmc.usp.br/index.php/SSC-300-2013\(fosorio\)](http://wiki.icmc.usp.br/index.php/SSC-300-2013(fosorio))

Programas na Linguagem C - Sub-rotinas / Módulos e Variáveis:

```
#include <stdio.h>
#include <stdlib.h>

char Nome[30];
int Idade;

le_nome ( )
{ printf("Digite seu nome: ");
  scanf("%s", Nome);
}

le_idade( )
{ printf("Digite sua idade: ");
  scanf("%d", &Idade);
}

exibe_dados ( )
{ printf("\n");
  printf("Nome: %s\n", Nome);
  printf("Idade: %d\n", Idade);
  printf("\n");
}
```

```
main( )
{
    char resp[3];

    resp[0]='s';

    while (resp[0] == 's')
    {
        le_nome();
        le_idade();
        exhibe_dados();

        printf("Continuar? (s/n) ");
        scanf ("%s", resp);
    }
}
```

Sub-Rotinas e Variáveis

- Programas podem ter variáveis gerais, as denominadas ***variáveis globais*** que podem ser usadas por qualquer sub-rotina. *Todos módulos tem acesso as variáveis globais.*
- Variáveis globais são declarada FORA dos blocos, ou seja, fora das sub-rotinas.
- Programas podem ter variáveis proprietárias de um bloco, as denominadas ***variáveis locais*** que podem ser usadas apenas dentro da sub-rotina (bloco) onde foram criadas. *Variáveis locais são acessadas somente dentro do seu bloco.*

Modularização de Programas

Programas na Linguagem C - Sub-rotinas / Módulos e Variáveis:

```
#include <stdio.h>
#include <stdlib.h>
```

Variáveis
Globais

```
char Nome[30];
int Idade;
```

le_nome ()

```
{ printf("Digite seu nome: ");
  scanf("%s", Nome);
}
```

le_idade()

```
{ printf("Digite sua idade: ");
  scanf("%d", &Idade);
}
```

exibe_dados ()

```
{ printf("\n");
  printf("Nome: %s\n", Nome);
  printf("Idade: %d\n", Idade);
  printf("\n");
}
```

main ()

```
{ char resp[3];
```

Variável
Local do
Main()

```
resp[0]='s';
```

```
while (resp[0] == 's')
```

```
{
    le_nome();
    le_idade();
    exhibe_dados();
```

```
printf("Continuar? (s/n) ");
scanf ("%s", resp);
```

```
}
```

```
}
```

Modularização de Programas

Programas na Linguagem C - Sub-rotinas / Módulos e Variáveis:

```
#include <stdio.h>
#include <stdlib.h>
char Nome[30];
le_nome ( )
{ printf("Digite seu nome: ");
  scanf("%s", Nome);
}
le_idade ( )
{
  int idade;
  printf("Digite sua idade: ");
  scanf("%d", &idade);
}
exibe_dados ( )
{ printf("\n");
  printf("Nome: %s\n", Nome);
  printf("Idade: %d\n", Idade);
  printf("\n");
}
```

Variável Global

Variável Local de Le_Idade()

```
main ( )
{
  char resp[3];
  resp[0]='s';

  while (resp[0] == 's')
  {
    le_nome();
    le_idade();
    exibe_dados();

    printf("Continuar? (s/n) ");
    scanf ("%s", resp);
  }
}
```

Variável Local do Main()

Programando uma SUB-ROTINA

Sub-Rotinas na Linguagem “C” :

* Exemplos de sub-rotinas :

Passagem de parâmetros POR VALOR

By Value

- Passagem de parâmetros deve respeitar o **tipo** de cada parâmetro !
- Passagem de parâmetros deve respeitar a **ordem** dos parâmetros !
- Passagem de parâmetros deve cuidar a **quantidade** de parâmetros !
- É passada apenas uma **cópia** dos parâmetros originais...
(*por valor = através de uma cópia*)

