

```
1
2
3  Algoritmo 'Lógica de' {
4
5      [ Programação 1 ]
6
7
8
9
10
11
12 }
13
14
```

Unidade 01

Prof. Daniel Caixeta



```
1  'Sumário' {
2
3      </01> 0 início [...]
4          < 1.1. Uma breve introdução. 1.2. Alguns conceitos [...]. >
5
6      </02> Como a máquina vê [...]
7          < 2.1. Bits e Bytes. 2.2. A representação [...]. >
8
9      </03> Como a máquina pensa [...]
10          < 3.1. Sistema posicional. > < 3.2. As bases [...]. 3.3. Conversão entre
11          bases 2, 8 e 16. 3.4. Conversão da base decimal (10) para binária (2). >
12
13      </04> A Lógica binária
14          < 4.1. Sistemas dicotômicos e a Lógica de Boole. 4.2. A lógica binária [...]. >
15
16      Referência
17  }
```

1 01 {

2
3
4
5 [Iniciando ]

6
7
8 < Introdução aos estudos [...] >

9
10
11
12 }
13
14

1.1. Uma breve introdução

- Os computadores desempenham um papel crucial na sociedade atual, especialmente em áreas como comunicação, comércio eletrônico, redes sociais, análise de comportamento e tomada de decisões.
- “Quase tudo” é computacionalmente viável, e por meio de algoritmos, esses processos estão se tornando cada vez mais autônomos e automatizados.
- Então a pergunta é:

O que poderia vir após a Era Digital?



1.2. Alguns conceitos [...]

- A informática pode ser entendida como a ciência que estuda o conjunto de informações e conhecimentos por meios digitais.

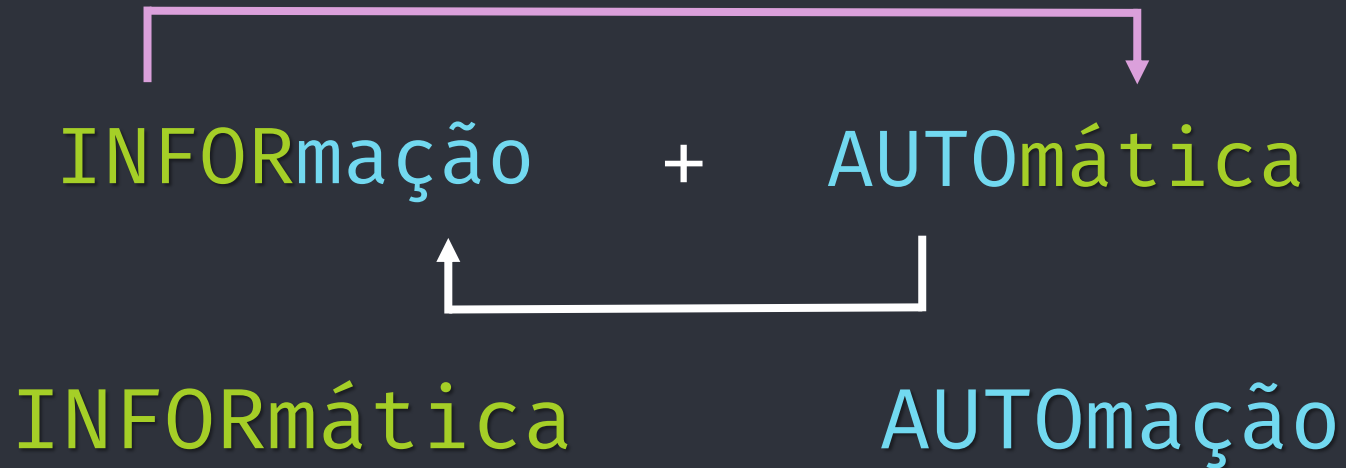


Figura 1. Informática e Automação - Conceitos.

- A informática descreve o conjunto de disciplinas e ciências que, por meio de suas interações, desenvolvem soluções para problemas relacionados à computação e ao processamento de informações.



Figura 2. Tarefas realizadas por máquinas/computadores.

- A computação pode ser definida como a busca de solução para um problema a partir de entradas (*inputs*), de forma a obter resultados (*outputs*) depois de processado os dados através de um algoritmo.

