DEFINIÇÃO

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

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

MÓDULO 2

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

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

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.

Fonte: Wikpedia

FORDISMO

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

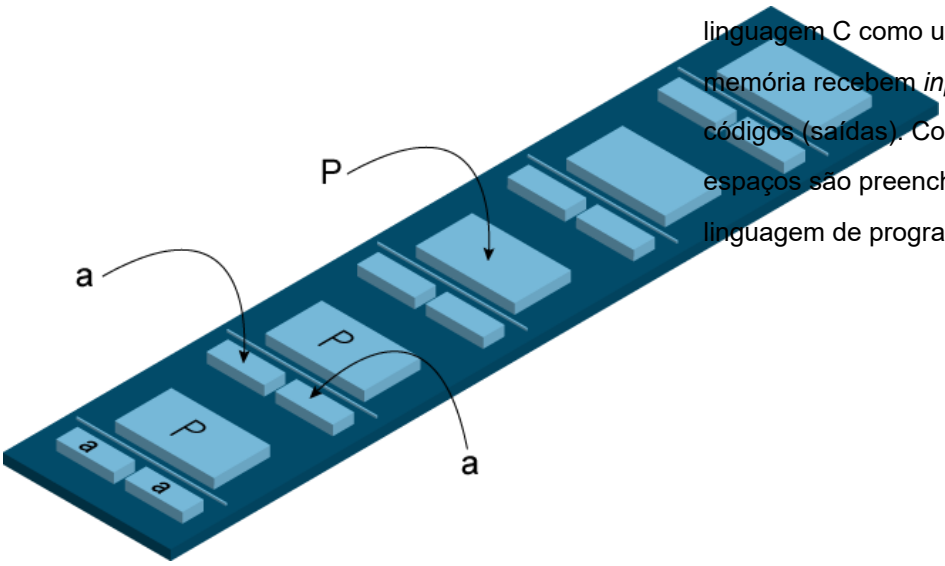


Imagem: Shutterstock.com

Veja 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.

EXEMPLO

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:

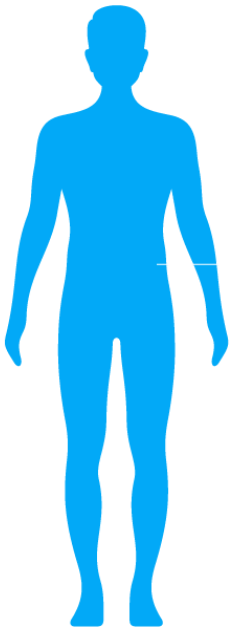


Imagem: Shutterstock.com

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

Fonte: Shutterstock

Para desenvolver um algoritmo, precisamos:

1º

Identificar quais dados de entrada serão utilizados e como representá-los em nossa linguagem de programação.

2º

Fazer, com esses dados já identificados, as transformações necessárias para modificar ou realizar cálculos.

Imagem: Shutterstock.com

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...

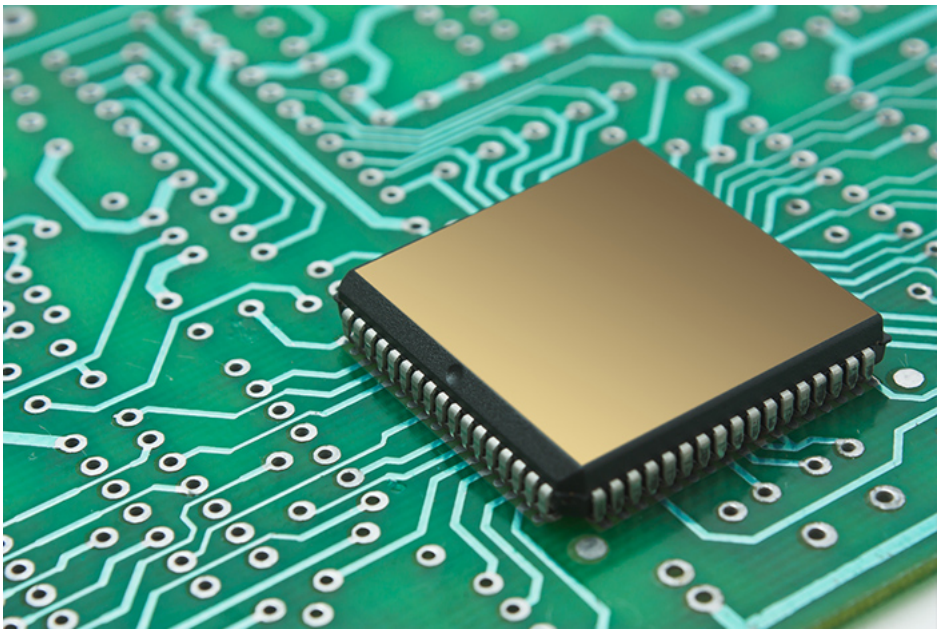


Imagem: Shutterstock.com

... 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: CHAR, INT, FLOAT E 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** com seus 256 símbolos. Observe-a a seguir:

ASCII

Sigla para American Standard Code for Information Interchange.

