VOLTAR

Passagem de parâmetros

Apresentar conceitos sobre a utilização de parâmetros em programação estruturada e aplicar tais conceitos utilizando a linguagem C.

NESTE TÓPICO

Introdução

Os parâmetros servem um ponto de comunicação bidirecional entre um módulo e o programa principal ou outro módulo hierarquicamente de nível mais alto. É possível passar valores de um módulo ou módulo chamador a outro módulo e vice-versa, utilizando parâmetros que podem ser formais ou reais.

Parâmetros formais e reais

Serão considerados parâmetros formais quando forem declarados por meio de variáveis juntamente com a identificação do nome do módulo, os quais serão tratados exatamente da mesma forma que são tratadas as variáveis globais ou locais. Serão considerados parâmetros reais quando substituírem os parâmetros formais, quando da utilização do módulo por um programa principal.

A forma geral do cabeçalho de um módulo com a declaração de parâmetros formais em linguagem C é:

void / tipo-retorno nome-módulo (tipo-parametro nome-parametro, ...)

Onde, void / tipo-retorno depende do tipo de módulo (procedimento ou função, respectivamente), nome-módulo é o nome que se deve dar ao módulo e tipo-parametro nome-parametro corresponde à declaração do parâmetro. Vale lembrar que podemos declarar quantos parâmetros forem necessários.

Exemplo:

- Declaração do cabeçalho do procedimento “Exemplo”, que terá como parâmetros duas variáveis inteiras x e y.

void Exemplo (int x, int y)

Exemplo 1

O programa abaixo mostra uma função cujo objetivo é verificar e retornar o maior valor entre duas variáveis (a e b), as quais são definidas como parâmetros formais. Na chamada da função, no programa principal, as variáveis p e q são definidas como parâmetros reais.

```
1.  #include <stdio.h>
2.  #include <stdlib.h>
3.
4.  int verifica_maior (int a, int b) //a e b sao Parametros FORMAIS
5.  {
6.           if (a > b)
7.                return (a);
8.           else return (b);
9.  }
10.
11. main ()
12. {
13.          int p=7, q=10;
14.          int maior;
15.          maior = verifica_maior (p, q); //p e q sao Parametros REAIS
16.          printf ("\n Maior elemento = %d", maior);
17. }
```

Passagem de parâmetros

A passagem de parâmetro ocorre quando é feita uma substituição dos parâmetros formais pelos reais no momento da execução do módulo. Esses parâmetros são passados por variáveis de duas formas: por valor e por referência.

Passagem de parâmetro por valor

A passagem de parâmetro por valor caracteriza-se pela não-alteração do valor do parâmetro real quando o parâmetro formal é manipulado dentro do módulo. Assim sendo, o valor passado pelo parâmetro real é copiado para o parâmetro formal, que no caso assume o papel de variável local do módulo. Qualquer modificação, que ocorra na variável local do módulo, não afeta o

valor do parâmetro real correspondente, ou seja, o processamento é executado somente dentro do módulo, ficando o resultado obtido "preso" no módulo.

Exemplo 2

No programa abaixo, o procedimento possui apenas um parâmetro formal (x), o qual é passado por valor. Este parâmetro é alterado pelo valor de seu quadrado dentro do procedimento, porém tal alteração só é válida dentro do procedimento.

```
1.  #include <stdio.h>
2.  #include <stdlib.h>
3.
4.  void quadrado (int x)
5.  {
6.           x = x*x;
7.           printf ("\n Valor de x = %d", x);
8.  }
9.
10. main ()
11. {
12.          int x = 4;
13.          quadrado (x);
14.          printf ("\n Valor de x = %d", x);
15.          system ("PAUSE");
16. }
```

Na Figura abaixo, é mostrada a simulação das alocações das variáveis na memória quando o programa acima é executado. Embora o parâmetro formal tenha o mesmo nome que o parâmetro real, as variáveis ocupam posições diferentes na memória. Assim, notamos que a alteração é feita somente no parâmetro formal, sendo que o parâmetro real mantém seu valor. Dessa forma, o programa exibirá duas mensagens na tela: “Valor de x = 16” e “Valor de x = 4”.

Simulação da alocação de variáveis na memória do computador

Fonte: Elaborado pelo autor

Passagem de parâmetro por referência

A passagem de parâmetro por referência ocorre quando o parâmetro real recebe o conteúdo do parâmetro formal e, após um certo processamento dentro do módulo, o parâmetro formal reflete a alteração de seu valor junto ao parâmetro real. Qualquer modificação feita no parâmetro formal implica em alteração do parâmetro real correspondente. A alteração efetuada é devolvida para a rotina chamadora.

Exemplo 3

No programa abaixo, o procedimento possui apenas um parâmetro (n), o qual é passado por referência. Para tanto, é necessário declarar o parâmetro formal como um ponteiro (asterisco antes do nome do parâmetro), que permitirá a alteração do parâmetro formal reflita sobre o parâmetro real. Na chamada do procedimento, é necessário passar o endereço de memória de n.

