OPERADORES MATEMÁTICOS, LÓGICOS, RELACIONAIS, ATRIBUIÇÃO E TABELA VERDADE

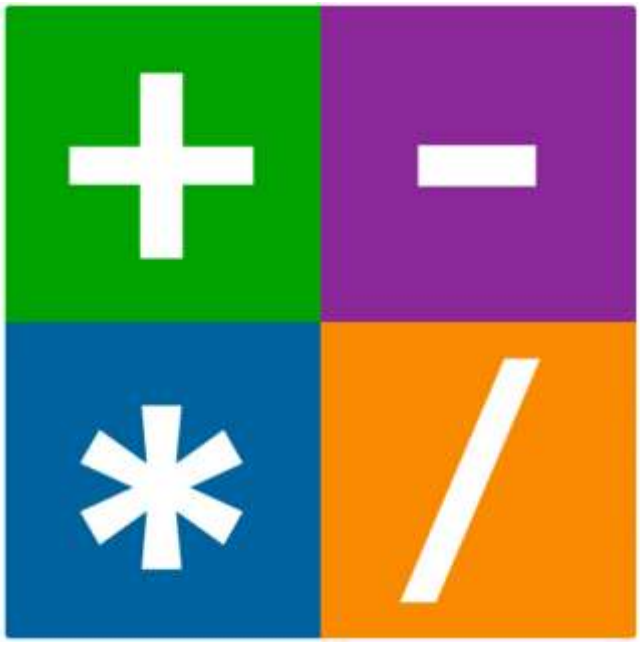
Manipulação dos dados

Operadores

Agora que já definimos os tipos de dados da linguagem C e apresentamos os conceitos de variáveis e constantes, precisamos aprender como manipular esses dados. Essa manipulação é realizada de acordo com os operadores disponibilizados pela linguagem.

Operadores matemáticos

O objetivo dos operadores matemáticos é representar as operações matemáticas do mundo real. Suas operações possuem peculiaridades para os quatro tipos de dados diferentes (char, int, float e double) da linguagem.



Os números reais são representados na linguagem C pelos tipos float e double por apresentarem uma maior similaridade com o mundo real.

Já aprendemos que a principal diferença entre ambos está em sua precisão:

float ≠ double

Operacionalidade

São ofertadas pela linguagem as seguintes operações:

Soma

Representada pelo símbolo ‘+’.

Subtração

Símbolo ‘-’ .

Multiplicação

Representada pelo símbolo ‘*’.

Divisão

Símbolo ‘/’ .

Essas operações funcionam exatamente da mesma forma que no mundo real, possuindo como única diferença a precisão numérica calculada.

1. **Mundo real**
   Os números podem possuir representação infinita.
2. **No computador**
   Isso não é possível, pois, nesse ambiente, a representação é finita.

Desse modo, é possível ocorrer algum problema de precisão numérica ao serem realizados os cálculos matemáticos. Embora seja pouco significativo na maioria dos casos, esse problema acarreta uma decisão sobre o tipo de ponto flutuante utilizado, que pode ser o de precisão.

float (simples) ou *double* (dupla)

Vejamos uma tabela com um resumo do que estudamos até o momento:

Operação matemática	Símbolo utilizado	Exemplo	
		Equação	Resultado
Soma	+	1.2 + 3.4	4.6

Subtração	-	1.2 - 3.4	-2.2
Multiplicação	*	1.2 * 3.4	4.08
Divisão	/	1.2 / 3.4	-0.3529

Tabela: Resumo 1.
Anderson Fernandes Pereira dos Santos.

Para os inteiros, números sem casa decimal, as diferenças começam a aparecer. As operações de soma, subtração e multiplicação funcionam essencialmente conforme já explicamos, considerando que os dois operandos sejam números inteiros.

Se um desses operadores for um número inteiro (*int*) e o outro, um real (*float* ou *double*), seu resultado também será um número real (*float* ou *double*, respectivamente).

Para a operação de divisão, quando os dois operandos são números inteiros, o resultado também é um inteiro. Portanto, essa operação é chamada de **divisão inteira**, embora ela use o mesmo símbolo utilizado para números reais.