ASCII control characters

00	NULL	(Null character)
01	SOH	(Start of Header)
02	STX	(Start of Text)
03	ETX	(End of Text)
04	EOT	(End of Trans.)
05	ENQ	(Enquiry)
06	ACK	(Acknowledgement)
07	BEL	(Bell)
08	BS	(Backspace)
09	HT	(Horizontal Tab)
10	LF	(Line feed)
11	VT	(Vertical Tab)
12	FF	(Form feed)
13	CR	(Carriage return)
14	SO	(Shift Out)
15	SI	(Shift In)
16	DLE	(Data link escape)
17	DC1	(Device control 1)
18	DC2	(Device control 2)
19	DC3	(Device control 3)
20	DC4	(Device control 4)
21	NAK	(Negative acknowl.)
22	SYN	(Synchronous idle)
23	ETB	(End of trans. block)
24	CAN	(Cancel)
25	EM	(End of medium)
26	SUB	(Substitute)
27	ESC	(Escape)
28	FS	(File separator)
29	GS	(Group separator)
30	RS	(Record separator)
31	US	(Unit separator)
127	DEL	(Delete)

ASCII printable characters

32	space	64	@	96	`
33	!	65	A	97	a
34	"	66	B	98	b
35	#	67	C	99	c
36	$	68	D	100	d
37	%	69	E	101	e
38	&	70	F	102	f
39	'	71	G	103	g
40	(	72	H	104	h
41	)	73	I	105	i
42	*	74	J	106	j
43	+	75	K	107	k
44	,	76	L	108	l
45	-	77	M	109	m
46	.	78	N	110	n
47	/	79	O	111	o
48	0	80	P	112	p
49	1	81	Q	113	q
50	2	82	R	114	r
51	3	83	S	115	s
52	4	84	T	116	t
53	5	85	U	117	u
54	6	86	V	118	v
55	7	87	W	119	w
56	8	88	X	120	x
57	9	89	Y	121	y
58	:	90	Z	122	z
59	;	91	[	123	{
60	<	92	\	124	|
61	=	93	]	125	}
62	>	94	^	126	~
63	?	95	_		

Extended ASCII characters

128	Ç	160	á	192	└	224	Ó
129	ü	161	í	193	┴	225	ß
130	é	162	ó	194	┬	226	Ô
131	â	163	ú	195	├	227	Ò
132	ä	164	ñ	196	─	228	õ
133	à	165	Ñ	197	┼	229	Õ
134	å	166	ª	198	ã	230	µ
135	ç	167	º	199	Ã	231	þ
136	ê	168	¿	200	╚	232	Þ
137	ë	169	®	201	╔	233	Ú
138	è	170	¬	202	╩	234	Û
139	ï	171	½	203	╦	235	Ù
140	î	172	¼	204	╠	236	ý
141	ì	173	¡	205	═	237	Ý
142	Ä	174	«	206	╬	238	¯
143	Å	175	»	207	¤	239	´
144	É	176	░	208	ð	240	≡
145	æ	177	▒	209	Ð	241	±
146	Æ	178	▓	210	Ê	242	‗
147	ô	179	│	211	Ë	243	¾
148	ö	180	┤	212	È	244	¶
149	ò	181	Á	213	ı	245	§
150	û	182	Â	214	Í	246	÷
151	ù	183	À	215	Î	247	¸
152	ÿ	184	©	216	Ï	248	°
153	Ö	185	╣	217	┘	249	¨
154	Ü	186	║	218	┌	250	·
155	ø	187	╗	219	█	251	¹
156	£	188	╝	220	▄	252	³
157	Ø	189	¢	221	¦	253	²
158	×	190	¥	222	Ì	254	■
159	ƒ	191	┐	223	▀	255	nbsp

Imagem: computersciencewiki.org

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)

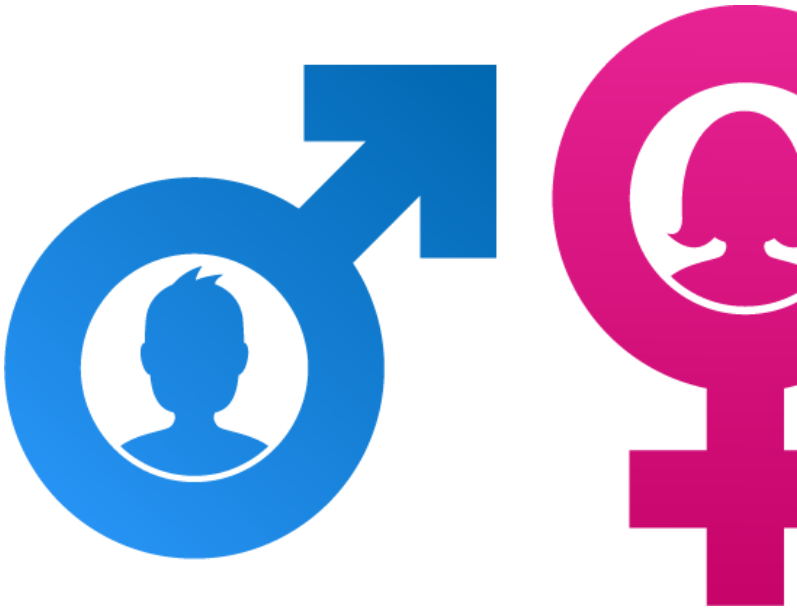
DO 128 AO 255

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’.

