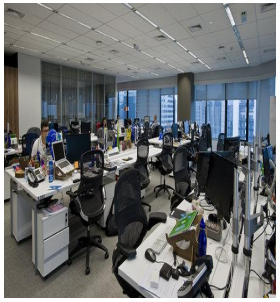
Arquitetura e Organização de Computadores

PROCESSAMENTO ELETRÔNICO DE DADOS – P.E.D

Entrada: Dados 1ª Etapa	Processamento: Dados 2ª Etapa	Resultado: informação 3ª Etapa

Informação □ subentende dados organizados (segundo uma orientação específica) para o atendimento ou emprego de uma pessoa ou grupo que os recebe.

Como o conhecimento e a tomada de decisão são importantes em várias áreas e em diferentes níveis hierárquicos de uma organização, a **informação** para uma determinada pessoa ou grupo pode ser considerado como um dado para outra.



Etapas - P.E.D de ítens do estoque de uma empresa

1ª Etapa □**Atualização** das informações de estoque para uso do almoxarifado e, nesse caso, os dados (de entrada) são ítens **recebidos** e **retirados** em um dia, bem como a posição do dia anterior.

2ª Etapa □ O **processamento** consistirá, basicamente, em operações aritméticas de adição e subtração. produzir informações para um outro nível de **tomada de decisão**.

3ª Etapa □ O **resultado** (de saída), obtêm-se informações sobre a nova posição do estoque.

Arquitetura e Organização de Computadores

PROCESSAMENTO ELETRÔNICO DE DADOS – P.E.D

Entrada: Dados	Processamento: Dados	Resultado: informação
Digitação do programa e dos dados	Cálculo e Testes	Impressão dos Resultados

Em geral, um sistema de processamento de dados compreende duas partes: o sistema de computação (o computador e os programas básicos) e os sistemas de aplicação. O primeiro, normalmente fornecido completo pelo fabricante, e os últimos, desenvolvidos pelo usuário ou por terceiros, especificamente dedicados à aplicação de interesse do usuário.

Qualquer processamento de dados requer a **execução** de uma série de etapas, que podem ser realizadas de forma manual ou automática por um **computador**. Tais etapas, elaboradas e executadas **passo** a **passo**, constituem o que se chama programa. Cada um dos passos mencionados é uma diferente **instrução**, ou ordem de comando, dada ao hardware, objetivando a realização de uma determinada ação (uma operação aritmética, uma transferência de informação etc.). O programa é o conjunto de instruções.

Algoritmo para soma de 100 números e imprimir o resultado:

I. Escrever e guardar N=0 e SOMA=0
2. Ler número da entrada
3. Somar valor do número ao de SOMA e guardar resultado como SOMA
4. Somar 1 ao valor de N e guardar resultado como novo N
5. Se valor de N for menor que 100, então passar para item 2
6. Senão : imprimir valor de SOMA
7. Parar

