Ponteiros e Funções: Passagem de Parâmetros por Valor e Referência

Cleo Zanella Billa

FURG - Fundação Universidade Federal de Rio Grande
C3 - Centro de Ciências Computacionais

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.
- Variáveis declaradas dentro de funções são chamadas variáveis locais.

Exemplo

Exemplo 1: Mostra o valor de B

```c
#include <stdio.h>

void FUNC1() {
  /* Essa Variavel B, so pode ser usada dentro de FUNC1() */
  int B;
  B = -100;
  printf("Valor de B dentro da funcao FUNC1: %d\n", B);
}

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

int main() {
  /* Essa Variavel B, so pode ser usada dentro de main() */
  int B;
  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);
  return 0;
}
```

Exemplo

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 Resultado: %d ",Resultado);
}
```

Variáveis Globais

- Até agora todas as variáveis que declaramos são denominadas variáveis locais, pois só podem ser vistas dentro da função onde foram declaradas.
- Mas a linguagem C permite que sejam declaradas variáveis globais, que podem ser utilizadas em todo o corpo do programa.
- **Não é recomendável a utilização de variáveis globais**, mas quando bem utilizadas podem ajudar bastante a resolver certos problemas.

Variáveis Globais - Exemplo

```c
#include <stdio.h>
int global=1;

void AlteraGlobal();
void NaoAlteraGlobal();

int main(){
  printf("global= %d \n",global); /* global=1 */
  AlteraGlobal();
  printf("global= %d \n",global); /* global=2 */
  NaoAlteraGlobal();
  printf("global= %d \n",global); /* global=3 */
  return 0;
}

void AlteraGlobal(){
  global=2;
}

void NaoAlteraGlobal(){
  int global;
  global=3;
}
```

Passagem por valor - Exemplo

Exemplo de função de fatorial:

```
#include<stdio.h>
double fatorial(int x) {
  double fat =1;
  int i;
  for (i=x;i>1;i--)
    fat=fat*i;

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

int main(){
  int n;
  printf("Digite n:");
  scanf("%d",&n);

  printf("%d!= %d",n,fatorial(n));
  return 0;
}
```

Passagem por valor

- No caso da função fatorial, o valor de `n` na chamada `fatorial(n)` é passado por valor para `x`.
- Ou seja, a variável `x` passa a ter o mesmo valor de `n`, mas são duas variáveis diferentes.
- 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:

Passagem por valor - Exemplo

```
#inculde<sdtio.h>
double fatorial(int x) {
  double fat =1;
  while (x>1){
     fat=fat*x;
     x--;
  }
  /* Retorna o valor de fat */
  return fat;
}

int main(){
  int n;
  printf("Digite n:");
  scanf("%d",&n);

  printf("%d!= %d",n,fatorial(n));
  return 0;
}
```

Passagem por Referência

- 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**, ou seja, usar ponteiros.
- 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 do próximo slide requer que $p \leq n$. Caso contrário podemos trocar o conteúdo dessas variáveis.

Passagem por Referência - Exemplo

```c
#include <stdio.h>



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

  int aux;

  /* o conteudo de x eh atribuido a aux. */
  aux = *x;
  /* o conteudo de y eh atribuido
     ao conteudo de y */
  *x = *y;
  /* o valor de aux eh atribuido
     ao conteudo de y */
  *y = aux;
}
```

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

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

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

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

Vetores como Parâmetros de Funções

- 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],...);
```

Vetores - Exemplo

```c
#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 i, vetor[10];
  for (i=0;i<10;i++)
      vetor[i]=i*i;
  MostraVet(vetor,TAM);
  printf("\n");
  MostraVet_2(vetor,TAM);
  printf("\n");
  MostraVet_3(vetor);
  return 0;
}
```

Matrizes e Funções

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

```
void MostraMatriz01( int m[][40],...);
void MostraMatriz02( int m[10][40],...);
void Mostra3D( int m[][10][20],...);
void Mostra4D( int m[][10][20][30],...);

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

Exercício 01

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 Exercício 01

```
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);
}
```

Exercício 02

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

Resposta Exercício 02

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

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

Exercício 03

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

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 imaginária de z; e a parte real e imaginária de w. O resultado deve ser retornado nos dois primeiros parâmetros (deve substituir o valor de z).

Resposta Exercício 03

```
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)));
}
```

Referências

- ASCENCIO, Ana F. G.; CAMPOS, Edilene A. V. Fundamentos da programa de computadores. Pearson Prentice Hall, 2007.
- Schildt, Herbert. C completo e total. Pearson Makron Books, 1997.
- Notas de aula do Prof. Flavio Keidi Miyazawa.
- Notas de aula do Prof. Emanuel Estrada.
- Notas de aula do Prof. Alessandro Bicho.