Programação em “C”: programa típico

By Value

```
#include <stdio.h>
#include <stdlib.h>

double calcula_media (double, double);
void main (void);

double calcula_media ( v1, v2 )
double v1, v2;
{
    double media;

    media = ( v1 + v2 ) / 2.0;
    return (media);
}

void main ( )
{
    double valor1, valor2;
    double m1, m2;

    printf (“Entre 2 números: “);
    scanf (“%lf %lf",&valor1, &valor2);
    m1 = calcula_media ( valor1, valor2 );
    m2 = calcula_media ( 10.0, 7.0 );
}
```


Programação em “C”: programa típico

By Value

```
#include <stdio.h>  
#include <stdlib.h>
```

Macros:
#include
#define

```
double calcula_media (double, double);  
void main (void);  
  
double calcula_media ( v1, v2 )  
double v1, v2;  
{  
    double media;  
    media = ( v1 + v2 ) / 2.0;  
    return (media);  
}  
  
void main ( )  
{  
    double valor1, valor2;  
    double m1, m2;  
  
    printf (“Entre 2 números: “);  
    scanf (“%lf %lf",&valor1, &valor2);  
    m1 = calcula_media ( valor1, valor2 );  
    m2 = calcula_media ( 10.0, 7.0 );  
}
```

Programação em “C”: programa típico

By Value

```
#include <stdio.h>
#include <stdlib.h>
```

```
double calcula_media (double, double);
void main (void);
```

```
double calcula_media ( v1, v2 )
double v1, v2;
{
    double media;

    media = ( v1 + v2 ) / 2.0;
    return (media);
}
```

```
void main ( )
{
    double valor1, valor2;
    double m1, m2;

    printf (“Entre 2 números: “);
    scanf (“%lf %lf",&valor1, &valor2);
    m1 = calcula_media ( valor1, valor2 );
    m2 = calcula_media ( 10.0, 7.0 );
}
```

Declaração:
“cabeçalho da rotina”

Tipo de Saída
Nome_da_Rotina
(Tipos de Entrada) ;

Programação em “C”: programa típico

By Value

```
#include <stdio.h>
#include <stdlib.h>

double calcula_media (double, double);
void main (void);

double calcula_media ( v1, v2 )
double v1, v2;
{
    double media;

    media = ( v1 + v2 ) / 2.0;
    return (media);
}

void main ( )
{
    double valor1, valor2;
    double m1, m2;

    printf (“Entre 2 números: “);
    scanf (“%lf %lf",&valor1, &valor2);
    m1 = calcula_media ( valor1, valor2 );
    m2 = calcula_media ( 10.0, 7.0 );
}
```

Definição:

“como funciona”

Tipo de Saída
Nome_da_Rotina
(Parâmetro1, ...)
Tipo Parâmetro1;
Tipo Parâmetro1; ...

Parâmetros:

- Variáveis locais
- Variáveis dinâmicas
- Cuidar: Quantidade, Tipo e Ordem!

Programação em “C”: programa típico

By Value

```
#include <stdio.h>
#include <stdlib.h>

double calcula_media (double, double);
void main (void);

double calcula_media ( v1, v2 )
double v1, v2;
{
    double media;

    media = ( v1 + v2 ) / 2.0;
    return (media);
}

void main ( )
{
    double valor1, valor2;
    double m1, m2;

    printf (“Entre 2 números: “);
    scanf (“%lf %lf”, &valor1, &valor2);
    m1 = calcula_media ( valor1, valor2 );
    m2 = calcula_media ( 10.0, 7.0 );
}
```

Definição:

“como funciona”

```
Nome_da_Rotina ( ... )
{
    Bloco da Rotina
    comando;
    comando;
    ...
}
```

Sub-Rotina:

É um “mini-programa”

Programação em “C”: programa típico

By Value

```
#include <stdio.h>
#include <stdlib.h>

double calcula_media (double, double);
void main (void);

double calcula_media ( v1, v2 )
double v1, v2;
{
    double media;

    media = ( v1 + v2 ) / 2.0;
    return (media);
}

void main ( )
{
    double valor1, valor2;
    double m1, m2;

    printf (“Entre 2 números: “);
    scanf (“%lf %lf",&valor1, &valor2);
    m1 = calcula_media ( valor1, valor2 );
    m2 = calcula_media ( 10.0, 7.0 );
}
```

Parâmetros de Entrada:

- Pertence a sub-rotina
- Variável interna
- Variável dinâmica

Neste exemplo:
Parâmetros
Por Valor (by value)

Programação em “C”: programa típico

By Value

```
#include <stdio.h>
#include <stdlib.h>

double calcula_media (double, double);
void main (void);

double calcula_media ( v1, v2 )
double v1, v2;
{
    double media;
    media = ( v1 + v2 ) / 2.0;
    return (media);
}

void main ( )
{
    double valor1, valor2;
    double m1, m2;

    printf (“Entre 2 números: “);
    scanf (“%lf %lf",&valor1, &valor2);
    m1 = calcula_media ( valor1, valor2 );
    m2 = calcula_media ( 10.0, 7.0 );
}
```

double media;



Variável Local:

- Pertence a sub-rotina
- Variável interna
- Variável dinâmica

Programação em “C”: programa típico

By Value

```
#include <stdio.h>
#include <stdlib.h>

double calcula_media (double, double);
void main (void);

double calcula_media ( v1, v2 )
double v1, v2;
{
    double media;

    media = ( v1 + v2 ) / 2.0;
    return (media);
}

void main ( )
{
    double valor1, valor2;
    double m1, m2;

    printf (“Entre 2 números: “);
    scanf (“%lf %lf",&valor1, &valor2);
    m1 = calcula_media ( valor1, valor2 );
    m2 = calcula_media ( 10.0, 7.0 );
}
```

Retorno do Resultado:

- Pertence a sub-rotina
- Variável interna
- Variável dinâmica

Programação em “C”: programa típico

By Value

```
#include <stdio.h>
#include <stdlib.h>

double calcula_media (double, double);
void main (void);

double calcula_media ( v1, v2 )
double v1, v2;
{
    double media;

    media = ( v1 + v2 ) / 2.0;
    return (media);
}
```

```
void main () ←
{
    double valor1, valor2;
    double m1, m2;

    printf (“Entre 2 números: “);
    scanf (“%lf %lf”, &valor1, &valor2);
    m1 = calcula_media ( valor1, valor2 );
    m2 = calcula_media ( 10.0, 7.0 );
}
```

Rotina Principal:

“Main()”

- Onde começa o programa
- Código principal

Programação em “C”: programa típico

By Value

```
#include <stdio.h>
#include <stdlib.h>

double calcula_media (double, double);
void main (void);

double calcula_media ( v1, v2 )
double v1, v2;
{
    double media;

    media = ( v1 + v2 ) / 2.0;
    return (media);
}

void main ( )
{
    double valor1, valor2;
    double m1, m2;

    printf (“Entre 2 números: “);
    scanf (“%lf %lf",&valor1, &valor2);
    m1 = calcula_media ( valor1, valor2 );
    m2 = calcula_media ( 10.0, 7.0 );
}
```

Variáveis Locais do “Main()”:

- Pertencem a este bloco
- Variáveis locais
- Variáveis dinâmicas

Programação em “C”: programa típico

By Value

```
#include <stdio.h>
#include <stdlib.h>

double calcula_media (double, double);
void main (void);

double calcula_media ( v1, v2 )
double v1, v2;
{
    double media;

    media = ( v1 + v2 ) / 2.0;
    return (media);
}

void main ( )
{
    double valor1, valor2;
    double m1, m2;

    printf (“Entre 2 números: “);
    scanf (“%lf %lf",&valor1, &valor2);
    m1 = calcula_media ( valor1, valor2 );
    m2 = calcula_media ( 10.0, 7.0 );
}
```

Chamada da Sub-Rotina

- Executa a sub-rotina
- Envia dados de entrada
- Recebe dados de saída



Programação em “C”: programa típico

By Value

```
1      #include <stdio.h>
2      #include <stdlib.h>
3
4      double calcula_media (double, double);
5      void main (void);
6
7      double calcula_media ( v1, v2 )
8      double v1, v2;
9      {
10         double media;
11         media = ( v1 + v2 ) / 2.0;
12         return (media);
13     }
14
15     void main ( )
16     {
17         double valor1, valor2;
18         double m1, m2;
19
20         printf (“Entre 2 números: “);
21         scanf (“%lf %lf”, &valor1, &valor2);
22         m1 = calcula_media ( valor1, valor2 );
23         m2 = calcula_media ( 10.0, 7.0 );
24     }
```