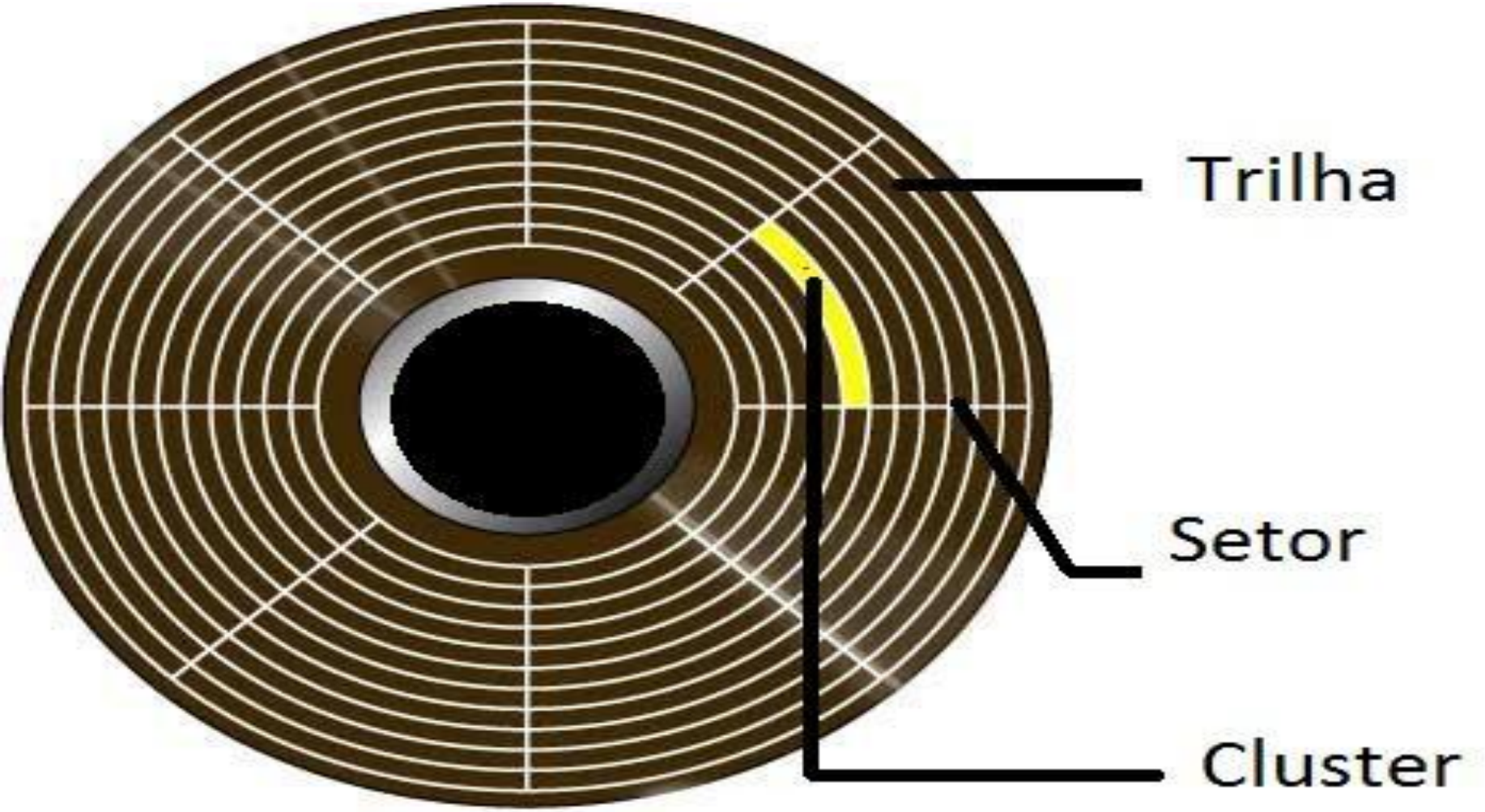
PROCESSAMENTO ELETRÔNICO DE DADOS – P.E.D

Entrada: Dados	Processamento: Dados	Resultado: informação
Digitação do programa e dos dados	Cálculo e Testes	Impressão dos Resultados

As **etapas** de um **algoritmo** são as **instruções** que deverão ser executadas por uma máquina (quando falamos de computadores); o **conjunto de instruções** constitui o que chamamos de **programa**. Um **programa** de computador é a formalização de um algoritmo em **linguagem inteligível** pelo computador.

Todo **dado** coletado pelos computadores, as **instruções** por ele executadas, bem como os **resultados** de um **processamento** são sempre constituídos de conjuntos ordenados de **zeros e uns.**

No entanto, essa linguagem, chamada de **linguagem de máquina**, é, para os seres humanos, tediosa de manipular, difícil de compreender e fácil de acarretar erros. Por essa razão, foram desenvolvidas outras linguagens, mais próximas do entendimento dos operadores, genericamente chamadas linguagens de programação. Atualmente, há dezenas dessas linguagens, tais como: Cobol, PL/I, Pascal, Fortran, Basic, Lisp, Assembly, C, C++, Java e etc.



