Universidade de São Paulo - São Carlos,SP
Instituto de Ciências Matemáticas e de Computação

USP - ICMC - SSC
SSC 0501 - 1o. Semestre 2011

Disciplina de
Introdução à Ciência da Computação
ICC 1 - Teoria

Prof. Fernando Santos Osório
Email: fosorio [at] { icmc. usp. br , gmail. com }
Página Pessoal: http://www.icmc.usp.br/~fosorio/
Web - WIKI ICMC: http://wiki.icmc.usp.br/index.php/SSC-501

PAE: Daniel Sales (Mestr. CCMC – LRM)
Email: dsales [at] icmc.usp.br
Monitor: Danilo Alvares da Silva
E-mail: danilo [at] grad.icmc.usp.br

Aula 11s

Linguagem de Programação "C"

Agenda:

- **Modularização de Programas:**
  - **Funções e Procedimentos da Linguagem C: Sub-rotinas**
  - **Blocos {} *versus* Sub-rotinas( )**
  - **Programas Seqüenciais e Programas Modulares**
- **Programando uma Sub-Rotina:**
  - **Declaração e Definição de uma Sub-rotina (Procedure)**
  - **Passagem de Parâmetros de Entrada**
- **Programando uma Função:**
  - **Declaração e Definição de Funções (Function)**
  - **Retorno de Valores: Parâmetro de Saída (Return)**
- **Entrada e Saída de Parâmetros sem restrições...**

Modularização de Programas

Sub-Rotinas:

- Programas podem ser quebrados em pequenos módulos, denominados de sub-rotinas (procedimentos e funções)
- Sub-Rotinas são blocos de código que recebem um nome, assim como o main que é também um módulo (principal)
  main( ) { ... } /* Main e seu bloco { ... } */
- O uso de sub-rotinas permite tornar o código modular, onde estas podem ser executadas (chamadas) em diferentes partes do programa e tantas vezes quanto for necessário.

Modularização de Programas

Quebrando programas na Linguagem C - Sub-rotinas / Módulos:

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

char Nome[30];
int  Idade;

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

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

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

  system("PAUSE");
  return 0;
}
```

Modularização de Programas

Quebrando programas na Linguagem C - Sub-rotinas / Módulos:

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

char Nome[30];
int  Idade;

main( )
{

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

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

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

 system("PAUSE");
 return 0;
}
```

```
#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");
}
```


Modularização de Programas

Quebrando programas na Linguagem C - Sub-rotinas / Módulos:

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

char Nome[30];
int  Idade;

main( )
{

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

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

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

 system("PAUSE");
 return 0;
}
```

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

char Nome[30];
int  Idade;

le_nome ( )
{  ... }

le_idade( )
{  ... }

exibe_dados ( )
{  ...  }

main( )
{
    le_nome();
    le_idade();
    exibe_dados();

   system("PAUSE");
   return 0;
}
```

Modularização de Programas

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();
       exibe_dados();

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

}
```


Modularização de Programas

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>
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");
}
```

Variáveis Globais

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

   resp[0]='s';

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

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

}
```

Variável Local do Main()

Modularização de Programas

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");
}
```

Variáveis Globais

```
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()

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()

Cuidado: Idade agora é local

```
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()

Modularização de Programas

Funções e Procedimentos da Linguagem C - Sub-rotinas:

Você já usa sub-rotinas a muito tempo!