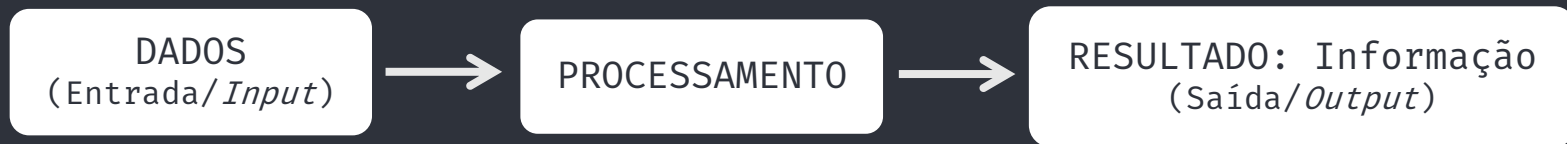
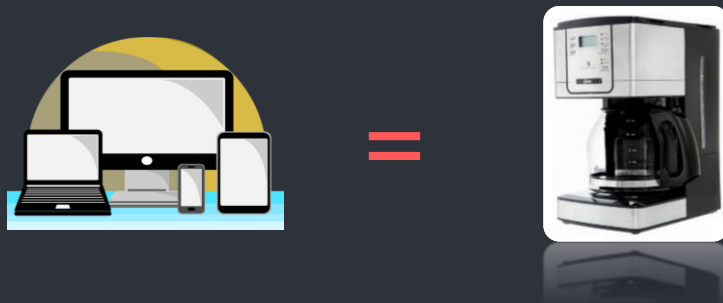


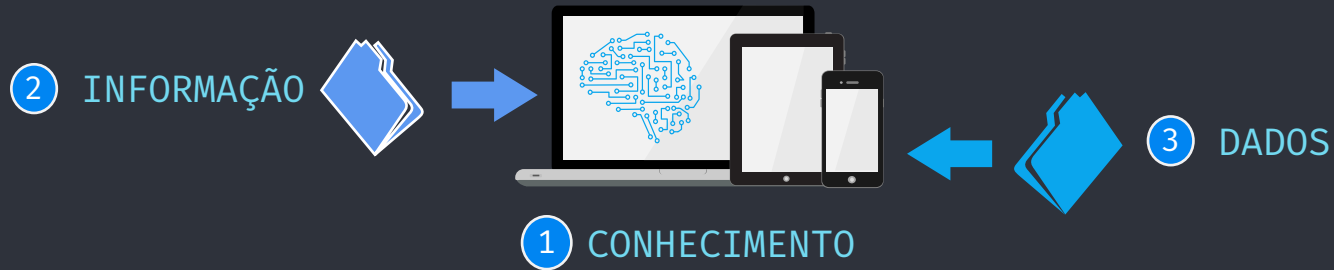
Figura 3. Etapas básicas de um processamento de dados. (MONTEIRO, 2014).

- Por milênios, a computação foi executada com lápis e papel, com giz e ardósia, ou mentalmente, por vezes com o auxílio de tabelas ou utensílios artesanais.

- A palavra computador significa aquele que faz cálculos, seja ele, pessoa ou máquina.
- Já para Monteiro (2014), trata-se de uma máquina (conjunto de partes eletrônicas e eletromecânicas) capaz de coletar, manipular e fornecer resultados informações para um ou mais objetivos (ver Figura 3).
- Por se tratar de uma máquina programável, essa pode executar diversas rotinas conforme programado.



- Já dados e informações podem ser tratados como sinônimos ou não. Dados é a matéria-prima, e informação é o resultado do processamento.
- Entende-se também que a informação consiste em dados organizados para cumprir um propósito específico, podendo ser empregada por indivíduos, usuários ou grupos.





- O próprio avanço tecnológico na área de telecomunicação também contribuiu para o crescimento do uso de computadores, *notebooks*, *tablets*, *smartphones*, etc., visto que permitiu a interligação entre as redes de comunicação de dados.
- O principal exemplo é a *internet*, que permite a comunicação entre qualquer dispositivo, em qualquer parte do planeta.



Figura 5. A *internet* - Rede de comunicação mundial.

02 {

[Como a máquina ]

< Dos *bits* aos *bytes* [...] >

}

2.1. Bits e Bytes

- Um *bit* ou dígito binário (*binary digit*), é:
[...] a unidade básica que os computadores e sistemas digitais utilizam para trabalhar, e pode assumir apenas dois valores, 0 ou 1. (FARIAS, 2013).
- Já um *byte* é uma sequência de 8 *bits*. Portanto, o *byte* é a menor unidade de armazenamento utilizada pelos computadores. Quer dizer, que jamais conseguiremos salvar menos do que 8 *bits* em uma informação.

