



Tipos de dados, expressões, operadores e tabela verdade

Prof.^o Anderson Fernandes Pereira dos Santos

Kleber de Aguiar

Descrição

Tipos de dados da linguagem C. Manipulação de variáveis e constantes, além de suas operações. Tipos de operadores (e sua precedência) e de expressões passíveis de uso. Conceito de tabela verdade na elaboração de expressões lógicas.

Propósito

Compreender os conceitos dos tipos de dados suportados pela linguagem e suas manipulações para o desenvolvimento de aplicações robustas e eficientes.

Preparação

Antes de iniciar seu estudo, instale e configure em seu computador ou smartphone o ambiente de desenvolvimento Dev C++, que é obtido gratuitamente na internet. Para fazer isso, pesquise e siga as instruções indicadas por Lucas Hort no vídeo Como baixar, instalar e configurar o Dev-C++ no Windows (2019).

Se não quiser realizar a instalação e configuração desse ambiente, você ainda pode usar a versão portátil (também disponível na internet). Basta apenas executá-la diretamente por intermédio de um pendrive.

Objetivos

Módulo 1

Conceitos de tipos de dados

Empregar os conceitos de tipos de dados por meio da manipulação de variáveis e constantes na linguagem C.

Módulo 2

Operadores matemáticos, lógicos, relacionais, atribuição e tabela verdade

Aplicar os operadores matemáticos, lógicos, relacionais e de atribuição, além dos conceitos de tabela verdade.



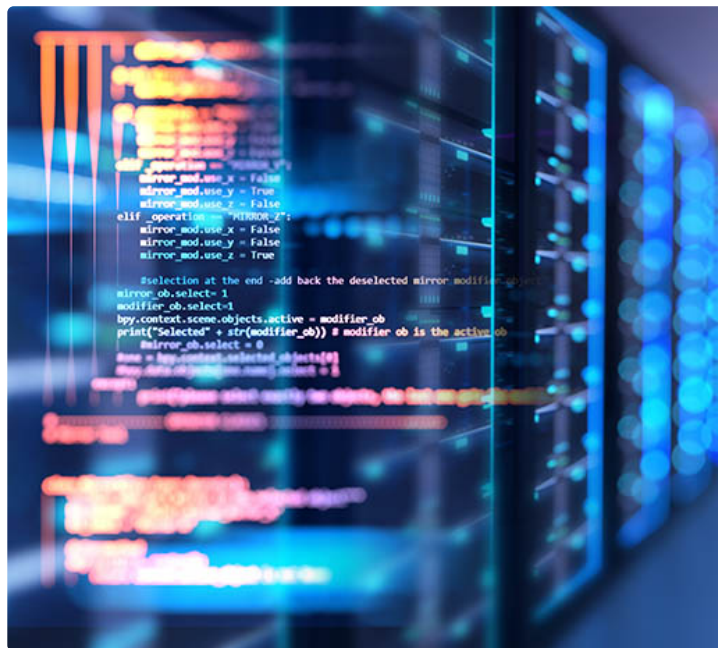
Introdução

Em função dos avanços tecnológicos, o uso de linguagem de programação vem crescendo. Nos cursos de graduação voltados à tecnologia, é frequente que se tenha contato com mais de uma disciplina focada em programação. Além disso, outras carreiras que compartilham de um aprendizado tecnológico têm contato com pelo menos uma linguagem de programação. Nessas diferentes perspectivas, várias linguagens de programação são usadas, tanto para desenvolvimento de projetos como para ensino.

Aprender uma linguagem de programação vem a ser, como em qualquer disciplina, aprender mecanismos que possibilitam desenvolver melhor uma atividade. Conhecer os conceitos básicos de programação, são os primeiros passos para desenvolver habilidades iniciais para escrever os primeiros programas.

Na linguagem C, compreender os conceitos dos tipos de dados que a linguagem suporta para o armazenamento das informações, bem como sua manipulação se faz necessário para o desenvolvimento de aplicações mais eficientes.

Fazendo uso da linguagem de programação C, vamos aprender conceitos simples de manipular variáveis e constantes, além de suas operações. Vamos também elaborar expressões lógicas fazendo uso de operadores lógicos.