Circuito – lei de Ohm

Fórmula da lei de Ohm:

$$V = I \times R$$

Manipulando a fórmula temos suas variantes:

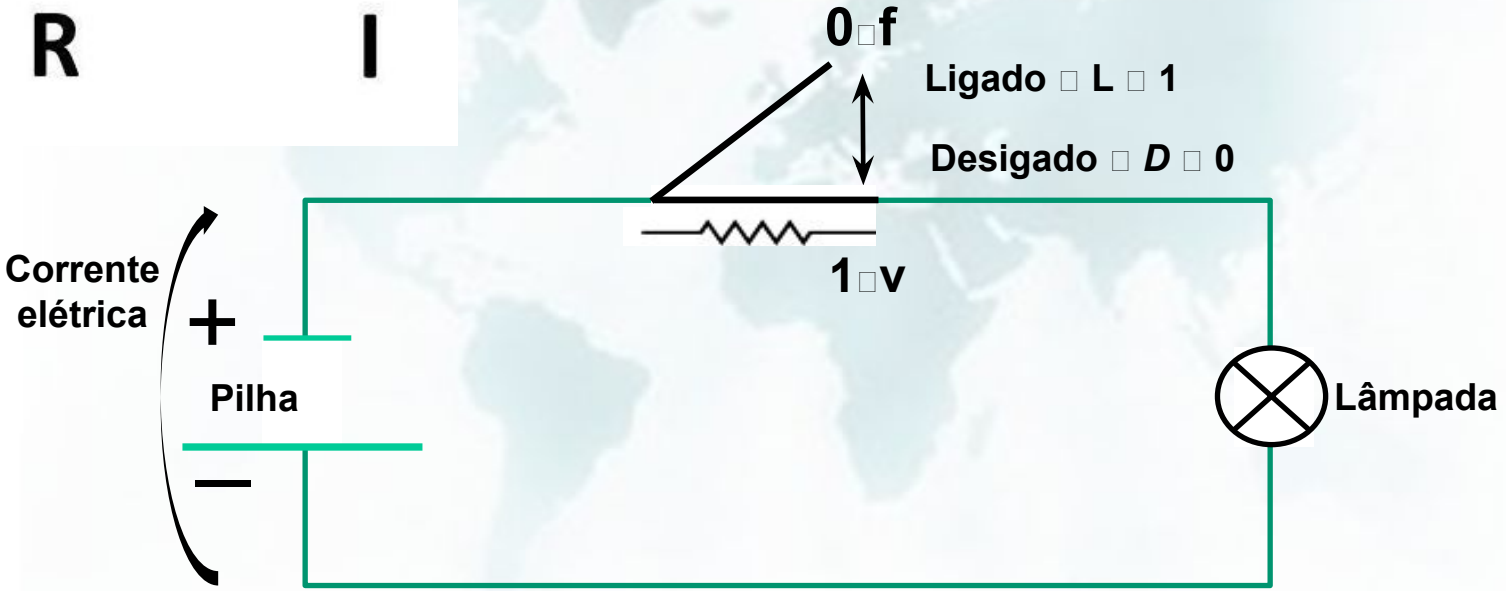
$$I = \frac{V}{R} \qquad R = \frac{V}{I}$$

