CAPÍTULO 2

OPERADORES

2.1 Operadores aritméticos

Os operadores aritméticos são os seguintes: + (adição), - (subtração), * (multiplicação) e / (divisão). No caso de divisão entre números inteiros, o resultado é truncado. O operador % fornece o resto da divisão e só funciona como operador entre tipos inteiros. Então:

Expressão	Tem o valor
22 / 3	7
22 % 3	1
14 / 4	3
14 % 4	2
21 / 7	3
21 % 7	0

2.2 Operador de atribuição

O operador de atribuição (=) copia o valor do lado direito para a variável, ou endereço, do lado esquerdo. Além disso, o operador = também retorna este valor. Isso significa que, ao fazer x = y, por exemplo, o valor da variável y será copiado na variável x, sendo este valor o resultado da operação.

Em outras palavras, se fizermos x = y = 0, será processada inicialmente a operação y = 0, atribuindo o valor 0 à variável y. Esta expressão fornece o resultado 0, que é atribuído à variável x. No final do processamento, ambas as variáveis terão o valor 0.

Expressão	Operação	Valor da expressão
i = 3	Coloca o valor 3 em i	3
i = 3 + 4	O valor 7 é colocado em i	7
i = k = 4	O valor 4 é colocado em k; o valor da atribuição (4) é então colocado em i	4
i = (k = 4) + 3	O valor 4 é colocado em k; a adição é realizada e o valor 7 é colocado em i	4

2.3 Operadores relacionais

Os operadores relacionais em C são: == (igual a), != (diferente de), > (maior que), < (menor que), >= (maior ou igual a) e <= (menor ou igual a). O resultado de dois valores conectados por um operador relacional será 0 para falso ou 1 para verdadeiro.

Expressão	Valor
5 < 3	0
3 < 5	1
5 == 5	1
3 == 5	0
5 <= 5	1

Em C, não existem variáveis lógicas. Qualquer valor pode ser testado como verdadeiro ou falso. Se ele for zero, é falso. Se for qualquer valor diferente de zero, é verdadeiro.

2.4 Operadores lógicos

Os dois operadores lógicos binários são && (e) e || (ou). O resultado de suas operações também será 0 (falso) ou 1 (verdadeiro).

Expressão	Valor
5 || 3	1
5 || 0	1
5 && 3	1
5 && 0	0
(i > 5) && (i <= 7)	1 se i for maior que 5 e menor ou igual a 7 0, qualque outro caso

O operador negação (!) inverte o sentido do valor que o segue. Qualquer valor não zero será convertido para 0 e um valor 0 será convertido para 1.

Expressão	Valor
!5	0
!0	1
!(i > 5)	1 se i não for maior que 5 0, se i for maior que 5

2.5 Operadores bit a bit

Os operadores bit a bit operam apenas com números inteiros. Os operadores são: & (e, bit a bit), | (ou, bit a bit), ^ (ou exclusivo, bit a bit), ~ (negação, bit a bit), << (deslocamento à esquerda) e >> (deslocamento à direita). São utilizados normalmente para ligar ou desligar bits ou para testar bits específicos em variáveis inteiras.

Expressão	Valor	Expressão binária	Valor binário
1 | 2	3	00000001 | 00000010	00000011
0xFF & 0x0F	0x0F	11111111 & 00001111	00001111
0x33 & 0xCC	0xFF	00110011 | 11001100	11111111
0x0F << 2	0x3C	00001111 << 2	00111100
0x1C >> 1	0x0E	00011100 >> 1	00001110

2.6 Atribuições reduzidas

C oferece muitos operadores de atribuição que são redução de outros operadores. Eles tomam a forma de *op*=, onde *op* pode ser +, -, *, /, %, <<, >>, &, ^, |. A expressão *f op= g* é análoga a *f = f op g*. Por exemplo:

Expressão	É igual a
a += 2	a = a + 2
i <<= 1	i = i << 1
s /= 7 + 2	s = s / (7 + 2)

2.7 Operadores pré e pós-fixados

Os operadores pré e pós-fixados incrementam (++) ou decrementam (--) uma variável. Uma operação prefixada é realizada antes que o valor da variável seja utilizado. Uma operação pós-fixada é efetuada após a utilização da variável. Por exemplo, para uma variável i inteira com valor 5:

Expressão	Valor de i utilizado na avaliação	Valor da expressão	Valor final de i
5 + (i++)	5	10	6
5 + (i--)	5	10	4
5 + (++i)	6	11	6
5 + (--i)	4	9	4

2.8 Operadores condicionais

Operadores condicionais são uma maneira rápida de selecionar um entre dois valores, baseado no valor de uma expressão. A sintaxe é:

```
expr1 ? expr2 : expr3
```

Se *expr1* for verdadeira (não zero), então o resultado é o valor de *expr2*. Caso contrário é o valor de *expr3*. Por exemplo:

Expressão	Valor
5 ? 1 : 2	1
0 ? 1 : 2	2
(a > b) ? a : b	1
(a > b) ? ((a > b) ? a : c) : ((b > c) ? b : c)	1

2.9 Operador vírgula

Numa seqüência de expressões separadas por vírgulas, as expressões são processadas da esquerda para a direita sendo retornado o valor da expressão mais à direita.

Expressão	Valor
5, 1, 2	2
i++, j + 2	j + 2
i++, j++, k++	valor de k (antes do incremento)
++i, ++j, ++k	valor de k (depois do incremento)

2.10 Precedência de operadores

Os operadores têm uma ordem de precedência. Isto é, sem parênteses, certas operações são efetuadas antes de outras com menor precedência. Para operadores com precedência igual, a associatividade é da esquerda para direita, com algumas exceções (mostradas na tabela abaixo). Para se ter certeza da interpretação, as expressões que se deseja interpretar primeiro devem ser agrupadas entre parênteses.

Cada conjunto de operadores na tabela possui a mesma precedência. Os símbolos ainda não mencionados serão descritos mais adiante (nos capítulos sobre funções, matrizes e estruturas).

Operador	Descrição
()	chamada de função
[]	elemento de matriz
->	ponteiro para membro de estrutura
.	membro de estrutura
!	negação lógica
~	negação bit a bit
++	incremento
--	decremento
-	menos unário
(tipo)	conversão (cast)
*	ponteiro
&	endereço
sizeof	tamanho do objeto
*	multiplicação
/	elemento de matriz
%	resto da divisão
+	adição
-	subtração
<<	deslocamento à esquerda
>>	deslocamento à direita
<	menor que
<=	menor ou igual a
>	maior que
>=	maior ou igual a
==	igualdade
!=	desigualdade
&	e, bit a bit
^	ou exclusivo, bit a bit
|	ou, bit a bit
&&	e, lógico
||	ou, lógico
? :	condicional
=	atribuição
op=	atribuição
,	vírgula