1 - Conceitos de tipos de dados

Ao final deste módulo, você será capaz de empregar os conceitos de tipos de dados por meio da manipulação de variáveis e constantes na linguagem C.

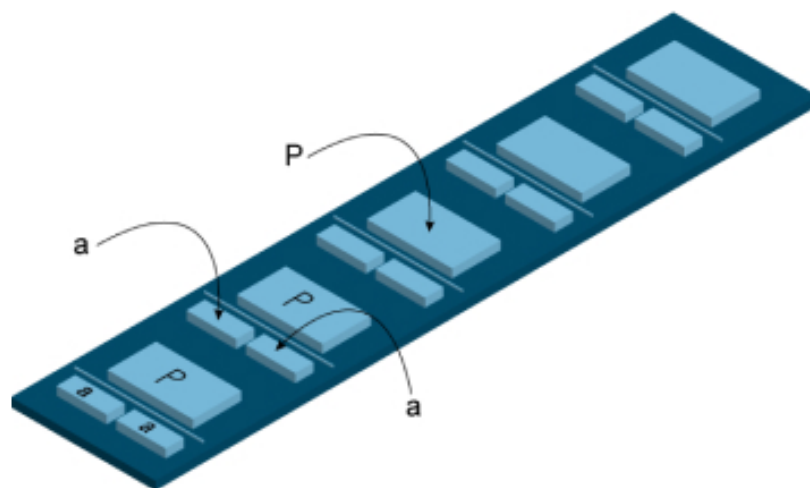
Dados

Tipos de dados

Você sabe como podemos representar a solução de um problema da vida real na linguagem de programação? É possível fazer isso por meio de uma sequência finita de passos conhecida como algoritmo.

Um algoritmo pode ser entendido como uma linha de produção fabril semelhante à do **Fordismo**, uma vez que transformamos os dados de entrada para alcançarmos ou calcularmos determinado valor.

Veja na imagem ao lado a representação do carregamento do código na linguagem C como uma linha de produção. Os espaços de memória recebem inputs (entradas) para transformá-los em códigos (saídas). Com a execução do código, esses espaços são preenchidos por variáveis que dão origem à linguagem de programação.

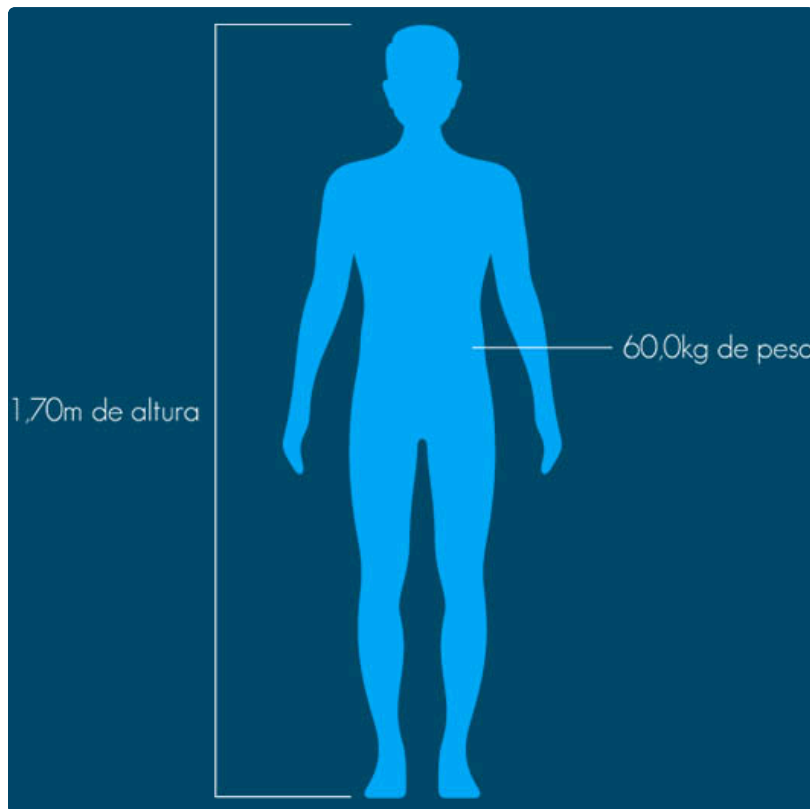


Representação do carregamento do código.