U = R x I

U → V - Volts

R → Ω - Resistencia

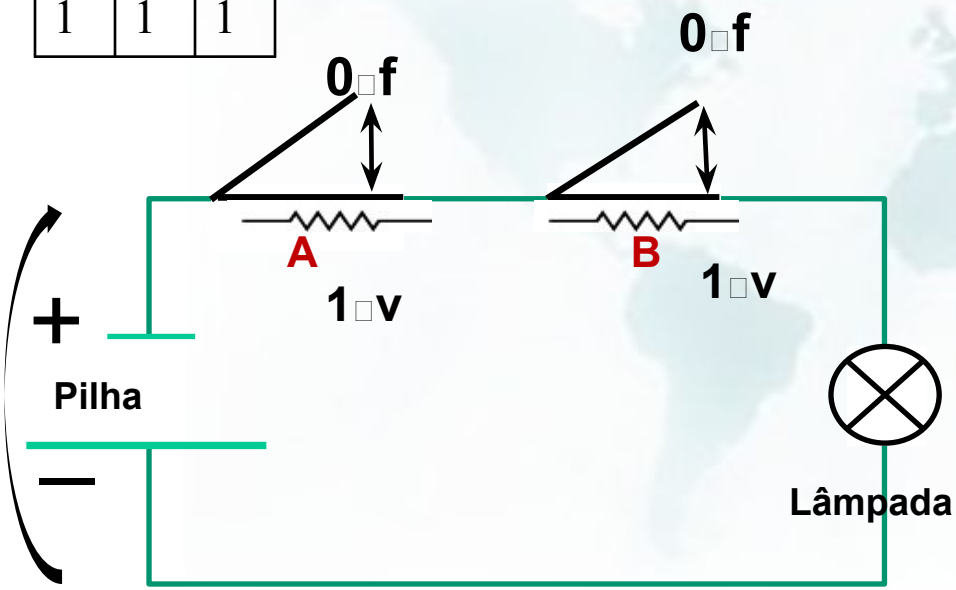
I → a - Ampere



Portas Lógicas - And

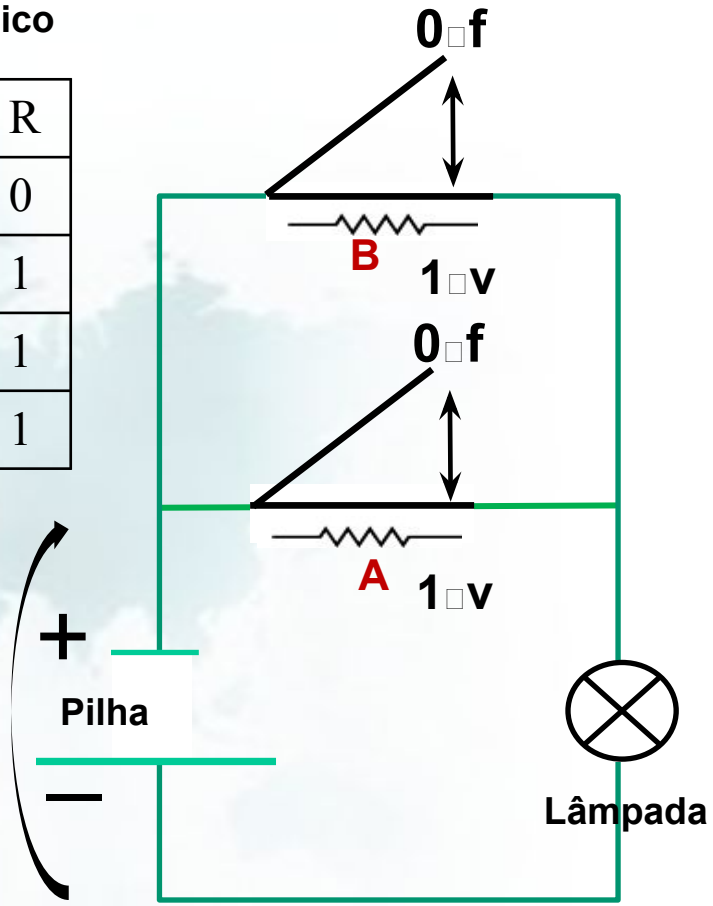
And - Lógico