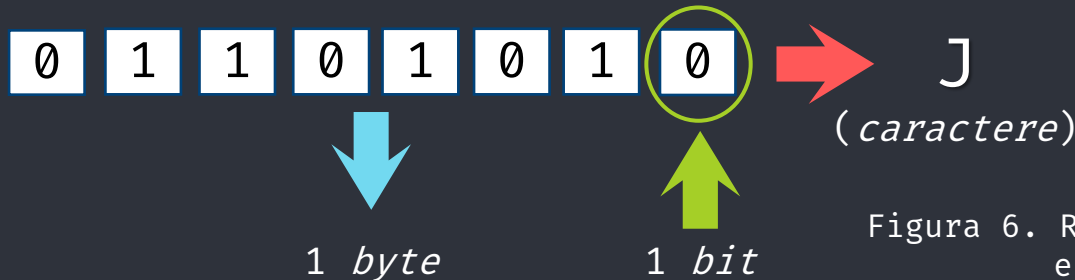


Figura 6. Representação de um *byte* e um *bit*.

2.2. A representação [...]





- Um *bit* só pode assumir dois valores (0 ou 1), portanto, só será possível representar exatamente dois estados distintos. (FARIAS, 2013).

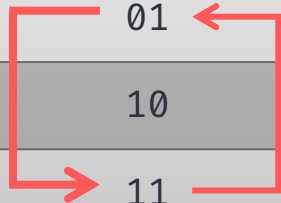
Tabela 1. Representação com um *bit*.

Bit	Porta	Lâmpada	Detector de movimento	Estado civil
0	Fechada	Desligada	Sem movimento	Solteiro
1	Aberta	Ligada	Com movimento	Casado

- Para representar mais de dois valores distintos precisamos de uma sequência de *bits* maior. Na Tabela 2, é apresentado exemplos utilizando uma sequência de 2 *bits*, obtendo assim 4 possibilidades.

Tabela 2. Representação com dois *bits*.

Sequência de bits	Semáforo
00	Desligado 
01	Pare 
10	Atenção 
11	Siga 



- Segundo Farias (2013), o número de possibilidades diferentes que podemos representar depende do tamanho da sequência que estamos utilizando, mais precisamente:

$$P = 2^n, \text{ onde } n = \text{tamanho de } \textit{bits}.$$

- Exemplos:

$$\begin{array}{llll} 2^1 = 2 & 2^2 = 4 & 2^3 = 8 & 2^4 = 16 \\ 2^5 = 32 & 2^6 = 64 & 2^7 = 128 & 2^8 = 256 \\ & & & (1 \text{ byte}) \end{array}$$

$$16 \text{ bits} = 65.535$$

$$32 \text{ bits} = 4.294.967.295$$

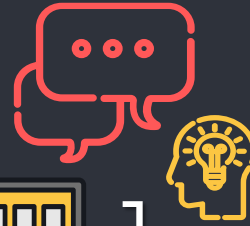
$$64 \text{ bits} = 18.446.744.073.709.551.615$$

03 {

[Como a máquina



]



< Como o computador pensa e executa? >

}

3.1. Sistema posicional

- O método de numeração de quantidades que adotamos, utiliza um sistema de numeração posicional.
- Significa que a posição ocupada por cada algarismo em um número altera seu valor de uma potência decimal (base 10) para cada casa à esquerda. Vejamos o exemplo abaixo:

$$128_{10} = \underbrace{1 \times 10^2}_{100 \text{ Centena}} + \underbrace{2 \times 10^1}_{20 \text{ Dezena}} + \underbrace{8 \times 10^0}_{8 \text{ Unidade}}$$

3.2. As bases [...]

- A base de um sistema é a quantidade de algarismos disponíveis em sua representação.
- Citemos alguns exemplos:

</1> Base decimais (b_{10}) \rightarrow 0 1 2 3 4 5 6 7 8 9

</2> Base binária (b_2) \rightarrow 0 1

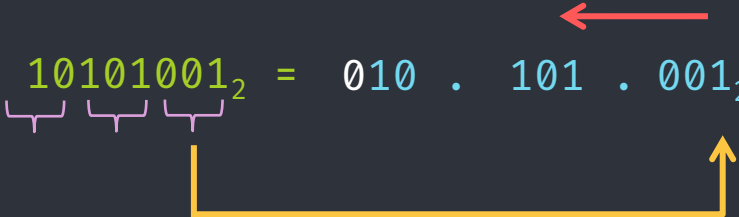
</3> Base octal (b_8) \rightarrow 0 1 2 3 4 5 6 7

