The background of the slide is a dark, blue-toned image of an industrial setting, likely a factory or laboratory. Several large, complex robotic arms are visible, some with multiple joints and sensors. A semi-transparent world map is overlaid on the left side of the image. The text is centered and rendered in a clean, white, sans-serif font.

Paradigmas de Linguagens

Aula 02

BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

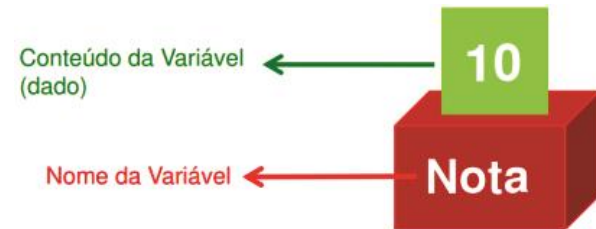
Variável

Uma variável de programa é uma abstração de uma célula de memória de um computador ou de uma coleção de células.

Os programadores geralmente pensam em variáveis como nomes para locais de memória, mas existe muito mais acerca de uma variável do que apenas um nome.

Uma variável pode ser caracterizada como um conjunto de seis atributos (**nome**, endereço, **valor**, **tipo**, tempo de vida, escopo).

float pi =3.14; //float é tipo de dado, pi é o nome e 3.14 é o conteúdo



```
const Pi float64 = 3.14
```

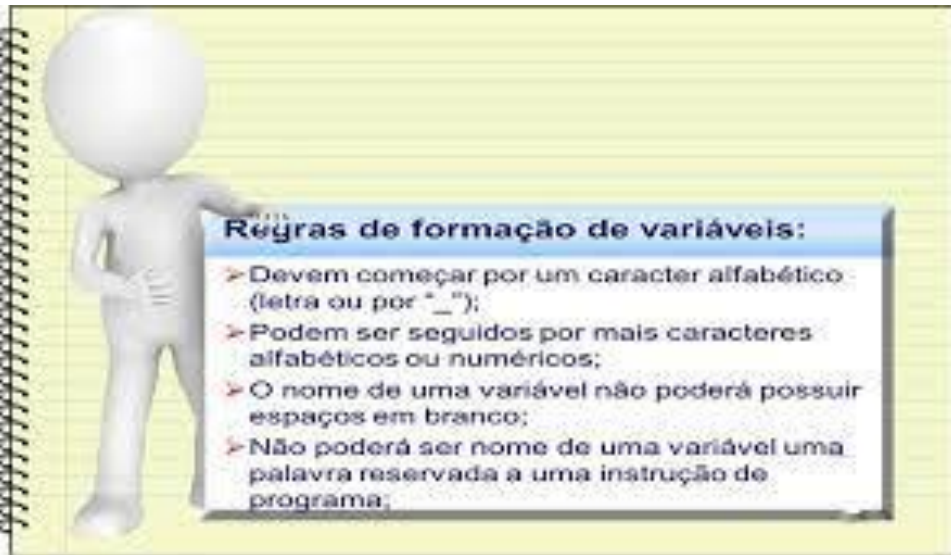
```
graph TD; A[const] --> B[declare]; A[Pi] --> C[name]; A[float64] --> D[type]; A["= 3.14"] --> E[value];
```

declare

name

type

value
64-bit precision



Nomes de variáveis

Um nome é uma cadeia de caracteres usada para identificar alguma entidade em um programa. O termo identificador é muito usado como sinônimo de nome.

Algumas linguagens impõem limite no tamanho do identificador, Fortran 95 por exemplo permite até 31 caracteres.

Outras linguagens não têm restrição de tamanho, como é o caso do Java e do C#, porém temos ainda algumas linguagens que embora não tenham tal limite de caracteres, consideram apenas os primeiros caracteres, como é o caso do C99, onde apenas os 63 primeiros são significativo.

C++ KEYWORDS

asm	continue	float	new	signed	try
auto	default	for	operator	sizeof	typedef
break	delete	friend	private	static	union
case	do	goto	protected	struct	unsigned
catch	double	if	public	switch	virtual
char	Else	Inline	register	template	void
class	enum	int	return	this	volatile
const	extern	long	short	throw	while

Nomes de variáveis

Os nomes, na maioria das linguagens de programação, têm o mesmo formato: uma letra seguida por uma cadeia de letras, dígitos e sublinhados (_).

Atualmente, nas linguagens baseadas em C, o uso do sublinhados foi substituído pela chamada de camelo (camelCase), na qual todas as palavras de um nome contendo múltiplas palavras têm sua primeira letra em maiúsculo, exceto a primeira palavra, exemplo: nomeCompleto.