Ordem de Execução:

```
main ( ):
12, 13, 14, 15, 16, 17, 18
m1 = calcula_media ( valor1, valor2 ):
5, 6, 7, 8, 9, 10, 11
main ( ):
19
m2 = calcula_media ( 10.0, 7.0 ):
5, 6, 7, 8, 9, 10, 11
main ( ):
20 < FIM>
```

Programando uma SUB-ROTINA

Sub-Rotinas na Linguagem “C” : Procedures

* Exemplos de sub-rotinas :

Passagem de parâmetros POR VALOR

By Value

- Passagem de parâmetros deve respeitar o **tipo** de cada parâmetro !
- Passagem de parâmetros deve respeitar a **ordem** dos parâmetros !
- Passagem de parâmetros deve cuidar a **quantidade** de parâmetros !
- É passada apenas uma **cópia** dos parâmetros originais...
(*exceto quando são passados ponteiros/endereços*)

Programando uma SUB-ROTINA

Sub-Rotinas na Linguagem “C” : Procedures

By Value

* Exemplos de sub-rotinas : *passagem de parâmetros por valor*

```
void exibe_media ( v1, v2 )           /* Rotina:   exibe_media           */
int v1, v2;                          /* Entrada: v1, v2 - inteiros      */
{                                     /* Passagem de params. por valor */
    double media;                    /* v1 e v2 recebem uma cópia de n1 e n2 */

    media = ( v1 + v2 ) / 2.0;
    printf (“Média = %lf \n”, media); /* Exibe o resultado na tela      */
    v1 = v2 = 0;                     /* Zera v1 e v2... Não afeta n1, n2 */
}

main ( )
{
    int n1, n2;
    printf (“Entre 2 números inteiros: “);
    scanf (“%d %d”, &n1, &n2);
    exibe_media ( n1, n2 );          /* Chama a procedure media */
}
```

Programando uma SUB-ROTINA

Sub-Rotinas na Linguagem “C” : Procedures

By Value

* Exemplos de sub-rotinas : *passagem de parâmetros por valor*

```
void exibe_media ( v1, v2 )  
int v1, v2;  
{  
    double media;  
  
    media = ( v1 + v2 ) / 2.0;  
    printf (“Média = %lf \n”, media);  
    v1 = v2 = 0;  
}  
  
main ( )  
{  
    int n1, n2;  
    printf (“Entre 2 números inteiros: “);  
    scanf (“%d %d”, &n1, &n2);  
    exibe_media ( n1, n2 );  
}
```

/* Rotina: **exibe_media** */
/* Entrada: v1, v2 - inteiros */
/* **Passagem de params. por valor** */
/* v1 e v2 recebem uma cópia de n1 e n2 */

/* Exibe o resultado na tela */
/* **Zera v1 e v2... Não afeta n1, n2** */

/* Chama a procedure media */

Programando uma SUB-ROTINA

Sub-Rotinas na Linguagem “C” : Procedures

* Exemplos de sub-rotinas :

Passagem de parâmetros POR REFERÊNCIA

By Reference

- Passagem de parâmetros deve respeitar o **tipo** de cada parâmetro !
- Passagem de parâmetros deve respeitar a **ordem** dos parâmetros !
- Passagem de parâmetros deve cuidar a **quantidade** de parâmetros !
- É passado o **ENDEREÇO** dos parâmetros originais...
PORTANTO, temos acesso total aos dados de entrada
- Parâmetros por Referência servem para:
Entrada de Dados: Podemos receber valores
Saída de Dados: *Podemos escrever e retornar valores através deles!*