Fordismo

Linha de produção do empreendedor e engenheiro mecânico estadunidense Henry Ford (1863-1947) implantada em 1914.

Para o cálculo do índice de massa corpórea (IMC) de uma pessoa, medimos sua altura e seu peso. Normalmente, ela é medida em metros e possui valores com casas decimais. Da mesma forma, ele o é em quilogramas, apresentando casas decimais. Suponhamos que essas medidas apresentem os seguintes números apresentados na imagem a seguir.



A representação em casas decimais de números tão comuns do nosso dia a dia também pode ser feita nas linguagens de programação.



Para desenvolver um algoritmo, precisamos:

1. Identificar quais dados de entrada serão utilizados e como representá-los em nossa linguagem de programação.
2. Fazer com esses dados já identificados as transformações necessárias para modificar ou realizar cálculos.

Da mesma forma que, para montar um carro, Ford iniciava o processo com uma carroceria a fim de poder agregar seus demais componentes, como portas, sistema de suspensão e motor, nós o começamos com uma região de memória na qual serão acrescentados os dados necessários para a realização do nosso cálculo.

Desde sua concepção, a linguagem C possui quatro tipos de dados básicos:

1. *char*;
2. *int*;
3. *float*;
4. *double*.

Por meio deles e de suas manipulações, é possível representar qualquer tipo de informação do mundo real.

Dica

Ainda existe nessa linguagem uma forma de identificar a ausência de valores. Tal situação será vivida quando posteriormente forem tratados os casos de modularização de códigos (funções e procedimentos). Neste caso, é usada a palavra reservada *void*.

Char

O tipo *char* representa um caractere (podendo ser uma letra, um número ou um símbolo) e ocupa um *byte* na memória. Em computação, ele é representado pela tabela ASCII (Sigla para American Standard Code for Information Interchange) com seus 256 símbolos. Observe-a a seguir:

ASCII control characters				ASCII printable characters				Extended ASCII characters			
00	NULL	(Null character)		32	space	64	@	96	.	128	Ç
01	SOH	(Start of Header)		33	!	65	A	97	a	129	ü
02	STX	(Start of Text)		34	"	66	B	98	b	130	é
03	ETX	(End of Text)		35	#	67	C	99	c	131	â
04	EOT	(End of Trans.)		36	\$	68	D	100	d	132	ä
05	ENQ	(Enquiry)		37	%	69	E	101	e	133	å
06	ACK	(Acknowledgement)		38	&	70	F	102	f	134	à
07	BEL	(Bell)		39	'	71	G	103	g	135	ç
08	BS	(Backspace)		40	(72	H	104	h	136	ê
09	HT	(Horizontal Tab)		41)	73	I	105	i	137	ë
10	LF	(Line feed)		42	*	74	J	106	j	138	è
11	VT	(Vertical Tab)		43	+	75	K	107	k	139	í
12	FF	(Form feed)		44	,	76	L	108	l	140	î
13	CR	(Carriage return)		45	-	77	M	109	m	141	ï
14	SO	(Shift Out)		46	.	78	N	110	n	142	Ä
15	SI	(Shift In)		47	/	79	O	111	o	143	Å
16	DLE	(Data link escape)		48	0	80	P	112	p	144	É
17	DC1	(Device control 1)		49	1	81	Q	113	q	145	æ
18	DC2	(Device control 2)		50	2	82	R	114	r	146	Æ
19	DC3	(Device control 3)		51	3	83	S	115	s	147	ó
20	DC4	(Device control 4)		52	4	84	T	116	t	148	ô
21	NAK	(Negative acknowl.)		53	5	85	U	117	u	149	ö
22	SYN	(Synchronous idle)		54	6	86	V	118	v	150	ù
23	ETB	(End of trans. block)		55	7	87	W	119	w	151	û
24	CAN	(Cancel)		56	8	88	X	120	x	152	ÿ
25	EM	(End of medium)		57	9	89	Y	121	y	153	Ö
26	SUB	(Substitute)		58	:	90	Z	122	z	154	Ü
27	ESC	(Escape)		59	;	91	[123	{	155	ø
28	FS	(File separator)		60	<	92	\	124		156	£
29	GS	(Group separator)		61	=	93]	125	}	157	ø
30	RS	(Record separator)		62	>	94	^	126	~	158	x
31	US	(Unit separator)		63	?	95	_			159	f
127	DEL	(Delete)								191	ÿ
										223	■
										255	nbsp

ASCII control characters / ASCII printable characters / Extended ASCII characters.