Exemplo

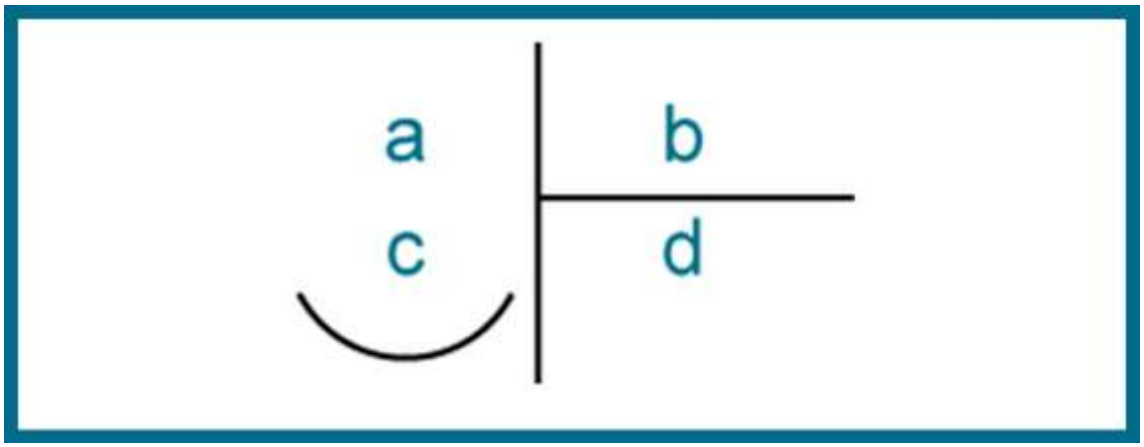
Observe que 5 / 2 tem como resultado o número 2, ou seja, o maior inteiro que pode ser obtido dentro do resultado matemático – que, neste caso, seria 2,5.

Caso a operação de divisão envolva dois números – um real (*float* ou *double*) e um inteiro (*int*) –, o resultado será um número real (*float* ou *double*). Ainda existe outra operação que é particular de números inteiros: resto da divisão. Usando o símbolo *%*, ela retorna o resto da divisão de dois números inteiros.

Exemplo

Note que 5 % 2 tem como resultado o número 1.

Observemos mais um resumo do que aprendemos.



Resumo 2.

A divisão inteira dos números inteiros (*int*) **a** e **b** resulta no valor **d** e no resto **c**. Assim, os valores c e d podem ser obtidos por meio das seguintes equações:

$$d = a / b$$

$$c = a \% b$$

Classificação

Perante a quantidade de operandos possíveis, os operadores podem ser classificados como:

1. Unários

   Só possuem um operando. O operando dos operadores unários é chamado de **incremento** ou **decremento**. Esses operadores podem ser usados de forma pré-fixa ou pós-fixa. Nas duas situações, os valores são acrescidos (incremento) ou decrescidos (decremento) de uma unidade. Desse modo, a expressão **a++** ou **++a** calcula o valor **a+1**.

2. Binários

   Possuem dois operandos.

3. Ternários

   Possuem três operandos.

Todos os operadores apresentados até aqui são considerados BINÁRIOS, pois eles possuem dois operandos.

Vejamos esta tabela com um resumo do que foi exposto:

Operação matemática	Símbolo utilizado	Exemplo	
		Equação	Resultado
Soma	+	1 + 2	3
Subtração	-	3 - 4	-1
Multiplicação	*	5 * 6	30
Divisão inteira	/	5 / 2	2
Resto da divisão	%	5 % 2	1
Incremento	++	2++	3
		++2	
Decremento	--	2--	1
		--2	

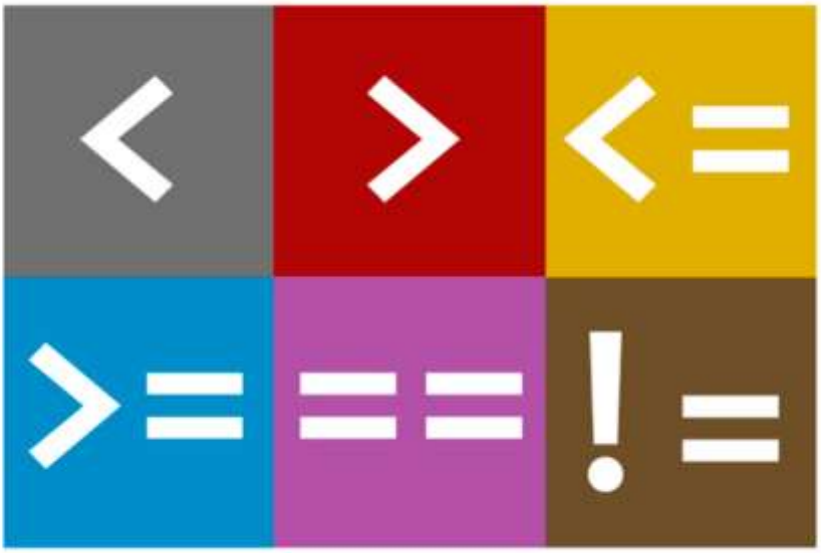
Tabela: Resumo 3.
Anderson Fernandes Pereira dos Santos.