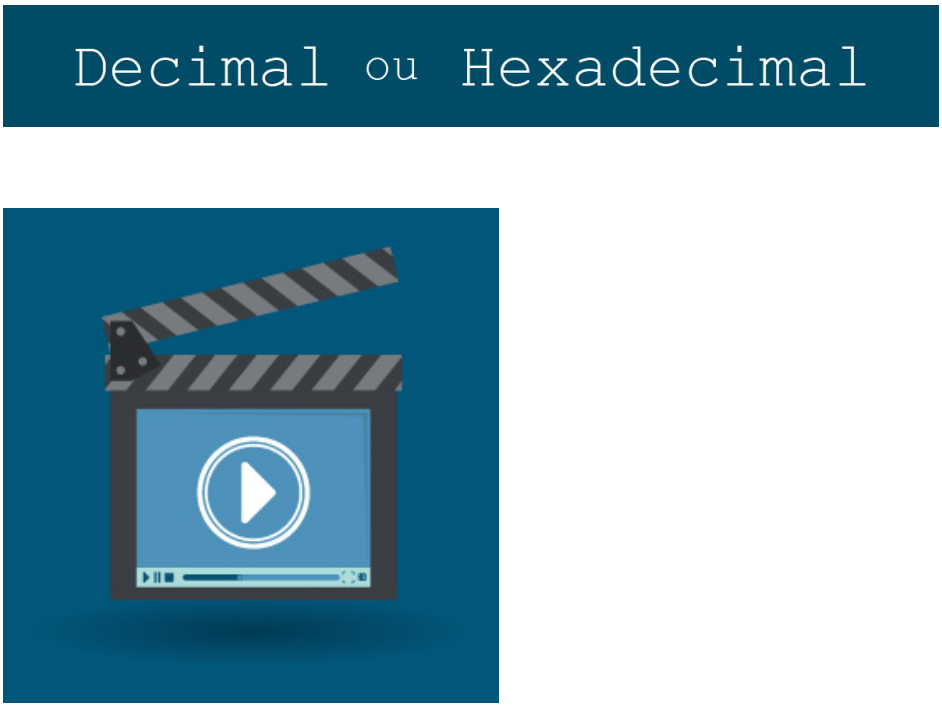


Fonte: Shutterstock

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

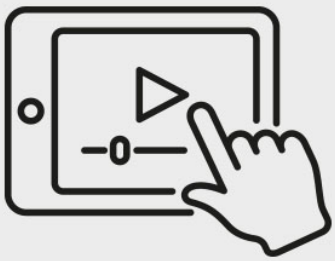
Assim, conforme pode ser visto, o ‘m’ (**m** minúsculo) é diferente de ‘M’ (**M** maiúsculo).

Notemos também que, para cada caractere da tabela, existe um índice representado em:



Neste vídeo, o professor **Humberto Henriques** discorre sobre 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.



MANIPULAÇÃO DE VARIÁVEIS E CONSTANTES

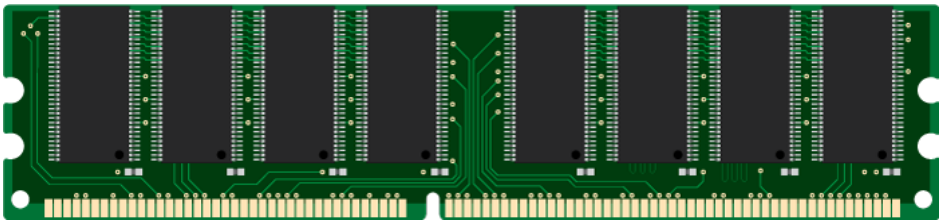
Já sabemos como representar os dados do mundo real na linguagem de programação C. Agora precisamos entender como eles podem ser manipulados. Para fazer isso, a linguagem trata os dados como variáveis e constantes.

1. CONCEITO

A **variável** é um tipo de espaço de memória que pode ser alterado a qualquer tempo.

≠

A **constante**, por sua vez, não pode.



Fonte: Shutterstock

As duas formas permitem a referenciação 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.

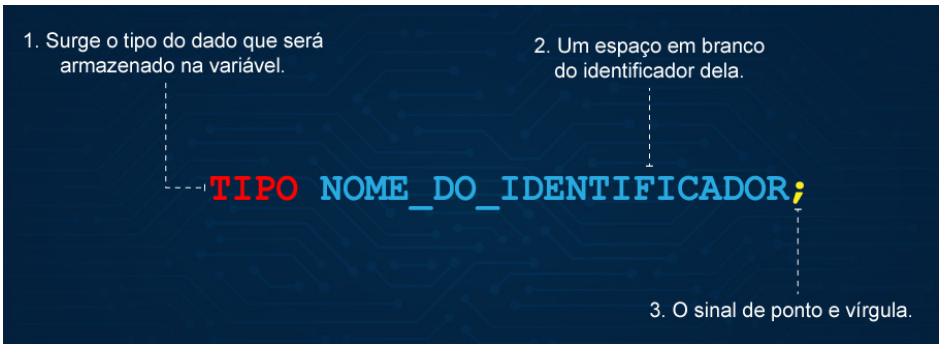
Fonte: Shutterstock

2. DEFINIÇÃO DAS VARIÁVEIS

Formalmente, um espaço de memória é rotulado por intermédio de um identificador quando as variáveis são definidas.

Fonte: Shutterstock

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?

FLOAT PESO;

FLOAT ALTURA;

Já sabemos que a linguagem C é considerada sensível a um **contexto**. Assim, ao escrevermos uma aplicação nessa linguagem, os identificadores...

CONTEXTO

A grafia que usa maiúsculas e minúsculas é diferente.

peso - Peso - PESO

... Serão diferentes.

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**.

FLOAT OU FLOAT

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:

TIPO NOME_DO_IDENTIFICADOR_1, NOME_DO_IDENTIFICADOR_2;

COMO ESTARIA DESCRITA, PORTANTO, A REPRESENTAÇÃO DOS VALORES DE PESO E ALTURA?