Sub-Rotinas da Biblioteca Padão do "C" (Libc). Exemplos:

printf *("Hello!\n");* => Executa uma sub-rotina de impressão na tela

scanf ("%d",valor); => Executa uma sub-rotina de entrada de dados

strcpy (Nome,"Fulano"); => Copia uma string para uma variável

num = *atoi* ("1234"); => Converte uma string em valor inteiro

preco = *atof* ("56.78"); => Converte uma string em valor float/double

x = *sqrt* (nro); => Extrai a raiz quadrada de um número

x = *sin* (angulo); => Calcula o seno dado um determinado ângulo

sorteio = *rand* ( ) % 10; => Gera um número pseudo-aleatório de 0 a 9

getchar ( ); => Espera que seja pressionada uma tecla

USP – SSC0501 - Semestre 2011-1 - ICC
Introdução à Ciência da Computação - Teoria
Prof. Fernando Osório / PAE Daniel Sales

Modularização de Programas

Sub-rotinas( ) e seus Blocos {} :

```
main ( )          => Main é um bloco que recebeu o nome especial "main"
{
   printf("1");    => Printf é uma sub-rotina executada pelo main
   getchar ( );    => Getchar é uma sub-rotina executada pelo main
   printf("2");    => Sub-rotinas podem ser usadas (chamadas) diversa vezes!
   getchar ( );
}

getchar ( )          => Getchar é um bloco que recebeu um nome
{ ... }                 Sub-Rotina: Getchar / Sem Parâmetros
strcpy (Dest, Org)   => Strcpy é um bloco que recebeu um nome
{ ... }                 Sub-Rotina: Strcpy / Com Parâmetros
```

Modularização de Programas

Funções e Procedimentos da Linguagem C - Sub-rotinas:

Exemplos:

MEU_STRCPY

```
strcpy (Destino, Origem)
char Destino[];
char Origem[];
{
   int indice;

   for (indice=0; Origem[indice] !=  '\0'; indice++)
        Destino[indice] = Origem [indice];
   Destino[indice]='\0';
}
```


Modularização de Programas

Funções e Procedimentos da Linguagem C - Sub-rotinas:

Exemplos:

MEU_STRCPY

```
strcpy (Destino, Origem)
char Destino[];
char Origem[];
{
   int indice;

   for (indice=0; Origem[indice] !=  '\0'; indice++)
        Destino[indice] = Origem [indice];
   Destino[indice]='\0';
}
```

```
strcpy (Destino, Origem)
char *Destino;
char *Origem;
{
   while (*Destino++ = *Origem++);
}
```

Modularização de Programas

Programas Seqüenciais e Programas Modulares:

```
double calcula_media (N1, N2, N3)
double N1, N2, N3;
{
  return ( ( N1 + N2 + N3 ) / 3.0 );
}

main ( )
{    double Nota1, Nota2, Nota3;  double Media;

     scanf ("%lf",&Nota1); scanf ("%lf",&Nota2); scanf ("%lf",&Nota3);
     Media = calcula_media (Nota1, Nota2, Nota3);
     prinf (" Media = %lf \n", Media);

     scanf ("%lf",&Nota1); scanf ("%lf",&Nota2); scanf ("%lf",&Nota3);
     Media = calcula_media (Nota1, Nota2, Nota3);
     prinf (" Media = %lf \n", Media);
}
```


Modularização de Programas

Programas Seqüenciais e Programas Modulares:

```
double calcula_media (N1, N2, N3)
double N1, N2, N3;
{
  return ( ( N1 + N2 + N3 ) / 3.0 );
}

faz_media ( )
{
     double Nota1, Nota2, Nota3;
     double Media;

     scanf ("%lf",&Nota1);
     scanf ("%lf",&Nota2);
     scanf ("%lf",&Nota3);
     Media = calcula_media (Nota1, Nota2, Nota3);
     prinf (" Media = %lf \n", Media);
}
```

```
main ( )
{
     int cont, total=10;

     faz_media ( );
     faz_media ( );
     faz_media ( );

     for (cont=1; cont <= total; cont ++)
          faz_media ( );
}
```

Simule o Fluxo de Execução!

