ALGORITMOS

Materiais de apoio

- Utilização dos materiais dos professores:
  - Lourival Coelho da Silva Filho
  - Marcelo Ferreira Siqueira

Origem

- A palavra algoritmo tem origem no sobrenome do matemático, astrônomo, geólogo, geógrafo e autor persa **Mohammed ibn-Musa al-Khwarizmi**, que viveu entre 790 e 840 d. C. (numerais indianos – notação posicional decimal para o Ocidente).

Algoritmos e problemas computacionais

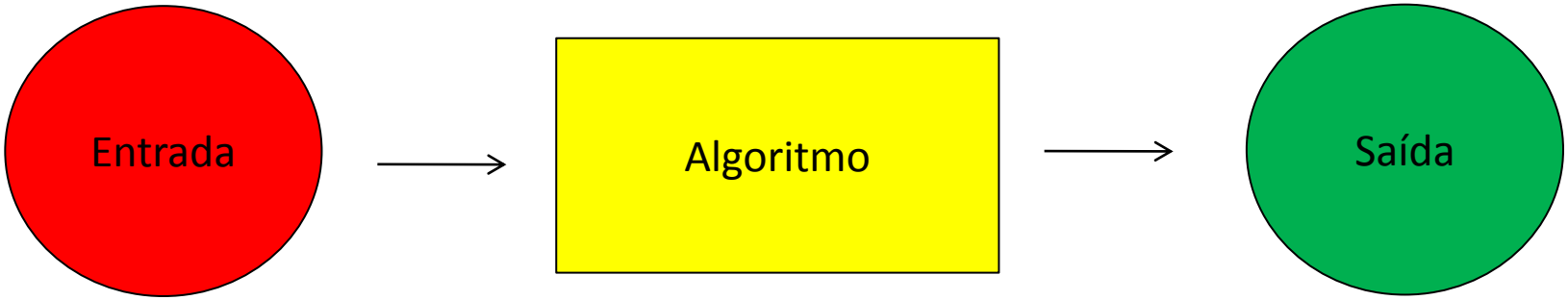
- Uma definição: é uma **sequência finita de instruções, bem definidas e não-ambíguas, para um dado problema**. Cada instrução de um algoritmo deve ser executada por um período de tempo finito.
- Algoritmos computacionais descrevem instruções a serem executadas por computador.
- **Um algoritmo é um caminho para a solução de um problema**.
- Um algoritmo é uma abstração, uma ideia ou uma estratégia.

Assim,

- Quando elaboramos um algoritmo devemos especificar **ações claras e precisas**, que a partir de um **estado inicial**, após um **período de tempo finito**, produzem um **estado final previsível e bem definido**.
- Isto significa que o algoritmo fixa um **padrão de comportamento a ser seguido**, uma norma de execução a ser trilhada, com vistas a **alcançar**, como **resultado final**, **a solução de um problema**, garantindo que **sempre** que executado, sob as **mesmas condições**, produza o mesmo resultado.

Componentes fundamentais

- **Entrada** – conjunto de dados que deve ser fornecido ao algoritmo.
- **Saída** – conjunto de dados produzidos pelo algoritmo.
- **Algoritmos computacionais** – agentes transformadores de dados de entrada em dados de saída (comumente denominado processamento).



Importante

- Um algoritmo **não é a solução de um problema**, mas sim **um processo para se obter a solução**.
- Problema computacional
  - Pode ser resolvido por um computador através de um algoritmo.
  - Ocorrência – uma instância qualquer da entrada do problema.
- Um algoritmo deve sempre ser construído para resolver **todas** as possíveis ocorrências de um problema.
- Assim, um **algoritmo** é dito **correto** se ele sempre **termina e produz a resposta correta para todas as ocorrências** de um dado problema.

Formas de representação

- Descrição narrativa:
  - Linguagem natural.
    - Inconveniente: má interpretação, originando ambiguidades e imprecisões.
    - Exemplo:
      - Algoritmo para trocar um pneu furado
        - » afrouxar ligeiramente as porcas;
        - » suspender o carro;
        - » retirar as porcas e o pneu;
        - » colocar o pneu reserva e as porcas;
        - » abaixar o carro;
        - » dar o aperto final nas porcas.

