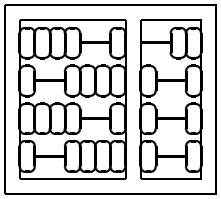

MC-102 Algoritmos e Programação
de Computadores
IC-UNICAMP

Aula 16 - Aula sobre Funções e Procedimentos - Continuação

1 Objetivos

Apresentar conceitos mais avançados de funções, como escopo de variáveis,passagem de parâmetros por valor e por referência e passagem de vetores.

2 Motivação

- Permitir a alteração de variáveis dentro de funções.
- Permitir que funções mais elaboradas sejam criadas, por exemplo, com a possibilidade de manipular vetores.

3 Procedimentos e Funções - Continuação

3.1 Escopo de Variáveis

Na linguagem C, duas ou mais variáveis não podem ser declaradas com o mesmo nome se fazem parte da mesma função, mas elas podem ter o mesmo nome se forem declaradas em funções diferentes. Nesse último caso, elas se comportarão como variáveis distintas, apesar de possuírem o mesmo nome. As variáveis da função *main()* funcionam da mesma forma.

Exemplo 1: Mostra o valor de B

```
#include <stdio.h>

void FUNC1()
{
  int B;        /* Essa á Varivel B, ós pode ser usada dentro de FUNC1() */

   B = -100;
   printf("Valor de B dentro da funcao FUNC1: %d\n", B);
}
void FUNC2()
{
```

```
    int B;     /* Essa á Varivel B, ós pode ser usada dentro de FUNC2() */

    B = -200;
    printf("Valor de B dentro da funcao FUNC2: %d\n", B);
}
void main()
{
     int B;    /* Essa á Varivel B, ós pode ser usada dentro de main() */

     B = 10;
     printf("Valor de B: %d\n", B);
     B = 20;
     FUNC1();
     printf("Valor de B: %d\n", B);
     B = 30;
     FUNC2();
     printf("Valor de B: %d\n", B);
}
```

Exemplo 2:Calcula a soma de 10 números digitados pelo usuário:

```
#include <stdio.h>

/* -----  Definicao da funcao --------  */
int soma(int x, int y)
{
  int A;
  A=x+y;
  return A;
}

/* -------- Programa Principal ---------- */
int main()
{
  int A;
  int Resultado;
  int x;

  for (x=0;x<10;x++) {
     printf("\n Entre com a: ");
     scanf("%d",&A);
     Resultado=soma(Resultado,A);
  }

  printf("\n\n   O Resultadoé %d ",Resultado);
}
```

3.2 Passagem por valor e por referência

Exemplo de função de fatorial:

```
double fatorial(int x)
{
  double fat =1;
  int i;
  for (i=x;i>1;i--)
    fat=fat*i;

  return fat; /* Retorna o valor de fat */
}
```

No caso da função fatorial, o valor de x na chamada *fatorial(x)* é passado para uma cópia x da variável n. Qualquer alteração em x não afeta o conteúdo de n no escopo da função principal. Dizemos então que o parâmetro é passado por valor. Isto nos permite, por exemplo, simplificar a função para:

```
double fatorial(int x)
{
  double fat =1;
  while (x>1){
     fat=fat*x;
     x—
  }
  return fat; /* Retorna o valor de fat */
}
```

Porém em várias situações desejamos alterar o conteúdo de uma ou mais variáveis no escopo da função principal. Neste caso, os parâmetros devem ser passados por **referência**. Isto é, a função cria uma cópia do endereço da variável correspondente na função principal em vez de uma cópia do seu conteúdo. Qualquer operação no conteúdo deste endereço é uma alteração direta do conteúdo da variável da função principal. Por exemplo, o programa anterior requer que $p \leq n$. Caso contrário podemos trocar o conteúdo dessas variáveis.

```
#include <stdio.h>

double fatorial(int x)
{
  double fat =1;

  while(x>1) {
    fat=fat*i;
    x--;
  }
  return fat; /* Retorna o valor de fat */
}
```

```
void troca (int *x, int *y) { /* x e y sao apontadores para enderecos de */
   int aux;                   /* memoria que guardam valores do tipo int */

   aux = *x;    /* o conteudo de xé  atribuido a aux. */
   *x = *y;     /* o conteudo de yé  atribuido ao conteudo de y */
   *y = aux;    /* o valor de auxé  atribuido ao conteudo de y */
}


/*---------- Programa Principal----------*/
int main()
{
  int n, p, C;
  scanf("%d %d",&n, &p);

  if (p>n)
    troca(&p,&n); /* Passa o endereco de p e n */

  if ((p>=0)&&(n>=0)){
    C= (int)(fatorial(n) / (fatorial(p) * fatorial (n-p)));
    printf("%d \n", C);
  }
  return 0;
}
```