Nos 256 símbolos listados, ocorre a seguinte divisão:

Do 0 ao 31

Os 32 iniciais são símbolos de controle.

Do 32 ao 127

Compõem a tabela ASCII (algumas vezes, chamada de normal).

Do 128 ao 255

Pertencem à tabela ASCII estendida.

Esses caracteres podem ser usados de diversas formas.

Exemplo

A representação dos termos masculino e feminino em um cadastro é feita pelos caracteres *M* e *F*. Utilizam-se aspas simples para a sua representação quando ambos forem mostrados em uma implementação. Desse modo, o caractere **M** de **masculino** é representado por '*M*' e o **F** de **feminino**, por '*F*'.

Observemos que a tabela ASCII representa os caracteres minúsculos e maiúsculos de forma distinta:

m ≠ **M**

Assim, conforme pode ser visto, o '*m*' (**m** minúsculo) é diferente de '*M*' (**M** maiúsculo).

Notemos também que, para cada caractere da tabela, existe um índice representado em decimal ou hexadecimal.



Os quatro tipos de dados básicos utilizados na linguagem c

Confira agora os outros três tipos de dados básicos da linguagem C: *int*, *float* e *double*.

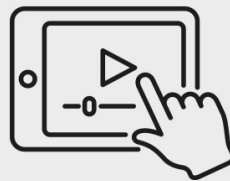
Para assistir a um vídeo sobre o assunto, acesse a versão online deste conteúdo.



Variáveis e constantes

Neste vídeo, vamos mergulhar na manipulação de variáveis e constantes em C. Exploraremos os conceitos fundamentais de declaração, atribuição e operações aritméticas em variáveis.

Para assistir a um vídeo sobre o assunto, acesse a versão online deste conteúdo.



Manipulação de variáveis e constantes

Já sabemos como representar os dados do mundo real na linguagem de programação C. Agora precisamos entender como eles podem ser manipulados. Para fazer isso, a linguagem trata os dados como variáveis e constantes.

Conceito

A variável é um tipo de espaço de memória que pode ser alterado a qualquer tempo. A constante, por sua vez, não pode. As duas formas permitem a referência deles em um espaço de memória. Esses espaços são identificados por meio de rótulos. Chamados de identificadores, eles possibilitam, a partir de seu uso, o acesso ao conteúdo armazenado em memória.

Exemplo

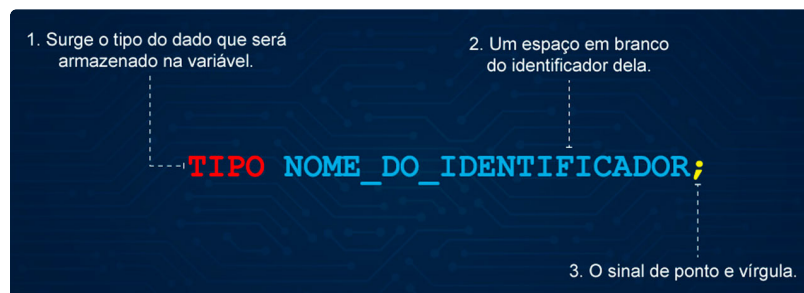
Caixas de correio que ficam em frente às residências.

Definição das variáveis

Formalmente, um espaço de memória é rotulado por intermédio de um identificador quando as variáveis são definidas. Veja uma ilustração de processo na próxima imagem.



Para criar uma variável, utiliza-se a seguinte notação:



O tipo do dado pode ser qualquer um dos quatro tipos já abordados: *char*, *int*, *float* e *double*.

De acordo com o que vimos, como estaria descrita a representação dos valores de peso e altura? Veja no próximo recurso.