</4> Base hexadecimal (b_{16}) \rightarrow 0 1 2 3 4 5 6 7 8 9 A B C D E F
(10 algarismos + 6 símbolos)

3.3. Conversão entre bases 2, 8 e 16

- As conversões mais simples são as que envolvem bases que são potências entre si. (FARIAS, 2013).
- Leva-se em consideração que $2^3 = 8$ e $2^4 = 16$.
- Exemplifiquemos a conversão 2^3 , que funciona da seguinte forma:
 - 1º. Separa-se os algarismos de um número binário em grupos de três (começando sempre da direita para a esquerda).
 - 2º. Converte cada grupo de três algarismos por seu equivalente em octal.
- Vejamos:

1
2 $10101001_2 = 010 . 101 . 001_2$
3
4
5
6
7
8
9
10
11
12
13
14



• Olhando a tabela de conversão direta temos:

$$010_2 = 2_8 \quad 101_2 = 5_8 \quad 001_2 = 1_8 \quad 251_8$$

$$10101001_2 = 251_8$$

Bin.	Octal
000	0
001 →	1
010 →	2
011	3
100	4
101 →	5
110	6
111	7

- Agora a conversão entre as bases 2 e 16.
- Como $2^4 = 16$, seguimos o mesmo processo anterior, bastando agora separarmos em grupos com quatro algarismos e converter cada grupo seguindo a Tabela ao lado. Por exemplo:

$$11010101101_2 = \underbrace{0110}_2 \cdot \underbrace{1010}_2 \cdot \underbrace{1101}_2$$

$$0110_2 = 6_{16}$$

$$1010_2 = A_{16}$$

$$1101_2 = D_{16}$$

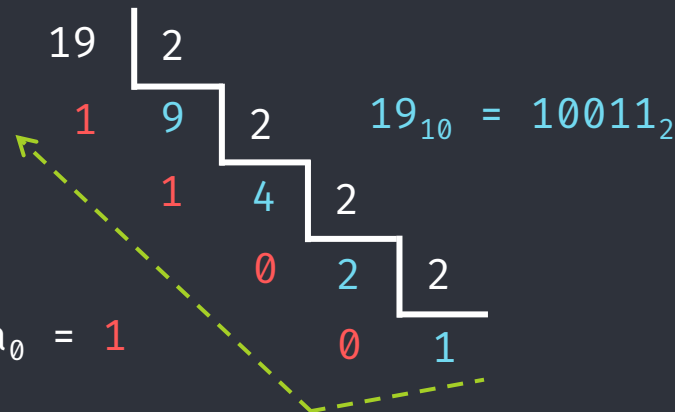
$$011010101101_2 = 6AD_{16}$$

Bin.	Hexa.	Bin.	Hexa.
0000	0	1000	8
0001	1	1001	9
0010	2	1010	A
0011	3	1011	B
0100	4	1100	C
0101	5	1101	D
0110	6	1110	E
0111	7	1111	F

3.4. Conversão da base 10 para bin.

- 1
- 2 • Já a conversão de números da base 10 para uma base qual-
- 3 quer, emprega-se algoritmos que serão de ordem inversa das
- 4 anteriores.
- 5 • O número decimal será dividido sucessivas vezes pela base
- 6 2, o resto de cada divisão ocupará sucessivamente as posi-
- 7 ções de ordem 0 e 1, e assim por diante, até que o resto da
- 8 última divisão resulte em quociente 0.
- 9 • Esse último quociente irá ocupar a posição mais alta ordem.
- 10
- 11
- 12
- 13
- 14

- Exemplo: Converta o numero 19_{10} para a base 2.



$$a_4 = 1 \quad a_3 = 0 \quad a_2 = 0 \quad a_1 = 1 \quad a_0 = 1$$

- Usando a conversão anterior como prova real, temos:

$$10011_2 = (1 \times 2^4) + (0 \times 2^3) + (0 \times 2^2) + (1 \times 2^1) + (1 \times 2^0) = 19_{10}$$

04 {

[A Lógica binária]

< Operadores e circuitos lógicos >

}

4.1. Sistemas dicotômicos e a Álgebra de Boole

- Segundo Daghlia (2008), a maioria das situações mundo apresenta uma natureza dualística, caracterizada por dois estados que se excluem mutuamente.