A	B	R
0	0	0
0	1	0
1	0	0
1	1	1



Or - Lógico

A	B	R
0	0	0
0	1	1
1	0	1
1	1	1



Código Hexadecimal

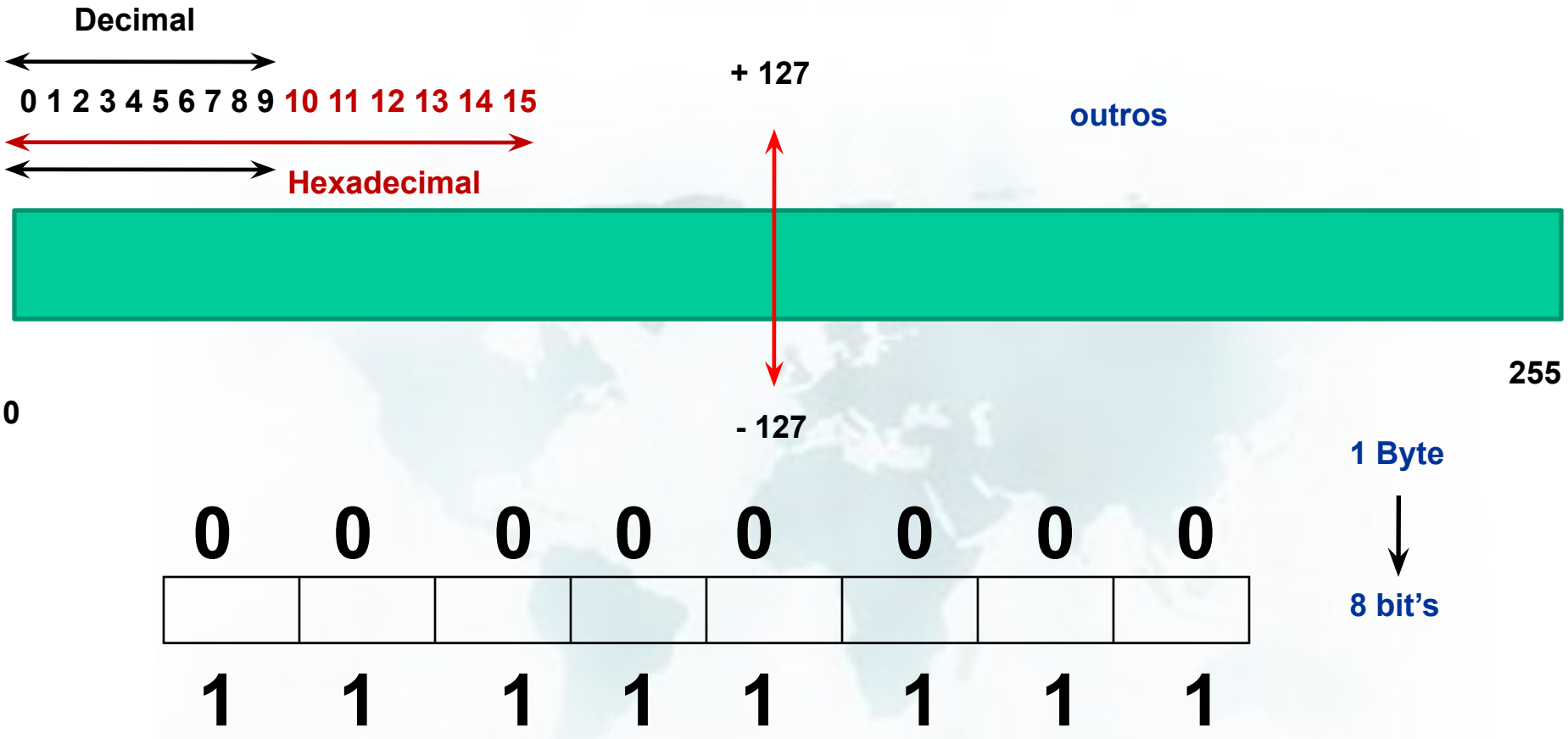
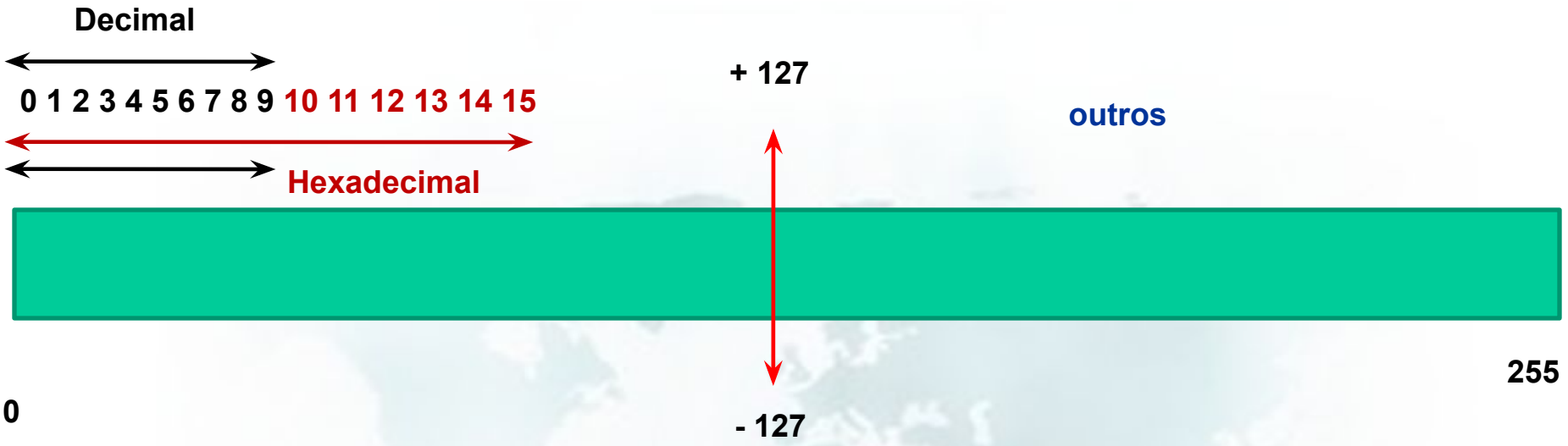