LINGUAGEM C



Já sabemos que a linguagem C é considerada sensível a um contexto. Assim, ao escrevermos uma aplicação nessa linguagem, os identificadores serão diferentes. Veja um exemplo a seguir:

Contexto

A grafia que usa maiúsculas e minúsculas é diferente.

peso - Peso - PESO

Além disso, o próprio tipo de dado utilizado possui a mesma regra.

Desse modo:

***Float* (todas as letras são minúsculas) = ao tipo de dado, constituindo uma palavra reservada da linguagem.**

Quaisquer representações diferentes não correspondem a ele, podendo, dessa forma, ser utilizadas como identificadores, a exemplo de *Float* ou *FLOAT*.

Recomendação

Não constitui uma boa prática de programação usar identificadores que sejam variantes em minúsculas ou maiúsculas de palavras reservadas. Por exemplo, não é recomendável o uso de identificadores como *Float* ou *FLOAT*.

Ainda podemos definir as variáveis com outro formato:

LINGUAGEM C



Como estaria descrita, portanto, a representação dos valores de peso e altura? Vamos conferir!

LINGUAGEM C



Também é possível estabelecer uma quantidade maior de variáveis separando-as sempre das demais pelo uso de vírgula, enquanto a última deve conter um ponto e vírgula para finalizar.

Atenção!

Não é recomendável definir uma quantidade muito grande de variáveis de uma só vez, pois isso dificulta o entendimento do código-fonte da aplicação.

Recomendamos a definição de poucas variáveis por vez. Caso haja algum tipo de relação entre elas, essa identificação deve ser feita por meio de comentários.

Seguindo o exemplo do caso de peso e altura, faríamos assim:

LINGUAGEM C



Outro ponto importante é que uma variável sempre deve ser definida antes de seu uso. Assim, quando formos usar determinada variável, sua definição deverá ocorrer previamente.

Coloquialmente conhecidos como **nomes de variáveis**, os identificadores podem ter até 32 caracteres formados por:

1. Letras do alfabeto (maiúsculas e minúsculas);
2. Dígitos (0-9);
3. Símbolo de underscore _ .

O primeiro caractere deve ser uma letra do alfabeto ou o underscore.

Não usamos caracteres acentuados ao definirmos um identificador.

Saiba mais

Pesquise na internet sobre a notação húngara criada por Charles Simonyi.

Além das variáveis, há situações em que é necessário usar valores fixos em toda a aplicação. Conhecidos como constantes, esses valores são definidos por intermédio da palavra reservada "const" antes do tipo de acordo com o seguinte formato:

```
const TIPO NOME_DO_IDENTIFICADOR;
```

Nele, o NOME_DO_IDENTIFICADOR segue as mesmas regras relativas ao identificador descritas anteriormente. Por exemplo, é possível definir o valor π usando o seguinte exemplo:

```
const float pi = 3.141592;
```

Conceitos

Aplicação dos conceitos apresentados

Da teoria da informação, surgem os conceitos de:

Dados

Considerado um valor sem contextualização.

Informação

Quando é contextualizado, o dado transforma-se em informação.

Hoje em dia, a informação é o principal fator de destaque em empresas vencedoras. A partir dos dados contextualizados, é possível compreender tudo à nossa volta. Afinal, eles dão origem a áreas que estão revolucionando o mercado nos últimos anos.

Exemplo

Big data, ciência de dados e inteligência artificial.

Só será possível analisar os dados, entendendo suas correlações e regras de formação, se eles forem tratados da melhor forma à medida que estiverem sendo capturados no mundo real.

Falta pouco para atingir seus objetivos.

Vamos praticar alguns conceitos?

Questão 1

(Adaptada de: MPU - FCC - Analista de Informática - Desenvolvimento de Sistemas - 2007) O tipo de dados *float* refere-se aos dados do tipo:

- A Caractere
- B Inteiro
- C Booleano
- D Real
- E Vazio (sem valor)

Parabéns! A alternativa D está correta.

Os do tipo *float* são dados com casas decimais. Por isso, eles são representados na Matemática como números reais.