Quando utilizados como operandos de operação matemática, os dados do tipo char são traduzidos para números inteiros, possuindo o mesmo funcionamento descrito anteriormente. Essa tradução é realizada graças ao uso da tabela ASCII apresentada no módulo 1.

Operadores relacionais

Os operadores relacionais permitem a realização de comparações entre valores. Elas são expressas por meio dos valores verdadeiro e falso.

Um dos tipos utilizados no desenvolvimento de algoritmos é o lógico. Ele modela a álgebra de Boole ou álgebra booleana, base para o desenvolvimento da eletrônica presente na computação. Nessa álgebra, são utilizados os valores verdadeiro e falso. Na linguagem C, não há um tipo de dado que represente tais valores diretamente. Essa representação ocorre pela interpretação do valor da variável. Assim, os valores 0, vazio ou *null* são interpretados como falso; os outros, como verdadeiro.



As operações são:

Menor

Expressa pelo símbolo '<'.< /p>

Maior

Símbolo '>'.

Menor ou igual

Combinação dos símbolos '<='.< /p>

Maior ou igual

Combinação dos símbolos '>='.

Igualdade

Combinação dos símbolos '=='.

Desigualdade

Combinação dos símbolos '!='.

Caso seja necessário verificar se uma pessoa tem mais de 1,90m de altura, o que podemos fazer?

Resposta

Devemos comparar os valores de altura da pessoa pela variável **a** e pelo valor de referência 1,90m. Na linguagem C, esse conhecimento é representado por: **a** > 1.9

Note que a unidade de medida não é expressa na equação. Caso fosse realizada a comparação anterior, seria necessário manter essa unidade.

Demonstraremos a seguir uma tabela com um resumo do assunto abordado:

Operação matemática	Símbolo utilizado	Exemplo	
		Equação	Resultado
Maior	>	1.2 > 3.4	0 (falso)
Menor	<	1.2 < 3.4	1 (verdadeiro)
Menor ou igual	<=	1.2 <= 3.4	1 (verdadeiro)

Maior ou igual	>=	1.2 >= 3.4	0 (falso)
Igualdade	==	1.2 == 3.4	0 (falso)
Desigualdade	!=	1.2 != 3.4	1 (verdadeiro)

Tabela: Resumo 4.
Anderson Fernandes Pereira dos Santos.

Operadores lógicos

Eles possuem como operandos os tipos verdadeiro e falso apresentados anteriormente. Existem dois tipos de operadores lógicos:

Unários

Possuem apenas um operando. Exemplo: Negação (representado pelo símbolo!).

Quando é aplicado a uma variável lógica, o operador **negação** (!) retorna o oposto dela.

Exemplo: Caso a variável a seja falsa (0, vazio ou *null*), sua negação valerá **verdadeiro** (valor diferente de 0, vazio ou *null*).

Binários

Têm dois operandos.

Exemplo: Trata-se do **e-lógico** (representado pela combinação dos símbolos &&) e do ou-lógico (combinação dos símbolos ||).

Quando for aplicado a dois valores lógicos, o operador **e-lógico** (&&) só retornará **verdadeiro** (1) se os dois operadores forem simultaneamente verdadeiros.

Da mesma forma, o operador **OU** (||) retornará verdadeiro nos casos em que, no mínimo, um dos operandos seja verdadeiro.

Verifiquemos um resumo sobre esse assunto:

Operador lógico	Símbolo utilizado	Exemplo	
		Equação	Resultado
Negação	!	!0	1
Operador E	&&	1 && 0	0
Operador OU	||	1 || 0	1

Tabela: Resumo 5.
Anderson Fernandes Pereira dos Santos.

Note que, durante o estudo, são usados os termos falso e verdadeiro.

Na linguagem C, os tipos de dados que representam o valor falso sempre são os valores:

1. 0: Caso a variável seja numérica, ou seja, *int*, *float* ou *double*;
2. *null*: Se for uma variável que armazene algum endereço de memória;
3. *null*: Quando for uma string, isto é, uma cadeia de caracteres.

O valor, portanto, será **verdadeiro** caso não seja **falso**, podendo assumir quaisquer valores numéricos, de endereço de memória ou de cadeia de caracteres.

Operadores bit a bit

Como vimos até agora, os tipos de dados apresentados ocupam espaço em memória.