Formas de representação (continuação)

- Maneira textual e informal (ou quase) - pseudocódigo: PORTUGOL
  - Linguagem que se assemelha àquela que usamos para nos comunicar.
  - Vocabulário destinado à descrição – extremamente pequeno.
  - Forma geral:

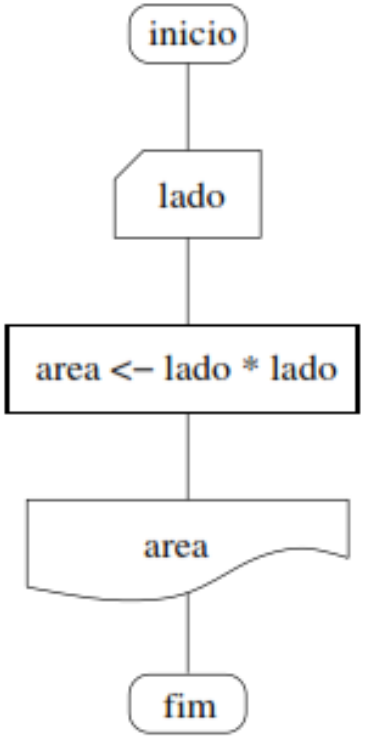
```
algoritmo < nome_do_algoritmo >
        < declaração_de_variáveis >   (cabeçalho)
início
        < instruções >  (processamento)
fim
```

Formas de representação: Portugol - exemplo

```
1  algoritmo "Area do quadrado''
2    var lado, area : real
3  inicio
4    escreva ( "Entre com o comprimento dos lados do quadrado: " )
5    leia ( lado )
6    area <- lado * lado
7    escreva ( "A area do quadrado e:   " ,  area )
8  fimalgoritmo
```

Formas de representação (continuação)

- Maneira gráfica: FLUXOGRAMA
  - Diagramas com figuras que identificam as várias instruções do algoritmo.



Formas de representação: Fluxograma - simbologia

Figura	Descrição
	Início ou fim do fluxograma
	Entrada de dados
	Cálculo de expressões
	Saída de resultados
	Tomada de decisão
	Fluxo

Resolução de problemas

- A capacidade para resolver problemas pode ser vista como uma habilidade a ser adquirida: **conhecimento** e **destreza**.
  - Conhecimento (**táticas e estratégias**) – adquirido pelo **estudo**.
  - Destreza (**experiência no uso do conhecimento**) – adquirida pela **prática**.

Mais alguns conceitos

- **Programação** – ato de instruir o computador para que ele resolva um determinado problema.
- **Linguagem de programação** – linguagem apropriada para descrever algoritmos computacionais que permitirão inserir ações e dados no computador.
- **Programa** – algoritmo em uma linguagem de programação.
- **Linguagem de máquina** – linguagem de programação própria de cada modelo de computador.
- **Compilador** (para uma linguagem de programação) – realiza a tradução automática de um programa escrito em uma certa linguagem para um programa equivalente escrito na linguagem de máquina.

Tipos de dados

- Qualquer trabalho realizado por um computador é baseado na manipulação das informações contidas em sua memória. A grosso modo, estas informações podem ser classificadas em dois tipos:
  - as **instruções**: leituras, atribuições, operações, etc.
  - os **dados**, propriamente dito: valores a serem processados.

Tipos de dados

- Um computador processa dados num sistema numérico diferente do utilizado pelo ser humano em geral.
    - O ser humano utiliza o **sistema decimal** (dez dígitos);
    - O computador utiliza o **sistema binário** (dois dígitos) – cada dígito binário (0 ou 1) é chamado de ***bit***.
- A memória de um computador é um conjunto de células identificadas univocamente por um número chamado endereço (número da célula).
    - Qualquer informação será armazenada na memória do computador utilizando, pelo menos, uma célula.
    - A cada célula, chamamos de *byte*, que corresponde a um conjunto de oito *bits*, e portanto um *byte* possui 2^8 estados possíveis, ou seja, 256 combinações dos *bits* 0 e 1 nas oitos posições do *byte*.

Como são representadas as informações?

- Os computadores podem manipular diversos tipos de arquivos (ou dados), incluindo: dados numéricos, texto, imagens, vídeos, som.
- Todos estes itens são armazenados em sua forma primitiva, ou seja, são representados em sua **forma binária** (**combinações de 0 e 1**). Nesse sistema numérico, os dados são transformados em 0 e 1, conhecidos por ***BInary digiT*** (BIT), para então serem armazenados na memória em grupo de ***oito bits***, agrupamento que forma **um Byte**.
- Cada *byte* é identificado e acessado por meio de um endereço.