Programando uma SUB-ROTINA

Sub-Rotinas na Linguagem “C” : Procedures e Funções

By Reference

* Exemplos de sub-rotinas : *passagem de parâmetros por referência (ponteiro)*

```
void calcula_media ( v1, v2, media )    /* Rotina : calcula_media */
int v1, v2;                            /* Passagem de params. por valor */
double *media;                         /* Passagem de param. por referência */
{                                       /* Media é um ponteiro para result */
    *media = ( v1 + v2 ) / 2.0;
    printf (“Média = %lf \n”, *media); /* Exibe o resultado na tela */
    v1 = v2 = 0;                       /* Zera v1 e v2... Não afeta n1, n2 */
}

main ( )
{
    int n1, n2;
    double result;

    printf (“Entre 2 números inteiros: “); scanf (“%d %d",&n1, &n2);
    calcula_media ( n1, n2, &result);    /* Chama a procedure media */
    printf (“Valores: %d- %d \nResultado: %lf \n”, n1, n2, result);
}
```


Programando uma SUB-ROTINA

Sub-Rotinas na Linguagem “C” : Procedures e Funções

By Reference

* Exemplos de sub-rotinas : *passagem de parâmetros por referência (ponteiro)*

```
void calcula_media ( v1, v2, media ) /* Rotina : calcula_media */
int v1, v2; /* Passagem de params. por valor */
double *media; /* Passagem de param. por referência */
{ /* Media é um ponteiro para result */
    *media = ( v1 + v2 ) / 2.0;
    printf (“Média = %lf \n”, *media); /* Exibe o resultado na tela */
    v1 = v2 = 0; /* Zera v1 e v2... Não afeta n1, n2 */
}

main ( )
{
    int n1, n2;
    double result;

    printf (“Entre 2 números inteiros: “); scanf (“%d %d”, &n1, &n2);
    calcula_media ( n1, n2, &result); /* Chama a procedure media */
    printf (“Valores: %d- %d \nResultado: %lf \n”, n1, n2, result);
}
```

Programando uma SUB-ROTINA

Sub-Rotinas na Linguagem “C” : Procedures e Funções

By Reference

* Exemplos de sub-rotinas : *passagem de parâmetros por referência (ponteiro)*

```
void calcula_media ( v1, v2, media )  
int v1, v2;  
double *media;  
{  
    *media = ( v1 + v2 ) / 2.0;  
    printf (“Média = %lf \n”, *media);  
    v1 = v2 = 0;  
}  
  
main ()  
{  
    int n1, n2;  
    double result;  
  
    printf (“Entre 2 números inteiros: “);  
    calcula_media ( n1, n2, &result);  
    printf (“Valores: %d- %d \nResultado: %lf \n”, n1, n2, result);  
}
```

/* Rotina : calcula_media */
/* Passagem de params. por valor */
/* **Passagem de param. por referência** */
/* Media é um ponteiro para result */
/* Exibe o resultado na tela */
/* Zera v1 e v2... Não afeta n1, n2 */

Scanf também usa passagem por referência

scanf (“%d %d”, &n1, &n2);
/* Chama a procedure media */

Programando uma SUB-ROTINA

Sub-Rotinas na Linguagem “C” : Procedures e Funções


By Reference

* Exemplos de sub-rotinas : *passagem de parâmetros por referência (ponteiro)*

```
void calcula_media ( v1, v2, media )    /* Rotina : calcula_media */
int v1, v2;                            /* Passagem de params. por valor */
double *media;                         /* Passagem de param. por referência */
{                                       /* Media é um ponteiro para result */
    *media = ( v1 + v2 ) / 2.0;
    printf (“Média = %lf \n”, *media); /* Exibe o resultado na tela */
    v1 = v2 = 0;                      /* Zera v1 e v2... Não afeta n1, n2 */
}

main ( )
{
    int n1, n2;
    double result;

    printf (“Entre 2 números inteiros: “); scanf (“%d %d”, &n1, &n2);
    calcula_media ( n1, n2, &result);    /* Chama a procedure media */
    printf (“Valores: %d- %d \nResultado: %lf \n”, n1, n2, result);
}
```