FLOAT PESO, ALTURA;

OU

FLOAT ALTURA, PESO;

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:

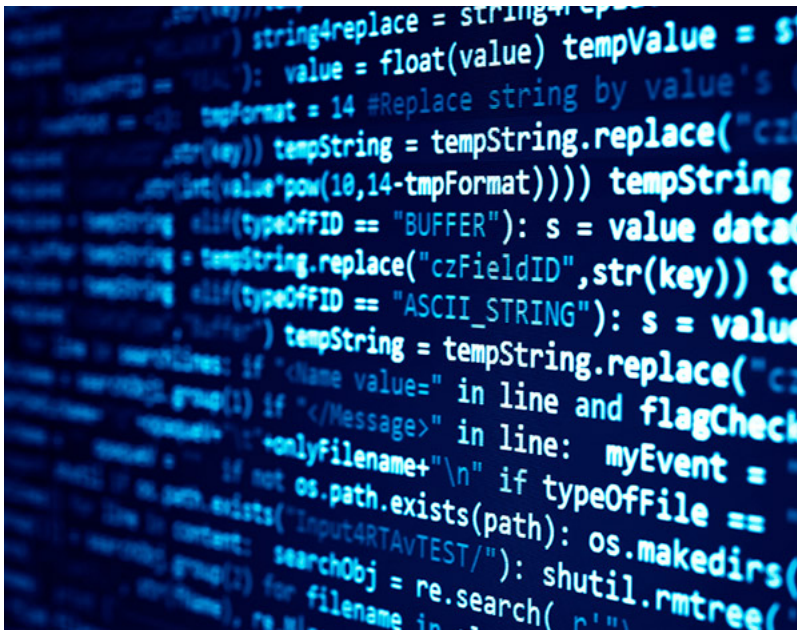
FLOAT ALTURA, PESO;

// VALORES DE ALTURA E PESO DO USUÁRIO,

// MEDIDOS EM METROS E QUILOGRAMA,

// RESPECTIVAMENTE.

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.



Fonte: Shutterstock

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

LETRAS DO ALFABETO (MAIÚSCULAS E MINÚSCULAS)

DÍGITOS (0-9)

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;

APLICAÇÃO DOS CONCEITOS APRESENTADOS

Da teoria da informação, advêm 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.

Fonte: Shutterstock

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.

VEM QUE EU TE EXPLICO!

Os vídeos a seguir abordam os assuntos mais relevantes do conteúdo que você acabou de estudar.

Tipo Char

Constantes e Variáveis

Questão 4 / Variáveis

VERIFICANDO O APRENDIZADO

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

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

3. (FUNIVERSA - PC-DF - PERITO CRIMINAL - INFORMÁTICA - 2012) SÃO PALAVRAS-CHAVE DA LINGUAGEM C NO PADRÃO ANSI, NÃO PODENDO, PORTANTO, SER UTILIZADAS COMO NOMES PARA VARIÁVEIS:

A) typedef, master, core, newline

B) union, extern, main, core

C) int, long, static, void

D) signed, unsigned, master, main

4. EM UMA APLICAÇÃO PARA O CÁLCULO DO IMC, FORAM DEFINIDAS DUAS VARIÁVEIS: PESO E ALTURA. DEVIDO A REQUISITOS ORIUNDOS DO MUNDO REAL, ELAS SÃO DEFINIDAS COMO FLOAT. ASSINALE A ALTERNATIVA QUE APRESENTA A FORMA INCORRETA DE DEFINIÇÃO DESSAS VARIÁVEIS:

A) float peso; float altura;

B) float peso, altura;

C) float altura, peso;

D) float peso, float altura;

5. (ADAPTADA DE: CESPE - BANCO DA AMAZÔNIA - TÉCNICO-CIENTÍFICO - TECNOLOGIA DA INFORMAÇÃO - ADMINISTRAÇÃO DE DADOS - 2010) ACERCA DAS ESTRUTURAS DE

INFORMAÇÃO, CONSIDERE A SEGUINTE AFIRMAÇÃO: NOS TIPOS PRIMITIVOS DE DADOS DO TIPO INTEIRO, OS VALORES SÃO NÚMEROS INTEIROS PARA OS QUAIS SÃO DEFINIDAS OPERAÇÕES MATEMÁTICAS. ESSES TIPOS DE DADOS NÃO POSSUEM CASAS DECIMAIS.

NA IMPLEMENTAÇÃO DE UM APLICATIVO, PODEMOS REPRESENTAR A IDEIA APRESENTADA POR MEIO DO SEGUINTE TIPO INTEIRO:

A) A idade

B) A cor dos olhos

C) O peso

D) O endereço

GABARITO

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

A alternativa **"D "** está correta.

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

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

A alternativa **"B "** está correta.

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

3. (FUNIVERSA - PC-DF - Perito Criminal - Informática - 2012) São palavras-chave da linguagem C no padrão ANSI, não podendo, portanto, ser utilizadas como nomes para variáveis:

A alternativa **"C "** está correta.

Na letra A, são apresentados os termos master, core e newline, que não pertencem à linguagem C. Nas letras B e D, os termos core e master se repetem.

4. Em uma aplicação para o cálculo do IMC, foram definidas duas variáveis: peso e altura. Devido a requisitos oriundos do mundo real, elas são definidas como float. Assinale a alternativa que apresenta a forma incorreta de definição dessas variáveis:

A alternativa **"D "** está correta.