Byte como medida

Unidade	Capacidade
Byte	1 Byte = 8 bits
Quilobyte (kB)	1kB = 1024 Bytes ou 2^{10} Bytes ou 8.192 bits
Megabyte (MB)	1MB = 1024 kB ou 2^{20} Bytes ou 8.388.608 bits
Gigabyte (GB)	1GB = 1024 MB ou 1.048.576 kB ou 2^{30} Bytes ou 8.589.934.592 bits
Terabyte (TB)	1 TB = 1024 GB ou 1.048.576 MB ou 1.073.741.824 kB ou 2^{40} Bytes ou 8.796.093.022.208 bits
Petabyte (PB)	?
Exabyte (EB)	?
Zettabyte (ZB)	?
Yottabyte (YB)	?

Tabela ASCII

- Todos os caracteres existentes possuem um caractere numérico correspondente em uma tabela no computador na codificação *American Standard Coded for Information Interchange* (ASCII) e esses caracteres numéricos são transformados em binário para que a máquina entenda.
- Então, quando digitamos a letra “A”, o computador recebe um byte representando essa letra, ou seja, recebe “01000001” e assim para todos os caracteres.

Tabela ASCII (primeiras versões)

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

Source: www.LookupTables.com

Extended ASCII Codes

128	Ç	144	É	160	á	176	▒	192	└	208	╨	224	α	240	≡
129	ü	145	æ	161	í	177	▒	193	┴	209	╤	225	ß	241	±
130	é	146	Æ	162	ó	178	▓	194	┬	210	╥	226	Γ	242	≥
131	â	147	ô	163	ú	179	│	195	├	211	╙	227	π	243	≤
132	ä	148	ö	164	ñ	180	┤	196	─	212	╘	228	Σ	244	⌠
133	à	149	ò	165	Ñ	181	╡	197	┼	213	╒	229	σ	245	⌡
134	å	150	û	166	ª	182	╢	198	╞	214	╓	230	µ	246	÷
135	ç	151	ù	167	º	183	╖	199	╟	215	╫	231	τ	247	≈
136	ê	152	ÿ	168	¿	184	╕	200	╚	216	╪	232	Φ	248	°
137	ë	153	Ö	169	⌐	185	╣	201	╔	217	┘	233	Θ	249	∙
138	è	154	Ü	170	¬	186	║	202	╩	218	┌	234	Ω	250	·
139	ï	155	¢	171	½	187	╗	203	╦	219	█	235	δ	251	√
140	î	156	£	172	¼	188	╝	204	╠	220	▄	236	∞	252	n
141	ì	157	¥	173	¡	189	╜	205	═	221	▌	237	φ	253	²
142	Ä	158	₧	174	«	190	╛	206	╬	222	▐	238	ε	254	■
143	Å	159	ƒ	175	»	191	┐	207	╧	223	▀	239	∩	255	

Source: www.LookupTables.com

Tabela ASCII

Tabela ASCII -I

Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
0	00	Null	32	20	Space	64	40	@	96	60	`
1	01	Start of heading	33	21	!	65	41	A	97	61	a
2	02	Start of text	34	22	"	66	42	B	98	62	b
3	03	End of text	35	23	#	67	43	C	99	63	c
4	04	End of transmit	36	24	$	68	44	D	100	64	d
5	05	Enquiry	37	25	%	69	45	E	101	65	e
6	06	Acknowledge	38	26	&	70	46	F	102	66	f
7	07	Audible bell	39	27	'	71	47	G	103	67	g
8	08	Backspace	40	28	(	72	48	H	104	68	h
9	09	Horizontal tab	41	29	)	73	49	I	105	69	i
10	0A	Line feed	42	2A	*	74	4A	J	106	6A	j
11	0B	Vertical tab	43	2B	+	75	4B	K	107	6B	k
12	0C	Form feed	44	2C	,	76	4C	L	108	6C	l
13	0D	Carriage return	45	2D	-	77	4D	M	109	6D	m
14	0E	Shift out	46	2E	.	78	4E	N	110	6E	n
15	0F	Shift in	47	2F	/	79	4F	O	111	6F	o
16	10	Data link escape	48	30	0	80	50	P	112	70	p
17	11	Device control 1	49	31	1	81	51	Q	113	71	q
18	12	Device control 2	50	32	2	82	52	R	114	72	r
19	13	Device control 3	51	33	3	83	53	S	115	73	s
20	14	Device control 4	52	34	4	84	54	T	116	74	t
21	15	Neg. acknowledge	53	35	5	85	55	U	117	75	u
22	16	Synchronous idle	54	36	6	86	56	V	118	76	v
23	17	End trans. block	55	37	7	87	57	W	119	77	w
24	18	Cancel	56	38	8	88	58	X	120	78	x
25	19	End of medium	57	39	9	89	59	Y	121	79	y
26	1A	Substitution	58	3A	:	90	5A	Z	122	7A	z
27	1B	Escape	59	3B	;	91	5B	[	123	7B	{
28	1C	File separator	60	3C	<	92	5C	\	124	7C	|
29	1D	Group separator	61	3D	=	93	5D	]	125	7D	}
30	1E	Record separator	62	3E	>	94	5E	^	126	7E	~
31	1F	Unit separator	63	3F	?	95	5F	_	127	7F	⌂

