**CAPÍTULO 3**

**EXERCÍCIOS DE AUTORREVISÃO**

1. SENTENÇAS

O procedimento para resolver um problema em termos de ações a serem executadas e da ordem em que essas ações devem ser executadas é chamado de ALGORITMO.

Especificar a ordem de execução das instruções pelo computador é chamado de CONTROLE DE PROGRAMA.

Todos os programas podem ser escritos nos termos de três tipos de estruturas de controle:  
- SEQUÊNCIA  
- SELEÇÃO  
- REPETIÇÃO

A estrutura de seleção IF...ELSE é usada para executar uma ação quando uma condição é verdadeira e outra ação quando a condição é falsa.

Várias instruções agrupadas com chaves ( { e } ) são chamadas de INSTRUÇÃO COMPOSTA.

A estrutura de repetição WHILE especifica que uma instrução ou grupo de instruções deve ser executada repetidamente enquanto alguma condição permanecer verdadeira.

A repetição de um conjunto de instruções por um número específico de vezes é chamada de repetição CONTROLADA POR CONTADOR.

Quando não se sabe com antecedência quantas vezes um conjunto de instruções será repetido, um valor de SENTINELA pode ser usado para encerrar a repetição.

1. Abaixo vemos quatro instruções diferentes em C onde cada uma deve somar 1 à variável inteira x:

x = x + 1  
x += 1  
++ x  
x ++

1. Escreva uma única instrução em C que possibilite realizar as seguintes tarefas:

a) atribuir a soma de x e y e z e incrementar o valor de x em 1 após o cálculo.  
 z = x++ + y  
b) multiplicar a variável produto por 2 usando o operador \*=  
 produto \*= 2  
c) multiplicar a variável produto por 2 usando operadores = e \*  
 produto = produto \* 2  
d) testar se o valor da variável contador é maior do que 10. Se for, imprimir ‘Contador é maior do que 10’  
 if (contador > 10)  
 printf (“Contador é maior que 10.\n);  
e) decrementar a variável x em 1, depois subtraí-la da variável total  
 total -= --x;  
f) somar a variável x à variável total, depois decrementar x por 1  
 total += x--;  
g) calcular o resto após q ser dividido por divisor e atribuir o resultado a q; escreva essa instrução de duas maneiras diferentes  
 q %= divisor;  
 q = q % divisor;  
h) imprimir o valor 123.4567 com 2 dígitos de precisão. Que valor é impresso?  
 printf (“%.2f”,123.4567);  
 123.45 é exibido  
i) imprimir o ponto flutuante 3.14159 com 3 dígitos à direita do ponto decimal. Qual valor é impresso?  
 printf (“%.3f”,3.14159);  
 3.142 é exibido

4). Escreva uma instrução em C que possibilite realizar as seguintes tarefas:

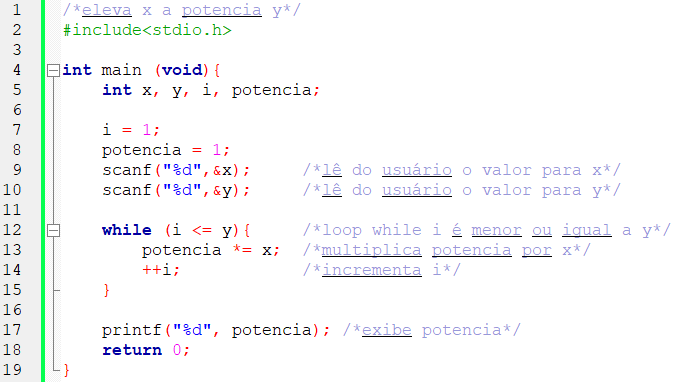
a) Declarar variáveis soma e x para que sejam do tipo int;  
 int soma, x;  
b) Inicializar a variável x em 1;  
 x = 1;  
c) Inicializar a variável soma em 0;  
 soma = 0;  
d) Somar a variável x à variável soma e atribuir o resultado à variável soma;  
 soma += x; OU   
 soma = soma + x;  
e) Imprimir ‘A soma é: ‘ seguido pelo valor da variável soma.  
 printf(“A soma é: %d\n”, soma);

5). Combine as instruções que você escreveu no exercício anterior em um programa que calcule a soma dos inteiros de 1 a 10. Use a estrutura while para realizar um loop pelas instruções de cálculo e incremento. O loop deverá terminar quando o valor x chegar a 11.  
*/\*Calcula a soma dos inteiros de 1 à 10\*/  
#****include****<stdio.h>****int*** *main (****void****){  
 int soma, x; /\*declara variáveis soma e x\*/  
 x = 1; /\*inicializa x\*/  
 soma = 0 /\*inicializa soma\*/* ***while*** *( x <= 10){  
 soma += x; /\*soma x à variável soma\*/  
 ++x; /\*incrementa x\*/  
 }  
 printf("A soma e: %d\n", soma); /\*exibe a soma\*/* ***return*** *0;  
}*

6). Determine os valores das variáveis ***produto*** e ***x*** depois que o cálculo a seguir for realizado. Suponha que o ***produto*** e ***x*** tenham o valor 5 quando a instrução começara ser executada.  
***produto \*= x++;***produto = 25, x = 6;

7) Escreva instruções em C que:  
a) leiam a variável inteira x com scanf  
 scanf(“%d”, &x);  
b) leiam a variável inteira y com scanf  
 scanf(“%d”, &y);  
c) inicializem a variável inteira i em 1  
 i = 1;  
d) inicializem a variável potência em 1  
 potência = 1;  
e) multipliquem a variável inteira potência por x e atribua o resultado à potência  
 potência \*= x;  
f) incrementem a variável i em 1  
 i ++;  
g) testem i para verificar se ele é maior ou igual a y na condição de uma estrutura while  
 if ( i <= y );  
h) exibam a variável inteira potência com printf  
 printf(“%d”, potência);

8). Escreva um programa em C que use as instruções do exercício anterior para calcular x elevado à potência y. O programa deverá ter uma estrutura de controle de repetição while.



9) Identifique e corrija os erros em casa um dos itens:  
a) while (c <= 5 ) {  
 produto \*= c;  
 ++c;  
 **}**  
b) scanf (“%.4f”, &valor);  
 scanf (“%f”, &valor);  
c) if ( sexo == 1)  
 printf(“Mulher\n”);  
 se não  
 printf(“Homem\n”);

if ( sexo == 1)  
 printf(“Mulher\n”);  
 se não  
 printf(“Homem\n”)

10) O que está errado na estrutura de repetição while a seguir (considere z com valor 100), que supõe-se calcular a soma dos inteiros de 100 para 1?  
while (z > = 10)  
soma + = z;  
O valor da variável Z nunca muda na estrutura while. Portanto, cria-se um loop infinito. Para impedir que isso aconteça, z precisa ser decrementado, de modo que seu valor chegue a 0.