3.3 Vetores como parâmetros

Vetores também podem ser passados como parâmetros para funções, mas eles sempre são passados por referência, portanto ao alterar os elementos do vetor dentro da função, eles também serão alterados no programa principal. A linguagem C aceita três formas para passar um vetor como parâmetro:

```
void MostraVetor(int v[],...);
void MostraVetor(int *v, ...);
void MosrtaVetor(int v[10],...);
```

O exemplo abaixo mostra que os três tipos funcionam da mesma forma:

```
#include <stdio.h>
#define TAM 10

void MostraVet(int v[],int n) {
  int j;
  for(j=0;j<n;j++)
    printf(" %d ",v[j]);
}

void MostraVet_2(int *v,int n){
```

```
  int j;
  for(j=0;j<n;j++)
    printf(" %d ",v[j]);
}

void MostraVet_3(int v[TAM]) {
  int j;
  for(j=0;j<TAM;j++)
    printf(" %d ",v[j]);
}

int main()
{
  int vetor[10];
  int i;
  for (i=0;i<10;i++)
      vetor[i]=i*i;

  MostraVet(vetor,TAM);
  printf("\n");
  MostraVet_2(vetor,TAM);
  printf("\n");
  MostraVet_3(vetor);
}
```

CUIDADO - Matrizes não podem seguir a mesma regra, somente o tamanho da primeira dimensão pode ser omitida, o tamanho das outras dimensões devem estar explícitos. Exemplos:

```
  void MostraMatriz( int m[][40],...);
  void MostraMatriz( int m[10][40],...);
  void Mostra3D( int m[][10][20],...);
  void Mostra4D( int m[][10][20][30],...);

*****  void MostraMatriz( int m[][],...); /* ERRADO - Atencao */
```

3.4 Exercícios:

1. Escreva um procedimento que recebe as 3 notas de um aluno por parâmetro e uma letra. Se a letra for A o procedimento calcula a média aritmética das notas do aluno e se for P, a sua média ponderada (pesos: 5, 3 e 2). A média calculada deve retornar por parâmetro.

   Resposta:

```
void media(float *media, float nota1, float nota2, float nota3, char opcao) {
  if (opcao=='A')
      *media=((nota1+nota2+nota3)/3);
  if (opcao=='P')
      *media=((nota1*5+nota2*3+nota3*2)/10);
}
```

2. Faça um procedimento que recebe 2 vetores A e B de tamanho 10 de inteiros, por parâmetro. Ao final do procedimento, B deve conter o cubo de cada elemento de A.

   R.

```
int cubo(int a){
  return (a*a*a);
}

void Cubo_Vetor(int A[10],int B[10]){
  int i;
  for (i=0;i<10;i++)
     B[i]=cubo(A[i]);
}
```

3. Faça duas funções que implementam as operações de soma e multiplicação de dois números complexos z ew.

   As operações são definidas por:

   Soma.....: $z + w = (a + bi) + (c + di) = (a + c) + (b + d)i$

   Multiplicação....: $z.w = (a + bi).(c + di) = (ac - bd) + (ad + bc)i$

   Portanto a função deve receber 4 parâmetros, a parte real e imaginaria de z; e a parte real e imaginaria de w. O resultado deve ser retornado nos dois primeiros parâmetros (deve substituir o valor de z).

   R.

```
void soma_complexo(int *z_real, int *z_i, int w_real, int w_i) {
             *zreal=*zreal+w_real;
             *z_i=*z_i+w_i;
}

void multiplica_complexo(int *z_real, int *z_i, int w_real, int w_i) {
             *zreal=((*zreal)*(w_real))-((*z_i)*(w_i));
             *z_i=(((*zreal)*(w_i))+((*z_i)*(w_real)));
}
```

4 Bibliografia Utilizada

Estas aulas foram baseadas nas notas de aula do **prof. Alexandre Falcão**. Disponível em: http://www.dcc.unicamp.br/ afalcao/mc102/notas-aula.pdf

Apostila do **prof. Flávio Keidi Miyazawa**.

Felipe Massia Pereira. MC102 Algoritmos e Programação. Disponível em:http://www.ic.unicamp.br/ massia/mc102/

Márcio Serolli Pinho. Laboratório de Programação I - Material de consulta de Linguagem C. Disponível em: http://www.inf.pucrs.br/ pinho/LaproI/

Fernando Lobo. Programação Imperativa, 2003/04. Disponível em :http://www.adeec.fct.ualg.pt/PI_flobo/

Material do Curso de C da UFMG - Disponível em:http://ead1.eee.ufmg.br/cursos/C/