Tabela ASCII -II

Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
128	80	Ç	160	A0	á	192	C0	└	224	E0	α
129	81	ü	161	A1	í	193	C1	┴	225	E1	ß
130	82	é	162	A2	ó	194	C2	┬	226	E2	Γ
131	83	â	163	A3	ú	195	C3	├	227	E3	π
132	84	ä	164	A4	ñ	196	C4	─	228	E4	Σ
133	85	à	165	A5	Ñ	197	C5	┼	229	E5	σ
134	86	å	166	A6	ª	198	C6	╞	230	E6	µ
135	87	ç	167	A7	º	199	C7	╟	231	E7	τ
136	88	ê	168	A8	¿	200	C8	╚	232	E8	Φ
137	89	ë	169	A9	⌐	201	C9	╔	233	E9	Θ
138	8A	è	170	AA	¬	202	CA	╩	234	EA	Ω
139	8B	ï	171	AB	½	203	CB	╦	235	EB	δ
140	8C	î	172	AC	¼	204	CC	╠	236	EC	∞
141	8D	ì	173	AD	¡	205	CD	═	237	ED	ø
142	8E	Ä	174	AE	«	206	CE	╬	238	EE	ε
143	8F	Å	175	AF	»	207	CF	╧	239	EF	∩
144	90	É	176	B0	░	208	D0	╨	240	F0	≡
145	91	æ	177	B1	▒	209	D1	╤	241	F1	±
146	92	Æ	178	B2	▓	210	D2	╥	242	F2	≥
147	93	ô	179	B3	|	211	D3	╙	243	F3	≤
148	94	ö	180	B4	┤	212	D4	╘	244	F4	⌠
149	95	ò	181	B5	╡	213	D5	╒	245	F5	⌡
150	96	û	182	B6	╢	214	D6	╓	246	F6	÷
151	97	ù	183	B7	╖	215	D7	╫	247	F7	≈
152	98	ÿ	184	B8	╕	216	D8	╪	248	F8	°
153	99	Ö	185	B9	╣	217	D9	┘	249	F9	∙
154	9A	Ü	186	BA	║	218	DA	┌	250	FA	·
155	9B	¢	187	BB	╗	219	DB	█	251	FB	√
156	9C	£	188	BC	╝	220	DC	▄	252	FC	ⁿ
157	9D	¥	189	BD	╜	221	DD	▌	253	FD	²
158	9E	₧	190	BE	╛	222	DE	▐	254	FE	■
159	9F	ƒ	191	BF	┐	223	DF	▀	255	FF	⌂

Tipos básicos de dados

- Os dados podem ser numéricos, literais e lógicos.
    - Os dados numéricos dividem-se em números inteiros e números reais.

Dados numéricos

- Inteiro: consiste dos números inteiros e das operações de adição, subtração, multiplicação, divisão inteira e resto.
    - Na linguagem Portugol, os números inteiros são escritos apenas como a concatenação dos dígitos 0, 1, 2, 3, 4, 5, 6, 7, 8 e 9, tal como em 5, 100 e 1678. Números negativos são representados com o sinal "–" na frente do número, tal como −23.
    - Número inteiro – 2 *bytes* e número inteiro longo – 4 *bytes*.
- Real: consiste dos números reais e das operações de adição, subtração, multiplicação, divisão.
    - Na linguagem Portugol, os números reais são caracterizados por possuírem uma parte inteira e uma parte fracionária. Por exemplo, as partes inteira e fracionária do número real 3.141596 são 3 e 141596, respectivamente. Note que um "ponto" e não uma vírgula é usado para separar as partes inteira e fracionária.
    - Número real – 4 *bytes* e número real de precisão dupla – 8 *bytes*.