Questão 2

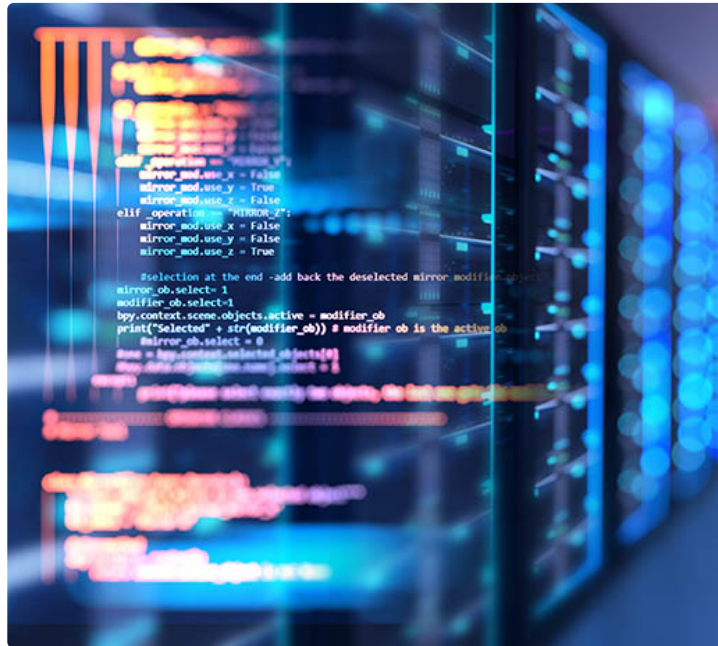
(Adaptada de: IBGC - Hemominas - Técnico de Informática - 2013)
Assinale a alternativa que apresenta um exemplo típico de dados numéricos sem casas decimais:

- A Rua Corrente Divina, 123.
- B 558.
- C 3,1415.
- D Um, dois e três.

E '7'.

Parabéns! A alternativa B está correta.

As letras A, D e E representam tipos de dados do tipo *char*. B apresenta um tipo de dado numérico sem casas decimais; C, tipos de dados com casas decimais.



2 - Operadores matemáticos, lógicos, relacionais, atribuição e tabela verdade

Ao final deste módulo, você será capaz de aplicar os operadores matemáticos, lógicos, relacionais e de atribuição, além dos conceitos de tabela verdade.

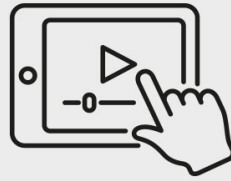
Manipulação dos dados



Manipulação de Dados e Operadores Matemáticos em C: Dominando as Operações Fundamentais.

Neste vídeo, vamos explorar a manipulação de dados e os operadores matemáticos em C. Aprenderemos como realizar as operações fundamentais, como soma, subtração, multiplicação e divisão, em variáveis numéricas.

Para assistir a um vídeo sobre o assunto, acesse a versão online deste conteúdo.

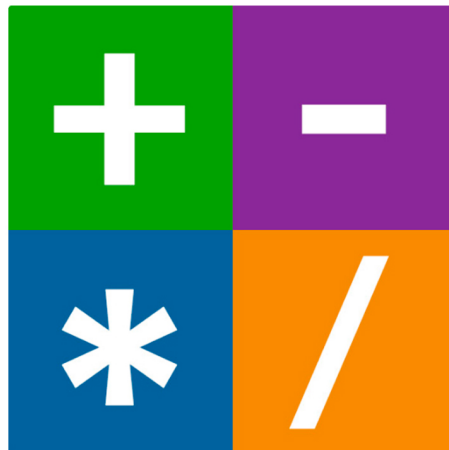


Operadores

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.