Depois de um identificador, só é possível haver um sinal de ponto e vírgula ou uma vírgula seguida de outro identificador.

5. (Adaptada de: CESPE - Banco da Amazônia - Técnico-científico - Tecnologia da Informação - Administração de Dados - 2010) Acerca das estruturas de informação, considere a seguinte afirmação: nos tipos primitivos de dados do tipo inteiro, os valores são números inteiros para os quais são definidas operações matemáticas. Esses tipos de dados não possuem casas decimais.

Na implementação de um aplicativo, podemos representar a ideia apresentada por meio do seguinte tipo inteiro:

A alternativa **"A "** está correta.

Exemplos de idades de pessoas são 20 ou 50 anos, ou seja, números sem casas decimais. Portanto, utilizamos o tipo primitivo inteiro. A cor dos olhos é medida em cadeia de caracteres (char), como, por exemplo azul, verde e castanho-claro. O peso, por sua vez, deve ser medido em quilogramas (ou libras) com um número real (tipo primitivo float ou double), como 70,0 Kg. Já o endereço é representado como uma cadeia de caracteres (char), como Av. Paulista, 100.

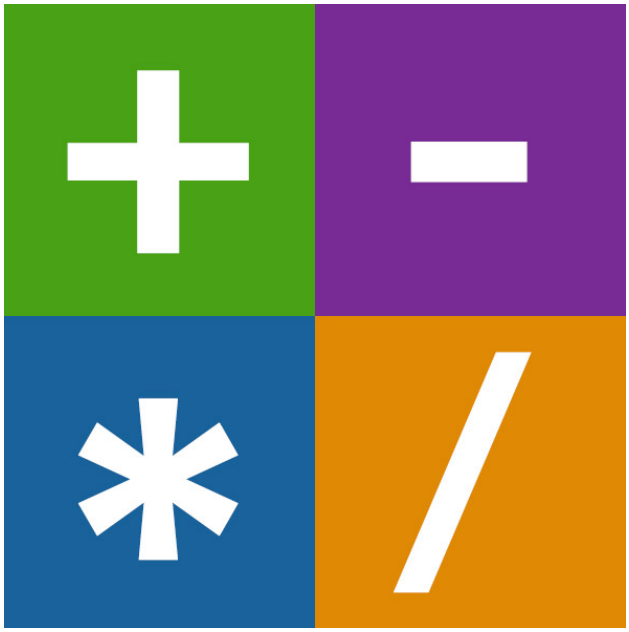
- Aplicar os operadores matemáticos, lógicos, relacionais e de atribuição, além dos conceitos de tabela verdade

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.

1. 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.



NÚMEROS REAIS (R)

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

A) OPERACIONALIDADE

São ofertadas pela linguagem as seguintes operações:

SOMA

Representada pelo símbolo '**+**'

SUBTRAÇÃO

Símbolo '**-**'

MULTIPLICAÇÃO

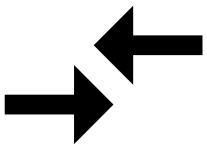
Representada pelo símbolo ‘*’

DIVISÃO

Símbolo ‘/’

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

No mundo real, os números podem possuir representação infinita.



No computador, isso não é possível, já que, em tal 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

Operação matemática	Símbolo utilizado	Exemplo	
		Equação	Resultado
Multiplicação	*	1.2 * 3.4	4.08
Divisão	/	1.2 / 3.4	-0.3529

Tabela resumo 1.

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.

Fonte: Shutterstock

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.

ESSA OPERAÇÃO, PORTANTO, NÃO ESTÁ DEFINIDA PARA NÚMEROS REAIS.

Observemos mais um resumo do que aprendemos.

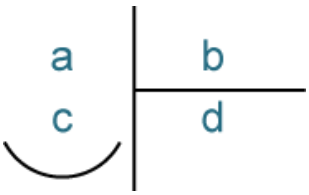


Figura: Resumo 2.

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

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

Têm dois operandos.

TERNÁRIOS

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	

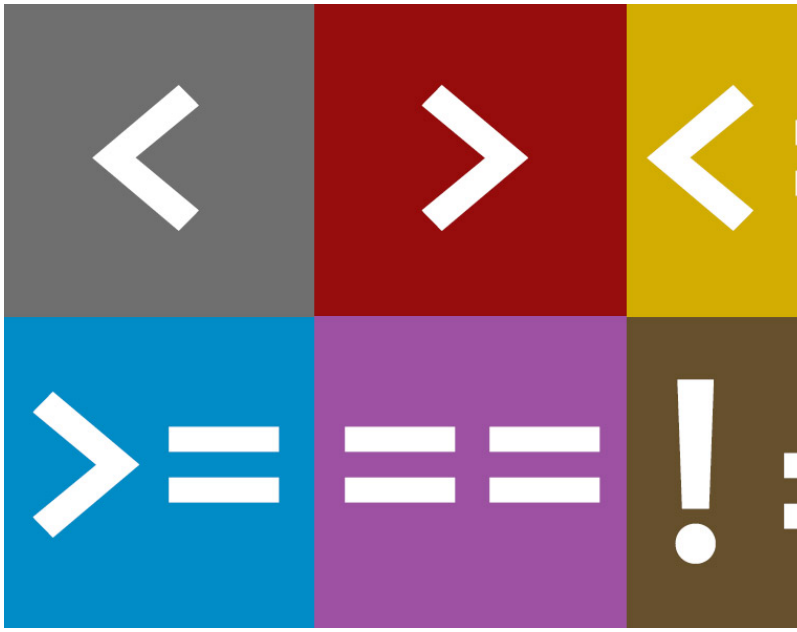
Tabela: Resumo 3.