Programando uma SUB-ROTINA

SUB-ROTINA ou PROCEDURE: Declaração, Definição e Parâmetros

```
[void] nome_da_rotina ( <nome_vars_parâmetros> )
<declaração_dos_parâmetros>;
{
   <variáveis_locais>;

   <comandos>;
}
```


USP – SSC0501 - Semestre 2011-1 - ICC
Introdução à Ciência da Computação - Teoria
Prof. Fernando Osório / PAE Daniel Sales

Programando uma SUB-ROTINA

SUB-ROTINA ou PROCEDURE: Declaração, Definição e Parâmetros

```
[void] nome_da_rotina ( <nome_vars_parâmetros> )
<declaração_dos_parâmetros>;
{
   <variáveis_locais>;

   <comandos>;
}
```

```
void calc_media (double, double);   /*declara a sub-rotina*/
```

```
void calc_media (v1, v2)                /* define a sub-rotina */
double v1, v2;
{
   double media;
   media = (v1 + v2 ) / 2.0;
   printf("Media=%.2lf",media);
}
```

Programando uma SUB-ROTINA

SUB-ROTINA ou PROCEDURE: Declaração, Definição e Parâmetros

```
[void] nome_da_rotina ( <nome_vars_parâmetros> )
<declaração_dos_parâmetros>;
{
   <variáveis_locais>;

   <comandos>;
}
```

Sintaxe Geral

COM ponto-e-vírgula ";"

```
void calc_media (double, double);  /*declara a sub-rotina*/
```

Declaração (opcional => .h)

SEM ponto-e-vírgula ";"

```
void calc_media (v1, v2)          /* define a sub-rotina */
double v1, v2;
{
   double media;
   media = (v1 + v2 ) / 2.0;
   printf("Media=%.2lf",media);
}
```

Definição da Sub-Rotina: programa

USP – SSC0501 - Semestre 2011-1 - ICC
Introdução à Ciência da Computação - Teoria
Prof. Fernando Osório / PAE Daniel Sales

Programando uma SUB-ROTINA

SUB-ROTINA ou PROCEDURE: Declaração, Definição e Parâmetros

```
[void] nome_da_rotina ( <nome_vars_parâmetros> )
<declaração_dos_parâmetros>;
{
   <variáveis_locais>;

   <comandos>;
}
```

Exemplo:

```
void calc_media (double, double);  /*declara a sub-rotina*/
```

```
main ( )
{
   /* chamada */
   calc_media(8.3, 4.7);
   system("pause");
}
```

executa a Sub-rotina

```
/* define a sub-rotina */
void calc_media (v1, v2)
double v1, v2;
{
   double media;
   media = (v1 + v2 ) / 2.0;
   printf("Media=%.2lf",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 )              /* 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 )              /* 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

*** 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 FUNÇÃO

FUNÇÃO ou FUNCTION: Declaração, Definição e Parâmetros

```
<tipo_retorno> nome_da_rotina ( <nome_vars_parâmetros> )
<declaração_dos_parâmetros>;
{
   <variáveis_locais>;

   <comandos>;
   return ( <resultado> );
}
```

Programando uma FUNÇÃO

FUNÇÃO ou FUNCTION: Declaração, Definição e Parâmetros

```
<tipo_retorno> nome_da_rotina ( <nome_vars_parâmetros> )
<declaração_dos_parâmetros>;
{
   <variáveis_locais>;

   <comandos>;
   return ( <resultado> );
}
```

```
double calc_media (double, double);   /*declara a função */
```

```
double calc_media (v1, v2)              /* define a função */
double v1, v2;
{
   double media;
   media = (v1 + v2 ) / 2.0;
   return (media);                      /* atenção: respeite o tipo do retorno! */
}
```


Programando uma FUNÇÃO

FUNÇÃO ou FUNCTION: Declaração, Definição e Parâmetros

```
<tipo_retorno> nome_da_rotina ( <nome_vars_parâmetros> )
<declaração_dos_parâmetros>;
{
   <variáveis_locais>;

   <comandos>;
   return ( <resultado> );
}
```

```
double calc_media (double, double);   /*declara a função */
```

```
/* define a função */
double calc_media (v1, v2)
double v1, v2;
{
   double media;
   media = (v1 + v2 ) / 2.0;
   return (media);
}
```