Mundo real		No computador
Os números podem possuir representação infinita.	×	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
Soma	+	1.2 + 3.4
Subtração	-	1.2 - 3.4
Multiplicação	*	1.2 * 3.4
Divisão	/	1.2 / 3.4

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

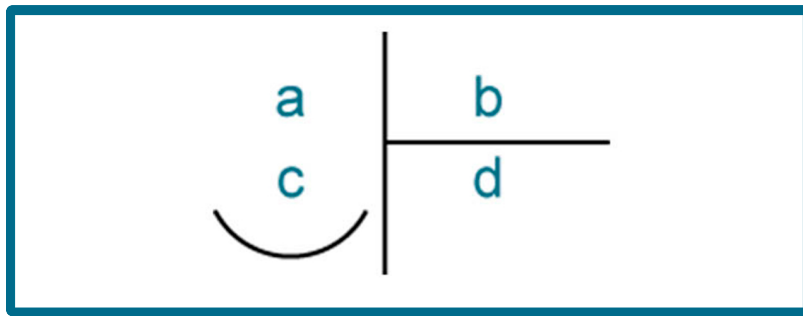
Observe que 5 / 2 tem como resultado o número 2, ou seja, o maior inteiro que pode ser obtido dentro do resultado matemático – que, neste caso, seria 2,5.

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:



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**.



Binários

Possuem dois operandos.



Ternários

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
Soma	+	1 + 2
Subtração	-	3 - 4
Multiplicação	*	5 * 6
Divisão inteira	/	5 / 2
Resto da divisão	%	5 % 2
Incremento	++	2++
		++2
Decremento	--	2--
		--2



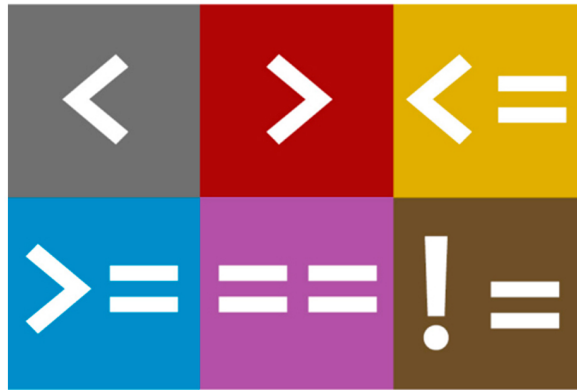
Tabela: Resumo 3.
Anderson Fernandes Pereira dos Santos.

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 '>'.< /p>

Menor ou igual

Combinação dos símbolos '<='.< /p>

Maior ou igual

Combinação dos símbolos '>='.< /p>

Igualdade

Combinação dos símbolos '=='.< /p>

Desigualdade

Combinação dos símbolos '!='.< /p>

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
Maior	>	1.2 > 3.4
Menor	<	1.2 < 3.4
Menor ou igual	<=	1.2 <= 3.4
Maior ou igual	>=	1.2 >= 3.4
Igualdade	==	1.2 == 3.4
Desigualdade	!=	1.2 != 3.4



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
Negação	!	!0
Operador E	&&	1 && 0
Operador OU		1 0

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
E lógico	a & b	2 & 6
OU lógico	a b	2 4
OU Exclusivo	a ^ b	2 ^ 6
Deslocamento à direita	a >> b	4 >> 2
Deslocamento à esquerda	a << b	2 << 4

Negação	$\sim a$	~ 2
---------	----------	----------

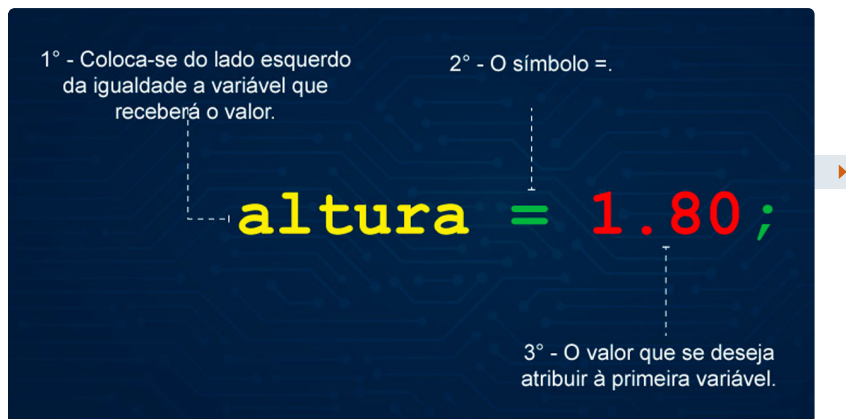
Tabela: Resumo 6.