Tabela ASCII



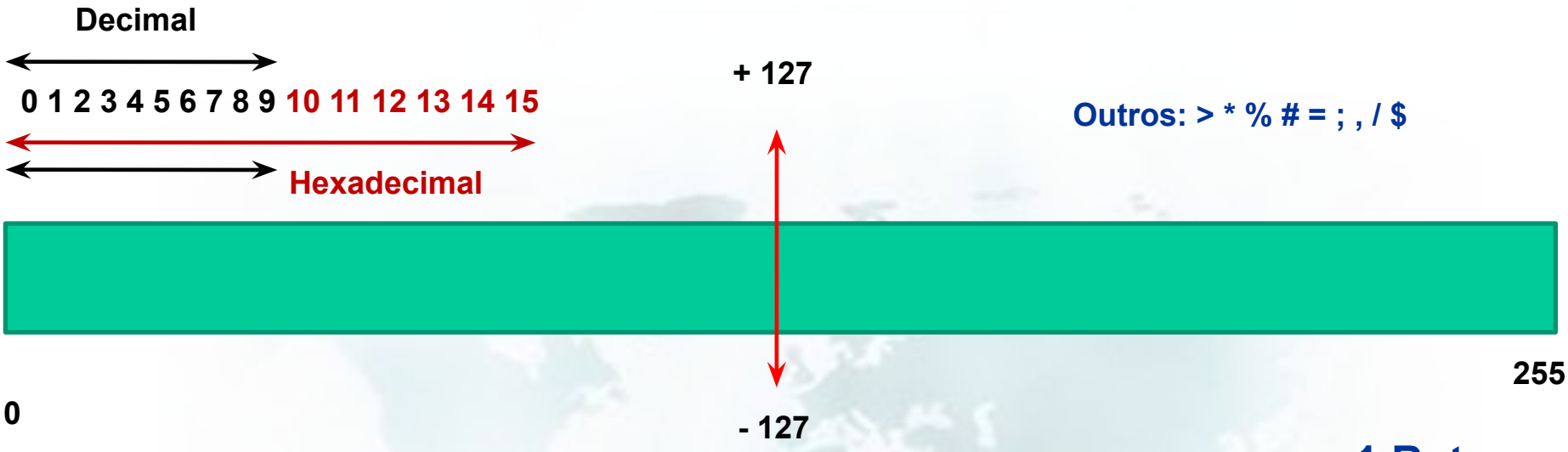
0 1 2 3 4 5 6 7 8 9 A B C D E F

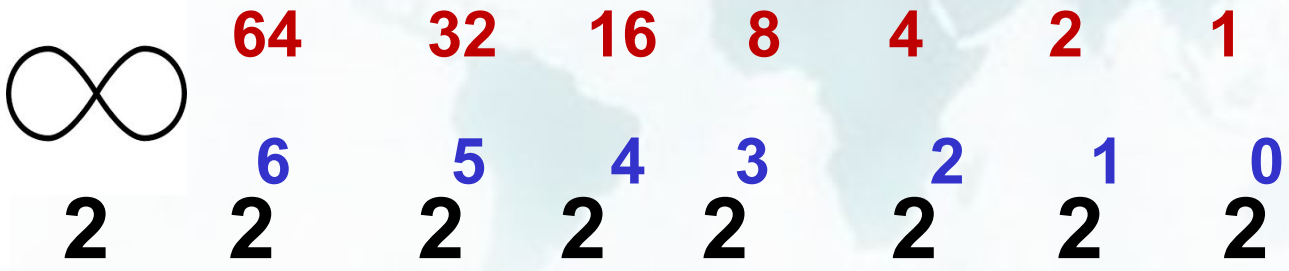
Hexadecimal

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

Decimal

Código Hexadecimal





Conversão de Base Binário□Decimal

Fatec Zona Sul
Centro Paula Souza
Governo do Estado São Paulo

Foco no essencial

1 Byte

8 bit's

Abstração

Porque sabemos que é uma zebra???

Lista

Conversão de Base Binário□Decimal

∞	64	32	16	8	4	2	1
2	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}

Conversão de Base Binário□Decimal

8	4	2	1

32 16 8 4 2 1

2^5 2^4 2^3 2^2 2^1 2^0

2 2 2 2 2 2

Tabela ASCII

Tabela ASCII A sigla ASCII deriva de American Standard Code for Information Interchange, ou seja, Código no Padrão Americano para Troca de Informações.

Como os computadores lidam apenas com números, um código ASCII é a representação numérica dos caracteres. Quando os computadores venceram a barreira dos 8 bits, a tabela ASCII pôde ser ampliada para receber mais 128 códigos. Esta tabela ficou conhecida como caracteres ASCII estendidos (ou ampliados).

Tabela ASCII

Fatec Zona Sul

Centro Paula Souza

Governo do Estado São Paulo

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	[NULL]	32	20	[SPACE]	64	40	@	96	60	`
1	1	[START OF HEADING]	33	21	!	65	41	A	97	61	a
2	2	[START OF TEXT]	34	22	"	66	42	B	98	62	b
3	3	[END OF TEXT]	35	23	#	67	43	C	99	63	c
4	4	[END OF TRANSMISSION]	36	24	$	68	44	D	100	64	d
5	5	[ENQUIRY]	37	25	%	69	45	E	101	65	e
6	6	[ACKNOWLEDGE]	38	26	&	70	46	F	102	66	f
7	7	[BELL]	39	27	'	71	47	G	103	67	g
8	8	[BACKSPACE]	40	28	(	72	48	H	104	68	h
9	9	[HORIZONTAL TAB]	41	29	)	73	49	I	105	69	i