Dados literais

- São chamados, também, “alfanuméricos”, “cadeia de caracteres” ou “*strings*”.
- Consiste de um único símbolo ou de uma concatenação de símbolos do alfabeto usado pela linguagem Portugol.
    - Este alfabeto inclui todas **as letras do alfabeto romano**, **todos os dígitos**, 0, 1, . . . , 9 e os **caracteres de pontuação**, tais como ?, ., . . ., **entre muitos outros símbolos**.
    - Os elementos do conjunto de valores do tipo caractere devem ser escritos, nos algoritmos, entre **aspas duplas**, como, por exemplo, "a”, "Esta e uma frase formada por caracteres”.
    - Há um elemento especial, **“”**, que é denominado de palavra **vazia**, pois não possui nenhum símbolo.
    - Ocupa 1 *byte* para cada caractere da informação

Tabela ASCII (*American Standart Code for Information Interchange*)

- Um conjunto de 256 caracteres – estados possíveis de um *byte* (0-255).

Exemplos de alguns caracteres na tabela ASCII.

“”	caractere vazio	00000000	0
“ “	espaço em branco	00100000	32
“#”	sustenido (jogo da velha)	00100011	35
“&”	clave de sol (e comercial)	00100110	38
“*”	asterisco	00101010	42
“+”	mais	00101011	43
“0”	dígito zero	00110000	48
“1”	dígito um	00110001	49
“9”	dígito nove	00111001	57
“=”	igual	00111101	61
“A”	letra A(maiúscula)	01000001	65
“B”	letra B	01000010	66
“Z”	letra Z	01011010	90
“a”	letra a(minúscula)	01100001	97
“b”	letra b	01100010	98
“z”	letra z	01111010	122

Armazenamento e representação

- Um exemplo:

"casa"

Endereço	Informação
1000	c (99)
1001	a (97)
1002	s (115)
1003	a (97)

Dados lógicos

- Inclui apenas os valores lógicos **falso** e **verdadeiro** e as **operações de negação**, **conjunção** e **disjunção**.
    - Estudaremos este tipo em maiores detalhes mais adiante.
    - Ocupa 1 *byte* na memória

Constante e variável

- Um algoritmo manipula dados, que podem ser dados **variáveis** ou **constantes**.
- Dados variáveis são representados por variáveis, enquanto dados constantes são representados por constantes.

Variável

- Uma variável pode ser imaginada como uma “caixa” para armazenar valores de dados.
  - Esta caixa só pode armazenar um **único valor por vez**. No entanto, o valor armazenado na caixa **pode mudar inúmeras vezes** durante a execução do algoritmo.
  - Em um ambiente computacional de verdade, a caixa correspondente a uma variável é uma **posição da memória do computador**.
  - Uma variável possui **nome, tipo** e **conteúdo**.

Variável (continuação)

- O **nome** de uma variável **deve ser único**, isto é, identificar, de forma única, a variável no algoritmo.
- O **tipo** de uma variável define os **valores que podem ser armazenados** na variável.
- O **conteúdo** de uma variável é o **valor que ela armazena**.
- O **ato de se criar uma variável** é conhecido como **declaração de variável**.
- Na linguagem Portugol, declaramos uma variável usando uma sentença da seguinte forma:

  <tipo> <nome> (onde tipo é o tipo da variável e nome é o nome da variável).

Associação de conteúdo

- Esta associação é denominada **definição** e deve ser realizada após a declaração da variável usando uma **instrução de leitura** ou um **comando de atribuição**.
- A instrução de leitura tem a forma

  leia (nome**)**
- A instrução de atribuição possui a forma

  nome **<-** valor
- Onde, nome é o nome de uma variável e valor é um valor do mesmo tipo de dados da variável.

Nome de variáveis

- Na linguagem Portugol, usamos as seguintes regras para criar um nome de variável:
  1. Nomes de variáveis devem possuir como **primeiro caractere uma letra ou o símbolo '_'** (sublinhado). Os demais caracteres, se algum, devem ser letras, números ou sublinhado.
  2. Nomes de variáveis **não podem ser iguais a palavras reservadas**.
  3. Nomes de variáveis podem ter, **no máximo**, **127 caracteres**.
  4. Não há diferença entre letras maiúsculas e minúsculas em nomes de variáveis.