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.

2. OPERADORES

RELACIONAIS

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



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 '<'

MAIOR

Símbolo '>'

MENOR OU IGUAL

Combinação dos símbolos '<='

MAIOR OU IGUAL:

Combinação dos símbolos '>='

IGUALDADE

Combinação dos símbolos '=='

DESIGUALDADE

Combinação dos símbolos '!='

EXEMPLO:

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

Fonte: Shutterstock

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

ATENÇÃO

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.

3. 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.

⊕ ATENÇÃO

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:

0: Caso a variável seja numérica, ou seja, int, float ou double;

null: Se for uma variável que armazene algum endereço de memória;

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.

4. 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

Operação	Expressão	Exemplo	
		Equação	Resultado
Deslocamento à esquerda	a >> b	4 >> 2	1
Deslocamento à direita	a << b	2 << 4	32
Negação	~a	~2	-3

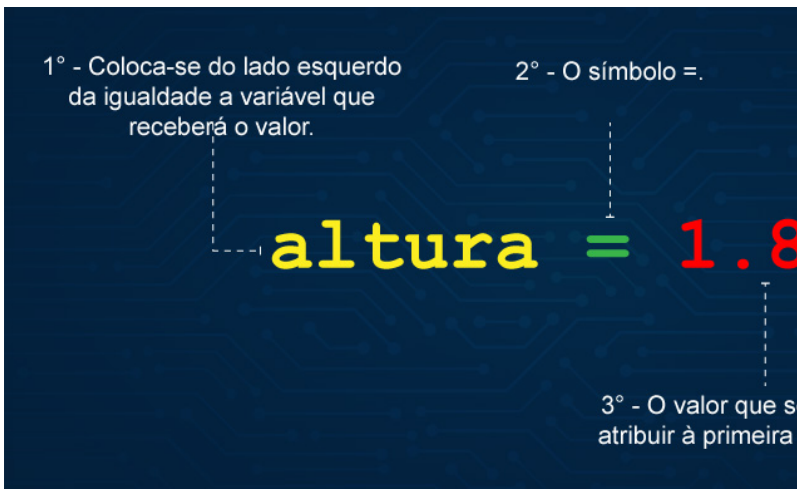
Tabela: Resumo 6.

5. 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**.

EXEMPLO:

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:

$$IMC = \frac{peso}{altura\ x\ altura}$$

Como representamos essa expressão matemática na linguagem C?

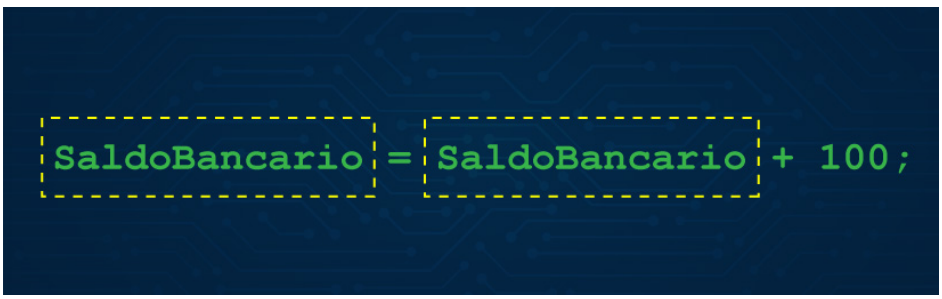
IMC = peso / (altura * altura);

NESTE CASO, AS OPERAÇÕES SÃO REALIZADAS DO LADO DIREITO DA EXPRESSÃO, ENQUANTO SEU RESULTADO É ARMAZENADO NA VARIÁVEL IMC.

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?



Neste 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:

SaldoBancario += 100;

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

Operação	Forma resumida
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
a=a<<b;	a<<=b;
a=a>>b;	a>>=b;

6. 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

COM PERDA DE INFORMAÇÃO

A) 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:

idade = 20;

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

Desse modo, teríamos:

idade_real = (float) idade;

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

B) COM PERDA DE INFORMAÇÃO

Converte 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:

float pi = 3.1415;

Se quisermos converter este 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 que:

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

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

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

TABELA VERDADE

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 a variável float pi com valor 3,1415:

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

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

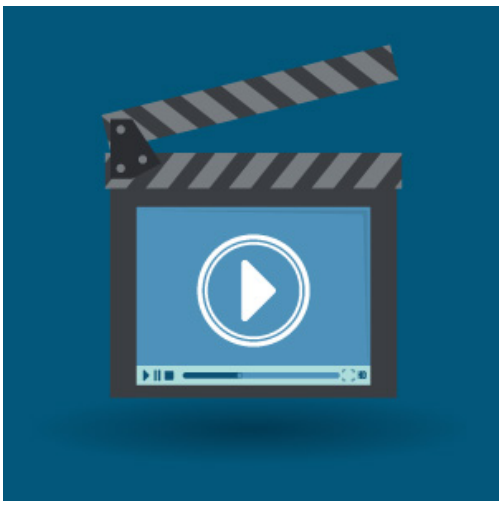
a	b	a||b

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

a	b	a → b
verdadeiro	verdadeiro	verdadeiro
verdadeiro	falso	falso
falso	verdadeiro	verdadeiro
falso	falso	verdadeiro

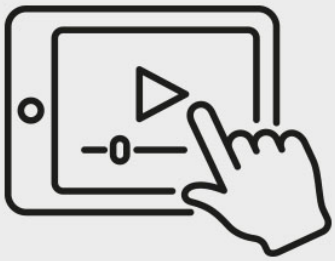
a	b	a ^ b
verdadeiro	verdadeiro	falso
verdadeiro	falso	verdadeiro
falso	verdadeiro	verdadeiro
falso	falso	falso

a	~a
verdadeiro	falso
falso	verdadeiro



Neste vídeo, o professor **Humberto Henriques** reforça, por meio de exemplos, 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.