10	A	[LINE FEED]	42	2A	*	74	4A	J	106	6A	j
11	B	[VERTICAL TAB]	43	2B	+	75	4B	K	107	6B	k
12	C	[FORM FEED]	44	2C	,	76	4C	L	108	6C	l
13	D	[CARRIAGE RETURN]	45	2D	-	77	4D	M	109	6D	m
14	E	[SHIFT OUT]	46	2E	.	78	4E	N	110	6E	n
15	F	[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	[NEGATIVE 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	[ENG OF 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	[SUBSTITUTE]	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	[DEL]

Tabela ASCII

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	

Codigo Hexadecimal - \x

Linguagem C

#include <stdio.h> .h->Biblioteca padrão std-> standart i/o-> input / output

#include <conio.h> .h->Biblioteca console con-> console i/o-> input / output

main() -> Função principal

{ -> Inicio

-> Corpo do programa

} -> Fim

Codigo Hexadecimal - \x

Linguagem C

```
#include <stdio.h>
#include <conio.h>

main()
{
    printf("\n\n");
    printf("\n\xC9\xCD\xBB");
    printf( "\n\xBA\XBA");
    printf("\n\xC8\xCD\xBC");
        getch();
}
```

Codigo Hexadecimal - \x Linguagem C

```
#include <stdio.h>
#include <conio.h>
main()
{
    printf("\n\n");
    printf("\n\xDC\XDC\XDB\xDB\xDB\xDB\xDC\xDC");
    printf( "\n\XDFO\xDF\xDF\xDF\xDFBA\XDFOO\XDF");
    printf("\n\\n");
    printf("\n\xDC\XDC\xDB\xDB\xDB\xDB\xDB\Xdb\xDB");
    printf( "\n\XDFO\xDF\xDF\xDF\xDFO\xDFOO\XDF");
    printf("\n\\n");
        getch();
}
```

```
#include <stdio.h>
#include <conio.h>

main()
{
     int x;
     puts(“DIGITE UM VALOR QUALQUER”);
     scanf(“%d”,&x);
     puts(“O valor digitado foi: ”);
     printf(“%d\n”,x);
          getch();
}
```