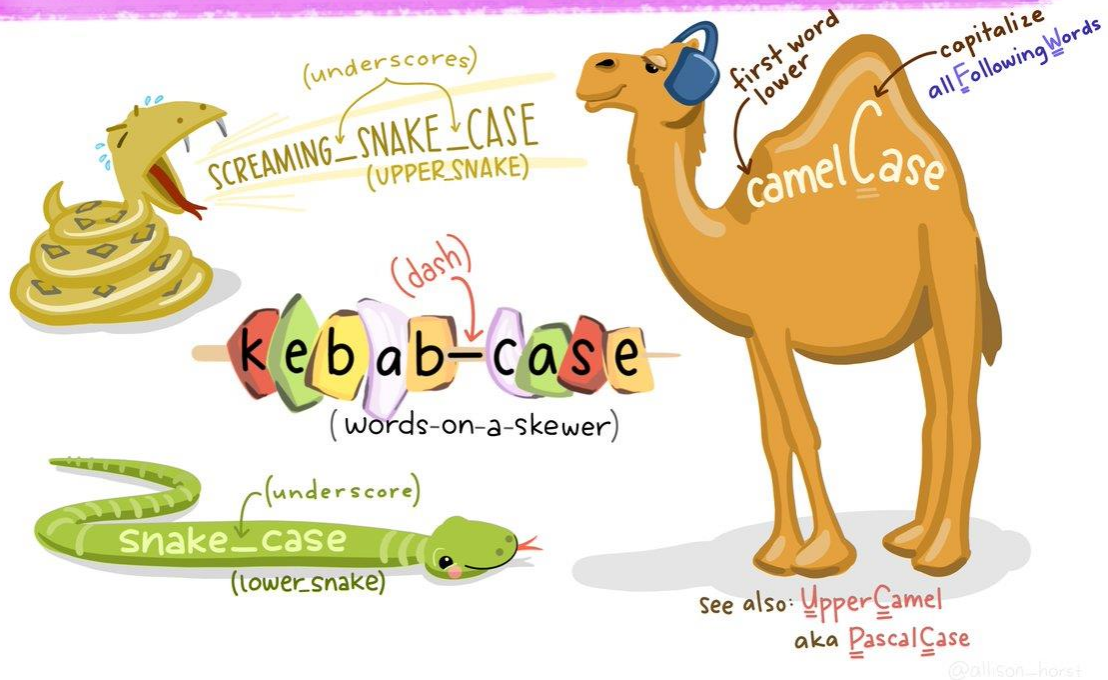
Note que o uso de sublinhados e de capitalização mista em nomes é uma questão de estilo de programação, não de projeto de linguagem.

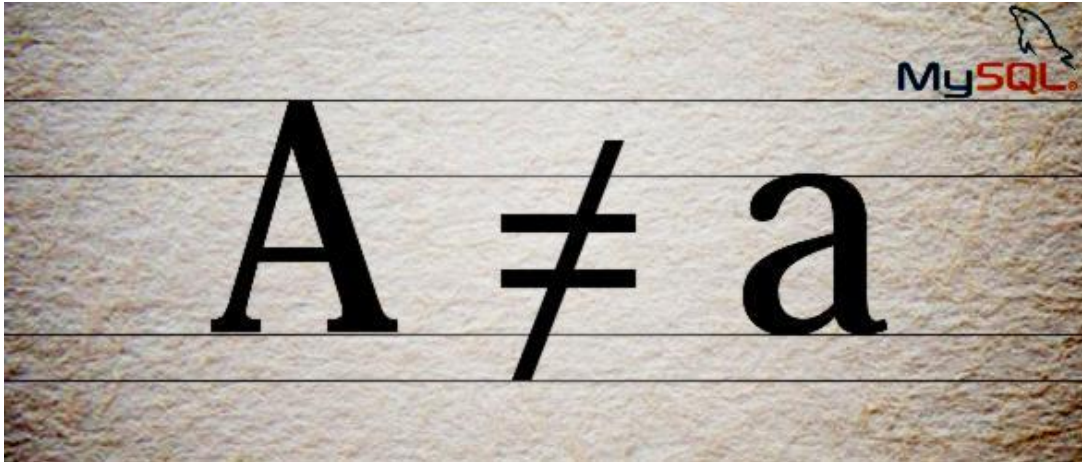


Camel
case?

Snake
case!

in that case...



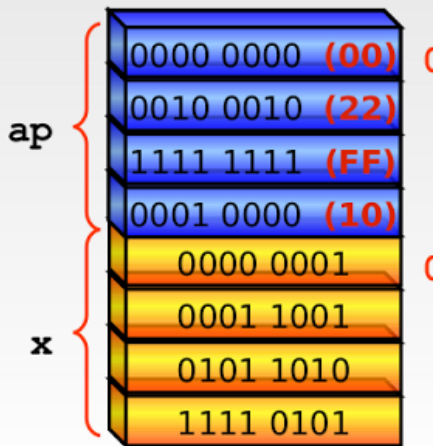


Nomes de variáveis

Em muitas linguagens, mais notavelmente nas baseadas em C, as letras maiúsculas e minúsculas nos nomes são distintas; ou seja, são sensíveis à capitalização (Case-sensitive) .

Por exemplo, os três nomes seguintes são distintos em C++: rosa , ROSA e Rosa .


```
int x;  
int *ap;    // apontador p  
ap = &x;    // ap aponta p
```



Endereço de variáveis

Endereço de uma variável é o endereço de memória de máquina ao qual ela está associada.

Em muitas linguagens, é possível que a mesma variável seja associada com diferentes endereços em vários momentos em um programa.

Por exemplo, se um subprograma tem uma variável local alocada a partir da pilha de tempo de execução quando o subprograma é chamado, diferentes chamadas podem resultar em a variável ter diferentes endereços.

Essas são, em certo sentido, instâncias diferentes da mesma variável.

Endereço de variáveis

É possível haver múltiplas variáveis com o mesmo endereço.

Quando mais de uma variável pode ser usada para acessar a mesma posição de memória, elas são chamadas de apelidos (aliases).

O uso de apelidos é um problema para a legibilidade porque permite que uma variável tenha seu valor modificado por uma atribuição à uma variável diferente.