1 *byte*

O tipo caractere ocupa um *byte* na memória.

4 *bytes*

O tipo *float* ocupa quatro *bytes* na memória.

Já sabemos que 1 byte é igual a 8 bits. Em algumas situações, no entanto, é necessário realizar uma manipulação bit a bit.

Exemplo

Esses casos ocorrem quando manipulamos tráfegos em redes de computadores, obtemos valores armazenados em memória e desejamos fazer alguma leitura ou escrita direta em dispositivos físicos (*hardware*).

Essas operações podem ser resumidas de acordo com a seguinte tabela:

Operação	Expressão	Exemplo	
		Equação	Resultado
E lógico	a&b	2&6	2
OU lógico	a | b	2 | 4	6
OU Exclusivo	a ^ b	2 ^ 6	4
Deslocamento à direita	a >> b	4 >> 2	1
Deslocamento à esquerda	a << b	2 << 4	32
Negação	~a	~2	-3

Tabela: Resumo 6.
Anderson Fernandes Pereira dos Santos.

Operadores de atribuição

As operações observadas até aqui permitiram a realização de cálculos, comparações e manipulações dos dados. Agora, contudo, é necessário apresentar a maneira de armazenar esses valores em memória – e isso é feito em função dos **operadores de atribuição**.

Em computação, como podemos atribuir um valor 1,80m a uma variável altura e como representamos essa expressão na linguagem C?

Além disso, podemos armazenar o resultado de uma operação em determinada variável. Em nosso exemplo, o IMC é calculado pela divisão do peso pela altura ao quadrado:

IMC = peso / altura × altura

Como representamos essa expressão matemática na linguagem C?

LINGUAGEM C

IMC = peso / (altura * altura);

Nesse caso, as operações são realizadas do lado direito da expressão, enquanto seu resultado é armazenado na variável IMC.

Vamos a um exemplo:

Em uma aplicação, existe a necessidade de adicionar R$100,00 ao saldo bancário de uma pessoa. Para isso, deve-se recuperar o valor do saldo bancário dela, somar o de R$100,00 e, na sequência, armazenar o resultado na mesma variável.

Caso esse saldo fosse representado pela variável *SaldoBancario*, como poderíamos representar sua expressão na linguagem C?

Nesse caso, a variável é utilizada dos dois lados da expressão.

Dessa forma, do lado direito da equação, temos o valor inicial da variável antes de a expressão ser executada e, no esquerdo, o da variável após a sua execução.

Essa forma de operação e atribuição sequencial pode ser substituída por outra mais resumida na qual não haja a necessidade de repetir o nome da variável dos dois lados da expressão:

LINGUAGEM C

SaldoBancario += 100;

Esta tabela demonstra que a forma resumida também pode ser utilizada em outras operações:

Operação	Forma resumida
a = a + b;	a + = b;
a = a - b;	a - = b;
a = a * b;	a* = b;
a = a/b;	a/ = b;
a = a % b	a % = b;

Operação	Forma resumida
a = a & b;	a & = b;
a = a|b;	a| = b;
a = a^b;	a^ = b
a = a << b;	a <<= b;
a = a >> b;	a >>= b;

Tabela: Operação / Forma resumida.
Anderson Fernandes Pereira dos Santos.

Operadores de conversão

Este tipo de operador permite uma “tradução” entre valores diferentes. Com ele, é possível converter valores de tipos de dados diferentes. Essa conversão pode ocorrer de duas formas sem perda de informação e com perda de informação. Veja mais a seguir.

Sem perda de informação

Converte um tipo que ocupa uma quantidade menor de memória para outro com uma quantidade maior.

Exemplo: considere que a variável idade, do tipo inteiro, tenha valor 20.

Desse modo, temos:

LINGUAGEM C

```
idade = 20;
```

Caso fosse necessário convertê-la para outra variável do tipo *float*, *idade_real*, essa conversão não apresentaria problema.

Desse modo, teríamos:

LINGUAGEM C

idade_real = (float) idade;

Nesse caso, a variável *idade_real* fica com o valor 20.0. Portanto, não há perda de informação.

Com perda de informação

Converte-se um tipo de dado de maior tamanho ocupado em memória para outro com um tamanho menor.:

Exemplo: considere a variável float pi com valor 3,1415

LINGUAGEM

float pi = 3.1415;

Se quisermos converter esse número para uma variável inteira p, como a variável pi possui uma parte inteira (antes da vírgula) e outra decimal (depois dela), a inteira será copiada para a nova variável, enquanto a decimal ficará perdida.