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:

$$IMC = \frac{\text{peso}}{\text{altura} \times \text{altura}}$$

Como representamos essa expressão matemática na linguagem C?

LINGUAGEM C



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?

```
SaldoBancario = SaldoBancario + 100;
```

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



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;

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
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



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

Desse modo, teríamos:

LINGUAGEM C



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 C



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



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:

$$\begin{aligned} &a + b \% c \\ &1 + 2 \% 3 \\ &\quad 1 + 2 \\ &\quad \quad 3 \end{aligned}$$

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

Por meio de exemplos, confira neste vídeo o entendimento sobre os conceitos de precedência de operadores e de tabela verdade.

Para assistir a um vídeo sobre o assunto, acesse a versão online deste conteúdo.



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 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	$a \wedge b$
verdadeiro	verdadeiro	falso
verdadeiro	falso	verdadeiro
falso	verdadeiro	verdadeiro
falso	falso	falso



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

a	$\sim a$
verdadeiro	falso
falso	verdadeiro

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).

CERN

Sigla para a Organização Europeia de Pesquisa Nuclear, um dos maiores e mais respeitados centros de pesquisa científica do mundo.



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.

Falta pouco para atingir seus objetivos.


Vamos praticar alguns conceitos?

Questão 1

(FCC - TRF - 4ª Região - Técnico Judiciário - Tecnologia da Informação - 2014) O tipo booleano é um tipo de dado utilizado na programação de computadores. Em operações lógicas, o resultado será sempre um valor boolean TRUE ou FALSE.

Muitas vezes, tais operações são apresentadas em uma tabela conhecida como tabela verdade. Observe este exemplo:

A	B	A E B
TRUE	FALSE	
FALSE	TRUE	III



As lacunas I, II ou III são preenchidas, correta e respectivamente, por:

- A *True, true e false.*
- B *True, false e false.*
- C *False, true e true.*
- D *True, true e true.*
- E *True, false, true.*

Parabéns! A alternativa A está correta.

Os valores calculados derivam diretamente das tabelas verdade apresentadas anteriormente:

a	b	a b
verdadeiro	verdadeiro	verdade
verdadeiro	falso	verdade
falso	verdadeiro	verdade
falso	falso	falso



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



Questão 2

(Adaptado de: NUCEPE - SEDUC-PI - Professor de Informática - 2009) Assinale a alternativa que mostra o operador lógico OU em linguagem C:

A \$\$

B ||

C &&

D Or

E //

Parabéns! A alternativa B está correta.

Os símbolos das letras A, D e E não pertencem à linguagem C. O símbolo da letra C é o e-lógico.

Considerações finais

Versamos sobre quatro tipos de dados primitivos utilizados na linguagem C: char, que representa um caractere; int, um número inteiro; float e double, que representam os números ponto flutuante de precisão simples e dupla. Além de descrevermos suas características e funcionalidades, falamos sobre a precedência deles.

Apresentamos ainda seis tipos de operação que trabalham com esses operadores. São as operações matemáticas, relacionais, lógicas, bit a bit, de conversão e de atribuição. Por fim, estabelecemos os conceitos de tabela verdade.



Podcast

Agora, ouça um resumo dos principais tópicos abordados.

Para ouvir o *áudio*, acesse a versão online deste conteúdo.



Explore +

Para explorar mais os conceitos da álgebra booleana e dos circuitos lógicos, sugerimos que assista ao seguinte filme:

O jogo da imitação, dirigido por Morten Tyldum em 2014.

A obra retrata o desenvolvimento de uma tecnologia capaz de decifrar os códigos da máquina alemã, um grande enigma durante a Segunda Guerra Mundial. Projetada a partir da combinação de circuitos lógicos, a Máquina de Turing foi desenvolvida pelo matemático e cientista da computação Alan Turing (1912-1954).

Referências

DAMAS, L. **Linguagem C**. 10. ed. Rio de Janeiro: LTC, 2006.

SCHILD, H. **C completo e total**. 3. ed. São Paulo: Makron Books, 1996.

Material para download

Clique no botão abaixo para fazer o download do conteúdo completo em formato PDF.

Download material

O que você achou do conteúdo?



 Relatar problema