A forma mais simples de um alias é equivalente ao `typedef` mecanismo do c++ 03:

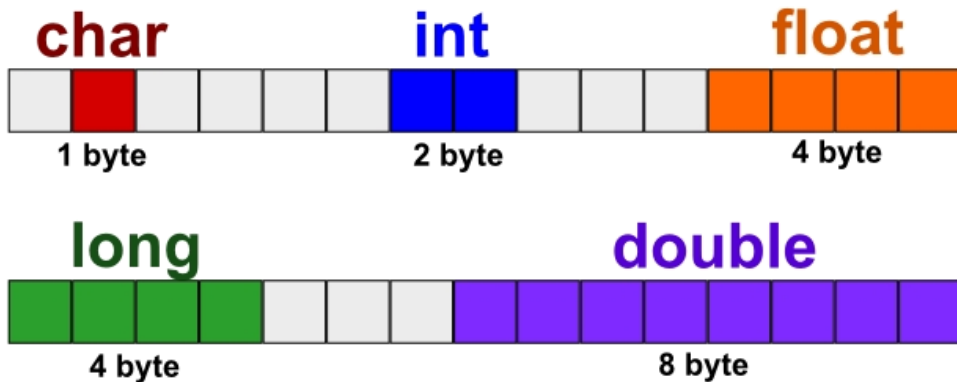
```
C++  
  
// C++11  
using counter = long;  
  
// C++03 equivalent:  
// typedef long counter;
```

 Copiar

Os dois permitem a criação de variáveis do tipo "counter". Algo mais útil seria um alias de tipo para `std::ios_base::fmtflags` como este:

```
C++  
  
// C++11  
using fmtfl = std::ios_base::fmtflags;  
  
// C++03 equivalent:  
// typedef std::ios_base::fmtflags fmtfl;  
  
fmtfl fl_orig = std::cout.flags();  
fmtfl fl_hex = (fl_orig & ~std::cout.basefield) | std::cout.showbase | std::cout.hex;  
// ...  
std::cout.flags(fl_hex);
```

 Copiar



PI=3.141592653

valor

Valor de variáveis

O VALOR DE UMA VARIÁVEL É O CONTEÚDO DA(S) CÉLULA(S) DE MEMÓRIA ASSOCIADA(S) À ELA. CADA TIPO DE DADO POSSUI UM TAMANHO MÁXIMO QUE PODE ALOCAR E ARMAZENAR DADOS

Tipo de variáveis

O tipo de uma variável determina a faixa de valores que ela pode armazenar e o conjunto de operações definidas para valores do tipo.

Por exemplo, o tipo `int` em Java especifica uma faixa de valores de -2147483648 a 2147483647 e operações aritméticas para adição, subtração, multiplicação, divisão e módulo.

Tipo	Em bit	Bits	Bytes																																																																
char	8	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>1</td><td>1</td></tr></table>	0	0	0	0	1	1	1	1	8 bits = 1 byte																																																								
0	0	0	0	1	1	1	1																																																												
int	16	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>1</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	16 bits = 2 bytes																																																
0	0	0	0	1	1	1	1																																																												
1	1	1	1	0	0	0	0																																																												
float	32	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>1</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	32 bits = 4 bytes																																
0	0	0	0	1	1	1	1																																																												
1	1	1	1	0	0	0	0																																																												
1	1	1	1	1	1	1	0																																																												
0	0	0	0	0	0	0	0																																																												
double	64	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>1</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>1</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	64 bits = 8 bytes
0	0	0	0	1	1	1	1																																																												
1	1	1	1	0	0	0	0																																																												
1	1	1	1	1	1	1	0																																																												
0	0	0	0	0	0	0	0																																																												
0	0	0	0	1	1	1	1																																																												
1	1	1	1	0	0	0	0																																																												
1	1	1	1	1	1	1	0																																																												
0	0	0	0	0	0	0	0																																																												
void	0	0	Nada																																																																

C++

Tipo de variáveis - Tipo de dados primitivos

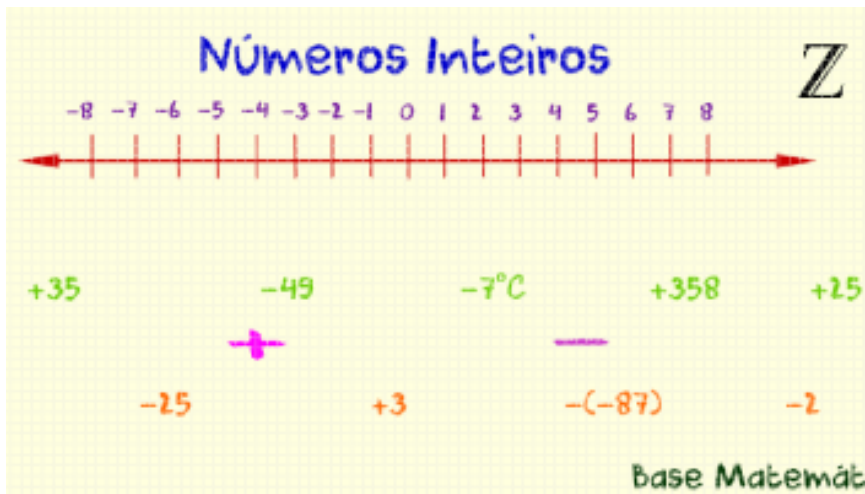
<i>Tipo</i>	<i>Bytes</i>	<i>Valores limites</i>
char	1	-128 a 127
unsigned char	1	0 a 255
short	2	-32768 a 32767
unsigned short	2	0 a 65535
int	4	-2147483648 a 2147483647
unsigned int	4	0 a 4294967295
long	4	-2147483648 a 2147483647
unsigned long	4	0 a 4294967295
float	4	1.175494×10^{-38} a $3.402823 \times 10^{+38}$
double	8	$2.225074 \times 10^{-308}$ a $1.797693 \times 10^{+308}$
long double	8	$2.225074 \times 10^{-308}$ a $1.797693 \times 10^{+308}$

São aqueles cujos valores não podem ser decompostos em outros valores de tipos mais simples.