APLICAÇÃO DOS CONCEITOS APRESENTADOS

Imagem: Shutterstock.com

Desde o final da década de 1990, a internet dominou todos os campos de nossa vida. Hoje em dia, 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).

Imagem: Shutterstock.com

Com a finalidade de apresentar uma forma mais dinâmica de divulgação dos dados, Berns-Lee desenvolveu o **protocolo HTTP**, 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.

CERN

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

PROTOCOLO HTTP

Ele permite o acesso a *sites* na 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.

VEM QUE EU TE EXPLICO!

Os vídeos a seguir abordam os assuntos mais relevantes do conteúdo que você acabou de estudar.

Operadores Relacionais

Precedências de Operadores

Tabela Verdade

VERIFICANDO O APRENDIZADO

1. SEJAM AS VARIÁVEIS *A* , *B* E *C* DO TIPO INT (INTEIRO). CONSIDERANDO AS LINHAS DE CÓDIGO NA LINGUAGEM C ABAIXO:

A = 3, B = 2, C = 2;

C++;

C += !A >= B && C != B;

ASSINALE A ALTERNATIVA QUE CORRESPONDA AO VALOR DA VARIÁVEL *C* AO FINAL DA EXECUÇÃO:

A) 4.

B) 3.

C) 2.

D) 0.

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

3. (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	A OU B	NÃO (A E B)
TRUE	*FALSE*		I	II
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*

4. (FUNDATEC - PREFEITURA DE CHUÍ - RS - FISCAL DE TRIBUTOS - 2019) OBSERVE A SEGUINTE TABELA VERDADE:

P	Q	R	P^Q	(P^Q)⇒R
VERDADEIRO	VERDADEIRO	VERDADEIRO	VERDADEIRO	VERDADEIRO
(1)	VERDADEIRO	FALSO	FALSO	(2)

OS VALORES LÓGICOS QUE PREENCHEM (1) E (2) SÃO, RESPECTIVAMENTE:

A) Falso - Verdadeiro

B) Verdadeiro - Verdadeiro

C) Verdadeiro - Falso

D) Falso - Falso

5. CONSIDERE O SEGUINTE SEGMENTO DE CÓDIGO NA LINGUAGEM C:

```
INT A=5, B=2;
FLOAT C=7, D=3;
INT E, F;
FLOAT G, H;
E=A/B;
F=C/D;
G=A/B;
```

H=C/D;

ASSINALE A ALTERNATIVA QUE APRESENTA OS VALORES DAS VARIÁVEIS TÉRMINO DA EXECUÇÃO:

A) 5, 2, 7.0, 3.0, 2, 2, 2.0, 2.3

B) 5, 2, 7.0, 3.0, 3, 4, 2.0, 2.3

C) 5, 2, 7.0, 3.0, 2, 2, 2.5, 2.3

D) 5, 2, 7.0, 3.0, 3, 4, 2.5, 2.3

6. (UFTM - ENGENHARIA DA COMPUTAÇÃO OU ENGENHARIA DA PRODUÇÃO - 2018) APONTE, ENTRE AS ALTERNATIVAS ABAIXO, OS RESULTADOS DA RESOLUÇÃO DA SEGUINTE EXPRESSÃO LÓGICA (ESCRITA NA LINGUAGEM C) PARA OS VALORES DE A, B E C DEFINIDOS NOS CENÁRIOS I, II E III:

(A && B) && ((C || A || B) || (!A && C))

I: A=*TRUE*, B=*TRUE*, C=*FALSE*
II: A=*FALSE*, B=*TRUE*, C=*TRUE*
III: A=*FALSE*, B=*TRUE*, C=*FALSE*

A) I: *true*, II: *false*, III: *false*

B) I: *true*, II: *true*, III: *false*

C) I: *false*, II: *false*, III: *false*

D) I: *false*, II: *true*, III: *false*

GABARITO

1. Sejam as variáveis *a* , *b* e *c* do tipo int (inteiro). Considerando as linhas de código na linguagem C abaixo:

```
a = 3, b = 2, c = 2;

c++;
```

c += !a >= b && c != b;

Assinale a alternativa que corresponda ao valor da variável *c* ao final da execução:

A alternativa **"B "** está correta.