Código Hexadecimal Pausa

```
main()
{
    int  x;
    for(x=1; x<20;x++)
    {
        printf("\xDB");
        sleep(20);
    }
   getch();
}
```

Código Hexadecimal Som

```
main()
{
    int  x;
    for(x=1; x<20;x++)
    {
        printf("\x7");
    }
}
```

Código Hexadecimal Funções-Caixa

```
#include <stdio.h>
#include <conio.h>
main()
{
  linha();
  printf("\xDB Um programa da galera de ADS \xDB \n);
  linha();
  getch();
}
```

```
linha()
{
    int  x;
    for(x=1; x<=20;x++)
    {
        printf("\xDB");
        sleep(20);
    }
}
```

Código Hexadecimal Funções-Som

```
#include <stdio.h>
#include <conio.h>
      main()
      {
        som();
        printf("\xDB Digite um caractere:" );
        getche();
       som();
      }
```

```
som()
{
   int  y;
   printf("\x7");
    for(y=1; y<5000;y++)
   {
     printf("\x7");
    }
}
```

Código Hexadecimal Funções-Planta

```
#include <stdio.h>
#include <conio.h>
main()
{
  printf("\n Sala \n" );
  retangulo(22,13);
  printf("\n Cozinha\n" );
  retangulo(16,16);
  printf("\n Banheiro\n" );
  retangulo(6,8);
  printf("\n Quarto\n" );
  retangulo(12,12);
}
```

```
retangulo(largura,altura)
 int largura,altura;
{
  int j,k;
  largura /=2;
  altura /=4;
  for(j=1;j<=altura;j++)
  {
   printf("\t \t" );
    for(k=1;j<=largura;k++)
    {
      printf("\xDB");
      printf("\n");
    }
  }
}
```

FÓRMULA CONDICIONAL SE

Sintaxe: **SE(teste_lógico; valor_se_verdadeiro; valor_se_falso)**
Teste_lógico é qualquer valor ou expressão que pode ser avaliada como VERDADEIRO ou FALSO.

Valor_se_verdadeiro: é o valor fornecido se teste_lógico for VERDADEIRO. Se teste_lógico for VERDADEIRO e valor_se_verdadeiro for omitido, VERDADEIRO será fornecido.

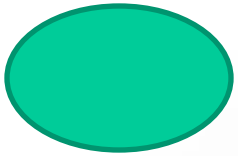
Valor_se_falso: é o valor fornecido se teste_lógico for FALSO. Se teste_lógico for FALSO e valor_se_falso for omitido, FALSO será fornecido.

Exemplos:

=SE(B2>C2;"Acima do orçamento";"OK")

=SE(Média>89;"A";SE(Média>79;"B"; SE(Média>69;"C";SE(Média>59;"D";"F"))))

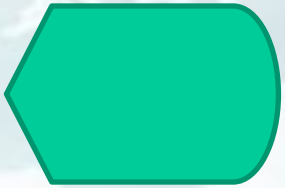
No exemplo anterior, a segunda instrução SE também é o argumento valor_se_falso para a primeira instrução SE. Da mesma maneira, a terceira instrução SE é o argumento valor_se_falso para a segunda instrução SE. Por exemplo, se o primeiro teste_lógico (Média>89) for VERDADEIRO, "A" será fornecido. Se o primeiro teste_lógico for FALSO, a segunda instrução SE é avaliada e assim por diante.



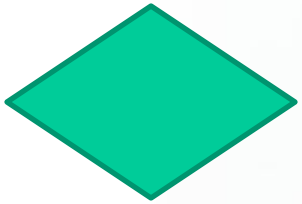
- Inicio e fim do programa

- Processamento de dados

- Entrada de dados



- Exibição de dados



- Tomada de decisão