Praticamente todas as linguagens de programação fornecem um conjunto de tipos de dados primitivos.

Os tipos de dados primitivos de uma linguagem são usados, com um ou mais construtores de tipo, para fornecer os tipos estruturados.

Exemplo: Inteiro, Ponto Flutuante, Decimal, Booleano, Caractere



Inteiro

O tipo de dados primitivo numérico mais comum é o inteiro.

Muitos computadores agora suportam diversos tamanhos de inteiros.

Por exemplo, Java inclui quatro tamanhos inteiros com sinal: byte, short, int e long.

Algumas linguagens, como C#, incluem tipos inteiros sem sinal, simplesmente tipos para valores inteiros sem sinal.

<https://docs.microsoft.com/pt-br/dotnet/csharp/language-reference/keywords/integral-types-table>

Ponto Flutuante

Tipos de dados de ponto flutuante modelam números reais, mas as representações são apenas aproximações para muitos valores reais.

Como esta representação é finita, os números como Pi, somente podem ser aproximados (arredondado).

A maioria das linguagens de programação inclui dois tipos de ponto flutuante frequentemente chamados float e double.


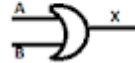

1.0	a number with a fractional part of zero
-3.141593	an approximation of $-\pi$
6.02E+23	6.02×10^{23}
0.0	a real zero

Decimal

Tipos de dados decimais armazenam um número fixo de dígitos decimais, com o ponto decimal em uma posição fixa no valor.

Esses são os tipos de dados primários para processamento de dados de negócio.

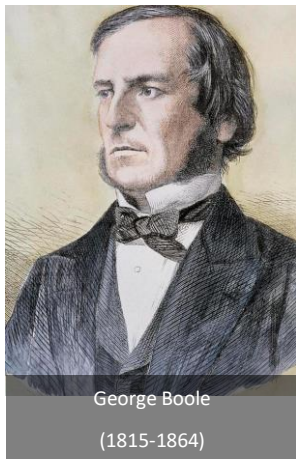
Tipos decimais têm a vantagem de serem capazes de armazenar precisamente valores decimais, ao menos dentro de uma faixa restrita, o que não pode ser feito com tipos de ponto flutuante.

PORTA LÓGICA	SÍMBOLO MATEMÁTICO	SÍMBOLO GRÁFICO
AND	$X=A.B$	
OR	$X=A+B$	
NOT	$X=A$	

Booleano

Tipos booleanos são talvez os mais simples. Sua faixa de valores tem apenas dois elementos: um para verdadeiro e outro para falso.

Em C89, no qual expressões numéricas são usadas como condicionais. Em tais expressões, todos os operandos com valores diferentes de zero são considerados verdadeiros, e zero é considerado falso.



Negação

P	$\sim P$
V	F
F	V

Conjunção

P	Q	$P \wedge Q$
V	V	V
V	F	F
F	V	F
F	F	F

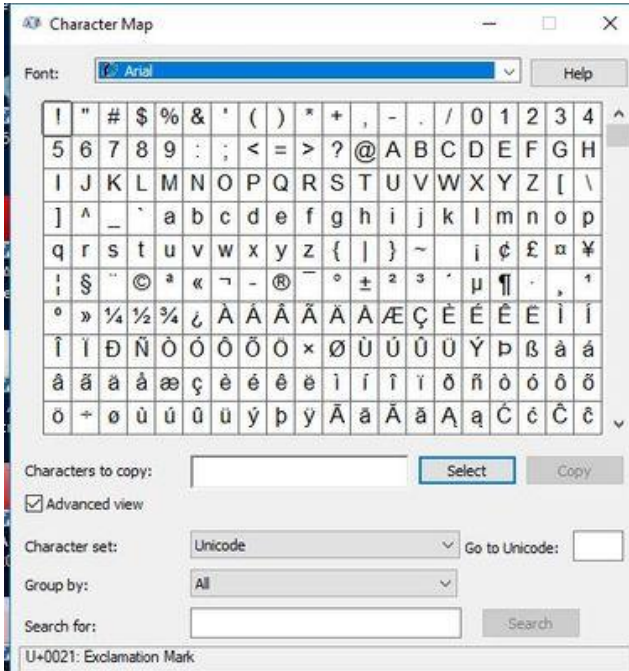
Disjunção

P	Q	$P \vee Q$
V	V	V
V	F	V
F	V	V
F	F	F

Caractere

Dados na forma de caracteres são armazenados nos computadores como codificações numéricas.

Tradicionalmente, a codificação mais usada era o ASCII (Padrão Americano de Codificação para Intercâmbio de Informação – American Standard Code for Information Interchange) de 8 bits, que usa os valores de 0 a 127 para codificar 128 caracteres diferentes.