Vamos analisar a execução das linhas de código:

Na primeira linha as variáveis ***a*** **,** ***b*** **,** ***c*** (todas do tipo **int**) são inicializadas com os valores **3**, **2** e **2**, respectivamente;

Na segunda linha a variável *c* **é incrementada em 1 unidade**. Logo, ***c*** **= 3** e as demais variáveis permanecem inalteradas (***a*** **= 3** e ***b*** **= 2**);

Na terceira linha temos a expressão lógica ***!a >= b && c != b*** sendo somada ao valor de ***c*** e atribuída à própria variável ***c*** ;

Do lado esquerdo da expressão temos a verificação se a **negação de *a (!a)* é maior ou igual a *b*** . Como o valor de ***a*** é diferente de zero, portanto considerado na linguagem C como um valor **verdadeiro**, a sua negação será considerada como **falso**, ou seja, **0 (zero)**. E como ***b*** **tem valor igual a 2**, o resultado de ***!a >= b*** **(0 >= 2)** será **falso (F)**;

Do lado direito da expressão temos a verificação se a variável ***c*** **é diferente de *b*** . Após ter sido incrementada, ***c*** **= 3**. E como ***b*** **= 2,** ***c != b*** será **verdadeiro (V)**;

Substituindo na expressão ***!a <= b && c != b*** , temos **F && V**, ou seja, **falso E verdadeiro**, o que resulta em **falso**;

Resta atribuir o resultado da expressão lógica à variável ***c*** .

Como ***c*** **é do tipo inteiro**, o valor **falso** será representado pela linguagem C como **0 (zero)**;

Substituindo em ***c += !a >= b && c != b*** , temos c += 0, que é o mesmo que c = c + 0.

Assim, c = 3 + 0 = 3.

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

A alternativa **"B "** está correta.

Os símbolos das letras A e D não pertencem à linguagem C. O símbolo de C é o e-lógico.

3. (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	A OU B	NÃO (A E B)
TRUE	***FALSE***		**I**	**II**
FALSE	***TRUE***	**III**		

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

A alternativa **"A "** está correta.

Os valores calculados derivam diretamente das tabelas verdade apresentadas anteriormente:

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

a	b	a && b	!(a && b)
verdadeiro	verdadeiro	verdadeiro	falso
verdadeiro	falso	falso	verdadeiro

a	b	a && b	!(a && b)
falso	verdadeiro	falso	verdadeiro
falso	falso	falso	verdadeiro

4. (FUNDATEC - Prefeitura de Chuí - RS - Fiscal de Tributos - 2019) Observe a seguinte tabela verdade:

P	Q	R	P^Q	(P^Q)⇒R
verdadeiro	**verdadeiro**	**verdadeiro**	**verdadeiro**	**verdadeiro**
(1)	**verdadeiro**	**falso**	**falso**	**(2)**

Os valores lógicos que preenchem (1) e (2) são, respectivamente:

A alternativa **"A "** está correta.

Os casos (1) e (2) são obtidos na verificação dos valores de P ^ Q nas tabelas verdade já apresentadas.

5. Considere o seguinte segmento de código na linguagem C:

```
int a=5, b=2;
float c=7, d=3;
int e, f;
float g, h;
e=a/b;
f=c/d;
g=a/b;
h=c/d;
```

Assinale a alternativa que apresenta os valores das variáveis término da execução:

A alternativa **"A "** está correta.

Vemos que a/b é uma divisão inteira. Assim, seu resultado é inteiro; portanto, a/b=2. Quando esse valor é atribuído ao inteiro e, tal inteiro apenas recebe seu valor: e=2. Quando essa atribuição é feita a um número ponto flutuante, ele é convertido; logo, g=2.0.

Observamos que c/d é uma divisão de números ponto flutuante sendo atribuída a um número inteiro, ou seja, depois da divisão com ponto flutuante (2.33333). Quando atribuído a um número inteiro, ele é convertido: f=2; já para um ponto flutuante, não há conversão: h=2.3.

6. (UFTM - Engenharia da Computação ou Engenharia da Produção - 2018) Aponte, entre as alternativas abaixo, os resultados da resolução da seguinte expressão lógica (escrita na linguagem C) para os valores de A, B e C definidos nos cenários I, II e III:

(A && B) && ((C || A || B) || (!A && C))

I: A=*true*, B=*true*, C=*false*
II: A=*false*, B=*true*, C=*true*
III: A=*false*, B=*true*, C=*false*

A alternativa **"A "** está correta.

Devemos inicialmente montar a tabela verdade:

A	B	C	!A	C=A&&B	D=C||A||B	E=!A&&C	D||E	C&&F
1	1	1	0	1	1	0	1	1
1	1	0	0	1	1	0	1	1
1	0	1	0	0	1	0	1	0
1	0	0	0	0	1	0	1	0
0	1	1	1	0	1	1	1	0
0	1	0	1	0	1	0	1	0
0	0	1	1	0	1	1	1	0

A	B	C	!A	C=A&&B	D=C||A||B	E=!A&&C	D||E	C&&F
0	0	0	1	0	0	0	0	0

Basta agora confrontar os valores da tabela verdade com os dados do enunciado apresentados nos itens I, II e III para encontrar o valor da expressão e os valores das variáveis mais adequados.

CONCLUSÃO

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. Trata-se das operações matemáticas, relacionais, lógicas, *bit* a *bit*, de conversão e de atribuição. Por fim, estabelecemos os conceitos de tabela verdade.

Para ouvir um *podcast* sobre o assunto, acesse a versão online deste conteúdo.

AVALIAÇÃO DO TEMA:

REFERÊNCIAS

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

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

EXPLORE+

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

O jogo da imitação. Direção: Morten Tyldum. Estados Unidos: Diamond Films, 2014. 115 min, son., color.

A obra retrata o desenvolvimento de uma tecnologia capaz de decifrar os códigos da máquina alemã 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).

CONTEUDISTA

Anderson Fernandes Pereira dos Santos

🔗 CURRÍCULO LATTES