Assim, a conversão *int* **p** = (*int*)**pi**; resultaria no seguinte valor final de **p** = 3. Haveria, portanto, uma perda da parte decimal 0.1415.

Precedência dos operadores

Precisamos definir a ordem em que os operadores podem ser aplicados. Imagine uma expressão do tipo:

a+**b**%**c**

O que seria executado primeiramente? A soma ou a operação resto de divisão? Exemplo: considere o seguinte:

LINGUAGEM C

int a=1, b=2, c=3;

Qual seria o valor da expressão **a**+**b**%**c**?

Como o operador resto da divisão tem precedência, ele é executado primeiramente e seu resultado, adicionado à variável soma. Desse modo, tal expressão seria executada pelo ambiente de desenvolvimento da seguinte forma:

a+**b**%**c**

1+2%3

1+2

3

A precedência de todos os operadores é apresentada nesta tabela:

Prioridade	Precedência
12	() [] . -> Expressão++ Expressão--
11	* & + - ! ~ ++Expressão --Expressão (Conversão) sizeof
10	* / %
9	+ -
8	>> <<
7	<> <=>=
6	== !=
5	& ^ |
4	&&
3	||
2	?:

1	= += == *= /+ %= >>= <<= &=^=|=,

Tabela: Prioridade / Precedência.
Anderson Fernandes Pereira dos Santos.

Nas primeiras linhas, são exibidos os itens com maior prioridade (menor número). Desse modo, aqueles com uma escala 10 possuem prioridade maior que outros com uma 5.

Tabela verdade

Precedência de operadores

Já dissertamos sobre o funcionamento dos operadores lógicos e relacionais. Tais operadores são utilizados para desenvolver expressões lógicas a serem utilizadas em instruções de fluxo de execução, constituindo parte essencial no desenvolvimento de uma aplicação.

Para analisar o resultado de uma expressão lógica, deve ser elaborada uma tabela conhecida como tabela verdade (ou tabela veritativa).

São utilizadas nela todas as combinações possíveis de entrada, sendo calculados, consequentemente, todos os valores possíveis da expressão lógica.

Exemplo: considere as variáveis ***a*** e ***b*** a seguir.

LINGUAGEM C

a && b

A tabela verdade montada para tal expressão deve considerar que as variáveis **a** e **b** possam assumir os valores verdadeiro e falso, tendo o resultado expresso na última coluna desta tabela:

a	b	a && b
verdadeiro	verdadeiro	verdadeiro
verdadeiro	falso	falso

falso	verdadeiro	falso
falso	falso	falso

Tabela: Tabela verdade para a && b.
Anderson Fernandes Pereira dos Santos.

Apresentaremos a seguir as tabelas verdade para as expressões dos operadores lógicos anteriormente citados:

a	b	a||b
verdadeiro	verdadeiro	verdadeiro
verdadeiro	falso	verdadeiro
falso	verdadeiro	verdadeiro
falso	falso	falso

Tabela: Tabela verdade 1.
Anderson Fernandes Pereira dos Santos.

a	b	$a \rightarrow b$
verdadeiro	verdadeiro	verdadeiro
verdadeiro	falso	falso
falso	verdadeiro	verdadeiro
falso	falso	verdadeiro

Tabela: Tabela verdade 2.
Anderson Fernandes Pereira dos Santos.

a	b	a ^ b
verdadeiro	verdadeiro	falso
verdadeiro	falso	verdadeiro
falso	verdadeiro	verdadeiro
falso	falso	falso

Tabela: Tabela verdade 3.
Anderson Fernandes Pereira dos Santos.

a	~a
verdadeiro	falso
falso	verdadeiro

Tabela: Tabela verdade 4.
Anderson Fernandes Pereira dos Santos.

Aplicação dos conceitos apresentados

Desde o final da década de 1990, a internet dominou todos os campos de nossa vida. Hoje, é praticamente impossível vivermos sem o uso das ferramentas proporcionadas pela web (gerada nos laboratórios da CERN graças ao trabalho do físico britânico e cientista da computação Tim Berns-Lee).

Com a finalidade de apresentar uma forma mais dinâmica de divulgação dos dados, Berns-Lee desenvolveu o protocolo HTTP (permite o acesso a sites na internet), que constitui a base de praticamente todas as tecnologias na área da informática. Esse desenvolvimento esteve fundamentado na confiabilidade dos protocolos que permitem o acesso aos sites feito basicamente por meio do envio de bits e bytes pela internet.

Graças à representação de dados (*int*, *float*, *double* e *char*) em linguagens de programação, foi possível obter todo o *boom* tecnológico dos últimos anos.