ASCII Table

Dec	Hex	Oct	Char	Dec	Hex	Oct	Char	Dec	Hex	Oct	Char	Dec	Hex	Oct	Char
0	0	0		32	20	40	[space]	64	40	100	@	96	60	140	`
1	1	1		33	21	41	!	65	41	101	A	97	61	141	a
2	2	2		34	22	42	"	66	42	102	B	98	62	142	b
3	3	3		35	23	43	#	67	43	103	C	99	63	143	c
4	4	4		36	24	44	\$	68	44	104	D	100	64	144	d
5	5	5		37	25	45	%	69	45	105	E	101	65	145	e
6	6	6		38	26	46	&	70	46	106	F	102	66	146	f
7	7	7		39	27	47	'	71	47	107	G	103	67	147	g
8	8	10		40	28	50	(72	48	110	H	104	68	150	h
9	9	11		41	29	51)	73	49	111	I	105	69	151	i
10	A	12		42	2A	52	*	74	4A	112	J	106	6A	152	j
11	B	13		43	2B	53	+	75	4B	113	K	107	6B	153	k
12	C	14		44	2C	54	,	76	4C	114	L	108	6C	154	l
13	D	15		45	2D	55	-	77	4D	115	M	109	6D	155	m
14	E	16		46	2E	56	.	78	4E	116	N	110	6E	156	n
15	F	17		47	2F	57	/	79	4F	117	O	111	6F	157	o
16	10	20		48	30	60	0	80	50	120	P	112	70	160	p
17	11	21		49	31	61	1	81	51	121	Q	113	71	161	q
18	12	22		50	32	62	2	82	52	122	R	114	72	162	r
19	13	23		51	33	63	3	83	53	123	S	115	73	163	s
20	14	24		52	34	64	4	84	54	124	T	116	74	164	t
21	15	25		53	35	65	5	85	55	125	U	117	75	165	u
22	16	26		54	36	66	6	86	56	126	V	118	76	166	v
23	17	27		55	37	67	7	87	57	127	W	119	77	167	w
24	18	30		56	38	70	8	88	58	130	X	120	78	170	x
25	19	31		57	39	71	9	89	59	131	Y	121	79	171	y
26	1A	32		58	3A	72	:	90	5A	132	Z	122	7A	172	z
27	1B	33		59	3B	73	;	91	5B	133	[123	7B	173	{
28	1C	34		60	3C	74	<	92	5C	134	\	124	7C	174	
29	1D	35		61	3D	75	=	93	5D	135]	125	7D	175	}
30	1E	36		62	3E	76	>	94	5E	136	^	126	7E	176	~
31	1F	37		63	3F	77	?	95	5F	137	_	127	7F	177	

Caractere

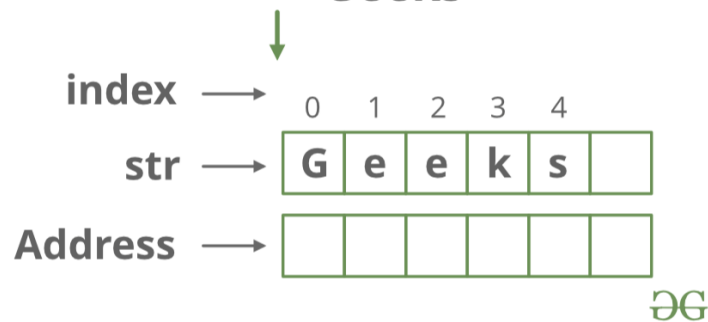
Por causa da globalização dos negócios e da necessidade de os computadores se comunicarem com outros computadores pelo mundo, o conjunto de caracteres ASCII se tornou inadequado.

Em 1991, o Consórcio Unicode publicou o padrão UCS-2, um conjunto de caracteres de 16 bits. Essa codificação de caracteres é geralmente chamada de Unicode e inclui os caracteres da maioria das linguagens naturais do mundo.

Após 1991, o Consórcio Unicode, em cooperação com a Organização Internacional de Padrões (ISO – International Standards Organization) desenvolveu uma codificação de caracteres de 4 bytes chamada UCS-4 ou UTF-32, que é descrita pelo padrão ISO/IEC 10646, publicado em 2000.

String in C

```
char str[] = "Geeks"
```



Cadeias de caracteres

Um tipo cadeia de caracteres (String) é um tipo no qual os valores consistem em sequências de caracteres.

Constantes do tipo cadeias de caracteres são usadas para rotular a saída, e a entrada e saída de todos os tipos de dados é geralmente feita em termos de cadeias.

É claro, cadeias de caracteres são também um tipo essencial para todos os programas que realizam manipulação de caracteres.

Cadeias de caracteres

As operações comuns em cadeias são: atribuição, concatenação, referência a subcadeias, comparação e casamento de padrões.

Se as cadeias não são definidas como um tipo primitivo, os dados da cadeia são normalmente armazenados em vetores de caracteres e referenciados como tal na linguagem. Essa é a abordagem usada por C e C++.

concatenado

Que se conseguiu concatenar; que está unido; encadeado.

Que se liga de modo lógico; que possui uma relação sequencial (lógica ou orgânica): raciocínio concatenado.