Situações dualísticas.

Valor_1	Valor_2
1	0
Sim	Não
Dia	Noite
Preto	Branco
Ligado	Desligado

- Em 1854, George Boole formalizou o tratamento da lógica, chamada de Álgebra Booleana, e definida como um conjunto de operadores e axiomas, assumidos como verdadeiros, e sem a necessidade de prova. (Güntzel & Nascimento, 2001).
- Em 1938, C. E. Shannon aplicou esta álgebra para mostrar que os circuitos elétricos de chaveamento podem ser representadas com dois valores. (*ibidem*).
- Para Güntzel & Nascimento (2001), diferentemente da álgebra ordinária, onde as variáveis podem assumir valores no intervalo $(-\infty ; +\infty)$, as variáveis booleanas só podem assumir um número finito de valores.

0 ou 1

- Em particular, na álgebra booleana, cada variável pode assumir um dentre dois possíveis. Por exemplo:

Exemplo de operadores.

Operador	Valores
V ou F	Verdadeiro ou Falso
C ou E	Certo ou Errado
0 ou 1	–

- Portanto são sistemas dicotômicos. Computacionalmente dizendo, 0 e 1, no qual é também utilizada na eletrônica digital de circuitos.

4.2. A lógica binária: Operadores e Tabela Verdade

- Na lógica booleana, três operadores básicos (NOT, AND e OR) permitem a construção de circuitos lógicos para diversas operações computacionais. Os valores finitos que as variáveis podem assumir levam a um número limitado de estados para uma função booleana, permitindo sua descrição por meio de tabelas, chamadas de Tabelas Verdade, que listam todas as combinações de valores de entrada e suas saídas.

</1> Operador NOT

O operador NOT (negação binária) gera o complemento do operando: um *bit* 1 se o operando for 0 e 0 caso contrário, como mostrado na tabela ao lado, onde A é o *bit* de entrada e S é o resultado, ou *bit* de saída.

Tabela verdade do operador NOT

A	S ou A'
0	1
1	0

Tabela verdade do operador AND

A	B	S
0	0	0
0	1	0
1	0	0
1	1	1

</2> Operador AND (A . B)

O operador AND (conjunção binária) retorna um *bit* 1 apenas quando ambos os operandos são 1, como ilustrado na tabela ao lado, onde A e B são *bits* de entrada, e S é o resultado, ou *bit* de saída.

</3> Operador OR (A + B)

O operador OR retorna um *bit* 1 quando pelo menos um dos operandos é 1, como mostrado na tabela ao lado, onde A e B são os *bits* de entrada, e S é o resultado, ou *bit* de saída.

Tabela verdade do operador OR

A	B	S
0	0	0
0	1	1
1	0	1
1	1	1

Operador NOT

A	S ou A'
0	1
1	0

Operador AND

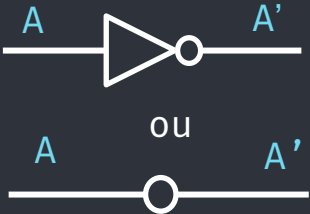
A	B	S
0	0	0
0	1	0
1	0	0
1	1	1

Operador OR

A	B	S
0	0	0
0	1	1
1	0	1
1	1	1

$(A \cdot B)$

$(A + B)$



Representação gráfica

</1> DAGHLIAN, Jacob. Lógica e Álgebra de Boole. 4ª ed. 12ª reimpr. São Paulo : Atlas, 2008.

</2> FARIAS, Gilberto. Introdução à computação. UFPB, 2013.

</3> GÜNTZEL, José Luís; NASCIMENTO, Francisco de Assis. Introdução aos Sistemas Digitais (v.2001/1). Disponível in:
<https://www.inf.ufsc.br/~j.guntzel/isd/isd.html>.

</4> TANENBAUM, Andrew S. AUSTIN, Todd. Organização Estruturada de Computadores. 6ª ed. Pearson, 2013.

</5> MONTEIRO, Mário A. Introdução à Organização dos Computadores. 5ª ed. LTC. 2014.

A 'Obrigado' Is {

Algoritmo & Linguagem de Programação 1

|
}