Exemplos de nomes de variáveis

- **VÁLIDOS**:
  - **_12234**
  - **fruta**
  - **x123**
- **NÃO-VÁLIDOS**:
  - **maria bonita**
  - **pi**
  - **fru?ta**
  - **1xed**

Exercícios

1. Escreva a declaração de uma variável do tipo real de nome x.
2. Escreva a declaração de uma variável do tipo caractere de nome carro.
3. Escreva a instrução de atribuição que atribui o valor 2.3 à variável do problema 1.
4. Escreva a instrução de atribuição que atribui o valor "corsa" à variável do problema 2.

Resolução dos exercícios

1. Escreva a **declaração** de uma variável do tipo **real** de nome **x**.
2. Escreva a **declaração** de uma variável do tipo **caractere** de nome **carro**.
3. Escreva a **instrução de atribuição** que atribui o valor **2.3** à **variável do problema 1**.
4. Escreva a **instrução de atribuição** que atribui o valor "**corsa**" à variável do **problema 2**.

Resolução dos exercícios

1. Escreva a **declaração** de uma variável do tipo **real** de nome **x**.

   real x

2. Escreva a **declaração** de uma variável do tipo **caractere** de nome **carro**.

   caractere carro

3. Escreva a **instrução de atribuição** que atribui o valor **2.3** à **variável do problema 1**.

   x ← 2.3

4. Escreva a **instrução de atribuição** que atribui o valor "**corsa**" à variável do **problema 2**.

   carro ← “corsa”

Exercícios

1. Quais dos seguintes nomes são válidos como nomes de variáveis?

   (a) xyz_2
   (b) _
   (c) ____
   (d) x123
   (e) 1_2
   (f) 123y
   (g) salute!
   (h) x_y_1
   (i) 34
   (j) meucarro

Palavras reservadas no Portugol

- Palavra reservada é uma palavra que possui um significado especial para a linguagem Portugol.
  - Em geral, uma palavra reservada identifica uma instrução.

abs	**eco**	**inicio**	**pi**
aleatorio	**enquanto**	**int**	**pos**
algoritmo	**entao**	**interrompa**	**procedimento**
arcos	**escolha**	**leia**	**quad**
arcsen	**escreva**	**literal**	**radpgrau**
arctan	**exp**	**log**	**raizq**
arquivo	**faca**	**logico**	**rand**
asc	**fim**	**logn**	**randi**
ate	**falso**	**maiusc**	**repita**
caractere	**fimalgoritmo**	**mensagem**	**se**
caso	**fimenquanto**	**minusc**	**sen**
compr	**fimescolha**	**nao**	**senao**
copia	**fimfuncao**	**numerico**	**timer**
cos	**fimpara**	**numpcarac**	**tan**
cotan	**fimprocedimento**	**ou**	**var**
cronometro	**fimrepita**	**outrocaso**	**verdadeiro**
debug	**fimse**	**para**	**xou**
declare	**funcao**	**passo**	
e	**grauprad**	**pausa**	

Métodos para a construção de algoritmo

- Compreender o problema
- Identificar os dados de entrada e objetos desse cenário-problema
- Definir o processamento
- Identificar/definir os dados de saída
- Construir o algoritmo
- Testar o algoritmo

Paradigmas de programação

- Paradigma $\rightarrow$ algo que serve de exemplo geral ou de modelo (padrão).
- Em programação $\rightarrow$ relacionado à **forma de pensar** do programador e como ele **busca a solução** para os problemas apresentados utilizando linguagem de programação.
  - Paradigma *versus* utilização do ferramental disponível.
  - Exemplos: estruturado, orientado a objetos, lógico, funcional, etc.

Paradigmas de programação

- Estruturado (imperativo) $\rightarrow$ **divide** um **problema** complexo em **pequenas partes** mais simples.
  - Qualquer problema pode ser resolvido utilizando três: **sequencial**, **condicional, iterativa** (repetição).
  - Modularização e parametrização.
- Orientado a objetos (OO) $\rightarrow$ o **problema** como uma **coleção de objetos** interagindo por meio de **trocas de mensagem**.
  - Procurar objetos $\rightarrow$ determinar características e responsabilidades $\rightarrow$ estabelecer como ocorrerá a interação entre os objetos