 Dicio.com.br

	A	B	C
1	Texto 1	EXEMPLO	EXEMPLO
2	Texto 2	DE	DE
3	Texto 3	CONCATENAR	CONCATENAR
4		EXEMPLO DE CONCATENAR	EXEMPLO-DE-CONCATENAR
5		=CONCAT(B1;" ";B2;" ";B3)	=CONCAT(C1;"-";C2;"-";C3)
6			

Um jeito tradicional de se declarar constantes em C e C++ é o seguinte:

```
1 #define BRASIL 0
2 #define ITALIA 1
3 #define PORTUGAL 2
4 #define ALEMANHA 3
```

E eis a maneira de fazer o mesmo utilizando `enums`:

```
1 enum
2 {
3     BRASIL,
4     ITALIA,
5     PORTUGAL,
6     ALEMANHA
7 };
```

Pode-se, alternativamente, criar um tipo para seu `enum`:

```
1 enum Paises
2 {
3     BRASIL,
4     ITALIA,
5     PORTUGAL,
6     ALEMANHA
7 };
```

Tipos Ordinais Definidos pelo Usuário

Um tipo ordinal é um no qual a faixa de valores possíveis pode ser facilmente associada com o conjunto dos inteiros positivos.

Existem dois tipos ordinais definidos pelo usuário suportados pelas linguagens de programação:

enumerações e subfaixas.

<https://murilo.wordpress.com/2009/05/29/o-que-sao-enums-e-como-utiliza-los-melhor-em-c/>

Enumerações

Um tipo enumeração é um no qual todos os valores possíveis, os quais são constantes nomeadas, são fornecidos, ou enumerados, na definição.

Tipos enumeração fornecem uma maneira de definir e agrupar coleções de constantes nomeadas, chamadas de constantes de enumeração.

A definição de um tipo enumeração típico é mostrada neste exemplo em C#:

```
enum days {Mon, Tue, Wed, Thu, Fri, Sat, Sun};
```

As constantes de enumeração são preenchidas implicitamente por atribuições de valores inteiros, 0, 1, ... mas podem ser atribuídos explicitamente quaisquer literais inteiros na definição do tipo.

Subfaixas

5B - Subranges (author: Tao Yue, state: unchanged)

A *subrange* type is defined in terms of another ordinal data type. The type specification is:

```
lowest_value .. highest_value
```

where `lowest_value < highest_value` and the two values are both in the range of another ordinal data type.

For example, you may want to declare the days of the week as well as the work week:

```
type
  DaysOfWeek = (Sunday, Monday, Tuesday, Wednesday,
                Thursday, Friday, Saturday);
  DaysOfWorkWeek = Monday..Friday;
```

You can also use subranges for built-in ordinal types such as `char` and `integer`.

Um tipo subfaixa é uma subsequência contígua de um tipo ordinal.

Por exemplo, `12..14` é uma subfaixa do tipo inteiro.

Tipos subfaixa foram introduzidos pelo Pascal e incluídos em Ada.

Exemplos:

type Days is (Mon, Tue, Wed, Thu, Fri, Sat, Sun);
subtype Weekdays is Days range Mon..Fri;
subtype Index is Integer range 1..100;

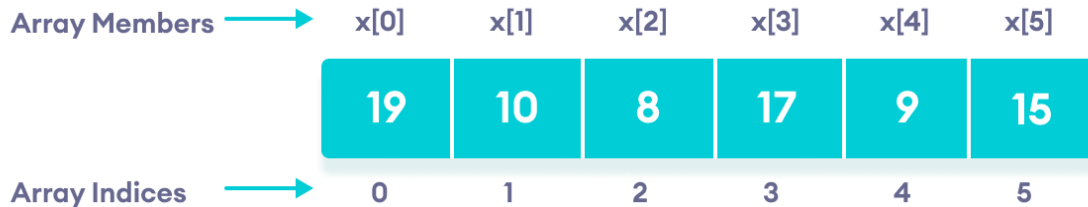
Tipos Array

É um tipo de dado estruturado unidimensional que consiste de um número fixo de elementos, sendo que todos os elementos devem ser do mesmo tipo (char, integer, real, string)

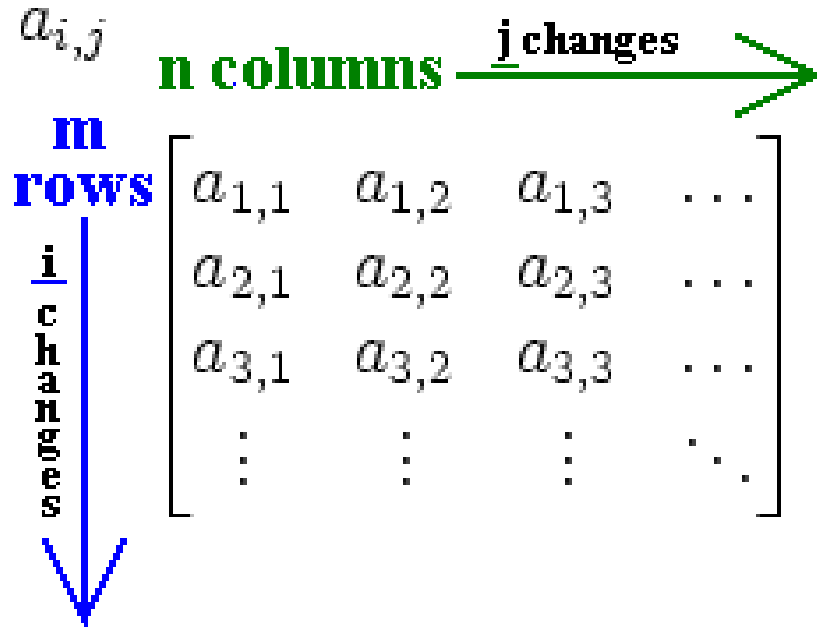
Exemplo:

```
int[] meuArray = {0,1,2,3,4}
```

```
int meuOutroArray = new int[5];
```



m-by-n matrix



Matrizes

Podemos definir um array com mais de uma dimensão, também chamado de matriz.

