Universidade Católica de Brasília

Algoritmo e Programação Estruturada

Aula 9

Prof. Osmam Brás de Souto

Universidade Católica de Brasília

Macros

- Macros são definidas através da diretiva *define* podendo ter parâmetros;
- Cada vez que o nome da macro é encontrado, os parâmetros usados na sua definição são substituídos pelos parâmetros encontrados no programa antes da compilação do programa;
- Não é especificado o tipo de dado do parâmetro;
- As macros diminuem o tempo de execução do programa, mas aumentam o seu código (duplicação);
- Cuidados com a sintaxe na realização de expressões.

Universidade Católica de Brasília

Macros

A diretiva *#define* pode ser usada na substituição de textos que tenham um significado explicito ao compilador.

Por exemplo:

```
#include<stdio.h>
#include<conio.c>
#define ERRO printf("\nVALOR INVALIDO.\n")
int main(void)
{
    int nro;
    printf("Informe um numero de 1 a 10: ");
    scanf("%d",&nro);
    if ((nro < 1) || (nro > 10))
        ERRO;
    else
        printf("Valor pertencente ao intervalo.");
    getch();
}   // encerra o programa
```

Universidade Católica de Brasília

Macros

```c
#include <stdio.h>
#include <stdlib.h>
#define ERRO printf("\nVALOR INVALIDO.\n")
int main(void) {
    int nro;
    printf("Informe um numero de 1 a 10: ");
    scanf("%d",&nro);
    if ((nro < 1) || (nro > 10))
    ERRO;
    else
    printf("Valor pertencente ao intervalo.");
    getch();
}
```

Universidade Católica de Brasília

Macros

- Esta diretiva também pode receber argumentos para serem utilizados no momento de sua execução no programa.

```
#define MOSTRA(p1) printf("%f\n", p1)
int main (void)      {        // abrindo o bloco de instruções
   float valor;
   printf("Informe o valor desejado: ");
   scanf("%f", &valor);
   MOSTRA(valor);
   valor = valor * 3;
   MOSTRA(valor);
   getch();    }        // encerra o bloco de instruções
```

VALOR é substituído pelo nome usado na macro

Universidade Católica de Brasília

Macros

```c
#include <stdio.h>
#include <stdlib.h>
#define MOSTRA(p1) printf("%f\n", p1)
int main(void) {
    float valor;
    printf("Informe o valor desejado: ");
    scanf("%f", &valor);
    MOSTRA(valor);
    valor = valor * 3;
    MOSTRA(valor);
    getch();
}
```

Universidade Católica de Brasília

Macros

A não utilização de parênteses nas macros pode resultar em erros na efetivação de uma expressão.

Exemplo:

```
#define SOMAR(p1,p2)  p1 + p2
int main (void)       {          // abrindo o bloco de instruções
  int valor_1, valor_2, total;
  printf("Informe os valores: ");
  scanf("%d %d",&valor_1, &valor_2);
  total = 10 * SOMAR(valor_1,valor_2);
  printf("Resultado = %d", total);
  getch();            }          // encerra o bloco de instruções
```

Resultado = 35
ao invés de
Resultado = 80

Supondo que os valores informados sejam 3 para valor_1 e 5 para valor_2 o resultado é 35 ao invés de 80 como esperado.

Universidade Católica de Brasília

Macros

```c
#include <stdio.h>
#include <stdlib.h>
#define SOMAR(p1,p2) p1 + p2
int main(void) {
    int valor_1, valor_2, total;
    printf("Informe os valores: ");
    scanf("%d %d",&valor_1, &valor_2);
    total = 10 * SOMAR(valor_1,valor_2);
    printf("Resultado = %d", total);
    getch();
}
```

Universidade Católica de Brasília

Macros

A solução neste exemplo seria incluir todo o texto envolvido por parênteses .

```
#define SOMAR(p1,p2) (p1 + p2)
```

No exemplo anterior a solução acima funcionaria e o resultado seria o esperado, porém envolvendo todo o texto entre parênteses não soluciona todos os problemas. Por exemplo:

```
#define  MULTIPLICAR(p1,p2) (p1 * p2)
...
Resultado = MULTIPLICAR (2+3,4);
```

Após a substituição do texto tem-se a expressão:

```
Resultado = (2+3*4);     // resultado 14
```

onde o valor desejado seria vinte (20).