Programando uma SUB-ROTINA

Sub-Rotinas na Linguagem “C” : Procedures e Funções


By Reference

* Exemplos de sub-rotinas : *passagem de parâmetros por referência (ponteiro)*

```
void calcula_media ( v1, v2, media )    /* Rotina : calcula_media */
int *v1, *v2;                          /* Passagem de param. por referência */
double *media;                         /* Passagem de param. por referência */
{                                       /* Media é um ponteiro para result */
    *media = ( *v1 + *v2 ) / 2.0;      /* V1 e V2 são ponteiros para N1 e N2 */
    printf (“Média = %lf \n”, *media); /* Exibe o resultado na tela */
    *v1 = *v2 = 0;                   /* Zera v1 e v2... AFETA n1, n2 */
}

main ( )
{
    int n1, n2;
    double result;

    printf (“Entre 2 números inteiros: “); scanf (“%d %d”, &n1, &n2);
    calcula_media ( &n1, &n2, &result); /* Chama a procedure media */
    printf (“Valores: %d- %d \nResultado: %lf \n”, n1, n2, result);
}
```



Programando uma SUB-ROTINA

Sub-Rotinas na Linguagem “C” : Procedures e Funções

By Reference

* Exemplos de sub-rotinas : *passagem de parâmetros por referência (ponteiro)*

```
void calcula_media ( v1, v2, media )    /* Rotina : calcula_media */
int *v1, *v2;                          /* Passagem de param. por referência */
double *media;                         /* Passagem de param. por referência */
{                                       /* Media é um ponteiro para result */
    *media = (*v1 + *v2) / 2.0;        /* V1 e V2 são ponteiros para N1 e N2 */
    printf (“Média = %lf \n”, *media); /* Exibe o resultado na tela */
    *v1 = *v2 = 0;                    /* Zera v1 e v2... AFETA n1, n2 */
}

main ( )
{
    int n1, n2;
    double result;

    printf (“Entre 2 números inteiros: “); scanf (“%d %d”, &n1, &n2);
    calcula_media ( &n1, &n2, &result); /* Chama a procedure media */
    printf (“Valores: %d- %d \nResultado: %lf \n”, n1, n2, result);
}
```

Programando uma SUB-ROTINA

Sub-Rotinas na Linguagem “C” : Procedures e Funções

By Reference

* Exemplos de sub-rotinas : *passagem de parâmetros por referência (ponteiro)*

```
void calcula_media ( v1, v2, media )    /* Rotina : calcula_media */
int *v1, *v2;                          /* Passagem de param. por referência */
double *media;                         /* Passagem de param. por referência */
{                                       /* Media é um ponteiro para result */
    *media = (*v1 + *v2) / 2.0;        /* V1 e V2 são ponteiros para N1 e N2 */
    printf (“Média = %lf \n”, *media); /* Exibe o resultado na tela */
    *v1 = *v2 = 0;                    /* Zera v1 e v2... AFETA n1, n2 */
}
```

```
main ()
```

```
{
    int n1, n2;
    double result;

    printf (“Entre 2 números inteiros: “); scanf (“%d %d”, &n1, &n2);
    calcula_media ( &5, &(n1+5), &result); /* Chama a procedure media */
    printf (“Valores: %d - %d \nResultado: %lf \n”, n1, n2, result);
}
```

Atenção: Não tem como obter o endereço de um valor constante ou de uma expressão!
Somente variáveis possuem endereço!

Passagem de Parâmetros por Referência: Ponteiros e Endereços

Revisão de Ponteiros em "C"

By Reference = Passagem do Endereço

Operadores de Endereço: Ponteiros

- Realizam operações com endereços de memória.

SINTAXE:

& → Obtém o endereço de uma variável. Exemplo:

```
int dado=51;  
int *x;  
x = &dado;
```

***** → Escreve/Lê o conteúdo do endereço especificado. Exemplo:

```
dado = *x;   ou   *x = dado;
```

	100
dado 51	101
	102
	103
	104
	105

Variável Dado End.

Ponteiros são TIPADOS: Apontam para um determinado tipo de dado

Declaração:

```
{ int *ptr_valint;  
  double *ptr_valreal; ...
```

Utilização:

```
*ptr_valint = dado;      /* escreve */  
ptr_valreal = &dado;     /* aponta */
```

Variáveis em Programas Modulares:

VARIÁVEIS GLOBAIS [aloc. estática]

São variáveis declaradas fora de qualquer bloco {} do programa, usualmente declaradas no início do código fora do main().

São acessíveis em qualquer parte do programa: **uso livre**.