Exemplo:

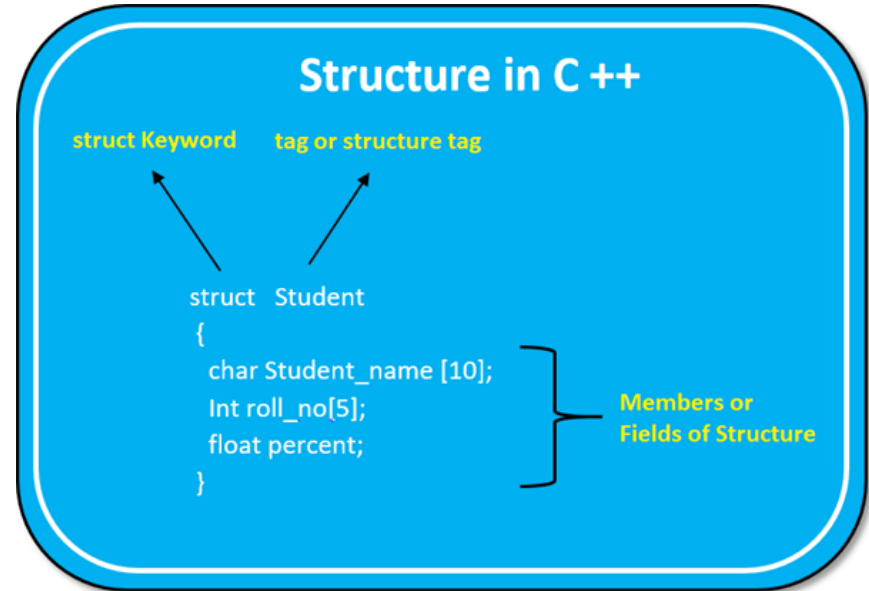
```
float[][] notas = new float[2][5];
```

```
float[][] notas = { {8.0, 7.5, 8.5, 9.0, 8.0 },  
                    {8.9, 9.0, 8.6, 8.4, 8.0 } };
```

Tipo Registro

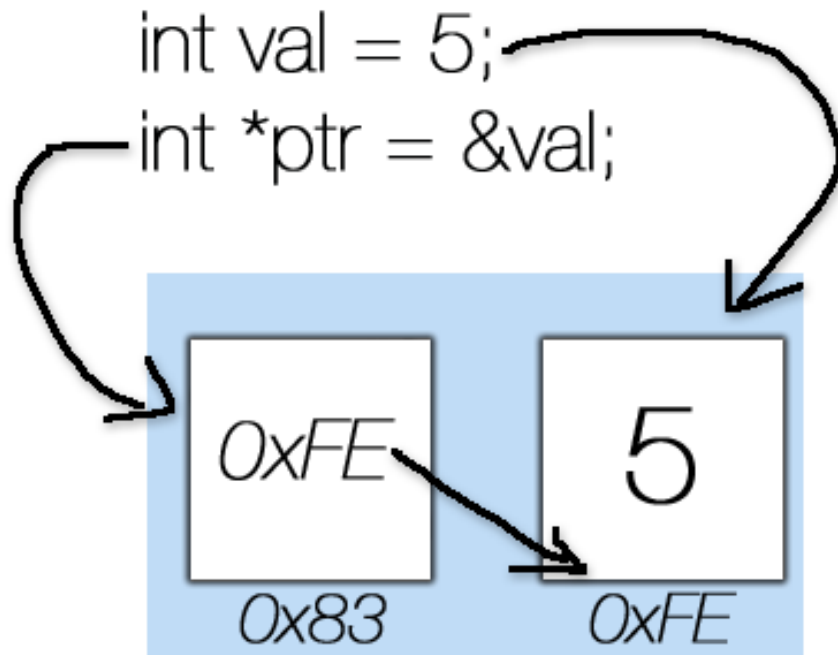
Um registro é um agregado de elementos de dados no qual os elementos individuais são identificados por nomes e acessados por meio de deslocamentos a partir do início da estrutura.

Em geral, existe uma necessidade nos programas de modelar coleções de dados que não são do mesmo tipo ou tamanho. Por exemplo, informações sobre um estudante universitário podem incluir seu nome, disciplina, nota das provas e assim por diante.



Tipo Registro

```
struct ficha_de_aluno
{
    char nome[50];
    char disciplina[30];
    float nota_prova1;
    float nota_prova2;
};
```



Tipos Ponteiro

Um ponteiro é um valor que representa uma referência ou um endereço de memória. São usados comumente em C, C++, Ada e Perl.

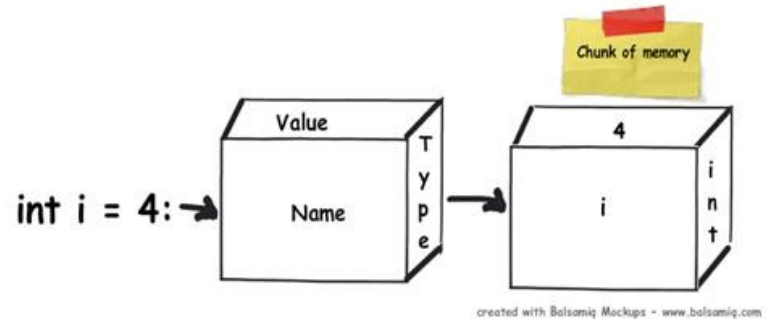
C fornece duas operações com ponteiros:

- Operador Unário "Endereço de" (&): Recebe uma variável como argumento e retorna o endereço dessa variável
- Operador Unário de "Desreferenciação" (*): Recebe uma referência e produz o valor dessa referência.

```
char *nome = (char *) malloc(21*sizeof(char));
```

Tempo de vida de variáveis

O tempo de vida de uma variável é o durante o qual ela está vinculada a uma posição específica da memória. Então, o tempo de vida de uma variável começa quando ela é vinculada a uma célula específica e termina quando ela é desvinculada dessa célula.