A solução é envolver cada parâmetro por parênteses

```
#define MULTIPLICAR (p1,p2) ( (p1) * (p2) )
Resultado = ((2+3)*(4));// resultado 20
```

Universidade Católica de Brasília

Funções Recursivas

Uma função é recursiva quando dentro dela existe uma chamada a ela própria.

Um código recursivo precisa usar mais memória, o que torna a execução mais lenta. Quando uma função chama a si mesmo novos parâmetros e variáveis locais são alocados na pilha e o código da função é executado com essas novas variáveis.

Uma chamada recursiva não faz uma nova cópia da função, mas apenas os seus argumentos são novos. Quando a função recursiva retorna, as variáveis locais e os parâmetros são removidos da pilha, a execução recomeça do ponto da chamada à função dentro da função.

O objetivo de uma função *recursiva*, também denominada de *recorrência*, é fazer com que ela passe por uma seqüência de chamadas até que um certo ponto seja atingido (resolvido).

Universidade Católica de Brasília

Funções Recursivas

Toda função recursiva deve possuir um comando condicional (*if*) em algum lugar, forçando a função a retornar sem que a chamada recursiva ocorra (*condição de escape*).

Exemplo:

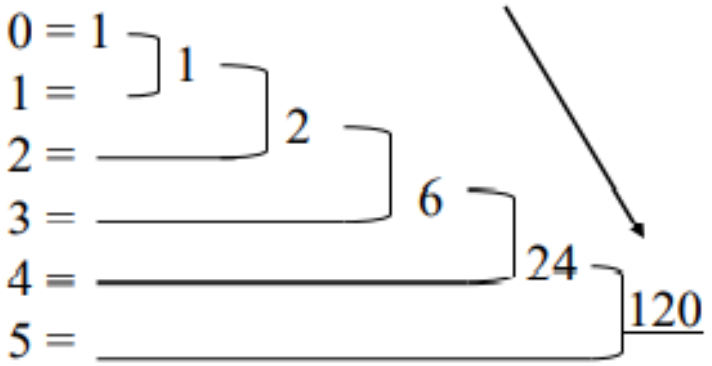
```
#include<stdio.h>
#include<conio.c>

int fatorial(int n) ;  // protótipo

int main(void)    {
  int um = 1;
  while(num)     {
    printf("\nDigite um numero: ");
    scanf("%d",&num);
    printf("Fatorial = %ld“,fatorial(num));
   }
   getch();
}
```

```
// Definição da função
long fatorial (int n)
{
  if (n == 0)  return (1) ;
  else
    return( fatorial(n-1) * n);
  }
```

Saída => 1*2*3*4*5 = ?



Universidade Católica de Brasília

Funções Recursivas

```c
#include<stdio.h>
long fatorial(int n) ; // protótipo
int main(void) {
 int num = 1;
 while(num) {
 printf("\nDigite um numero: ");
 scanf("%d",&num);
 printf("Fatorial = %ld",fatorial(num));
 }
 getch();
}
// Definição da função
long fatorial (int n)
{
 if (n == 0) return (1) ;
 else
 return( fatorial(n-1) * n);
 }
```

Referência de Criação e Apoio ao Estudo

Material para Consulta e Apoio ao Conteúdo

- FARRER, H. et all, Algoritmos Estruturados, Editora LTC, 3ª . edição, 1999. - livro
  - Capítulo 0

- MANZANO, J. e Oliveira, J., Algoritmos, Lógica para desenvolvimento de programação, Editora Ética, 1996.
  - Capítulo 1

Universidade Católica de Brasília

Obrigado(a)!