VARIÁVEIS LOCAIS [aloc. dinâmica]

São variáveis declaradas dentro de um bloco {} do programa, podendo ser variáveis locais do main(), de um procedimento, de uma função, ou mesmo de um bloco de um dado comando.

São acessíveis somente dentro do bloco onde estão declaradas: **uso limitado**.

VARIÁVEIS DO TIPO PARÂMETRO (By Value) [aloc. dinâmica]

São variáveis de parâmetro de uma função ou procedimento (“cópias”).

São acessíveis só dentro da sub-rotina onde estão declaradas: **uso limitado**.

Variáveis em Programas Modulares:

VARIÁVEIS LOCAIS [aloc. dinâmica]

São variáveis declaradas dentro de um bloco {} do programa, podendo ser variáveis locais do main(), de um procedimento, de uma função, ou mesmo de um bloco de um dado comando.

São acessíveis somente dentro do bloco onde estão declaradas: **uso limitado.**

VARIÁVEIS DO TIPO PARÂMETRO (By Value) [aloc. dinâmica]

São variáveis de parâmetro de uma função ou procedimento (“cópias”).

São acessíveis só dentro da sub-rotina onde estão declaradas: **uso limitado.**

VARIÁVEIS DO TIPO PARÂMETRO (By Reference) [aloc. estática]

São variáveis de parâmetro de uma função ou procedimento (“ponteiros”).

São externas a função, com uso livre de leitura e escrita: **uso livre.**

By Value => Usa a CÓPIA

By Reference => Usa a ORIGINAL

Comparativo: Passagem de Parâmetros

Por Valor

DECLARA

```
double calcula_media (double, double);
```

DEFINE

```
double calcula_media ( v1, v2 )  
double v1, v2;  
{  
    double media;  
  
    media = ( v1 + v2 ) / 2.0;  
    v1 = 0.0 ; v2 = 999.99 ;  
    return (media);  
}
```

UTILIZA

```
m1 = calcula_media ( valor1, valor2 );
```

Por Referência

DECLARA

```
double calcula_media (double *, double * );
```

DEFINE

```
double calcula_media ( v1, v2 )  
double *v1, *v2;  
{  
    double media;  
  
    media = ( *v1 + *v2 ) / 2.0;  
    *v1 = 0.0 ; *v2 = 999.99 ;  
    return (media);  
}
```

UTILIZA

```
m1 = calcula_media ( &valor1, &valor2 );
```

Exercício

Faça uma sub-rotina que receba como parâmetros 3 variáveis inteiras e retorne estas variáveis com seus valores ordenados, ou seja, o menor valor na primeira variável, o segundo menor valor na variável do meio, e o maior valor na última variável. A rotina deve retornar o valor 1 se os três valores forem iguais e 0 se existirem valores diferentes.

Esta rotina deve ser usada da seguinte forma:

```
printf ("Digite 3 valores: ");  
scanf ("%d", &A); scanf ("%d", &B); scanf ("%d", &C);  
if (Ordena3 (&A, &B, &C) == 1)  
    printf ("Os valores são todos iguais!\n");  
else  
    printf ("Valores ordenados: %d , %d, %d \n", A, B, C );
```



INFORMAÇÕES SOBRE A DISCIPLINA

USP - Universidade de São Paulo - São Carlos, SP
ICMC - Instituto de Ciências Matemáticas e de Computação
SSC - Departamento de Sistemas de Computação

Prof. Fernando Santos OSÓRIO

Web institucional: <http://www.icmc.usp.br/>

Página pessoal: <http://www.icmc.usp.br/~fosorio/>

Página do Grupo de Pesquisa: <http://www.lrm.icmc.usp.br/>

E-mail: fosorio [at] icmc. usp. br ou fosorio [at] gmail. com

Disciplina de Linguagem de Programação e Aplicações SSC300

WIKI - [http://wiki.icmc.usp.br/index.php/SSC-300-2013\(fosorio\)](http://wiki.icmc.usp.br/index.php/SSC-300-2013(fosorio))

> Programa, Material de Aulas, Critérios de Avaliação,

> Trabalhos Práticos, Datas das Provas, Notas