```
/* programa principal */
main ( )
{
   double m;
   m = calc_media (8.3, 4.7);
   printf ("%.2lf \n", m);
}
```

executa a Função

Modularização de Programas

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 por 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.

Modularização de Programas

Entrada e Saída de Parâmetros em Procedures e Functions...

- Procedures podem ter um ou mais parâmetros de entrada
  Não existe uma limitação maior no número de parâmetros de entrada;
- Passagem de parâmetros BY VALUE passa a cópia do dado, ou seja,
  se o dado passado for alterado, isto não irá afetar o dado original;
- Functions podem retornar apenas um valor.

Entrada e Saída de Parâmetros sem restrições...

Aguarde: Parâmetros BY REFERENCE
>>>> NA PRÓXIMA SEMANA !!! <<<<

Modularidade: Procedimentos e Funções

Exercícios

Ex. 1

Considere o programa ao lado e responda as perguntas que seguem...

Faça um "Teste de Mesa" (Simulação da Execução)

Enumere quais são as variáveis deste programa:
Globais
Locais
Parâmetros

```
1   double calcula_media ( v1, v2 )
2   int v1, v2;
3   {
4       double media;
5       media = ( v1 + v2 ) / 2.0;
6       v1 = v2 = 0;
7       return (media);
8   }
9   main ( )
10  {
11      int  n1, n2;
12      double m1, m2;
13      printf ("Entre 2 números inteiros: ");
14      scanf  ("%d %d",&n1, &n2);
15      m1 = calcula_media ( n1, n2 );
16      m2 = calcula_media ( 10, 7 );
17  }
```


Modularidade: Procedimentos e Funções

Exercícios

1) Enumere a seqüência de passos que foram executados pelo programa. Use a numeração ao lado do código para descrever a seqüência de comandos que foram executados.
   A execução começa por: 9, 10, 11, 12, 13 ...

2) Enumere em que pontos do programa podem ocorrer erros referentes aos parâmetros (detalhar os tipos de erro que podem ser cometidos)

3) Faça um teste de mesa e indique o valor das variáveis abaixo quando for executada cada uma das linhas de comando indicadas abaixo:
   Linha 16 – Valor de m1?
   Linha 16 – Valor de media?
   Linha 17 – Valor de m2?
   Linha 17 – Valor de v1 e v2?
   (Supor que o usuário digitou: 5 e 6)
   Linha 6 - Valor de n1 e n2?

Modularidade: Procedimentos e Funções

Exercícios Ex.2

Faça um programa com 2 sub-rotinas: le_notas, calc_media.
A sub-rotina le_notas recebe como parâmetro de entrada o número de um aluno (os alunos são numerados de 1 a 10), realiza a leitura das notas deste aluno (notas P1 e P2) e chama uma outra sub-rotina que realize o cálculo da média.
A sub-rotina calc_media recebe como parâmetro as duas notas do aluno e calcula uma média ponderada das 2 notas. Os pesos de cada prova também são passados como parâmetros. A sub-rotina devolve o valor da média ponderada calculada.
Faça um programa que tenha um laço para ler as notas dos 10 alunos e calcule as médias destes alunos. Do aluno 1 ao 5 a média usa pesos iguais para ambas provas, e do aluno 6 ao 10 a prova P2 tem peso dobrado.

Universidade de São Paulo - São Carlos,SP
Instituto de Ciências Matemáticas e de Computação

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/ssc/
Página pessoal: http://www.icmc.usp.br/~fosorio/
E-mail: fosorio [at] icmc. usp. br ou fosorio [at] gmail. com
PAE Daniel Sales – E-mail: dsales [at] icmc.usp.br
Monitor Danilo Alvares – E-mail: danilo [at] grad.icmc.usp.br

Disciplina de Introdução a Ciência da Computação
Web disciplina: Wiki ICMC - Http://wiki.icmc.usp.br
> Programa, Material de Aulas, Critérios de Avaliação,
> Trabalhos Práticos, Datas das Provas, Notas