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

void dobro (int *n)
{
          *n = 2*(*n);
          printf ("\n Valor de n = %d", *n);
}

main ()
{
         int n = 8;
         dobro (&n);
         printf ("\n Valor de n = %d", n);
         system ("PAUSE");
}
```

Na Figura abaixo, é mostrada a simulação das alocações das variáveis na memória quando o programa acima é executado. Embora o parâmetro formal tenha o mesmo nome que o parâmetro real, as variáveis ocupam posições diferentes na memória. Assim, notamos que a alteração do parâmetro real é feita de forma indireta por meio do parâmetro formal, que é definido como um ponteiro. Por fim, o programa exibirá duas mensagens na tela: “Valor de n = 16” e “Valor de n = 16”.

Simulação da alocação de variáveis na memória do computador

Fonte: Elaborado pelo autor

Exemplo 4

Neste exemplo, o programa contém os seguintes módulos:

1. Uma função com dois parâmetros, por valor, denominados “ini” e “fim”. É feito o cálculo e o valor da soma de todos os números pares entre “ini” e “fim” é retornado.

2. Um procedimento com dois parâmetros, sendo o primeiro “opcao” por valor e o segundo “n” por referência. Se “opcao” é igual a 1, então, é feito o cálculo do dobro do valor de “n”. Se “opcao” é igual a 2, o triplo do valor de “n” é calculado. O valor de n, alterado por referência dentro do procedimento, é mostrado no "main".

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

int soma_valores (int ini, int fim) {
       int i, soma = 0;
       for (i=ini;i<=fim;i++)
            soma = soma + i;
       return (soma);
}

void calcula_dobro_triplo (int opcao, int *n) {
       if (opcao==1)
           *n = *n * 2;
       else if (opcao==2)
           *n = *n * 3;
}

main () {
        int ini = 1, fim = 10, n=4, soma, opcao;

        soma = soma_valores (ini, fim);
        printf ("\n Valor da soma = %d", soma);

        printf ("\n Digite a opcao =");
        scanf ("%d", &opcao);
        calcula_dobro_triplo (opcao, &n);
        printf("Valor de n = %d", n);

        system ("pause");
}
```

Passagem de vetores como parâmetros

Como uma variável simples, um vetor também pode ser passado como parâmetro para uma função. O tipo de passagem de parâmetro de um vetor para uma função é sempre por referência.

Para indicarmos que um parâmetro de uma função é um vetor, basta colocarmos colchetes após o nome do parâmetro. Na especificação da função não é necessário colocar o tamanho do vetor. O exemplo a seguir define o protótipo de uma função f que tem como parâmetro um vetor v do tipo inteiro.

void f (int v []);

Quando chamamos uma função que tem como parâmetro um vetor, basta apenas passar o nome do vetor para a função. Analise o trecho de código seguinte.

```
// declaração e inicialização do vetor
int vetor [3] = {2, 4, 6};
// passagem do vetor para a função f definida anteriormente
f (vetor);
```

No exemplo anterior criamos e inicializamos um vetor de três posições. Esse vetor foi posteriormente passado como parâmetro para a função *f*, definida anteriormente. Note que é só passar o nome do vetor como argumento para a função.

Exemplo 5

A seguir apresentamos um exemplo completo de um programa em linguagem C, que utiliza passagem de vetor como parâmetro para função. Vamos criar duas funções: uma que inicializa todas as posições do vetor com valor 1 e outra que apresenta os dados do vetor na tela. Analise o trecho de código.

```
#include <stdio.h>
#include <stdlib.h>
// constante simbolica que representa o tamanho do vetor
#define TAM 10
// definição da função inicializaVetor
void inicializaVetor(int v[]) {
        int i;
        for (i = 0; i < TAM; i++) {
             v[i] = 1;
        }
}
// definição da função imprimeVetor
void imprimeVetor(int v[ ]) {
        int i;
        for (i = 0; i < TAM; i++) {
             printf ("%d ", v[i]);
        }
}
main () {
       int vetor[TAM];
       // chama a funcao inicializaVetor
       inicializaVetor(vetor);
       // chama a função imprimeVetor
       imprimeVetor(vetor);
       printf ("\n\n");
       system("PAUSE");
}
```

Exemplo 6

Neste exemplo, é criada uma função que recebe como parâmetro um vetor de valores inteiros com 5 posições e retorna a soma dos valores do vetor. A função é chamada no "main" e o valor da soma é mostrado na tela.

```c
#include <stdio.h>
#include <stdlib.h>
// constante simbólica para representar o tamanho do vetor
#define TAM 5

// definição da função que recebe como parametro um vetor de inteiros
int somaValores (int vet[]) {
     int i, soma = 0;
     for (i = 0; i < TAM; i++) {
            soma += vet[i];
     }
     return soma;
}

main () {
        int vetor[TAM];
        int i;
        // leitura dos dados para o vetor
        printf ("*** Informe os dados para o vetor ***\n");
        for (i = 0; i < TAM; i++) {
             printf ("Valor[%d]: ", i);
             scanf ("%d", &vetor[i]);
        }
        // chamada da funcao somaValores
        int s = somaValores (vetor);
        // impressao do retorno na tela
        printf ("\nSoma = %d\n\n", s);
        system ("PAUSE");
}
```

Agora que você já estudou essa aula acesse a plataforma AVA, resolva os exercícios e verifique o seu conhecimento. Caso fique alguma dúvida, leve a questão ao Fórum e divida com seus colegas e professor.

Quiz

Exercício

Passagem de parâmetros

INICIAR

Quiz

Exercício Final

Passagem de parâmetros

INICIAR

Referências

MIZRAHI, V. V. *Treinamento em linguagem* C, São Paulo: Pearson, 2008.

SCHILDT, H. C — *Completo e Total*. São Paulo: Pearson, 2006.

Avalie este tópico

ANTERIOR
Programação Estruturada (Modular)

Índice

PRÓXIMO
Ordenação de vetores

Ajuda?
(https://ava.uninove.br/seu/AVA/topico/topico.php?idCurso=)