**Repetições encaixadas**

Até o momento, nossos programas foram simples, e os blocos de comandos não exigiam uma longa lista de comandos. Mas os blocos podem se tornar bastante complexos, pois a linguagem C permite que cada bloco contenha comandos com mais blocos, e assim sucessivamente. Uma forma de simplificar esses blocos de comandos e melhorar a estrutura do programa é através do uso de funções, que veremos em breve. Nessa aula, vamos ver como alguns problemas exigem a criação de blocos dentro de blocos, em particular, de repetições dentro de repetições (ou repetições encaixadas). A forma mais simples de apresentar esse recurso é através de um problema.

**Um exemplo:**

Dado um número natural n calcule os fatores de n. Por exemplo, se n=45, a saída deve ser 3x2 5x1, ou seja, o número 45 é fatorado em 3 (2 vezes) e 5 (uma vez).

**Solução:**

Quando fatoramos na mão, fazemos algo como:

45 | 3  
15 | 3  
5 | 5  
1 |

Nós na verdade testamos o 2 antes do 3, mas na fatoração só mostramos os números relevantes. O computador não sabe quais os números relevantes, assim é necessário testar todos os números possíveis. Por exemplo, podemos começar testando quantas vezes o 2 é fator de n, depois 3, depois 4, etc, até que o número seja completamente fatorado.   
  
Pensando alto, com um exemplo real:

1. se n=45, começar testando o número 2.
2. Como o 2 não é fator, testamos o proximo número, ou seja, o 3.
3. Como 3 é fator de 45, contamos quantas vezes o 3 é fator de 45.
4. Imprimimos o fator e o número de vezes, e continuamos o processo com o fator 4, e o número 5 a ser fatorado.
5. Como o 4 não é fator de 5, testamos o proximo número, ou seja, o 5.
6. Como o 5 é fator de 5, contamos quantas vezes o 3 é fator de 45, e continuamos com o número 6 e o número 1 a ser fatorado.
7. Como o número é 1, podemos encerrar o processo (ou como o fator é maior que n).

Como o problema é complexo, vamos começar tentando colocar o pensamento acima em um rascunho que se pareça com um programa, que segue abaixo:

leia n  
 fator = 2; /\* começo a testar os fatores a partir de 2 \*/  
  
 while ( n != 1 ) {  
 while (n é multiplo de fator) { /\* contar quantas vezes \*/  
 incrementa o contador;  
 atualiza n ( n = n / fator )  
 }  
 se (cont não é zero) {  
 /\* o fator testado anteriormente é realmente um fator de n \*/  
 imprima (fator x cont);  
 cont = 0;  
 }  
 fator++;  
}

Agora é fácil colocar o rascunho na forma de um programa, como abaixo:

#include <stdio.h>  
#include <stdlib.h>  
int main ()   
{  
 int n, num, fator = 2, cont = 0;  
 printf("Entre com o numero a ser fatorado: ");  
 scanf ("%d", &n);  
 while ( n != 1 ) {  
 while (n%fator == 0) { /\* contar quantas vezes \*/  
 cont++;  
 n = n / fator;  
 }  
 if (cont != 0) {  
 /\* o fator testado anteriormente é realmente um fator de n \*/  
 printf ("fator %d x %d\n", fator, cont);  
 cont = 0;  
 }  
 fator++;  
 }  
 system("pause");  
 return 0;  
}

**Exercícios recomendados:**

Dados uma seqüência de números naturais terminada por zero, determinar os primos.