Escopo de variáveis

O escopo de uma variável é a faixa de sentenças nas quais ela é visível.

Uma variável é visível em uma sentença se ela pode ser referenciada nesta sentença.

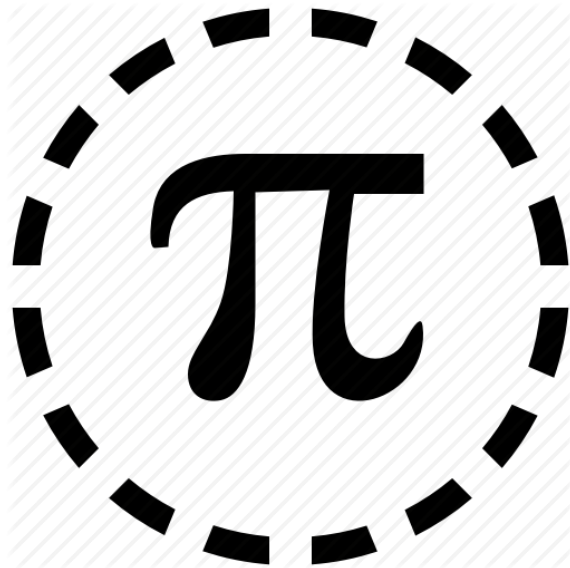
As regras de escopo de uma linguagem determinam como uma ocorrência em particular de um nome é associada com uma variável.

Constantes nomeadas

Uma constante nomeada é uma variável vinculada a um valor apenas uma vez.

Constantes nomeadas são úteis para auxiliar a legibilidade e a confiabilidade dos programas. A legibilidade pode ser melhorada, por exemplo, ao ser usado o nome `pi` em vez de constante `3.14159`.

```
1  const PI = Math.PI;  
2  
3  console.log({ PI });  
4  const fn = ({PI}) => {};
```



Exercícios Resolvidos

1. Qual tipo de dados primitivo é indicado para a manipulação de informações no formato moeda e possui precisão de duas casas decimais?

- A. Inteiro
- B. Ponto-Flutuante
- C. Caractere
- D. **Decimal**
- E. Booleano

Resposta: Alternativa D

Exercícios Resolvidos

1. Qual tipo de dados primitivo é indicado para a manipulação de informações no formato moeda e possui precisão de duas casas decimais?

Resposta: Alternativa D (Decimal)

O tipo de dados decimal é indicado para aplicações comerciais e financeiras que manipulam informações no formato moeda. Um exemplo de valor para este tipo é R\$ 50.32 (cinquenta reais e trinta e dois centavos). A vantagem de utilizar este tipo de dados, é a economia de memória para o armazenamento da parte decimal.

Exercícios Resolvidos

2. Qual das operações com o tipo de dados String permite a união de duas cadeias de caracteres?

- A. Comparação
- B. Concatenação
- C. Cópia
- D. Ordenação
- E. Organização

Resposta: Alternativa B

Exercícios Resolvidos

2. Qual das operações com o tipo de dados String permite a união de duas cadeias de caracteres?

Resposta: Alternativa B (Concatenação)

A operação de concatenação permite a união de duas cadeias de caracteres. Essa operação, em geral, é realizada por meio do operador representado pelo símbolo '+'.

Exercícios Resolvidos

03 - As variáveis de um programa são responsáveis pela manipulação de informações e representam uma abstração de um espaço na memória do computador. Os principais atributos que definem uma variável são:

- A. endereço principal e endereço virtual
- B. nome, endereço, valor, tipo e escopo - (CORRETA)
- C. nome simples e nome composto
- D. valor público, valor privado e valor protegido
- E. tipo estático, tipo dinâmico e tipo virtual

Exercícios Resolvidos

04 - Qual tipo de dados primitivo é comumente utilizado para representar valores que não possuem a parte fracionária?

- A. Inteiro - (CORRETA)
- B. Ponto-Flutuante
- C. Decimal
- D. Booleano
- E. Caractere

Exercícios Resolvidos

05 - Qual tipo de dados primitivo utilizado para representar valores do tipo verdadeiro ou falso?

- A. Inteiro
- B. Ponto-Flutuante
- C. Decimal
- D. Booleano - (CORRETA)
- E. Caractere

Exercícios Resolvidos

06 - Qual tipo de dados primitivo utilizado para representar valores que possuem uma parte inteira e outra parte fracionária?

- A. Inteiro
- B. Ponto-Flutuante - (CORRETA)
- C. Decimal
- D. Booleano
- E. Caractere

Exercícios Resolvidos

07 - Considerando um tipo de dados Array, em que a indexação é iniciada na posição zero, contendo os valores {'A','B','C','D','E'}, qual valor está armazenado na posição 2 do array?

- A. 'A'
- B. 'B'
- C. 'C' - (CORRETA)
- D. 'D'
- E. 'E'